



GumNaaM_Helpers



Mega File for MiD TerM

All Files in ONE FILE

(Handouts_waqar_moaz_junaaid_vu toper_orange_monkey)

Alhamdulillah GumNaaM_Helpers Bs..IT/CS ky Tamam Semesters ki tamam Books ki help krty hn.

Group Join krty hi description Read krna na bholyn

Semester Name	Group Link
Bs..IT/CS...smstr:1	https://chat.whatsapp.com/Ciya2X0DI0I8iQ14m0VNNj
Bs..IT/CS...smstr:2	https://chat.whatsapp.com/Kwh85sRK02z1Qk5B7aMR19
Bs..IT/CS...smstr:3	https://chat.whatsapp.com/ljlpdzi8pIA4IQVNUMmNYE
Bs..IT/CS...smstr:4	https://chat.whatsapp.com/CiGpHsSftir0oyhM47WfKh
Bs..IT/CS...smstr:5	https://chat.whatsapp.com/Hf7NWWhdW06uCW7Mbt1n4Db
Bs..IT/CS...smstr:6	https://chat.whatsapp.com/DyAoT7VFkwy8DuvrYDgm2L
Bs..IT/CS...smstr:7	https://chat.whatsapp.com/ENiyfIDRWatJFPbXCUGYNN
Bs..IT/CS...smstr:8	https://chat.whatsapp.com/Bnkk4n7utpdEuXazHLxRcA

...Paid Service Available for LMS Activities...

Only for those students who are so busy in office or work or job that they can't watch their LMS due to their Busyness

FOR MORE INFO. CONTACT US ONLY WHAT'S APP 0302-1500025

Assembly Language Programming Lecture Notes

Delivered by
Belal Hashmi

Compiled by
Junaid Haroon

Preface

Assembly language programming develops a very basic and low level understanding of the computer. In higher level languages there is a distance between the computer and the programmer. This is because higher level languages are designed to be closer and friendlier to the programmer, thereby creating distance with the machine. This distance is covered by translators called compilers and interpreters. The aim of programming in assembly language is to bypass these intermediates and talk directly with the computer.

There is a general impression that assembly language programming is a difficult chore and not everyone is capable enough to understand it. The reality is in contrast, as assembly language is a very simple subject. The wrong impression is created because it is very difficult to realize that the real computer can be so simple. Assembly language programming gives a freehand exposure to the computer and lets the programmer talk with it in its language. The only translator that remains between the programmer and the computer is there to symbolize the computer's numeric world for the ease of remembering.

To cover the practical aspects of assembly language programming, IBM PC based on Intel architecture will be used as an example. However this course will not be tied to a particular architecture as it is often done. In our view such an approach does not create versatile assembly language programmers. The concepts of assembly language that are common across all platforms will be developed in such a manner as to emphasize the basic low level understanding of the computer instead of the peculiarities of one particular architecture. Emphasis will be more on assembly language and less on the IBM PC.

Before attempting this course you should know basic digital logic operations of AND, OR, NOT etc. You should know binary numbers and their arithmetic. Apart from these basic concepts there is nothing much you need to know before this course. In fact if you are not an expert, you will learn assembly language quickly, as non-experts see things with simplicity and the basic beauty of assembly language is that it is exceptionally simple. Do not ever try to find a complication, as one will not be there. In assembly language what is written in the program is all that is there, no less and no more.

After successful completion of this course, you will be able to explain all the basic operations of the computer and in essence understand the psychology of the computer. Having seen the computer from so close, you will understand its limitations and its capabilities. Your logic will become fine grained and this is one of the basic objectives of teaching assembly language programming.

Then there is the question that why should we learn assembly language when there are higher level languages one better than the other; C, C++, Java, to name just a few, with a neat programming environment and a simple way to write programs. Then why do we need such a freehand with the computer that may be dangerous at times? The answer to this lies in a very simple example. Consider a translator translating from English to Japanese. The problem faced by the translator is that every language has its own vocabulary and grammar. He may need to translate a word into a sentence and destroy the beauty of the topic. And given that we do not know

Japanese, so we cannot verify that our intent was correctly conveyed or not. Compiler is such a translator, just a lot dumber, and having a scarce number of words in its target language, it is bound to produce a lot of garbage and unnecessary stuff as a result of its ignorance of our program logic. In normal programs such garbage is acceptable and the ease of programming overrides the loss in efficiency but there are a few situations where this loss is unbearable.

Think about a four color picture scanned at 300 dots per inch making 90000 pixels per square inch. Now a processing on this picture requires 360000 operations per square inch, one operation for each color of each pixel. A few extra instructions placed by the translator can cost hours of extra time. The only way to optimize this is to do it directly in assembly language. But this doesn't mean that the whole application has to be written in assembly language, which is almost never the case. It's only the performance critical part that is coded in assembly language to gain the few extra cycles that matter at that point.

Consider an arch just like the ones in mosques. It cannot be made of big stones alone as that would make the arch wildly jagged, not like the fine arch we are used to see. The fine grains of cement are used to smooth it to the desired level of perfection. This operation of smoothing is optimization. The core structure is built in a higher level language with the big blocks it provides and the corners that need optimization are smoothed with the fine grain of assembly language which allows extreme control.

Another use of assembly language is in a class of time critical systems called real time systems. Real time systems have time bound responses, with an upper limit of time on certain operations. For such precise timing requirement, we must keep the instructions in our total control. In higher level languages we cannot even tell how many computer instructions were actually used, but in assembly language we can have precise control over them. Any reasonable sized application or a serious development effort has nooks and corners where assembly language is needed. And at these corners if there is no assembly language, there can be no optimization and when there is no optimization, there is no beauty. Sometimes a useful application becomes useless just because of the carelessness of not working on these jagged corners.

The third major reason for learning assembly language and a major objective for teaching it is to produce fine grained logic in programmers. Just like big blocks cannot produce an arch, the big thick grained logic learnt in a higher level language cannot produce the beauty and fineness assembly language can deliver. Each and every grain of assembly language has a meaning; nothing is presumed (e.g. div and mul for input and out put of decimal number). You have to put together these grains, the minimum number of them to produce the desired outcome. Just like a "for" loop in a higher level language is a block construct and has a hundred things hidden in it, but using the grains of assembly language we do a similar operation with a number of grains but in the process understand the minute logic hidden beside that simple "for" construct.

Assembly language cannot be learnt by reading a book or by attending a course. It is a language that must be tasted and enjoyed. There is no other way to learn it. You will need to try every example, observe and verify the things you are told about it, and experiment a lot with the computer. Only then you will know and become able to appreciate how powerful, versatile, and simple this language is; the three properties that are hardly ever present together.

Whether you program in C/C++ or Java, or in any programming paradigm be it object oriented or declarative, everything has to boil down to the bits and bytes of assembly language before the computer can even understand it.

Table of Contents

Preface	i
Table of Contents	iii
1 Introduction to Assembly Language	1
1.1. Basic Computer Architecture	1
1.2. Registers	3
1.3. Instruction Groups	5
1.4. Intel iapx88 Architecture	6
1.5. History	6
1.6. Register Architecture	7
1.7. Our First Program	9
1.8. Segmented Memory Model	12
2 Addressing Modes	17
2.1. Data Declaration	17
2.2. Direct Addressing	17
2.3. Size Mismatch Errors	21
2.4. Register Indirect Addressing	22
2.5. Register + Offset Addressing	25
2.6. Segment Association	25
2.7. Address Wraparound	26
2.8. Addressing Modes Summary	27
3 Branching	31
3.1. Comparison and Conditions	31
3.2. Conditional Jumps	33
3.3. Unconditional Jump	36
3.4. Relative Addressing	37
3.5. Types of Jump	37
3.6. Sorting Example	38
4 Bit Manipulations	43
4.1. Multiplication Algorithm	43
4.2. Shifting and Rotations	43
4.3. Multiplication in Assembly Language	46
4.4. Extended Operations	47
4.5. Bitwise Logical Operations	50
4.6. Masking Operations	51
5 Subroutines	55
5.1. Program Flow	55
5.2. Our First Subroutine	57
5.3. Stack	59
5.4. Saving and Restoring Registers	62
5.5. Parameter Passing Through Stack	64
5.6. Local Variables	67
6 Display Memory	71

6.1. ASCII Codes	71
6.2. Display Memory Formation	72
6.3. Hello World in Assembly Language	74
6.4. Number Printing in Assembly	76
6.5. Screen Location Calculation	79
7 String Instructions	83
7.1. String Processing	83
7.2. STOS Example – Clearing the Screen	85
7.3. LODS Example – String Printing	86
7.4. SCAS Example – String Length	87
7.5. LES and LDS Example	89
7.6. MOVS Example – Screen Scrolling	90
7.7. CMPS Example – String Comparison	92
8 Software Interrupts	95
8.1. Interrupts	95
8.2. Hooking an Interrupt	98
8.3. BIOS and DOS Interrupts	99
9 Real Time Interrupts and Hardware Interfacing	105
9.1. Hardware Interrupts	105
9.2. I/O Ports	106
9.3. Terminate and Stay Resident	111
9.4. Programmable Interval Timer	114
9.5. Parallel Port	116
10 Debug Interrupts	125
10.1. Debugger using single step interrupt	125
10.2. Debugger using breakpoint interrupt	128
11 Multitasking	131
11.1. Concepts of Multitasking	131
11.2. Elaborate Multitasking	133
11.3. Multitasking Kernel as TSR	135
12 Video Services	141
12.1. BIOS Video Services	141
12.2. DOS Video Services	144
13 Secondary Storage	147
13.1. Physical Formation	147
13.2. Storage Access Using BIOS	148
13.3. Storage Access using DOS	153
13.4. Device Drivers	158
14 Serial Port Programming	163
14.1. Introduction	163
14.2. Serial Communication	165
15 Protected Mode Programming	167
15.1. Introduction	167
15.2. 32bit Programming	170
15.3. VESA Linear Frame Buffer	172
15.4. Interrupt Handling	174
16 Interfacing with High Level Languages	179

16.1. Calling Conventions	179
16.2. Calling C from Assembly	179
16.3. Calling Assembly from C	181
17 Comparison with Other Processors	183

17.1. Motorola 68K Processors	183
17.2. Sun SPARC Processor	184

Introduction to Assembly Language

1.1. BASIC COMPUTER ARCHITECTURE

Address, Data, and Control Buses

A computer system comprises of a processor, memory, and I/O devices. I/O is used for interfacing with the external world, while memory is the processor's internal world. Processor is the core in this picture and is responsible for performing operations. The operation of a computer can be fairly described with processor and memory only. I/O will be discussed in a later part of the course. Now the whole working of the computer is performing an operation by the processor on data, which resides in memory.

The scenario that the processor executes operations and the memory contains data elements requires a mechanism for the processor to read that data from the memory. "That data" in the previous sentence much be rigorously explained to the memory which is a dumb device. Just like a postman, who must be told the precise address on the letter, to inform him where the destination is located. Another significant point is that if we only want to read the data and not write it, then there must be a mechanism to inform the memory that we are interested in reading data and not writing it. Key points in the above discussion are:

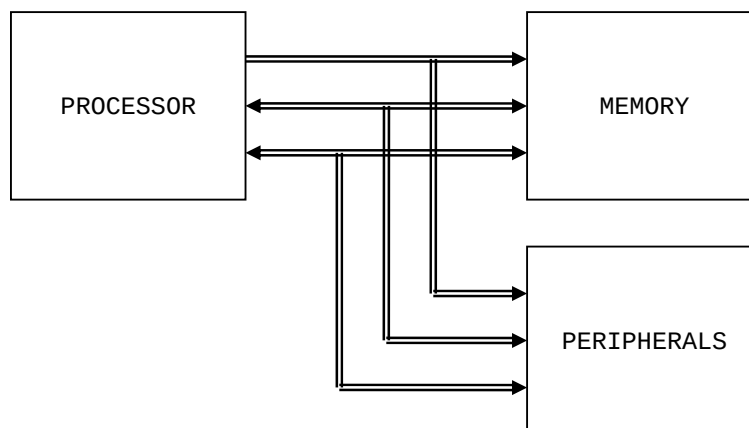
- There must be a mechanism to inform memory that we want to do the read operation
- There must be a mechanism to inform memory that we want to read precisely which element
- There must be a mechanism to transfer that data element from memory to processor

The group of bits that the processor uses to inform the memory about which element to read or write is collectively known as the *address bus*. Another important bus called the *data bus* is used to move the data from the memory to the processor in a read operation and from the processor to the memory in a write operation. The third group consists of miscellaneous independent lines used for control purposes. For example, one line of the bus is used to inform the memory about whether to do the read operation or the write operation. These lines are collectively known as the *control bus*.

These three buses are the eyes, nose, and ears of the processor. It uses them in a synchronized manner to perform a meaningful operation. Although the programmer specifies the meaningful operation, but to fulfill it the processor needs the collaboration of other units and peripherals. And that collaboration is made available using the three buses. This is the very basic description of a computer and it can be extended on the same lines to I/O but we are leaving it out just for simplicity for the moment.

The address bus is unidirectional and address always travels from processor to memory. This is because memory is a dumb device and cannot predict which element the processor at a particular instant of time needs. Data moves from both, processor to memory and memory to processor, so the data bus is bidirectional. Control bus is special and relatively complex, because different lines comprising it behave differently. Some take

information from the processor to a peripheral and some take information from the peripheral to the processor. There can be certain events outside the processor that are of its interest. To bring information about these events the data bus cannot be used as it is owned by the processor and will only be used when the processor grants permission to use it. Therefore certain processors provide control lines to bring such information to processor's notice in the control bus. Knowing these signals in detail is unnecessary but the general idea of the control bus must be conceived in full.



We take an example to explain the collaboration of the processor and memory using the address, control, and data buses. Consider that you want your uneducated servant to bring a book from the shelf. You order him to bring the fifth book from top of the shelf. All the data movement operations are hidden in this one sentence. Such a simple everyday phenomenon seen from this perspective explains the seemingly complex working of the three buses. We told the servant to “bring a book” and the one which is “fifth from top,” precise location even for the servant who is much more intelligent than our dumb memory. The dumb servant follows the steps one by one and the book is in your hand as a result. If however you just asked him for a book or you named the book, your uneducated servant will stand there gazing at you and the book will never come in your hand.

Even in this simplest of all examples, mathematics is there, “fifth from top.” Without a number the servant would not be able to locate the book. He is unable to understand your will. Then you tell him to put it with the seventh book on the right shelf. Precision is involved and only numbers are precise in this world. One will always be one and two will always be two. So we tell in the form of a number on the address bus which cell is needed out of say the 2000 cells in the whole memory.

A binary number is generated on the address bus, fifth, seventh, eighth, tenth; the cell which is needed. So the cell number is placed on the address bus. A memory cell is an n-bit location to store data, normally 8-bit also called a byte. The number of bits in a cell is called the *cell width*. The two dimensions, cell width and number of cells, define the memory completely just like the width and depth of a well defines it completely. 200 feet deep by 15 feet wide and the well is completely described. Similarly for memory we define two dimensions. The first dimension defines how many parallel bits are there in a single memory cell. The memory is called 8-bit or 16-bit for this reason and this is also the word size of the memory. This need not match the size of a processor word which has other parameters to define it. In general the memory cell cannot be wider than the width of the data bus. Best and simplest operation requires the same size of data bus and memory cell width.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

As we previously discussed that the control bus carries the intent of the processor that it wants to read or to write. Memory changes its behavior in response to this signal from the processor. It defines the direction of data flow. If processor wants to read but memory wants to write, there will be no communication or useful flow of information. Both must be synchronized, like a speaker speaks and the listener listens. If both speak simultaneously or both listen there will be no communication. This precise synchronization between the processor and the memory is the responsibility of the control bus.

Control bus is only the mechanism. The responsibility of sending the appropriate signals on the control bus to the memory is of the processor. Since the memory never wants to listen or to speak of itself. Then why is the control bus bidirectional. Again we take the same example of the servant and the book further to elaborate this situation. Consider that the servant went to fetch the book just to find that the drawing room door is locked. Now the servant can wait there indefinitely keeping us in surprise or come back and inform us about the situation so that we can act accordingly. The servant even though he was obedient was unable to fulfill our orders so in all his obedience, he came back to inform us about the problem. Synchronization is still important, as a result of our orders either we got the desired cell or we came to know that the memory is locked for the moment. Such information cannot be transferred via the address or the data bus. For such situations when peripherals want to talk to the processor when the processor wasn't expecting them to speak, special lines in the control bus are used. The information in such signals is usually to indicate the incapability of the peripheral to do something for the moment. For these reasons the control bus is a bidirectional bus and can carry information from processor to memory as well as from memory to processor.

1.2. REGISTERS

The basic purpose of a computer is to perform operations, and operations need operands. Operands are the data on which we want to perform a certain operation. Consider the addition operation; it involves adding two numbers to get their sum. We can have precisely one address on the address bus and consequently precisely one element on the data bus. At the very same instant the second operand cannot be brought inside the processor. As soon as the second is selected, the first operand is no longer there. For this reason there are temporary storage places inside the processor called *registers*. Now one operand can be read in a register and added into the other which is read directly from the memory. Both are made accessible at one instance of time, one from inside the processor and one from outside on the data bus. The result can be written to at a distinct location as the operation has completed and we can access a different memory cell. Sometimes we hold both operands in registers for the sake of efficiency as what we can do inside the processor is undoubtedly faster than if we have to go outside and bring the second operand.

Registers are like a scratch pad ram inside the processor and their operation is very much like normal memory cells. They have precise locations and remember what is placed inside them. They are used when we need more than one data element inside the processor at one time. The concept of registers will be further elaborated as we progress into writing our first program.

Memory is a limited resource but the number of memory cells is large. Registers are relatively very small in number, and are therefore a very scarce and precious resource. Registers are more than one in number, so we have to precisely identify or name them. Some manufacturers number their registers like r0, r1, r2, others name them like A, B, C, D etc. Naming is useful since the registers are few in number. This is called the nomenclature of the

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

particular architecture. Still other manufacturers name their registers according to their function like X stands for an index register. This also informs us that there are special functions of registers as well, some of which are closely associated to the particular architecture. For example index registers do not hold data instead they are used to hold the address of data. There are other functions as well and the whole spectrum of register functionalities is quite large. However most of the details will become clear as the registers of the Intel architecture are discussed in detail.

Accumulator

There is a central register in every processor called the accumulator. Traditionally all mathematical and logical operations are performed on the accumulator. The word size of a processor is defined by the width of its accumulator. A 32bit processor has an accumulator of 32 bits.

Pointer, Index, or Base Register

The name varies from manufacturer to manufacturer, but the basic distinguishing property is that it does not hold data but holds the address of data. The rationale can be understood by examining a “for” loop in a higher level language, zeroing elements in an array of ten elements located in consecutive memory cells. The location to be zeroed changes every iteration. That is the address where the operation is performed is changing. Index register is used in such a situation to hold the address of the current array location. Now the value in the index register cannot be treated as data, but it is the address of data. In general whenever we need access to a memory location whose address is not known until runtime we need an index register. Without this register we would have needed to explicitly code each iteration separately.

In newer architectures the distinction between accumulator and index registers has become vague. They have general registers which are more versatile and can do both functions. They do have some specialized behaviors but basic operations can be done on all general registers.

Flags Register or Program Status Word

This is a special register in every architecture called the flags register or the program status word. Like the accumulator it is an 8, 16, or 32 bits register but unlike the accumulator it is meaningless as a unit, rather the individual bits carry different meanings. The bits of the accumulator work in parallel as a unit and each bit mean the same thing. The bits of the flags register work independently and individually, and combined its value is meaningless.

An example of a bit commonly present in the flags register is the carry flag. The carry can be contained in a single bit as in binary arithmetic the carry can only be zero or one. If a 16bit number is added to a 16bit accumulator, and the result is of 17 bits the 17th bit is placed in the carry bit of the flags register. Without this 17th bit the answer is incorrect. More examples of flags will be discussed when dealing with the Intel specific register set.

Program Counter or Instruction Pointer

Everything must translate into a binary number for our dumb processor to understand it, be it an operand or an operation itself. Therefore the instructions themselves must be translated into numbers. For example to add numbers we understand the word “add.” We translate this word into a number to make the processor understand it. This number is the actual instruction for the computer. All the objects, inheritance and encapsulation constructs in higher level languages translate down to just a number in assembly language in the end. Addition, multiplication, shifting; all big

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

programs are made using these simple building blocks. A number is at the bottom line since this is the only thing a computer can understand.

A program is defined to be “an ordered set of instructions.” Order in this definition is a key part. Instructions run one after another, first, second, third and so on. Instructions have a positional relationship. The whole logic depends on this positioning. If the computer executes the fifth instructions after the first and not the second, all our logic is gone. The processor should ensure this ordering of instructions. A special register exists in every processor called the program counter or the instruction pointer that ensures this ordering. “The program counter holds the address of the next instruction to be executed.” A number is placed in the memory cell pointed to by this register and that number tells the processor which instruction to execute; for example 0xEA, 255, or 152. For the processor 152 might be the add instruction. Just this one number tells it that it has to add, where its operands are, and where to store the result. This number is called the *opcode*. The instruction pointer moves from one opcode to the next. This is how our program executes and progresses. One instruction is picked, its operands are read and the instruction is executed, then the next instruction is picked from the new address in instruction pointer and so on.

Remembering 152 for the add operation or 153 for the subtract operation is difficult. To make a simple way to remember difficult things we associate a symbol to every number. As when we write “add” everyone understands what we mean by it. Then we need a small program to convert this “add” of ours to 152 for the processor. Just a simple search and replace operation to translate all such symbols to their corresponding opcodes. We have mapped the numeric world of the processor to our symbolic world. “Add” conveys a meaning to us but the number 152 does not. We can say that add is closer to the programmer’s thinking. This is the basic motive of adding more and more translation layers up to higher level languages like C++ and Java and Visual Basic. These symbols are called *instruction mnemonics*. Therefore the mnemonic “add a to b” conveys more information to the reader. The dumb translator that will convert these mnemonics back to the original opcodes is a key program to be used throughout this course and is called the *assembler*.

1.3. INSTRUCTION GROUPS

Usual opcodes in every processor exist for moving data, arithmetic and logical manipulations etc. However their mnemonics vary depending on the will of the manufacturer. Some manufacturers name the mnemonics for data movement instructions as “move,” some call it “load” and “store” and still other names are present. But the basic set of instructions is similar in every processor. A grouping of these instructions makes learning a new processor quick and easy. Just the group an instruction belongs tells a lot about the instruction.

Data Movement Instructions

These instructions are used to move data from one place to another. These places can be registers, memory, or even inside peripheral devices. Some examples are:

```
mov ax, bx
lad 1234
```

Arithmetic and Logic Instructions

Arithmetic instructions like addition, subtraction, multiplication, division and Logical instructions like logical and, logical or, logical xor, or complement are part of this group. Some examples are:

```
and ax, 1234
add bx, 0534
add bx, [1200]
```

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

The bracketed form is a complex variation meaning to add the data placed at address 1200. Addressing data in memory is a detailed topic and is discussed in the next chapter.

Program Control Instructions

The instruction pointer points to the next instruction and instructions run one after the other with the help of this register. We can say that the instructions are tied with one another. In some situations we don't want to follow this implied path and want to order the processor to break its flow if some condition becomes true instead of the spatially placed next instruction. In certain other cases we want the processor to first execute a separate block of code and then come back to resume processing where it left.

These are instructions that control the program execution and flow by playing with the instruction pointer and altering its normal behavior to point to the next instruction. Some examples are:

```
cmp ax, 0
jne 1234
```

We are changing the program flow to the instruction at 1234 address if the condition that we checked becomes true.

Special Instructions

Another group called special instructions works like the special service commandos. They allow changing specific processor behaviors and are used to play with it. They are used rarely but are certainly used in any meaningful program. Some examples are:

```
cli
sti
```

Where cli clears the interrupt flag and sti sets it. Without delving deep into it, consider that the cli instruction instructs the processor to close its ears from the outside world and never listen to what is happening outside, possibly to do some very important task at hand, while sti restores normal behavior. Since these instructions change the processor behavior they are placed in the special instructions group.

1.4. INTEL IAPX88 ARCHITECTURE

Now we select a specific architecture to discuss these abstract ideas in concrete form. We will be using IBM PC based on Intel architecture because of its wide availability, because of free assemblers and debuggers available for it, and because of its wide use in a variety of domains. However the concepts discussed will be applicable on any other architecture as well; just the mnemonics of the particular language will be different.

Technically iAPX88 stands for "Intel Advanced Processor Extensions 88." It was a very successful processor also called 8088 and was used in the very first IBM PC machines. Our discussion will revolve around 8088 in the first half of the course while in the second half we will use iAPX386 which is very advanced and powerful processor. 8088 is a 16bit processor with its accumulator and all registers of 16 bits. 386 on the other hand, is a 32bit processor. However it is downward compatible with iAPX88 meaning that all code written for 8088 is valid on the 386. The architecture of a processor means the organization and functionalities of the registers it contains and the instructions that are valid on the processor. We will discuss the register architecture of 8088 in detail below while its instructions are discussed in the rest of the book at appropriate places.

1.5. HISTORY

Intel did release some 4bit processors in the beginning but the first meaningful processor was 8080, an 8bit processor. The processor became

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

popular due to its simplistic design and versatile architecture. Based on the experience gained from 8080, an advanced version was released as 8085. The processor became widely popular in the engineering community again due to its simple and logical nature.

Intel introduced the first 16bit processor named 8088 at a time when the concept of personal computer was evolving. With a maximum memory of 64K on the 8085, the 8088 allowed a whole mega byte. IBM embedded this processor in their personal computer. The first machines ran at 4.43 MHz; a blazing speed at that time. This was the right thing at the right moment. No one expected this to become the biggest success of computing history. IBM PC XT became so popular and successful due to its open architecture and easily available information.

The success was unexpected for the developers themselves. As when Intel introduced the processor it contained a timer tick count which was valid for five years only. They never anticipated the architecture to stay around for more than five years but the history took a turn and the architecture is there at every desk even after 25 years and the tick is to be specially handled every now and then.

1.6. REGISTER ARCHITECTURE

The iAPX88 architecture consists of 14 registers.

CS	SP	
DS	BP	
SS	SI	
ES	DI	
	AH	AL (AX)
IP	BH	BL (BX)
	CH	CL (CX)
FLAGS	DH	DL (DX)

General Registers (AX, BX, CX, and DX)

The registers AX, BX, CX, and DX behave as general purpose registers in Intel architecture and do some specific functions in addition to it. X in their names stand for extended meaning 16bit registers. For example AX means we are referring to the extended 16bit "A" register. Its upper and lower byte are separately accessible as AH (A high byte) and AL (A low byte). All general purpose registers can be accessed as one 16bit register or as two 8bit registers. The two registers AH and AL are part of the big whole AX. Any change in AH or AL is reflected in AX as well. AX is a composite or extended register formed by gluing together the two parts AH and AL.

The A of AX stands for Accumulator. Even though all general purpose registers can act as accumulator in most instructions there are some specific variations which can only work on AX which is why it is named the accumulator. The B of BX stands for Base because of its role in memory addressing as discussed in the next chapter. The C of CX stands for Counter as there are certain instructions that work with an automatic count in the CX register. The D of DX stands for Destination as it acts as the destination in I/O operations. The A, B, C, and D are in letter sequence as well as depict some special functionality of the register.

Index Registers (SI and DI)

SI and DI stand for source index and destination index respectively. These are the index registers of the Intel architecture which hold address of data and used in memory access. Being an open and flexible architecture, Intel allows many mathematical and logical operations on these registers as well like the general registers. The source and destination are named because of their implied functionality as the source or the destination in a special class of instructions called the string instructions. However their use is not at all restricted to string instructions. SI and DI are 16bit and cannot be used as 8bit register pairs like AX, BX, CX, and DX.

Instruction Pointer (IP)

This is the special register containing the address of the next instruction to be executed. No mathematics or memory access can be done through this register. It is out of our direct control and is automatically used. Playing with it is dangerous and needs special care. Program control instructions change the IP register.

Stack Pointer (SP)

It is a memory pointer and is used indirectly by a set of instructions. This register will be explored in the discussion of the system stack.

Base Pointer (BP)

It is also a memory pointer containing the address in a special area of memory called the stack and will be explored alongside SP in the discussion of the stack.

Flags Register

The flags register as previously discussed is not meaningful as a unit rather it is bit wise significant and accordingly each bit is named separately. The bits not named are unused. The Intel FLAGS register has its bits organized as follows:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0	D	I	T	S	Z		A		P		C

The individual flags are explained in the following table.

C	Carry	When two 16bit numbers are added the answer can be 17 bits long or when two 8bit numbers are added the answer can be 9 bits long. This extra bit that won't fit in the target register is placed in the carry flag where it can be used and tested.
P	Parity	Parity is the number of "one" bits in a binary number. Parity is either odd or even. This information is normally used in communications to verify the integrity of data sent from the sender to the receiver.
A	Auxiliary Carry	A number in base 16 is called a hex number and can be represented by 4 bits. The collection of 4 bits is called a nibble. During addition or subtraction if a carry goes from one nibble to the next this flag is set. Carry flag is for the carry from the whole addition while auxiliary carry is the carry from the first nibble to the second.
Z	Zero Flag	The Zero flag is set if the last mathematical or logical instruction has produced a zero in its destination.

S	Sign Flag	A signed number is represented in its two's complement form in the computer. The most significant bit (MSB) of a negative number in this representation is 1 and for a positive number it is zero. The sign bit of the last mathematical or logical operation's destination is copied into the sign flag.
T	Trap Flag	The trap flag has a special role in debugging which will be discussed later.
I	Interrupt Flag	It tells whether the processor can be interrupted from outside or not. Sometimes the programmer doesn't want a particular task to be interrupted so the Interrupt flag can be zeroed for this time. The programmer rather than the processor sets this flag since the programmer knows when interruption is okay and when it is not. Interruption can be disabled or enabled by making this bit zero or one, respectively, using special instructions.
D	Direction Flag	Specifically related to string instructions, this flag tells whether the current operation has to be done from bottom to top of the block (D=0) or from top to bottom of the block (D=1).
O	Overflow Flag	The overflow flag is set during signed arithmetic, e.g. addition or subtraction, when the sign of the destination changes unexpectedly. The actual process sets the overflow flag whenever the carry into the MSB is different from the carry out of the MSB

Segment Registers (CS, DS, SS, and ES)

The code segment register, data segment register, stack segment register, and the extra segment register are special registers related to the Intel segmented memory model and will be discussed later.

1.7. OUR FIRST PROGRAM

The first program that we will write will only add three numbers. This very simple program will clarify most of the basic concepts of assembly language. We will start with writing our algorithm in English and then moving on to convert it into assembly language.

English Language Version

"Program is an ordered set of instructions for the processor." Our first program will be instructions manipulating AX and BX in plain English.

```

move 5 to ax
move 10 to bx
add bx to ax
move 15 to bx
add bx to ax

```

Even in this simple reflection of thoughts in English, there are some key things to observe. One is the concept of destination as every instruction has a "to destination" part and there is a source before it as well. For example the second line has a constant 10 as its source and the register BX as its destination. The key point in giving the first program in English is to convey that the concepts of assembly language are simple but fine. Try to understand them considering that all above is everyday English that you know very well and every concept will eventually be applicable to assembly language.

Assembly Language Version

Intel could have made their assembly language exactly identical to our program in plain English but they have abbreviated a lot of symbols to avoid unnecessarily lengthy program when the meaning could be conveyed with less effort. For example Intel has named their move instruction “mov” instead of “move.” Similarly the Intel order of placing source and destination is opposite to what we have used in our English program, just a change of interpretation. So the Intel way of writing things is:

```
operation destination, source
operation destination
operation source
operation
```

The later three variations are for instructions that have one or both of their operands implied or they work on a single or no operand. An implied operand means that it is always in a particular register say the accumulator, and it need not be mentioned in the instruction. Now we attempt to write our program in actual assembly language of the iapx88.

Example 1.1	
001	; a program to add three numbers using registers
002	[org 0x0100]
003	mov ax, 5 ; load first number in ax
004	mov bx, 10 ; load second number in bx
005	add ax, bx ; accumulate sum in ax
006	mov bx, 15 ; load third number in bx
007	add ax, bx ; accumulate sum in ax
008	
009	mov ax, 0x4c00 ; terminate program
010	int 0x21
001	To start a comment a semicolon is used and the assembler ignores everything else on the same line. Comments must be extensively used in assembly language programs to make them readable.
002	Leave the org directive for now as it will be discussed later.
003	The constant 5 is loaded in one register AX.
004	The constant 10 is loaded in another register BX.
005	Register BX is added to register AX and the result is stored in register AX. Register AX should contain 15 by now.
006	The constant 15 is loaded in the register BX.
007	Register BX is again added to register AX now producing 15+15=30 in the AX register. So the program has computed 5+10+15=30.
008	Vertical spacing must also be used extensively in assembly language programs to separate logical blocks of code.
009-010	The ending lines are related more to the operating system than to assembly language programming. It is a way to inform DOS that our program has terminated so it can display its command prompt again. The computer may reboot or behave improperly if this termination is not present.

Assembler, Linker, and Debugger

We need an assembler to assemble this program and convert this into executable binary code. The assembler that we will use during this course is “Netwide Assembler” or NASM. It is a free and open source assembler. And the tool that will be most used will be the debugger. We will use a free debugger called “A fullscreen debugger” or AFD. These are the whole set of

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

weapons an assembly language programmer needs for any task whatsoever at hand.

To assemble we will give the following command to the processor assuming that our input file is named EX01.ASM.

```
nasm ex01.asm -o ex01.com -l ex01.lst
```

This will produce two files EX01.COM that is our executable file and EX01.LST that is a special listing file that we will explore now. The listing file produced for our example above is shown below with comments removed for neatness.

```

1
2                                [org 0x0100]
3 00000000 B80500                mov ax, 5
4 00000003 BB0A00                mov bx, 10
5 00000006 01D8                  add ax, bx
6 00000008 BB0F00                mov bx, 15
7 0000000B 01D8                  add ax, bx
8
9 0000000D B8004C                mov ax, 0x4c00
10 00000010 CD21                 int 0x21

```

The first column in the above listing is offset of the listed instruction in the output file. Next column is the opcode into which our instruction was translated. In this case this opcode is B8. Whenever we move a constant into AX register the opcode B8 will be used. After it 0500 is appended which is the immediate operand to this instruction. An immediate operand is an operand which is placed directly inside the instruction. Now as the AX register is a word sized register, and one hexadecimal digit takes 4 bits so 4 hexadecimal digits make one word or two bytes. Which of the two bytes should be placed first in the instruction, the least significant or the most significant? Similarly for 32bit numbers either the order can be most significant, less significant, lesser significant, and least significant called the big-endian order used by Motorola and some other companies or it can be least significant, more significant, more significant, and most significant called the little-endian order and is used by Intel. The big-endian have the argument that it is more natural to read and comprehend while the little-endian have the argument that this scheme places the less significant value at a lesser address and more significant value at a higher address.

Because of this the constant 5 in our instruction was converted into 0500 with the least significant byte of 05 first and the most significant byte of 00 afterwards. When read as a word it is 0005 but when written in memory it will become 0500. As the first instruction is three bytes long, the listing file shows that the offset of the next instruction in the file is 3. The opcode BB is for moving a constant into the BX register, and the operand 0A00 is the number 10 in little-endian byte order. Similarly the offsets and opcodes of the remaining instructions are shown in order. The last instruction is placed at offset 0x10 or 16 in decimal. The size of the last instruction is two bytes, so the size of the complete COM file becomes 18 bytes. This can be verified from the directory listing, using the DIR command, that the COM file produced is exactly 18 bytes long.

Now the program is ready to be run inside the debugger. The debugger shows the values of registers, flags, stack, our code, and one or two areas of the system memory as data. Debugger allows us to step our program one instruction at a time and observe its effect on the registers and program data. The details of using the AFD debugger can be seen from the AFD manual.

After loading the program in the debugger observe that the first instruction is now at 0100 instead of absolute zero. This is the effect of the org directive at the start of our program. The first instruction of a COM file must be at

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

offset 0100 (decimal 255) as a requirement. Also observe that the debugger is showing your program even though it was provided only the COM file and neither of the listing file or the program source. This is because the translation from mnemonic to opcode is reversible and the debugger mapped back from the opcode to the instruction mnemonic. This will become apparent for instructions that have two mnemonics as the debugger might not show the one that was written in the source file.

As a result of program execution either registers or memory will change. Since our program yet doesn't touch memory the only changes will be in the registers. Keenly observe the registers AX, BX, and IP change after every instruction. IP will change after every instruction to point to the next instruction while AX will accumulate the result of our addition.

1.8. SEGMENTED MEMORY MODEL

Rationale

In earlier processors like 8080 and 8085 the linear memory model was used to access memory. In linear memory model the whole memory appears like a single array of data. 8080 and 8085 could access a total memory of 64K using the 16 lines of their address bus. When designing iAPX88 the Intel designers wanted to remain compatible with 8080 and 8085 however 64K was too small to continue with, for their new processor. To get the best of both worlds they introduced the segmented memory model in 8088.

There is also a logical argument in favor of a segmented memory model in addition to the issue of compatibility discussed above. We have two logical parts of our program, the code and the data, and actually there is a third part called the program stack as well, but higher level languages make this invisible to us. These three logical parts of a program should appear as three distinct units in memory, but making this division is not possible in the linear memory model. The segmented memory model does allow this distinction.

Mechanism

The segmented memory model allows multiple functional windows into the main memory, a code window, a data window etc. The processor sees code from the code window and data from the data window. The size of one window is restricted to 64K. 8085 software fits in just one such window. It sees code, data, and stack from this one window, so downward compatibility is attained.

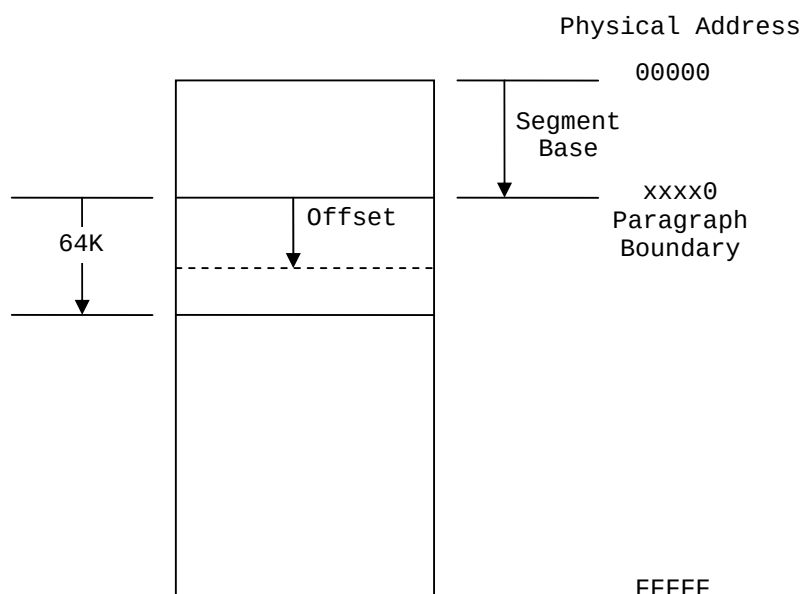
However the maximum memory iAPX88 can access is 1MB which can be accessed with 20 bits. Compare this with the 64K of 8085 that were accessed using 16 bits. The idea is that the 64K window just discussed can be moved anywhere in the whole 1MB. The four segment registers discussed in the Intel register architecture are used for this purpose. Therefore four windows can exist at one time. For example one window that is pointed to by the CS register contains the currently executing code.

To understand the concept, consider the windows of a building. We say that a particular window is 3 feet above the floor and another one is 20 feet above the floor. The reference point, the floor is the base of the segment called the datum point in a graph and all measurement is done from that datum point considering it to be zero. So CS holds the zero or the base of code. DS holds the zero of data. Or we can say CS tells how high code from the floor is, and DS tells how high data from the floor is, while SS tells how high the stack is. One extra segment ES can be used if we need to access two distant areas of memory at the same time that both cannot be seen through the same window. ES also has special role in string instructions. ES is used as an extra data segment and cannot be used as an extra code or stack segment.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

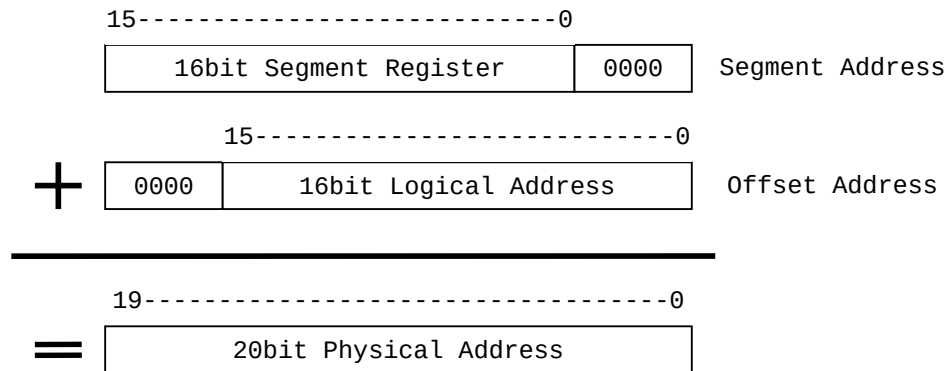
Revisiting the concept again, like the datum point of a graph, the segment registers tell the start of our window which can be opened anywhere in the megabyte of memory available. The window is of a fixed size of 64KB. Base and offset are the two key variables in a segmented address. Segment tells the base while offset is added into it. The registers IP, SP, BP, SI, DI, and BX all can contain a 16bit offset in them and access memory relative to a segment base.

The IP register cannot work alone. It needs the CS register to open a 64K window in the 1MB memory and then IP works to select code from this window as offsets. IP works only inside this window and cannot go outside of this 64K in any case. If the window is moved i.e. the CS register is changed, IP will change its behavior accordingly and start selecting from the new window. The IP register always works relatively, relative to the segment base stored in the CS register. IP is a 16bit register capable of accessing only 64K memory so how the whole megabyte can contain code anywhere. Again the same concept is there, it can access 64K at one instance of time. As the base is changed using the CS register, IP can be made to point anywhere in the whole megabyte. The process is illustrated with the following diagram.



Physical Address Calculation

Now for the whole megabyte we need 20 bits while CS and IP are both 16bit registers. We need a mechanism to make a 20bit number out of the two 16bit numbers. Consider that the segment value is stored as a 20 bit number with the lower four bits zero and the offset value is stored as another 20 bit number with the upper four bits zeroed. The two are added to produce a 20bit absolute address. A carry if generated is dropped without being stored anywhere and the phenomenon is called address wraparound. The process is explained with the help of the following diagram.



Therefore memory is determined by a segment-offset pair and not alone by any one register which will be an ambiguous reference. Every offset register is assigned a default segment register to resolve such ambiguity. For example the program we wrote when loaded into memory had a value of 0100 in IP register and some value say 1DDD in the CS register. Making both 20 bit numbers, the segment base is 1DDD0 and the offset is 00100 and adding them we get the physical memory address of 1DED0 where the opcode B80500 is placed.

Paragraph Boundaries

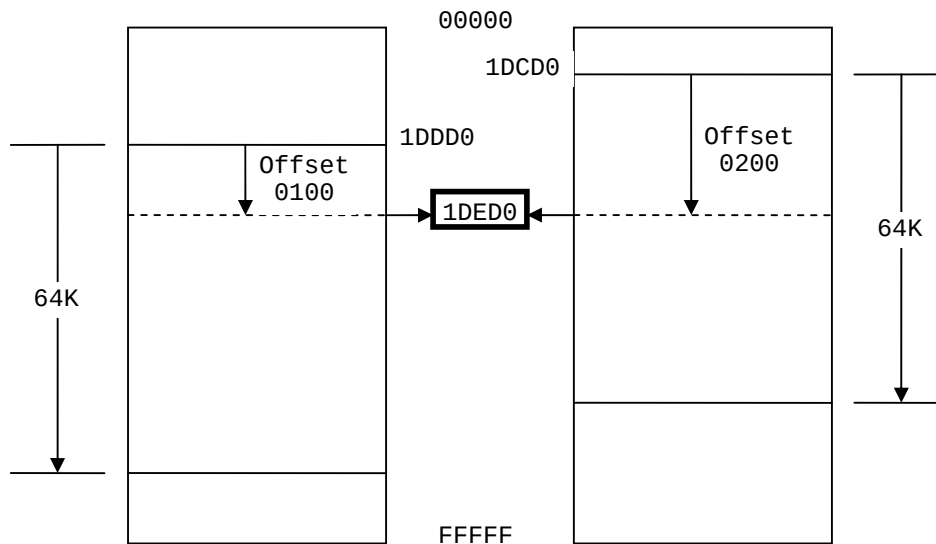
As the segment value is a 16bit number and four zero bits are appended to the right to make it a 20bit number, segments can only be defined a 16byte boundaries called paragraph boundaries. The first possible segment value is 0000 meaning a physical base of 00000 and the next possible value of 0001 means a segment base of 00010 or 16 in decimal. Therefore segments can only be defined at 16 byte boundaries.

Overlapping Segments

We can also observe that in the case of our program CS, DS, SS, and ES all had the same value in them. This is called overlapping segments so that we can see the same memory from any window. This is the structure of a COM file.

Using partially overlapping segments we can produce a number of segment, offset pairs that all access the same memory. For example 1DDD:0100 and IDED:0000 both point to the same physical memory. To test this we can open a data window at 1DED:0000 in the debugger and change the first three bytes to "90" which is the opcode for NOP (no operation). The change is immediately visible in the code window which is pointed to by CS containing 1DDD. Similarly IDCD:0200 also points to the same memory location. Consider this like a portion of wall that three different people on three different floors are seeing through their own windows. One of them painted the wall red; it will be changed for all of them though their perspective is different. It is the same phenomenon occurring here.

The segment, offset pair is called a logical address, while the 20bit address is a physical address which is the real thing. Logical addressing is a mechanism to access the physical memory. As we have seen three different logical addresses accessed the same physical address.



EXERCISES

- How the processor uses the address bus, the data bus, and the control bus to communicate with the system memory?
- Which of the following are unidirectional and which are bidirectional?
 - Address Bus
 - Data Bus
 - Control Bus
- What are registers and what are the specific features of the accumulator, index registers, program counter, and program status word?
- What is the size of the accumulator of a 64bit processor?
- What is the difference between an instruction mnemonic and its opcode?
- How are instructions classified into groups?
- A combination of 8bits is called a byte. What is the name for 4bits and for 16bits?
- What is the maximum memory 8088 can access?
- List down the 14 registers of the 8088 architecture and briefly describe their uses.
- What flags are defined in the 8088 FLAGS register? Describe the function of the zero flag, the carry flag, the sign flag, and the overflow flag.
- Give the value of the zero flag, the carry flag, the sign flag, and the overflow flag after each of the following instructions if AX is initialized with 0x1254 and BX is initialized with 0xFFFF.
 - add ax, 0xEDAB
 - add ax, bx
 - add bx, 0xF001
- What is the difference between little endian and big endian formats? Which format is used by the Intel 8088 microprocessor?
- For each of the following words identify the byte that is stored at lower memory address and the byte that is stored at higher memory address in a little endian computer.
 - 1234
 - ABFC
 - B100
 - B800

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

14. What are the contents of memory locations 200, 201, 202, and 203 if the word 1234 is stored at offset 200 and the word 5678 is stored at offset 202?
15. What is the offset at which the first executable instruction of a COM file must be placed?
16. Why was segmentation originally introduced in 8088 architecture?
17. Why a segment start cannot start from the physical address 55555.
18. Calculate the physical memory address generated by the following segment offset pairs.
 - a. 1DDD:0436
 - b. 1234:7920
 - c. 74F0:2123
 - d. 0000:6727
 - e. FFFF:4336
 - f. 1080:0100
 - g. AB01:FFFF
19. What are the first and the last physical memory addresses accessible using the following segment values?
 - a. 1000
 - b. 0FFF
 - c. 1002
 - d. 0001
 - e. E000
20. Write instructions that perform the following operations.
 - a. Copy BL into CL
 - b. Copy DX into AX
 - c. Store 0x12 into AL
 - d. Store 0x1234 into AX
 - e. Store 0xFFFF into AX
21. Write a program in assembly language that calculates the square of six by adding six to the accumulator six times.

2

Addressing Modes

2.1. DATA DECLARATION

The first instruction of our first assembly language program was “mov ax, 5.” Here MOV was the opcode; AX was the destination operand, while 5 was the source operand. The value of 5 in this case was stored as part of the instruction encoding. In the opcode B80500, B8 was the opcode and 0500 was the operand stored immediately afterwards. Such an operand is called an immediate operand. It is one of the many types of operands available.

Writing programs using just the immediate operand type is difficult. Every reasonable program needs some data in memory apart from constants. Constants cannot be changed, i.e. they cannot appear as the destination operand. In fact placing them as destination is meaningless and illegal according to assembly language syntax. Only registers or data placed in memory can be changed. So real data is the one stored in memory, with a very few constants. So there must be a mechanism in assembly language to store and retrieve data from memory.

To declare a part of our program as holding data instead of instructions we need a couple of very basic but special assembler directives. The first directive is “define byte” written as “db.”

```
db    somevalue
```

As a result a cell in memory will be reserved containing the desired value in it and it can be used in a variety of ways. Now we can add variables instead of constants. The other directive is “define word” or “dw” with the same syntax as “db” but reserving a whole word of 16 bits instead of a byte. There are directives to declare a double or a quad word as well but we will restrict ourselves to byte and word declarations for now. For single byte we use db and for two bytes we use dw.

To refer to this variable later in the program, we need the address occupied by this variable. The assembler is there to help us. We can associate a symbol with any address that we want to remember and use that symbol in the rest of the code. The symbol is there for our own comprehension of code. The assembler will calculate the address of that symbol using our origin directive and calculating the instruction lengths or data declarations in-between and replace all references to the symbol with the corresponding address. This is just like variables in a higher level language, where the compiler translates them into addresses; just the process is hidden from the programmer one level further. Such a symbol associated to a point in the program is called a label and is written as the label name followed by a colon.

2.2. DIRECT ADDRESSING

Now we will rewrite our first program such that the numbers 5, 10, and 15 are stored as memory variables instead of constants and we access them from there.

Example 2.1	
001	; a program to add three numbers using memory variables
002	[org 0x0100]
003	mov ax, [num1] ; load first number in ax

004	mov	bx, [num2]	; load second number in bx
005	add	ax, bx	; accumulate sum in ax
006	mov	bx, [num3]	; load third number in bx
007	add	ax, bx	; accumulate sum in ax
008	mov	[num4], ax	; store sum in num4
009			
010	mov	ax, 0x4c00	; terminate program
011	int	0x21	
012			
013	num1:	dw	5
014	num2:	dw	10
015	num3:	dw	15
016	num4:	dw	0

002	Originate our program at 0100. The first executable instruction should be placed at this offset.
003	The source operand is changed from constant 5 to [num1]. The bracket is signaling that the operand is placed in memory at address num1. The value 5 will be loaded in ax even though we did not specify it in our program code, rather the value will be picked from memory. The instruction should be read as “read the contents of memory location num1 in the ax register.” The label num1 is a symbol for us but an address for the processor while the conversion is done by the assembler.
013	The label num1 is defined as a word and the assembler is requested to place 5 in that memory location. The colon signals that num1 is a label and not an instruction.

Using the same process to assemble as discussed before we examine the listing file generated as a result with comments removed.

```

1
2                                     [org 0x0100]
3 00000000 A1[1700]                mov ax, [num1]
4 00000003 8B1E[1900]              mov bx, [num2]
5 00000007 01D8                    add ax, bx
6 00000009 8B1E[1B00]              mov bx, [num3]
7 0000000D 01D8                    add ax, bx
8 0000000F A3[1D00]                mov [num4], ax
9
10 00000012 B8004C                 mov ax, 0x4c00
11 00000015 CD21                   int 0x21
12
13 00000017 0500                   num1:      dw 5
14 00000019 0A00                   num2:      dw 10
15 0000001B 0F00                   num3:      dw 15
16 0000001D 0000                   num4:      dw 0

```

The first instruction of our program has changed from B80500 to A11700. The opcode B8 is used to move constants into AX, while the opcode A1 is used when moving data into AX from memory. The immediate operand to our new instruction is 1700 or as a word 0017 (23 decimal) and from the bottom of the listing file we can observe that this is the offset of num1. The assembler has calculated the offset of num1 and used it to replace references to num1 in the whole program. Also the value 0500 can be seen at offset 0017 in the file. We can say contents of memory location 0017 are 0005 as a word. Similarly num2, num3, and num4 are placed at 0019, 001B, and 001D addresses.

When the program is loaded in the debugger, it is loaded at offset 0100, which displaces all memory accesses in our program. The instruction A11700 is changed to A11701 meaning that our variable is now placed at 0117 offset. The instruction is shown as `mov ax, [0117]`. Also the data window can be used to verify that offset 0117 contains the number 0005.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

Execute the program step by step and examine how the memory is read and the registers are updated, how the instruction pointer moves forward, and how the result is saved back in memory. Also observe inside the debugger code window below the code for termination, that the debugger is interpreting our data as code and showing it as some meaningless instructions. This is because the debugger sees everything as code in the code window and cannot differentiate our declared data from opcodes. It is our responsibility that we terminate execution before our data is executed as code.

Also observe that our naming of num1, num2, num3, and num4 is no longer there inside the debugger. The debugger is only showing the numbers 0117, 0119, 011B, and 011D. Our numerical machine can only work with numbers. We used symbols for our ease to label or tag certain positions in our program. The assembler converts these symbols into the appropriate numbers automatically. Also observe that the effect of “dw” is to place 5 in two bytes as 0005. Had we used “db” this would have been stored as 05 in one byte.

Given the fact that the assembler knows only numbers we can write the same program using a single label. As we know that num2 is two ahead of num1, we can use num1+2 instead of num2 and let the assembler calculate the sum during assembly process.

Example 2.2	
001	; a program to add three numbers accessed using a single label
002	[org 0x0100]
003	mov ax, [num1] ; load first number in ax
004	mov bx, [num1+2] ; load second number in bx
005	add ax, bx ; accumulate sum in ax
006	mov bx, [num1+4] ; load third number in bx
007	add ax, bx ; accumulate sum in ax
008	mov [num1+6], ax ; store sum at num1+6
009	
010	mov ax, 0x4c00 ; terminate program
011	int 0x21
012	
013	num1: dw 5
014	dw 10
015	dw 15
016	dw 0
004	The second number is read from num1+2. Similarly the third number is read from num1+4 and the result is accessed at num1+6.
013-016	The labels num2, num3, and num4 are removed and the data there will be accessed with reference to num1.

Every location is accessed with reference to num1 in this example. The expression “num1+2” comprises of constants only and can be evaluated at the time of assembly. There are no variables involved in this expression. As we open the program inside the debugger we see a verbatim copy of the previous program. There is no difference at all since the assembler catered for the differences during assembly. It calculated 0117+2=0119 while in the previous it directly knew from the value of num2 that it has to write 0119, but the end result is a ditto copy of the previous execution.

Another way to declare the above data and produce exactly same results is shown in the following example.

Example 2.3	
001	; a program to add three numbers accessed using a single label
002	[org 0x0100]
003	mov ax, [num1] ; load first number in ax
004	mov bx, [num1+2] ; load second number in bx

005		add	ax, bx	; accumulate sum in ax
006		mov	bx, [num1+4]	; load third number in bx
007		add	ax, bx	; accumulate sum in ax
008		mov	[num1+6], ax	; store sum at num1+6
009				
010		mov	ax, 0x4c00	; terminate program
011		int	0x21	
012				
013	num1:	dw	5, 10, 15, 0	
013	As we do not need to place labels on individual variables we can save space and declare all data on a single line separated by commas. This declaration will declare four words in consecutive memory locations while the address of first one is num1.			

The method used to access memory in the above examples is called direct addressing. In direct addressing the memory address is fixed and is given in the instruction. The actual data used is placed in memory and now that data can be used as the destination operand as well. Also the source and destination operands must have the same size. For example a word defined memory is read in a word sized register. A last observation is that the data 0500 in memory was corrected to 0005 when read in a register. So registers contain data in proper order as a word.

A last variation using direct addressing shows that we can directly add a memory variable and a register instead of adding a register into another that we were doing till now.

Example 2.4				
01			; a program to add three numbers directly in memory	
02		[org 0x0100]		
03		mov	ax, [num1]	; load first number in ax
04		mov	[num1+6], ax	; store first number in result
05		mov	ax, [num1+2]	; load second number in ax
06		add	[num1+6], ax	; add second number to result
07		mov	ax, [num1+4]	; load third number in ax
08		add	[num1+6], ax	; add third number to result
09				
10		mov	ax, 0x4c00	; terminate program
11		int	0x21	
12				
13	num1:	dw	5, 10, 15, 0	

We generate the following listing file as a result of the assembly process described previously. Comments are again removed.

1				
2			[org 0x0100]	
3	00000000	A1[1900]		mov ax, [num1]
4	00000003	A3[1F00]		mov [num1+6], ax
5	00000006	A1[1B00]		mov ax, [num1+2]
6	00000009	0106[1F00]		add [num1+6], ax
7	0000000D	A1[1D00]		mov ax, [num1+4]
8	00000010	0106[1F00]		add [num1+6], ax
9				
10	00000014	B8004C		mov ax, 0x4c00
11	00000017	CD21		int 0x21
12				
13	00000019	05000A000F000000	num1:	dw 5, 10, 15, 0

The opcode of add is changed because the destination is now a memory location instead of a register. No other significant change is seen in the listing file. Inside the debugger we observe that few opcodes are longer now and the location num1 is now translating to 0119 instead of 0117. This is done automatically by the assembler as a result of using labels instead of

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

hard coding addresses. During execution we observe that the word data as it is read into a register is read in correct order. The significant change in this example is that the destination of addition is memory. Method to access memory is direct addressing, whether it is the MOV instruction or the ADD instruction.

The first two instructions of the last program read a number into AX and placed it at another memory location. A quick thought reveals that the following might be a possible single instruction to replace the couple.

```
mov [num1+6], [num1] ; ILLEGAL
```

However this form is illegal and not allowed on the Intel architecture. None of the general operations of mov add, sub etc. allow moving data from memory to memory. Only register to register, register to memory, memory to register, constant to memory, and constant to register operations are allowed. The other register to constant, memory to constant, and memory to memory are all disallowed. Only string instructions allow moving data from memory to memory and will be discussed in detail later. As a rule one instruction can have at most one operand in brackets, otherwise assembler will give an error.

2.3. SIZE MISMATCH ERRORS

If we change the directive in the last example from DW to DB, the program will still assemble and debug without errors, however the results will not be the same as expected. When the first operand is read 0A05 will be read in the register which was actually two operands place in consecutive byte memory locations. The second number will be read as 000F which is the zero byte of num4 appended to the 15 of num3. The third number will be junk depending on the current state of the machine. According to our data declaration the third number should be at 0114 but it is accessed at 011D calculated with word offsets. This is a logical error of the program. To keep the declarations and their access synchronized is the responsibility of the programmer and not the assembler. The assembler allows the programmer to do everything he wants to do, and that can possibly run on the processor. The assembler only keeps us from writing illegal instructions which the processor cannot execute. This is the difference between a syntax error and a logic error. So the assembler and debugger have both done what we asked them to do but the programmer asked them to do the wrong chore.

The programmer is responsible for accessing the data as word if it was declared as a word and accessing it as a byte if it was declared as a byte. The word case is shown in lot of previous examples. If however the intent is to treat it as a byte the following code shows the appropriate way.

Example 2.5	
001	; a program to add three numbers using byte variables
002	[org 0x0100]
003	mov al, [num1] ; load first number in al
004	mov bl, [num1+1] ; load second number in bl
005	add al, bl ; accumulate sum in al
006	mov bl, [num1+2] ; load third number in bl
007	add al, bl ; accumulate sum in al
008	mov [num1+3], al ; store sum at num1+3
009	
010	mov ax, 0x4c00 ; terminate program
011	int 0x21
012	
013	num1: db 5, 10, 15, 0
003	The number is read in AL register which is a byte register since the memory location read is also of byte size.
005	The second number is now placed at num1+1 instead of num1+2 because of byte offsets.

013	To declare data db is used instead of dw so that each data declared occupies one byte only.

Inside the debugger we observe that the AL register takes appropriate values and the sum is calculated and stored in num1+3. This time there is no alignment or synchronization error. The key thing to understand here is that the processor does not match defines to accesses. It is the programmer's responsibility. In general assembly language gives a lot of power to the programmer but power comes with responsibility. Assembly language programming is not a difficult task but a responsible one.

In the above examples, the processor knew the size of the data movement operation from the size of the register involved, for example in "mov ax, [num1]" memory can be accessed as byte or as word, it has no hard and fast size, but the AX register tells that this operation has to be a word operation. Similarly in "mov al, [num1]" the AL register tells that this operation has to be a byte operation. However in "mov ax, bl" the AX register tells that the operation has to be a word operation while BL tells that this has to be a byte operation. The assembler will declare that this is an illegal instruction. A 5Kg bag cannot fit inside a 1Kg bag and according to Intel a 1Kg cannot also fit in a 5Kg bag. They must match in size. The instruction "mov [num1], [num2]" is illegal as previously discussed not because of data movement size but because memory to memory moves are not allowed at all.

The instruction "mov [num1], 5" is legal but there is no way for the processor to know the data movement size in this operation. The variable num1 can be treated as a byte or as a word and similarly 5 can be treated as a byte or as a word. Such instructions are declared ambiguous by the assembler. The assembler has no way to guess the intent of the programmer as it previously did using the size of the register involved but there is no register involved this time. And memory is a linear array and label is an address in it. There is no size associated with a label. Therefore to resolve its ambiguity we clearly tell our intent to the assembler in one of the following ways.

```
mov byte [num1], 5
mov word [num1], 5
```

2.4. REGISTER INDIRECT ADDRESSING

We have done very elementary data access till now. Assume that the numbers we had were 100 and not just three. This way of adding them will cost us 200 instructions. There must be some method to do a task repeatedly on data placed in consecutive memory cells. The key to this is the need for some register that can hold the address of data. So that we can change the address to access some other cell of memory using the same instruction. In direct addressing mode the memory cell accessed was fixed inside the instruction. There is another method in which the address can be placed in a register so that it can be changed. For the following example we will take 10 instead of 100 numbers but the algorithm is extensible to any size.

There are four registers in iAPX88 architecture that can hold address of data and they are BX, BP, SI, and DI. There are minute differences in their working which will be discussed later. For the current example, we will use the BX register and we will take just three numbers and extend the concept with more numbers in later examples.

Example 2.6	
001	; a program to add three numbers using indirect addressing
002	[org 0x100]
003	mov bx, num1 ; point bx to first number
004	mov ax, [bx] ; load first number in ax
005	add bx, 2 ; advance bx to second number

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

006	add ax, [bx]	; add second number to ax
007	add bx, 2	; advance bx to third number
008	add ax, [bx]	; add third number to ax
009	add bx, 2	; advance bx to result
010	mov [bx], ax	; store sum at num1+6
011		
012	mov ax, 0x4c00	; terminate program
013	int 0x21	
014		
015	num1: dw 5, 10, 15, 0	
003	Observe that no square brackets around num1 are used this time. The address is loaded in bx and not the contents. Value of num1 is 0005 and the address is 0117. So BX will now contain 0117.	
004	Brackets are now used around BX. In iapx88 architecture brackets can be used around BX, BP, SI, and DI only. In iapx386 more registers are allowed. The instruction will be read as “move into ax the contents of the memory location whose address is in bx.” Now since bx contains the address of num1 the contents of num1 are transferred to the ax register. Without square brackets the meaning of the instruction would have been totally different.	
005	This instruction is changing the address. Since we have words not bytes, we add two to bx so that it points to the next word in memory. BX now contains 0119 the address of the second word in memory. This was the mechanism to change addresses that we needed.	

Inside the debugger we observe that the first instruction is “mov bx, 011C.” A constant is moved into BX. This is because we did not use the square brackets around “num1.” The address of “num1” has moved to 011C because the code size has changed due to changed instructions. In the second instruction BX points to 011C and the value read in AX is 0005 which can be verified from the data window. After the addition BX points to 011E containing 000A, our next word, and so on. This way the BX register points to our words one after another and we can add them using the same instruction “mov ax, [bx]” without fixing the address of our data in the instructions. We can also subtract from BX to point to previous cells. The address to be accessed is now in total program control.

One thing that we needed in our problem to add hundred numbers was the capability to change address. The second thing we need is a way to repeat the same instruction and a way to know that the repetition is done a 100 times, a terminal condition for the repetition. For the task we are introducing two new instructions that you should read and understand as simple English language concepts. For simplicity only 10 numbers are added in this example. The algorithm is extensible to any size.

Example 2.7		
001	; a program to add ten numbers	
002	[org 0x0100]	
003	mov bx, num1	; point bx to first number
004	mov cx, 10	; load count of numbers in cx
005	mov ax, 0	; initialize sum to zero
006		
007	l1: add ax, [bx]	; add number to ax
008	add bx, 2	; advance bx to next number
009	sub cx, 1	; numbers to be added reduced
010	jnz l1	; if numbers remain add next
011		
012	mov [total], ax	; write back sum in memory
013		
014	mov ax, 0x4c00	; terminate program
015	int 0x21	
016		
017	num1: dw 10, 20, 30, 40, 50, 10, 20, 30, 40, 50	

018	total: dw 0
006	Labels can be used on code as well. Just like data labels they remember the address at which they are used. The assembler does not differentiate between code labels and data labels. The programmer is responsible for using a data label as data and a code label as code. The label l1 in this case is the address of the following instruction.
009	SUB is the counterpart to ADD with the same rules as that of the ADD instruction.
010	JNZ stands for “jump if not zero.” NZ is the condition in this instruction. So the instruction is read as “jump to the location l1 if the zero flag is not set.” And revisiting the zero flag definition “the zero flag is set if the last mathematical or logical operation has produced a zero in its destination.” For example “mov ax, 0” will not set the zero flag as it is not a mathematical or logical instruction. However subtraction and addition will set it. Also it is set even when the destination is not a register. Now consider the subtraction immediately preceding it. If the CX register becomes zero as a result of this subtraction the zero flag will be set and the jump will be taken. And jump to l1, the processor needs to be told each and everything and the destination is an important part of every jump. Just like when we ask someone to go, we mention go to this market or that house. The processor is much more logical than us and needs the destination in every instruction that asks it to go somewhere. The processor will load l1 in the IP register and resume execution from there. The processor will blindly go to the label we mention even if it contains data and not code.

The CX register is used as a counter in this example, BX contains the changing address, while AX accumulates the result. We have formed a loop in assembly language that executes until its condition remains true. Inside the debugger we can observe that the subtract instruction clears the zero flag the first nine times and sets it on the tenth time. While the jump instruction moves execution to address l1 the first nine times and to the following line the tenth time. The jump instruction breaks program flow.

The JNZ instruction is from the program control group and is a conditional jump, meaning that if the condition NZ is true (ZF=0) it will jump to the address mentioned and otherwise it will progress to the next instruction. It is a selection between two paths. If the condition is true go right and otherwise go left. Or we can say if the weather is hot, go this way, and if it is cold, go this way. Conditional jump is the most important instruction, as it gives the processor decision making capability, so it must be given a careful thought. Some processors call it branch, probably a more logical name for it, however the functionality is same. Intel chose to name it “jump.”

An important thing in the above example is that a register is used to reference memory so this form of access is called register indirect memory access. We used the BX register for it and the B in BX and BP stands for base therefore we call register indirect memory access using BX or BP, “based addressing.” Similarly when SI or DI is used we name the method “indexed addressing.” They have the same functionality, with minor differences because of which the two are called base and index. The differences will be explained later, however for the above example SI or DI could be used as well, but we would name it indexed addressing instead of based addressing.

2.5. REGISTER + OFFSET ADDRESSING

Direct addressing and indirect addressing using a single register are two basic forms of memory access. Another possibility is to use different combinations of direct and indirect references. In the above example we used BX to access different array elements which were placed consecutively in memory like an array. We can also place in BX only the array index and not the exact address and form the exact address when we are going to access the actual memory. This way the same register can be used for accessing different arrays and also the register can be used for index comparison like the following example does.

Example 2.8	
001	; a program to add ten numbers using register + offset addressing
002	[org 0x0100]
003	mov bx, 0 ; initialize array index to zero
004	mov cx, 10 ; load count of numbers in cx
005	mov ax, 0 ; initialize sum to zero
006	
007	l1: add ax, [num1+bx] ; add number to ax
008	add bx, 2 ; advance bx to next index
009	sub cx, 1 ; numbers to be added reduced
010	jnz l1 ; if numbers remain add next
011	
012	mov [total], ax ; write back sum in memory
013	
014	mov ax, 0x4c00 ; terminate program
015	int 0x21
016	
017	num1: dw 10, 20, 30, 40, 50, 10, 20, 30, 40, 50
018	total: dw 0

003	This time BX is initialized to zero instead of array base
007	The format of memory access has changed. The array base is added to BX containing array index at the time of memory access.
008	As the array is of words, BX jumps in steps of two, i.e. 0, 2, 4. Higher level languages do appropriate incrementing themselves and we always use sequential array indexes. However in assembly language we always calculate in bytes and therefore we need to take care of the size of one array element which in this case is two.

Inside the debugger we observe that the memory access instruction is shown as “mov ax, [011F+bx]” and the actual memory accessed is the one whose address is the sum of 011F and the value contained in the BX register. This form of access is of the register indirect family and is called base + offset or index + offset depending on whether BX or BP is used or SI or DI is used.

2.6. SEGMENT ASSOCIATION

All the addressing mechanisms in iAPX88 return a number called *effective address*. For example in base + offset addressing, neither the base nor the offset alone tells the desired cell in memory to be accessed. It is only after the addition is done that the processor knows which cell to be accessed. This number which came as the result of addition is called the effective address. But the effective address is just an offset and is meaningless without a segment. Only after the segment is known, we can form the physical address that is needed to access a memory cell.

We discussed the segmented memory model of iAPX88 in reasonable detail at the end of previous chapter. However during the discussion of addressing modes we have not seen the effect of segments. Segmentation is there and it's all happening relative to a segment base. We saw DS, CS, SS, and ES

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

inside the debugger. Everything is relative to its segment base, even though we have not explicitly explained its functionality. An offset alone is not complete without a segment. As previously discussed there is a default segment associated to every register which accesses memory. For example CS is associated to IP by default; rather it is tied with it. It cannot access memory in any other segment.

In case of data, there is a bit relaxation and nothing is tied. Rather there is a default association which can be overridden. In the case of register indirect memory access, if the register used is one of SI, DI, or BX the default segment is DS. If however the register used is BP the default segment used is SS. The stack segment has a very critical and fine use and there is a reason why BP is attached to SS by default. However these will be discussed in detail in the chapter on stack. IP is tied to CS while SP is tied to SS. The association of these registers cannot be changed; they are locked with no option. Others are not locked and can be changed.

To override the association for one instruction of one of the registers BX, BP, SI or DI, we use the segment override prefix. For example “mov ax, [cs:bx]” associates BX with CS for this one instruction. For the next instruction the default association will come back to act. The processor places a special byte before the instruction called a prefix, just like prefixes and suffixes in English language. No prefix is needed or placed for default association. For example for CS the byte 2E is placed and for ES the byte 26 is placed. Opcode has not changed, but the prefix byte has modified the default association to association with the desired segment register for this one instruction.

In all our examples, we never declared a segment or used it explicitly, but everything seemed to work fine. The important thing to note is that CS, DS, SS, and ES all had the same value. The value itself is not important but the fact that all had the same value is important. All four segment windows exactly overlap. Whatever segment register we use the same physical memory will be accessed. That is why everything was working without the mention of a single segment register. This is the formation of COM files in IBM PC. A single segment contains code, data, and the stack. This format is operating system dependant, in our case defined by DOS. And our operating system defines the format of COM files such that all segments have the same value. Thus the only meaningful thing that remains is the offset.

For example if BX=0100, SI=0200, and CS=1000 and the memory access under consideration is [cs:bx+si+0x0700], the effective address formed is $bx+si+0700 = 0100 + 0200 + 0700 = 0A00$. Now multiplying the segment value by 16 makes it 10000 and adding the effective address 00A00 forms the physical address 10A00.

2.7. ADDRESS WRAPAROUND

There are two types of wraparounds. One is within a single segment and the other is inside the whole physical memory. Segment wraparound occurs when during the effective address calculation a carry is generated. This carry is dropped giving the effect that when we try to access beyond the segment limit, we are actually wrapped around to the first cell in the segment. For example if BX=9100, DS=1500 and the access is [bx+0x7000] we form the effective address $9100 + 7000 = 10100$. The carry generated is dropped forming the actual effective address of 0100. Just like a circle when we reached the end we started again from the beginning. An arc at 370 degrees is the same as an arc at 10 degrees. We tried to cross the segment boundary and it pushed us back to the start. This is called segment wraparound. The physical address in the above example will be 15100.

The same can also happen at the time of physical address calculation. For example BX=0100, DS=FFFF and the access under consideration is [bx+0x0100]. The effective address will be 0200 and the physical address will

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

be 100100. This is a 21bit answer and cannot be sent on the address bus which is 20 bits wide. The carry is dropped and just like the segment wraparound our physical memory has wrapped around at its very top. When we tried to access beyond limits the actual access is made at the very start. This second wraparound is a bit different in newer processor with more address lines but that will be explained in later chapters.

2.8. ADDRESSING MODES SUMMARY

The iAPX88 processor supports seven modes of memory access. Remember that immediate is not an addressing mode but an operand type. Operands can be immediate, register, or memory. If the operand is memory one of the seven addressing modes will be used to access it. The memory access mechanisms can also be written in the general form “base + index + offset” and we can define the possible addressing modes by saying that any one, two, or none can be skipped from the general form to form a legal memory access.

There are a few common mistakes done in forming a valid memory access. Part of a register cannot be used to access memory. Like BX is allowed to hold an address but BL or BH are not. Address is 16bit and must be contained in a 16bit register. BX-SI is not possible. The only thing that we can do is addition of a base register with an index register. Any other operation is disallowed. BS+BP and SI+DI are both disallowed as we cannot have two base or two index registers in one memory access. One has to be a base register and the other has to be an index register and that is the reason of naming them differently.

Direct

A fixed offset is given in brackets and the memory at that offset is accessed. For example “mov [1234], ax” stores the contents of the AX registers in two bytes starting at address 1234 in the current data segment. The instruction “mov [1234], al” stores the contents of the AL register in the byte at offset 1234.

Based Register Indirect

A base register is used in brackets and the actual address accessed depends on the value contained in that register. For example “mov [bx], ax” moves the two byte contents of the AX register to the address contained in the BX register in the current data segment. The instruction “mov [bp], al” moves the one byte content of the AL register to the address contained in the BP register in the current stack segment.

Indexed Register Indirect

An index register is used in brackets and the actual address accessed depends on the value contained in that register. For example “mov [si], ax” moves the contents of the AX register to the word starting at address contained in SI in the current data segment. The instruction “mov [di], ax” moves the word contained in AX to the offset stored in DI in the current data segment.

Based Register Indirect + Offset

A base register is used with a constant offset in this addressing mode. The value contained in the base register is added with the constant offset to get the effective address. For example “mov [bx+300], ax” stores the word contained in AX at the offset attained by adding 300 to BX in the current data segment. The instruction “mov [bp+300], ax” stores the word in AX to the offset attained by adding 300 to BP in the current stack segment.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

Indexed Register Indirect + Offset

An index register is used with a constant offset in this addressing mode. The value contained in the index register is added with the constant offset to get the effective address. For example “mov [si+300], ax” moves the word contained in AX to the offset attained by adding 300 to SI in the current data segment and the instruction “mov [di+300], al” moves the byte contained in AL to the offset attained by adding 300 to DI in the current data segment.

Base + Index

One base and one index register is used in this addressing mode. The value of the base register and the index register are added together to get the effective address. For example “mov [bx+si], ax” moves the word contained in the AX register to offset attained by adding BX and SI in the current data segment. The instruction “mov [bp+di], al” moves the byte contained in AL to the offset attained by adding BP and DI in the current stack segment. Observe that the default segment is based on the base register and not on the index register. This is why base registers and index registers are named separately. Other examples are “mov [bx+di], ax” and “mov [bp+si], ax.” This method can be used to access a two dimensional array such that one dimension is in a base register and the other is in an index register.

Base + Index + Offset

This is the most complex addressing method and is relatively infrequently used. A base register, an index register, and a constant offset are all used in this addressing mode. The values of the base register, the index register, and the constant offset are all added together to get the effective address. For example “mov [bx+si+300], ax” moves the word contents of the AX register to the word in memory starting at offset attained by adding BX, SI, and 300 in the current data segment. Default segment association is again based on the base register. It might be used with the array base of a two dimensional array as the constant offset, one dimension in the base register and the other in the index register. This way all calculation of location of the desired element has been delegated to the processor.

EXERCISES

1. What is a label and how does the assembler differentiate between code labels and data labels?
2. List the seven addressing modes available in the 8088 architecture.
3. Differentiate between effective address and physical address.
4. What is the effective address generated by the following instructions? Every instruction is independent of others. Initially BX=0x0100, num1=0x1001, [num1]=0x0000, and SI=0x0100
 - a. mov ax, [bx+12]
 - b. mov ax, [bx+num1]
 - c. mov ax, [num1+bx]
 - d. mov ax, [bx+si]
5. What is the effective address generated by the following combinations if they are valid. If not give reason. Initially BX=0x0100, SI=0x0010, DI=0x0001, BP=0x0200, and SP=0xFFFF
 - a. bx-si
 - b. bx-bp
 - c. bx+10
 - d. bx-10
 - e. bx+sp
 - f. bx+di
6. Identify the problems in the following instructions and correct them by replacing them with one or two instruction having the same effect.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

- a. `mov [02], [ 22]`
 - b. `mov [wordvar], 20`
 - c. `mov bx, al`
 - d. `mov ax, [si+di+100]`
7. What is the function of segment override prefix and what changes it brings to the opcode?
8. What are the two types of address wraparound? What physical address is accessed with `[BX+SI]` if `FFFF` is loaded in `BX`, `SI`, and `DS`.
9. Write instructions to do the following.
 - a. Copy contents of memory location with offset `0025` in the current data segment into `AX`.
 - b. Copy `AX` into memory location with offset `0FFF` in the current data segment.
 - c. Move contents of memory location with offset `0010` to memory location with offset `002F` in the current data segment.
10. Write a program to calculate the square of 20 by using a loop that adds 20 to the accumulator 20 times.

3

Branching

3.1. COMPARISON AND CONDITIONS

Conditional jump was introduced in the last chapter to loop for the addition of a fixed number of array elements. The jump was based on the zero flag. There are many other conditions possible in a program. For example an operand can be greater than another operand or it can be smaller. We use comparisons and boolean expressions extensively in higher level languages. They must be available in some form in assembly language, otherwise they could not possibly be made available in a higher level language. In fact they are available in a very fine and purified form.

The basic root instruction for all comparisons is CMP standing for compare. The operation of CMP is to subtract the source operand from the destination operand, updating the flags without changing either the source or the destination. CMP is one of the key instructions as it introduces the capability of conditional routing in the processor.

A closer thought reveals that with subtraction we can check many different conditions. For example if a larger number is subtracted from a smaller number then borrow is needed. The carry flag plays the role of borrow during the subtraction operation. And in this condition the carry flag will be set. If two equal numbers are subtracted the answer is zero and the zero flag will be set. Every significant relation between the destination and source is evident from the sign flag, carry flag, zero flag, and the overflow flag. CMP is meaningless without a conditional jump immediately following it.

Another important distinction at this point is the difference between signed and unsigned numbers. In unsigned numbers only the magnitude of the number is important, whereas in signed numbers both the magnitude and the sign are important. For example -2 is greater than -3 but 2 is smaller than 3. The sign has affected our comparisons.

Inside the computer signed numbers are represented in two's complement notation. In essence a number in this representation is still a number, just that now our interpretation of this number will be signed. Whether we use jump above and below or we use jump greater or less will convey our intention to the processor. The jump above and greater operations at first sight seem to be doing the same operation, and similarly below and less operations seem to be similar. However for signed numbers JG and JL will work properly and for unsigned JA and JB will work properly and not the other way around.

It is important to note that at the time of comparison, the intent of the programmer to treat the numbers as signed or unsigned is not clear. The subtraction in CMP is a normal subtraction. It is only after the comparison, during the conditional jump operation, that the intent is conveyed. At that time with a specific combination of flags checked the intent is satisfied.

For example a number 2 is represented in a word as 0002 while the number -2 is represented as FFFE. In a byte they would be represented as 02 and FE. Now both have the same magnitude however the different sign has caused very different representation in two's complement form. Now if the intent is to use FFFE or decimal 65534 then the same data would be placed in the word as in case of -2. In fact if -2 and 65534 are compared the processor will set the zero flag signaling that they are exactly equal. As regards an unsigned comparison the number 65534 is much greater than 2.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

So if a JA is taken after comparing -2 in the destination with 2 in the source the jump will be taken. If however JG is used after the same comparison the jump will not be taken as it will consider the sign and with the sign -2 is smaller than 2. The key idea is that -2 and 65534 were both stored in memory in the same form. It was the interpretation that treated it as a signed or as an unsigned number.

The unsigned comparisons see the numbers as 0 being the smallest and 65535 being the largest with the order that $0 < 1 < 2 \dots < 65535$. The signed comparisons see the number -32768 which has the same memory representation as 32768 as the smallest number and 32767 as the largest with the order $-32768 < -32767 < \dots < -1 < 0 < 1 < 2 < \dots < 32767$. All the negative numbers have the same representation as an unsigned number in the range 32768 ... 65535 however the signed interpretation of the signed comparisons makes them be treated as negative numbers smaller than zero.

All meaningful situations both for signed and unsigned numbers than occur after a comparison are detailed in the following table.

DEST = SRC	ZF = 1	When the source is subtracted from the destination and both are equal the result is zero and therefore the zero flag is set. This works for both signed and unsigned numbers.
UDEST < USRC	CF = 1	When an unsigned source is subtracted from an unsigned destination and the destination is smaller, borrow is needed which sets the carry flag.
UDEST ≤ USRC	ZF = 1 OR CF = 1	If the zero flag is set, it means that the source and destination are equal and if the carry flag is set it means a borrow was needed in the subtraction and therefore the destination is smaller.
UDEST ≥ USRC	CF = 0	When an unsigned source is subtracted from an unsigned destination no borrow will be needed either when the operands are equal or when the destination is greater than the source.
UDEST > USRC	ZF = 0 AND CF = 0	The unsigned source and destination are not equal if the zero flag is not set and the destination is not smaller since no borrow was taken. Therefore the destination is greater than the source.
SDEST < SSRC	SF ≠ OF	When a signed source is subtracted from a signed destination and the answer is negative with no overflow than the destination is smaller than the source. If however there is an overflow meaning that the sign has changed unexpectedly, the meanings are reversed and a

		positive number signals that the destination is smaller.
$SDEST \leq SSRC$	$ZF = 1$ OR $SF \neq OF$	If the zero flag is set, it means that the source and destination are equal and if the sign and overflow flags differ it means that the destination is smaller as described above.
$SDEST \geq SSRC$	$SF = OF$	When a signed source is subtracted from a signed destination and the answer is positive with no overflow then the destination is greater than the source. When an overflow is there signaling that sign has changed unexpectedly, we interpret a negative answer as the signal that the destination is greater.
$SDEST > SSRC$	$ZF = 0$ AND $SF = OF$	If the zero flag is not set, it means that the signed operands are not equal and if the sign and overflow match in addition to this it means that the destination is greater than the source.

3.2. CONDITIONAL JUMPS

For every interesting or meaningful situation of flags, a conditional jump is there. For example JZ and JNZ check the zero flag. If in a comparison both operands are same, the result of subtraction will be zero and the zero flag will be set. Thus JZ and JNZ can be used to test equality. That is why there are renamed versions JE and JNE read as jump if equal or jump if not equal. They seem more logical in writing but mean exactly the same thing with the same opcode. Many jumps are renamed with two or three names for the same jump, so that the appropriate logic can be conveyed in assembly language programs. This renaming is done by Intel and is a standard for iAPX88. JC and JNC test the carry flag. For example we may need to test whether there was an overflow in the last unsigned addition or subtraction. Carry flag will also be set if two unsigned numbers are subtracted and the first is smaller than the second. Therefore the renamed versions JB, JNAE, and JNB, JAE are there standing for jump if below, jump if not above or equal, jump if not below, and jump if above or equal respectively. The operation of all jumps can be seen from the following table.

JC JB JNAE	Jump if carry Jump if below Jump if not above or equal	CF = 1	This jump is taken if the last arithmetic operation generated a carry or required a borrow. After a CMP it is taken if the unsigned source is smaller than the unsigned destination.
JNC JNB JAE	Jump if not carry Jump if not below Jump if above or equal	CF = 0	This jump is taken if the last arithmetic operation did not

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

			generated a carry or required a borrow. After a CMP it is taken if the unsigned source is larger or equal to the unsigned destination.
JE JZ	Jump if equal Jump if zero	ZF = 1	This jump is taken if the last arithmetic operation produced a zero in its destination. After a CMP it is taken if both operands were equal.
JNE JNZ	Jump if not equal Jump if not zero	ZF = 0	This jump is taken if the last arithmetic operation did not produce a zero in its destination. After a CMP it is taken if both operands were different.
JA JNBE	Jump if above Jump if not below or equal	ZF = 0 AND CF = 0	This jump is taken after a CMP if the unsigned source is larger than the unsigned destination.
JNA JBE	Jump if not above Jump if not below or equal	ZF = 1 OR CF = 1	This jump is taken after a CMP if the unsigned source is smaller than or equal to the unsigned destination.
JL JNGE	Jump if less Jump if not greater or equal	SF $\neq$ OF	This jump is taken after a CMP if the signed source is smaller than the signed destination.
JNL JGE	Jump if not less Jump if greater or equal	SF = OF	This jump is taken after a CMP if the signed source is larger than or equal to the signed destination.
JG JNLE	Jump if greater Jump if not less or equal	ZF = 0 AND SF = OF	This jump is taken after a CMP if the signed source is larger than the signed destination.
JNG JLE	Jump if not greater Jump if less or equal	ZF = 1 OR SF $\neq$ OF	This jump is taken after a CMP if the signed source is smaller than or equal to the signed destination.

JO	Jump if overflow.	OF = 1	This jump is taken if the last arithmetic operation changed the sign unexpectedly.
JNO	Jump if not overflow	OF = 0	This jump is taken if the last arithmetic operation did not change the sign unexpectedly.
JS	Jump if sign	SF = 1	This jump is taken if the last arithmetic operation produced a negative number in its destination.
JNS	Jump if not sign	SF = 0	This jump is taken if the last arithmetic operation produced a positive number in its destination.
JP JPE	Jump if parity Jump if even parity	PF = 1	This jump is taken if the last arithmetic operation produced a number in its destination that has even parity.
JNP JPO	Jump if not parity Jump if odd parity	PF = 0	This jump is taken if the last arithmetic operation produced a number in its destination that has odd parity.
JCXZ	Jump if CX is zero	CX = 0	This jump is taken if the CX register is zero.

The CMP instruction sets the flags reflecting the relation of the destination to the source. This is important as when we say jump if above, then what is above what. The destination is above the source or the source is above the destination.

The JA and JB instructions are related to unsigned numbers. That is our interpretation for the destination and source operands is unsigned. The 16th bit holds data and not the sign. In the JL and JG instructions standing for jump if lower and jump if greater respectively, the interpretation is signed. The 16th bit holds the sign and not the data. The difference between them will be made clear as an elaborate example will be given to explain the difference.

One jump is special that it is not dependant on any flag. It is JCXZ, jump if the CX register is zero. This is because of the special treatment of the CX register as a counter. This jump is regardless of the zero flag. There is no counterpart or not form of this instruction.

The adding numbers example of the last chapter can be a little simplified using the compare instruction on the BX register and eliminating the need for a separate counter as below.

Example 3.1	
<pre> 001 ; a program to add ten numbers without a separate counter 002 [org 0x0100] 003 mov bx, 0 ; initialize array index to zero 004 mov ax, 0 ; initialize sum to zero 005 006 l1: add ax, [num1+bx] ; add number to ax 007 add bx, 2 ; advance bx to next index 008 cmp bx, 20 ; are we beyond the last index 009 jne l1 ; if not add next number 010 011 mov [total], ax ; write back sum in memory 012 013 mov ax, 0x4c00 ; terminate program 014 int 0x21 015 016 num1: dw 10, 20, 30, 40, 50, 10, 20, 30, 40, 50 017 total: dw 0 </pre>	
006	The format of memory access is still base + offset.
008	BX is used as the array index as well as the counter. The offset of 11th number will be 20, so as soon as BX becomes 20 just after the 10th number has been added, the addition is stopped.
009	The jump is displayed as JNZ in the debugger even though we have written JNE in our example. This is because it is a renamed jump with the same opcode as JNZ and the debugger has no way of knowing the mnemonic that we used after looking just at the opcode. Also every code and data reference that we used till now is seen in the opcode as well. However for the jump instruction we see an operand of F2 in the opcode and not 0116. This will be discussed in detail with unconditional jumps. It is actually a short relative jump and the operand is stored in the form of positive or negative offset from this instruction.

With conditional branching in hand, there are just a few small things left in assembly language that fills some gaps. Now there is just imagination and the skill to conceive programs that can make you write any program.

3.3. UNCONDITIONAL JUMP

Till now we have been placing data at the end of code. There is no such restriction and we can define data anywhere in the code. Taking the previous example, if we place data at the start of code instead of at the end and we load our program in the debugger. We can see our data placed at the start but the debugger is intending to start execution at our data. The COM file definition said that the first executable instruction is at offset 0100 but we have placed data there instead of code. So the debugger will try to interpret that data as code and showed whatever it could make up out of those opcodes.

We introduce a new instruction called JMP. It is the unconditional jump that executes regardless of the state of all flags. So we write an unconditional jump as the very first instruction of our program and jump to the next instruction that follows our data declarations. This time 0100 contains a valid first instruction of our program.

Example 3.2	
<pre> 001 ; a program to add ten numbers without a separate counter 002 [org 0x0100] 003 jmp start ; unconditionally jump over data 004 005 num1: dw 10, 20, 30, 40, 50, 10, 20, 30, 40, 50 006 total: dw 0 </pre>	

007				
008	start:	mov bx, 0		; initialize array index to zero
009		mov ax, 0		; initialize sum to zero
010				
011	l1:	add ax, [num1+bx]		; add number to ax
012		add bx, 2		; advance bx to next index
013		cmp bx, 20		; are we beyond the last index
014		jne l1		; if not add next number
015				
016		mov [total], ax		; write back sum in memory
017				
018		mov ax, 0x4c00		; terminate program
019		int 0x21		
003	JMP jumps over the data declarations to the start label and execution resumes from there.			

3.4. RELATIVE ADDRESSING

Inside the debugger the instruction is shown as JMP 0119 and the location 0119 contains the original first instruction of the logic of our program. This jump is unconditional, it will always be taken. Now looking at the opcode we see F21600 where F2 is the opcode and 1600 is the operand to it. 1600 is 0016 in proper word order. 0119 is not given as a parameter rather 0016 is given.

This is position relative addressing in contrast to absolute addressing. It is not telling the exact address rather it is telling how much forward or backward to go from the current position of IP in the current code segment. So the instruction means to add 0016 to the IP register. At the time of execution of the first instruction at 0100 IP was pointing to the next instruction at 0103, so after adding 16 it became 0119, the desired target location. The mechanism is important to know, however all calculations in this mechanism are done by the assembler and by the processor. We just use a label with the JMP instruction and are ensured that the instruction at the target label will be the one to be executed.

3.5. TYPES OF JUMP

The three types of jump, near, short, and far, differ in the size of instruction and the range of memory they can jump to with the smallest short form of two bytes and a range of just 256 bytes to the far form of five bytes and a range covering the whole memory.

Short Jump

EB	Disp

Near Jump

EB	Disp Low	Disp High

Far Jump

EB	IP Low	IP High	CS Low	CS High

Near Jump

When the relative address stored with the instruction is in 16 bits as in the last example the jump is called a near jump. Using a near jump we can jump anywhere within a segment. If we add a large number it will wrap around to

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

the lower part. A negative number actually is a large number and works this way using the wraparound behavior.

Short Jump

If the offset is stored in a single byte as in 75F2 with the opcode 75 and operand F2, the jump is called a short jump. F2 is added to IP as a signed byte. If the byte is negative the complement is negated from IP otherwise the byte is added. Unconditional jumps can be short, near, and far. The far type is yet to be discussed. Conditional jumps can only be short. A short jump can go +127 bytes ahead in code and -128 bytes backwards and no more. This is the limitation of a byte in signed representation.

Far Jump

Far jump is not position relative but is absolute. Both segment and offset must be given to a far jump. The previous two jumps were used to jump within a segment. Sometimes we may need to go from one code segment to another, and near and short jumps cannot take us there. Far jump must be used and a two byte segment and a two byte offset are given to it. It loads CS with the segment part and IP with the offset part. Execution therefore resumes from that location in physical memory. The three instructions that have a far form are JMP, CALL, and RET, are related to program control. Far capability makes intra segment control possible.

3.6. SORTING EXAMPLE

Moving ahead from our example of adding numbers we progress to a program that can sort a list of numbers using the tools that we have accumulated till now. Sorting can be ascending or descending like if the largest number comes at the top, followed by a smaller number and so on till the smallest number the sort will be called descending. The other order starting with the smallest number and ending at the largest is called ascending sort. This is a common problem and many algorithms have been developed to solve it. One simple algorithm is the bubble sort algorithm.

In this algorithm we compare consecutive numbers. If they are in required order e.g. if it is a descending sort and the first is larger than the second, then we leave them as it is and if they are not in order, we swap them. Then we do the same process for the next two numbers and so on till the last two are compared and possibly swapped.

A complete iteration is called a pass over the array. We need N passes at least in the simplest algorithm if N is the number of elements to be sorted. A finer algorithm is to check if any swap was done in this pass and stop as soon as a pass goes without a swap. The array is now sorted as every pair of elements is in order.

For example if our list of numbers is 60, 55, 45, and 58 and we want to sort them in ascending order, the first comparison will be of 60 and 55 and as the order will be reversed to 55 and 60. The next comparison will be of 60 and 45 and again the two will be swapped. The next comparison of 60 and 58 will also cause a swap. At the end of first pass the numbers will be in order of 55, 45, 58, and 60. Observe that the largest number has bubbled down to the bottom. Just like a bubble at bottom of water. In the next pass 55 and 45 will be swapped. 55 and 58 will not be swapped and 58 and 60 will also not be swapped. In the next pass there will be no swap as the elements are in order i.e. 45, 55, 58, and 60. The passes will be stopped as the last pass did not cause any swap. The application of bubble sort on these numbers is further explained with the following illustration.

State of Data	Swap Done	Swap Flag				
Pass 1		Off				
<table><tr><td>60</td><td>55</td><td>45</td><td>58</td></tr></table>	60	55	45	58	Yes	On
60	55	45	58			
<table><tr><td>55</td><td>60</td><td>45</td><td>58</td></tr></table>	55	60	45	58	Yes	On
55	60	45	58			
<table><tr><td>55</td><td>45</td><td>60</td><td>58</td></tr></table>	55	45	60	58	Yes	On
55	45	60	58			
Pass 2		Off				
<table><tr><td>55</td><td>45</td><td>58</td><td>60</td></tr></table>	55	45	58	60	Yes	On
55	45	58	60			
<table><tr><td>45</td><td>55</td><td>58</td><td>60</td></tr></table>	45	55	58	60	No	On
45	55	58	60			
<table><tr><td>45</td><td>55</td><td>58</td><td>60</td></tr></table>	45	55	58	60	No	On
45	55	58	60			
Pass 3		Off				
<table><tr><td>45</td><td>55</td><td>58</td><td>60</td></tr></table>	45	55	58	60	No	Off
45	55	58	60			
<table><tr><td>45</td><td>55</td><td>58</td><td>60</td></tr></table>	45	55	58	60	No	Off
45	55	58	60			
<table><tr><td>45</td><td>55</td><td>58</td><td>60</td></tr></table>	45	55	58	60	No	Off
45	55	58	60			
No more passes since swap flag is Off						

Example 3.3

```

001 ; sorting a list of ten numbers using bubble sort
002 [org 0x0100]
003     jmp start
004
005 data:    dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
006 swap:   db 0
007
008 start:   mov bx, 0                ; initialize array index to zero
009         mov byte [swap], 0       ; rest swap flag to no swaps
010
011 loop1:   mov ax, [data+bx]        ; load number in ax
012         cmp ax, [data+bx+2]      ; compare with next number
013         jbe noswap              ; no swap if already in order
014
015         mov dx, [data+bx+2]      ; load second element in dx
016         mov [data+bx+2], ax      ; store first number in second
017         mov [data+bx], dx        ; store second number in first
018         mov byte [swap], 1       ; flag that a swap has been done
019
020 noswap:  add bx, 2                ; advance bx to next index
021         cmp bx, 18               ; are we at last index
022         jne loop1               ; if not compare next two
023
024         cmp byte [swap], 1       ; check if a swap has been done
025         je bsort                ; if yes make another pass
026
027         mov ax, 0x4c00           ; terminate program
028         int 0x21

```

003	The jump instruction is placed to skip over data.
006	The swap flag can be stored in a register but as an example it is stored in memory and also to extend the concept at a later stage.
011-012	One element is read in AX and it is compared with the next element because memory to memory comparisons are not allowed.
013	If the JBE is changed to JB, not only the unnecessary swap on equal will be performed, there will be a major algorithmic flaw due to a logical error as in the case of equal elements the algorithm will never stop. JBE won't swap in the case of equal elements.
015-017	The swap is done using DX and AX registers in such a way that the values are crossed. The code uses the information that one of the elements is already in the AX register.
021	This time BX is compared with 18 instead of 20 even though the number of elements is same. This is because we pick an element and compare it with the next element. When we pick the 9th element we compare it with the next element and this is the last comparison, since if we pick the 10th element we will compare it with the 11th element and there is no 11th element in our case.
024-025	If a swap is done we repeat the whole process for possible more swaps.

Inside the debugger we observe that the JBE is changed to JNA due to the same reason as discussed for JNE and JNZ. The passes change the data in the same manner as we presented in our illustration above. If JBE in the code is changed to JAE the sort will change from ascending to descending. For signed numbers we can use JLE and JGE respectively for ascending and descending sort.

To clarify the difference of signed and unsigned jumps we change the data array in the last program to include some negative numbers as well. When JBE will be used on this data, i.e. with unsigned interpretation of the data and an ascending sort, the negative numbers will come at the end after the largest positive number. However JLE will bring the negative numbers at the very start of the list to bring them in proper ascending order according to a signed interpretation, even though they are large in magnitude. The data used is shown as below.

```
data:      dw 60, 55, 45, 50, -40, -35, 25, 30, 10, 0
```

This data includes some signed numbers as well. The JBE instruction will treat this data as an unsigned number and will cater only for the magnitude ignoring the sign. If the program is loaded in the debugger, the numbers will appear in their hexadecimal equivalent. The two numbers -40 and -35 are especially important as they are represented as FFD8 and FFDD. This data is not telling whether it is signed or unsigned. Our interpretation will decide whether it is a very large unsigned number or a signed number in two's complement form.

If the sorting algorithm is applied on the above data with JBE as the comparison instruction to sort in ascending order with unsigned interpretation, observe the comparisons of the two numbers FFD8 and FFDD. For example it will decide that FFDD > FFD8 since the first is larger in magnitude. At the end of sorting FFDD will be at the end of the list being declared the largest number and FFD8 will precede it to be the second largest.

If however the comparison instruction is changed to JLE and sorting is done on the same data it works similarly except on the two numbers FFDD and FFD8. This time JLE declares them to be smaller than every other number and also declares FFDD < FFD8. At the end of sorting, FFDD is

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

declared to be the smallest number followed by FFD8 and then 0000. This is in contrast to the last example where JBE was used. This happened because JLE interpreted our data as signed numbers, and as a signed number FFDD has its sign bit on signaling that it is a negative number in two's complement form which is smaller than 0000 and every positive number. However JBE did not give any significance to the sign bit and included it in the magnitude. Therefore it declared the negative numbers to be the largest numbers.

If the required interpretation was of signed numbers the result produced by JLE is correct and if the required interpretation was of unsigned numbers the result produced by JBE is correct. This is the very difference between signed and unsigned integers in higher level languages, where the compiler takes the responsibility of making the appropriate jump depending on the type of integer used. But it is only at this level that we can understand the actual mechanism going on. In assembly language, use of proper jump is the responsibility of the programmer, to convey the intentions to use the data as signed or as unsigned.

The remaining possibilities of signed descending sort and unsigned descending sort can be done on the same lines and are left as an exercise. Other conditional jumps work in the same manner and can be studied from the reference at the end. Several will be discussed in more detail when they are used in subsequent chapters.

EXERCISES

1. Which registers are changed by the CMP instruction?
2. What are the different types of jumps available? Describe position relative addressing.
3. If AX=8FFF and BX=0FFF and "cmp ax, bx" is executed, which of the following jumps will be taken? Each part is independent of others. Also give the value of Z, S, and C flags.
 - a. jg greater
 - b. jl smaller
 - c. ja above
 - d. jb below
4. Write a program to find the maximum number and the minimum number from an array of ten numbers.
5. Write a program to search a particular element from an array using binary search. If the element is found set AX to one and otherwise to zero.
6. Write a program to calculate the factorial of a number where factorial is defined as:

$$\text{factorial}(x) = x*(x-1)*(x-2)*\dots*1$$

$$\text{factorial}(0) = 1$$

Bit Manipulations

4.1. MULTIPLICATION ALGORITHM

With the important capability of decision making in our repertoire we move on to the discussion of an algorithm, which will help us uncover an important set of instructions in our processor used for bit manipulations.

Multiplication is a common process that we use, and we were trained to do in early schooling. Remember multiplying by a digit and then putting a cross and then multiplying with the next digit and putting two crosses and so on and summing the intermediate results in the end. Very familiar process but we never saw the process as an algorithm, and we need to see it as an algorithm to convey it to the processor.

To highlight the important thing in the algorithm we revise it on two 4bit binary numbers. The numbers are 1101 i.e. 13 and 0101 i.e. 5. The answer should be 65 or in binary 01000001. Observe that the answer is twice as long as the multiplier and the multiplicand. The multiplication is shown in the following figure.

```

1101 = 13
0101 = 5
-----
1101
0000x
1101xx
0000xxx
-----
01000001 = 65

```

We take the first digit of the multiplier and multiply it with the multiplicand. As the digit is one the answer is the multiplicand itself. So we place the multiplicand below the bar. Before multiplying with the next digit a cross is placed at the right most place on the next line and the result is placed shifted one digit left. However since the digit is zero, the result is zero. Next digit is one, multiplying with which, the answer is 1101. We put two crosses on the next line at the right most positions and place the result there shifted two places to the left. The fourth digit is zero, so the answer 0000 is placed with three crosses to its right.

Observe the beauty of binary base, as no real multiplication is needed at the digit level. If the digit is 0 the answer is 0 and if the digit is 1 the answer is the multiplicand itself. Also observe that for every next digit in the multiplier the answer is written shifted one more place to the left. No shifting for the first digit, once for the second, twice for the third and thrice for the fourth one. Adding all the intermediate answers the result is 01000001=65 as desired. Crosses are treated as zero in this addition.

Before formulating the algorithm for this problem, we need some more instructions that can shift a number so that we use this instruction for our multiplicand shifting and also some way to check the bits of the multiplier one by one.

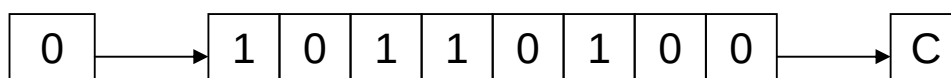
4.2. SHIFTING AND ROTATIONS

The set of shifting and rotation instructions is one of the most useful set in any processor's instruction set. They simplify really complex tasks to a very

neat and concise algorithm. The following shifting and rotation operations are available in our processor.

Shift Logical Right (SHR)

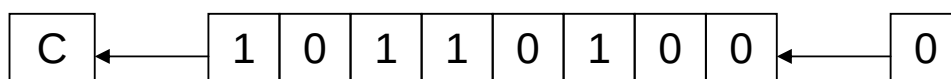
The shift logical right operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag. Imagine that there is a pipe filled to capacity with eight balls. The pipe is open from both ends and there is a basket at the right end to hold anything dropping from there. The operation of shift logical right is to force a white ball from the left end. The operation is depicted in the following illustration.



White balls represent zero bits while black balls represent one bits. Sixteen bit shifting is done the same way with a pipe of double capacity.

Shift Logical Left (SHL) / Shift Arithmetic Left (SAL)

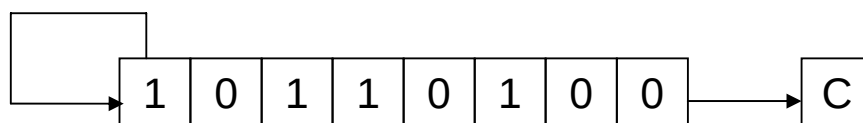
The shift logical left operation is the exact opposite of shift logical right. In this operation the zero bit is inserted from the right and every bit moves one position to its left with the most significant bit dropping into the carry flag. Shift arithmetic left is just another name for shift logical left. The operation is again exemplified with the following illustration of ball and pipes.



Shift Arithmetic Right (SAR)

A signed number holds the sign in its most significant bit. If this bit was one a logical right shifting will change the sign of this number because of insertion of a zero from the left. The sign of a signed number should not change because of shifting.

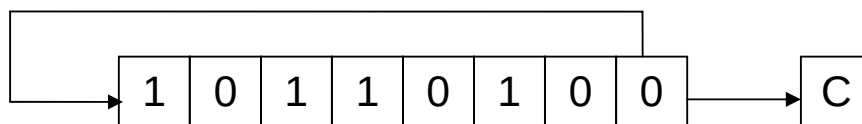
The operation of shift arithmetic right is therefore to shift every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket. The sign bit is retained in this operation. The operation is further illustrated below.



The left shifting operation is basically multiplication by 2 while the right shifting operation is division by two. However for signed numbers division by two can be accomplished by using shift arithmetic right and not shift logical right. The left shift operation is equivalent to multiplication except when an important bit is dropped from the left. The overflow flag will signal this condition if it occurs and can be checked with JO. For division by 2 of a signed number logical right shifting will give a wrong answer for a negative number as the zero inserted from the left will change its sign. To retain the sign flag and still effectively divide by two the shift arithmetic right instruction must be used on signed numbers.

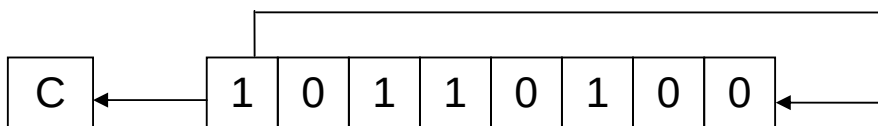
Rotate Right (ROR)

In the rotate right operation every bit moves one position to the right and the bit dropped from the right is inserted at the left. This bit is also copied into the carry flag. The operation can be understood by imagining that the pipe used for shifting has been molded such that both ends coincide. Now when the first ball is forced to move forward, every ball moves one step forward with the last ball entering the pipe from its other end occupying the first ball's old position. The carry basket takes a snapshot of this ball leaving one end of the pipe and entering from the other.



Rotate Left (ROL)

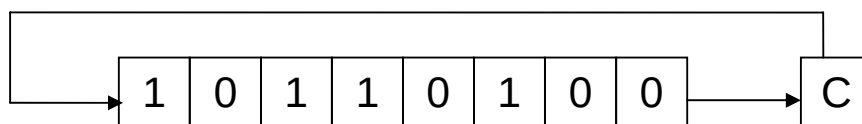
In the operation of rotate left instruction, the most significant bit is copied to the carry flag and is inserted from the right, causing every bit to move one position to the left. It is the reverse of the rotate right instruction. Rotation can be of eight or sixteen bits. The following illustration will make the concept clear using the same pipe and balls example.



Rotate Through Carry Right (RCR)

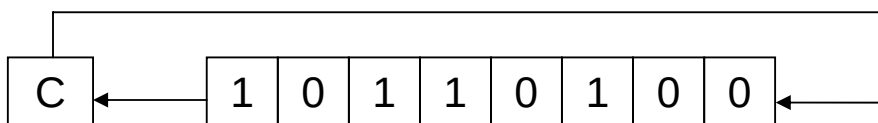
In the rotate through carry right instruction, the carry flag is inserted from the left, every bit moves one position to the right, and the right most bit is dropped in the carry flag. Effectively this is a nine bit or a seventeen bit rotation instead of the eight or sixteen bit rotation as in the case of simple rotations.

Imagine the circular molded pipe as used in the simple rotations but this time the carry position is part of the circle between the two ends of the pipe. Pushing the carry ball from the left causes every ball to move one step to its right and the right most bit occupying the carry place. The idea is further illustrated below.



Rotate Through Carry Left (RCL)

The exact opposite of rotate through carry right instruction is the rotate through carry left instruction. In its operation the carry flag is inserted from the right causing every bit to move one location to its left and the most significant bit occupying the carry flag. The concept is illustrated below in the same manner as in the last example.



4.3. MULTIPLICATION IN ASSEMBLY LANGUAGE

In the multiplication algorithm discussed above we revised the way we multiplied number in lower classes, and gave an example of that method on binary numbers. We make a simple modification to the traditional algorithm before we proceed to formulate it in assembly language.

In the traditional algorithm we calculate all intermediate answers and then sum them to get the final answer. If we add every intermediate answer to accumulate the result, the result will be same in the end, except that we do not have to remember a lot of intermediate answers during the whole multiplication. The multiplication with the new algorithm is shown below.

1101 = 13	Accumulated Result
0101 = 5	
-----	0 (Initial Value)
1101 = 13	0 + 13 = 13
0000x = 0	13 + 0 = 13
1101xx = 52	13 + 52 = 65
0000xxx = 0	65 + 0 = 65 (Answer)

We try to identify steps of our algorithm. First we set the result to zero. Then we check the right most bit of multiplier. If it is one add the multiplicand to the result, and if it is zero perform no addition. Left shift the multiplicand before the next bit of multiplier is tested. The left shifting of the multiplicand is performed regardless of the value of the multiplier's right most bit. Just like the crosses in traditional multiplication are always placed to mark the ones, tens, thousands, etc. places. Then check the next bit and if it is one add the shifted value of the multiplicand to the result. Repeat for as many digits as there are in the multiplier, 4 in our example. Formulating the steps of the algorithm we get:

- Shift the multiplier to the right.
- If CF=1 add the multiplicand to the result.
- Shift the multiplicand to the right.
- Repeat the algorithm 4 times.

For an 8bit multiplication the algorithm will be repeated 8 times and for a sixteen bit multiplication it will be repeated 16 times, whatever the size of the multiplier is.

The algorithm uses the fact that shifting right forces the right most bit to drop in the carry flag. If we test the carry flag using JC we are effectively testing the right most bit of the multiplier. Another shifting will cause the next bit to drop in the next iteration and so on. So our task of checking bits one by one is satisfied using the shift operation. There are many other methods to do this bit testing as well, however we exemplify one of the methods in this example.

In the first iteration there is no shifting just like there is no cross in traditional multiplication in the first pass. Therefore we placed the left shifting of the multiplicand after the addition step. However the right shifting of multiplier must be before the addition as the addition step's execution depends upon its result.

We introduce an assembly language program to perform this 4bit multiplication. The algorithm is extensible to more bits but there are a few complications, which are left to be discussed later. For now we do a 4bit multiplication to keep the algorithm simple.

Example 4.1	
01	; 4bit multiplication algorithm
02	[org 0x100]
03	jmp start
04	
05	multiplicand: db 13 ; 4bit multiplicand (8bit space)
06	multiplier: db 5 ; 4bit multiplier

07		result:	db	0		; 8bit result
08						
09		start:	mov	cl, 4		; initialize bit count to four
10			mov	bl, [multiplicand]		; load multiplicand in bl
11			mov	dl, [multiplier]		; load multiplier in dl
12						
13		checkbit:	shr	dl, 1		; move right most bit in carry
14			jnc	skip		; skip addition if bit is zero
15						
16			add	[result], bl		; accumulate result
17						
18		skip:	shl	bl, 1		; shift multiplicand left
19			dec	cl		; decrement bit count
20			jnz	checkbit		; repeat if bits left
21						
22			mov	ax, 0x4c00		; terminate program
23			int	0x21		

04-06	The numbers to be multiplied are constants for now. The multiplication is four bit so the answer is stored in an 8bit register. If the operands were 8bit the answer would be 16bit and if the operands were 16bit the answer would be 32bit. Since eight bits can fit in a byte we have used 4bit multiplication as our first example.
07	Since addition by zero means nothing we skip the addition step if the rightmost bit of the multiplier is zero. If the jump is not taken the shifted value of the multiplicand is added to the result.
14-16	The multiplicand is left shifted in every iteration regardless of the multiplier bit.
18	DEC is a new instruction but its operation should be immediately understandable with the knowledge gained till now. It simply subtracts one from its single operand.
19	The JNZ instruction causes the algorithm to repeat till any bits of the multiplier are left
20	

Inside the debugger observe the working of the SHR and SHL instructions. The SHR instruction is effectively dividing its operand by two and the remainder is stored in the carry flag from where we test it. The SHL instruction is multiplying its operand by two so that it is added at one place more towards the left in the result.

4.4. EXTENDED OPERATIONS

We performed a 4bit multiplication to explain the algorithm however the real advantage of the computer is when we ask it to multiply large numbers, Numbers whose multiplication takes real time. If we have an 8bit number we can do the multiplication in word registers, but are we limited to word operations? What if we want to multiply 32bit or even larger numbers? We are certainly not limited. Assembly language only provides us the basic building blocks. We build a plaza out of these blocks, or a building, or a classic piece of architecture is only dependant upon our imagination. With our logic we can extend these algorithms as much as we want.

Our next example will be multiplication of 16bit numbers to produce a 32bit answer. However for a 32bit answer we need a way to shift a 32bit number and a way to add 32bit numbers. We cannot depend on 16bit shifting as we have 16 significant bits in our multiplicand and shifting any bit towards the left may drop a valuable bit causing a totally wrong result. A valuable bit means any bit that is one. Dropping a zero bit doesn't cause any difference. So we place the 16it number in 32bit space with the upper 16 bits zeroed so that the sixteen shift operations don't cause any valuable bit to drop. Even though the numbers were 16bit we need 32bit operations to multiply correctly.

To clarify this necessity, we take example of a number 40000 or 9C40 in hexadecimal. In binary it is represented as 1001110001000000. To multiply

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

by two we shift it one place to the left. The answer we get is 0011100010000000 and the left most one is dropped in the carry flag. The answer should be the 17bit number 0x13880 but it is 0x3880, which are 14464 in decimal instead of the expected 80000. We should be careful of this situation whenever shifting is used.

Extended Shifting

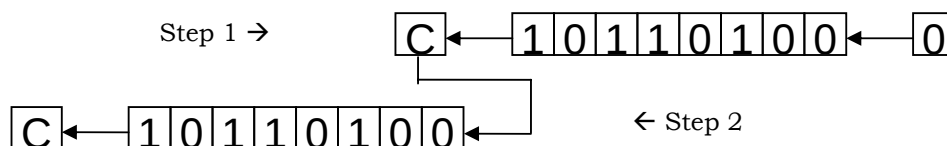
Using our basic shifting and rotation instructions we can effectively shift a 32bit number in memory word by word. We cannot shift the whole number at once since our architecture is limited to word operations. The algorithm we use consists of just two instructions and we name it extended shifting.

```
num1:    dd    40000
          shl   word [num1], 1
          rcl   word [num1+2], 1
```

The DD directive reserves a 32bit space in memory, however the value we placed there will fit in 16bits. So we can safely shift the number left 16 times. The least significant word is accessible at num1 and the most significant word is accessible at num1+2.

The two instructions are carefully crafted such that the first one shifts the lower word towards the left and the most significant bit of that word is dropped in carry. With the next instruction we push that dropped bit into the least significant bit of the next word effectively joining the two 16bit words. The final carry after the second instruction will be the most significant bit of the higher word, which for this number will always be zero.

The following illustration will clarify the concept. The pipe on the right contains the lower half and the pipe on the left contains the upper half. The first instruction forced a zero from the right into the lower half and the left most bit is saved in carry, and from there it is pushed into the upper half and the upper half is shifted as well.



For shifting right the exact opposite is done however care must be taken to shift right the upper half first and then rotate through carry right the lower half for obvious reasons. The instructions to do this are.

```
num1:    dd    40000
          shr   word [num1+2], 1
          rcr   word [num1], 1
```

The same logic has worked. The shift placed the least significant bit of the upper half in the carry flag and it was pushed from right into the lower half. For a signed shift we would have used the shift arithmetic right instruction instead of the shift logical right instruction.

The extension we have done is not limited to 32bits. We can shift a number of any size say 1024 bits. The second instruction will be repeated a number of times and we can achieve the desired effect. Using two simple instructions we have increased the capability of the operation to effectively an unlimited number of bits. The actual limit is the available memory as even the segment limit can be catered with a little thought.

Extended Addition and Subtraction

We also needed 32bit addition for multiplication of 16bit numbers. The idea of extension is same here. However we need to introduce a new instruction at this place. The instruction is ADC or “add with carry.” Normal addition has two operands and the second operand is added to the first

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

operand. However ADC has three operands. The third implied operand is the carry flag. The ADC instruction is specifically placed for extending the capability of ADD. Numbers of any size can be added using a proper combination of ADD and ADC. All basic building blocks are provided for the assembly language programmer, and the programmer can extend its capabilities as much as needed by using these fine instructions in appropriate combinations.

Further clarifying the operation of ADC, consider an instruction “ADC AX, BX.” Normal addition would have just added BX to AX, however ADC first adds the carry flag to AX and then adds BX to AX. Therefore the last carry is also included in the result.

The algorithm should be apparent by now. The lower halves of the two numbers to be added are first added with a normal addition. For the upper halves a normal addition would lose track of a possible carry from the lower halves and the answer would be wrong. If a carry was generated it should go to the upper half. Therefore the upper halves are added with an addition with carry instruction.

Since one operand must be in register, ax is used to read the lower and upper halves of the source one by one. The destination is directly updated. The set of instructions goes here.

```
dest:      dd  40000
src:       dd  80000
          mov  ax, [src]
          add  word [dest], ax
          mov  ax, [src+2]
          adc  word [dest+2], ax
```

To further extend it more addition with carries will be used. However the carry from last addition will be wasted as there will always be a size limit where the results and the numbers are stored. This carry will remain in the carry flag to be tested for a possible overflow.

For subtraction the same logic will be used and just like addition with carry there is an instruction to subtract with borrows called SBB. Borrow in the name means the carry flag and is used just for clarity. Or we can say that the carry flag holds the carry for addition instructions and the borrow for subtraction instructions. Also the carry is generated at the 17th bit and the borrow is also taken from the 17th bit. Also there is no single instruction that needs borrow and carry in their independent meanings at the same time. Therefore it is logical to use the same flag for both tasks.

We extend subtraction with a very similar algorithm. The lower halves must be subtracted normally while the upper halves must be subtracted with a subtract with borrow instruction so that if the lower halves needed a borrow, a one is subtracted from the upper halves. The algorithm is as under.

```
dest:      dd  40000
src:       dd  80000
          mov  ax, [src]
          sub  word [dest], ax
          mov  ax, [src+2]
          sbb  word [dest+2], ax
```

Extended Multiplication

We use extended shifting and extended addition to formulate our algorithm to do extended multiplication. The multiplier is still stored in 16bits since we only need to check its bits one by one. The multiplicand however cannot be stored in 16bits otherwise on left shifting its significant bits might get lost. Therefore it has to be stored in 32bits and the shifting and addition used to accumulate the result must be 32bits as well.

Example 4.2

01		; 16bit multiplication	
02		[org 0x0100]	
03		jmp	start
04			
05		multiplicand: dd	1300 ; 16bit multiplicand 32bit space
06		multiplier: dw	500 ; 16bit multiplier
07		result: dd	0 ; 32bit result
08			
09	start:	mov cl, 16	; initialize bit count to 16
10		mov dx, [multiplier]	; load multiplier in dx
11			
12	checkbit:	shr dx, 1	; move right most bit in carry
13		jnc skip	; skip addition if bit is zero
14			
15		mov ax, [multiplicand]	
16		add [result], ax	; add less significant word
17		mov ax, [multiplicand+2]	
18		adc [result+2], ax	; add more significant word
19			
20	skip:	shl word [multiplicand], 1	
21		rcl word [multiplicand+2], 1	; shift multiplicand left
22		dec cl	; decrement bit count
23		jnz checkbit	; repeat if bits left
24			
25		mov ax, 0x4c00	; terminate program
26		int 0x21	
05-07	The multiplicand and the multiplier are stored in 32bit space while the multiplier is stored as a word.		
10	The multiplier is loaded in DX where it will be shifted bit by bit. It can be directly shifted in memory as well.		
15-18	The multiplicand is added to the result using extended 32bit addition.		
20-21	The multiplicand is shifted left as a 32bit number using extended shifting operation.		

The multiplicand will occupy the space from 0103-0106, the multiplier will occupy space from 0107-0108 and the result will occupy the space from 0109-010C. Inside the debugger observe the changes in these memory locations during the course of the algorithm. The extended shifting and addition operations provide the same effect as would be provided if there were 32bit addition and shifting operations available in the instruction set.

At the end of the algorithm the result memory locations contain the value 0009EB10 which is 65000 in decimal; the desired answer. Also observe that the number 00000514 which is 1300 in decimal, our multiplicand, has become 05140000 after being left shifted 16 times. Our extended shifting has given the same result as if a 32bit number is left shifted 16 times as a unit.

There are many other important applications of the shifting and rotation operations in addition to this example of the multiplication algorithm. More examples will come in coming chapters.

4.5. BITWISE LOGICAL OPERATIONS

The 8088 processor provides us with a few logical operations that operate at the bit level. The logical operations are the same as discussed in computer logic design; however our perspective will be a little different. The four basic operations are AND, OR, XOR, and NOT.

The important thing about these operations is that they are bitwise. This means that if “and ax, bx” instruction is given, then the operation of AND is applied on corresponding bits of AX and BX. There are 16 AND operations as a result; one for every bit of AX. Bit 0 of AX will be set if both its original value and Bit 0 of BX are set, bit 1 will be set if both its original value and Bit 1 of BX are set, and so on for the remaining bits. These operations are conducted in parallel on the sixteen bits. Similarly the operations of other logical operations are bitwise as well.

AND operation

AND performs the logical bitwise *and* of the two operands (byte or word) and returns the result to the destination operand. A bit in the result is set if both corresponding bits of the original operands are set; otherwise the bit is cleared as shown in the truth table. Examples are “and ax, bx” and “and byte [mem], 5.” All possibilities that are legal for addition are also legal for the AND operation. The different thing is the bitwise behavior of this operation.

X	Y	X and Y
0	0	0
0	1	0
1	0	0
1	1	1

OR operation

OR performs the logical bitwise “inclusive or” of the two operands (byte or word) and returns the result to the destination operand. A bit in the result is set if either or both corresponding bits in the original operands are set otherwise the result bit is cleared as shown in the truth table. Examples are “or ax, bx” and “or byte [mem], 5.”

X	Y	X or Y
0	0	0
0	1	1
1	0	1
1	1	1

XOR operation

XOR (Exclusive Or) performs the logical bitwise “exclusive or” of the two operands and returns the result to the destination operand. A bit in the result is set if the corresponding bits of the original operands contain opposite values (one is set, the other is cleared) otherwise the result bit is cleared as shown in the truth table. XOR is a very important operation due to the property that it is a reversible operation. It is used in many cryptography algorithms, image processing, and in drawing operations. Examples are “xor ax, bx” and “xor byte [mem], 5.”

X	Y	X xor Y
0	0	0
0	1	1
1	0	1
1	1	0

NOT operation

NOT inverts the bits (forms the one’s complement) of the byte or word operand. Unlike the other logical operations, this is a single operand instruction, and is not purely a logical operation in the sense the others are, but it is still traditionally counted in the same set. Examples are “not ax” and “not byte [mem], 5.”

4.6. MASKING OPERATIONS

Selective Bit Clearing

Another use of AND is to make selective bits zero in its destination operand. The source operand is loaded with a mask containing one at positions which are retain their old value and zero at positions which are to be zeroed. The effect of applying this operation on the destination with mask in the source is to clear the desired bits. This operation is called masking. For example if the lower nibble is to be cleared then the operation can be applied with F0 in the source. The upper nibble will retain its old value and the lower nibble will be cleared.

Selective Bit Setting

The operation can be used as a masking operation to set selective bits. The bits in the mask are cleared at positions which are to retain their values, and are set at positions which are to be set. For example to set the lower nibble of the destination operand, the operation should be applied with a mask of 0F in the source. The upper nibble will retain its value and the lower nibble will be set as a result.

Selective Bit Inversion

XOR can also be used as a masking operation to invert selective bits. The bits in the mask are cleared at positions, which are to retain their values, and are set at positions, which are to be inverted. For example to invert the lower nibble of the destination operand, the operand should be applied with a mask of 0F in the source. The upper nibble will retain its value and the lower nibble will be set as a result. Compare this with NOT which inverts everything. XOR on the other hand allows inverting selective bits.

Selective Bit Testing

AND can be used to check whether particular bits of a number are set or not. Previously we used shifting and JC to test bits one by one. Now we introduce another way to test bits, which is more powerful in the sense that any bit can be tested anytime and not necessarily in order. AND can be applied on a destination with a 1-bit in the desired position and a source, which is to be checked. If the destination is zero as a result, which can be checked with a JZ instruction, the bit at the desired position in the source was clear.

However the AND operation destroys the destination mask, which might be needed later as well. Therefore Intel provided us with another instruction analogous to CMP, which is non-destructive subtraction. This is the TEST instruction and is a non-destructive AND operation. It doesn't change the destination and only sets the flags according to the AND operation. By checking the flags, we can see if the desired bit was set or cleared.

We change our multiplication algorithm to use selective bit testing instead of checking bits one by one using the shifting operations.

Example 4.3	
01	; 16bit multiplication using test for bit testing
02	[org 0x0100]
03	jmp start
04	
05	multiplicand: dd 1300 ; 16bit multiplicand 32bit space
06	multiplier: dw 500 ; 16bit multiplier
07	result: dd 0 ; 32bit result
08	
09	start: mov cl, 16 ; initialize bit count to 16
10	mov bx, 1 ; initialize bit mask
11	
12	checkbit: test bx, [multiplier] ; move right most bit in carry
13	jz skip ; skip addition if bit is zero
14	
15	mov ax, [multiplicand]
16	add [result], ax ; add less significant word
17	mov ax, [multiplicand+2]
18	adc [result+2], ax ; add more significant word
19	
20	skip: shl word [multiplicand], 1
21	rcl word [multiplicand+2], 1 ; shift multiplicand left
22	shl bx, 1 ; shift mask towards next bit
23	dec cl ; decrement bit count
24	jnz checkbit ; repeat if bits left
25	
26	mov ax, 0x4c00 ; terminate program
27	int 0x21
12	The test instruction is used for bit testing. BX holds the mask and in every next iteration it is shifting left, as our concerned bit is now the next bit.
22-24	We can do without counting in this example. We can stop as soon as our mask in BX becomes zero. These are the small tricks that assembly allows us to do and optimize our code as a result.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

Inside the debugger observe that both the memory location and the mask in BX do not change as a result of TEST instruction. Also observe how our mask is shifting towards the left so that the next TEST instruction tests the next bit. In the end we get the same result of 0009EB10 as in the previous example.

EXERCISES

- Write a program to swap every pair of bits in the AX register.
- Give the value of the AX register and the carry flag after each of the following instructions.

```

stc
mov ax, <your rollnumber>
adc ah, <first character of your name>
cmc
xor ah, al
mov cl, 4
shr al, cl
rcr ah, cl

```
- Write a program to swap the nibbles in each byte of the AX register.
- Calculate the number of one bits in BX and complement an equal number of least significant bits in AX.
HINT: Use the XOR instruction
- Write a program to multiply two 32bit numbers and store the answer in a 64bit location.
- Declare a 32byte buffer containing random data. Consider for this problem that the bits in these 32 bytes are numbered from 0 to 255. Declare another byte that contains the starting bit number. Write a program to copy the byte starting at this starting bit number in the AX register. Be careful that the starting bit number may not be a multiple of 8 and therefore the bits of the desired byte will be split into two bytes.
- AX contains a number between 0-15. Write code to complement the corresponding bit in BX. For example if AX contains 6; complement the 6th bit of BX.
- AX contains a non-zero number. Count the number of ones in it and store the result back in AX. Repeat the process on the result (AX) until AX contains one. Calculate in BX the number of iterations it took to make AX one. For example BX should contain 2 in the following case:
AX = 1100 0101 1010 0011 (input – 8 ones)
AX = 0000 0000 0000 1000 (after first iteration – 1 one)
AX = 0000 0000 0000 0001 (after second iteration – 1 one) STOP

5

Subroutines

5.1. PROGRAM FLOW

Till now we have accumulated the very basic tools of assembly language programming. A very important weapon in our arsenal is the conditional jump instruction. During the course of last two chapters we used these tools to write two very useful algorithms of sorting and multiplication. The multiplication algorithm is useful even though there is a MUL instruction in the 8088 instruction set, which can multiply 8bit and 16bit operands. This is because of the extensibility of our algorithm, as it is not limited to 16bits and can do 32bit or 64bit multiplication with minor changes.

Both of these algorithms will be used a number of times in any program of a reasonable size and complexity. An application does not only need to multiply at a single point in code; it multiplies at a number of places. If multiplication or sorting is needed at 100 places in code, copying it 100 times is a totally infeasible solution. Maintaining such a code is an impossible task.

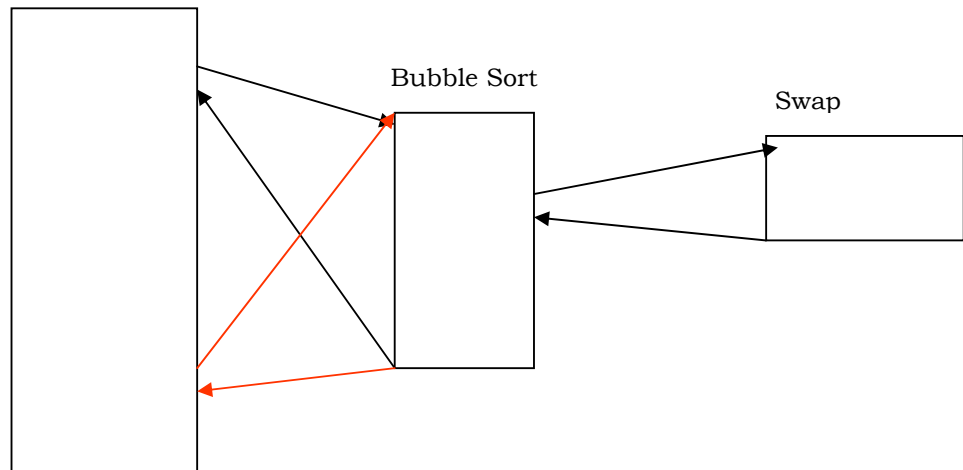
The straightforward solution to this problem using the concepts we have acquainted till now is to write the code at one place with a label, and whenever we need to sort we jump to this label. But there is problem with this logic, and the problem is that after sorting is complete how the processor will know where to go back. The immediate answer is to jump back to a label following the jump to bubble sort. But we have jumped to bubble sort from 100 places in code. Which of the 100 positions in code should we jump back? Jump back at the first invocation, but jump has a single fixed target. How will the second invocation work? The second jump to bubble sort will never have control back at the next line.

Instruction are tied to one another forming an execution thread, just like a knitted thread where pieces of cotton of different sizes are twisted together to form a thread. This thread of execution is our program. The jump instruction breaks this thread permanently, making a permanent diversion, like a turn on a highway. The conditional jump selects one of the two possible directions, like right or left turn on a road. So there is no concept of returning.

However there are roundabouts on roads as well that take us back from where we started after having traveled on the boundary of the round. This is the concept of a temporary diversion. Two or more permanent diversions can take us back from where we started, just like two or more road turns can take us back to the starting point, but they are still permanent diversions in their nature.

We need some way to implement the concept of temporary diversion in assembly language. We want to create a roundabout of bubble sort, another roundabout of our multiplication algorithm, so that we can enter into the roundabout whenever we need it and return back to wherever we left from after completing the round.

Program



Key point in the above discussion is returning to where we left from, like a loop in a knitted thread. Diversion should be temporary and not permanent. The code of bubble sort written at one place, multiply at another, and we temporarily divert to that place, thus avoiding a repetition of code at a 100 places.

CALL and RET

In every processor, instructions are available to divert temporarily and to divert permanently. The instructions for permanent diversion in 8088 are the jump instructions, while the instruction for temporary diversion is the CALL instruction. The word call must be familiar to the readers from subroutine call in higher level languages. The CALL instruction allows temporary diversion and therefore reusability of code. Now we can place the code for bubble sort at one place and reuse it again and again. This was not possible with permanent diversion. Actually the 8088 permanent diversion mechanism can be tricked to achieve temporary diversion. However it is not possible without getting into a lot of trouble. The key idea in doing it this way is to use the jump instruction form that takes a register as argument. Therefore this is not impossible but this is not the way it is done.

The natural way to do this is to use the CALL instruction followed by a label, just like JMP is followed by a label. Execution will divert to the code following the label. Till now the operation has been similar to the JMP instruction. When the subroutine completes we need to return. The RET instruction is used for this purpose. The word return holds in its meaning that we are to return from where we came and need no explicit destination. Therefore RET takes no arguments and transfers control back to the instruction following the CALL that took us in this subroutine. The actual technical process that informs RET where to return will be discussed later after we have discussed the system stack.

CALL takes a label as argument and execution starts from that label, until the RET instruction is encountered and it takes execution back to the instruction following the CALL. Both the instructions are commonly used as a pair, however technically they are independent in their operation. The RET works regardless of the CALL and the CALL works regardless of the RET. If you CALL a subroutine it will not complain if there is no RET present and similarly if you RET without being called it won't complain. It is a logical pair and is used as a pair in every decent code. However sometimes we play tricks with the processor and we use CALL or RET alone. This will become clear when we need to play such tricks in later chapters.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

Parameters

We intend to write the bubble sort code at one place and CALL it whenever needed. An immediately visible problem is that whenever we call this subroutine it will sort the same array in the same order. However in a real application we will need to sort various arrays of various sizes. We might sometimes need an ascending sort and descending at other times. Similarly our data may be signed or unsigned. Such pieces of information that may change from invocation to invocation and should be passed from the caller to the subroutine are called parameters.

There must be some way of passing these parameters to the subroutine. Revising the subroutine temporary flow breakage mechanism, the most straightforward way is to use registers. The CALL mechanism breaks the thread of execution and does not change registers, except IP which must change for processor to start executing at another place, and SP whose change will be discussed in detail later. Any of the other registers can hold parameters for the subroutine.

5.2. OUR FIRST SUBROUTINE

Now we want to modify the bubble sort code so that it works as a subroutine. We place a label at the start of bubble sort code, which works as the anchor point and will be used in the CALL instruction to call the subroutine. We also place a RET at the end of the algorithm to return from where we called the subroutine.

Example 5.1	
01	; bubble sort algorithm as a subroutine
02	[org 0x0100]
03	jmp start
04	
05	data: dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
06	swap: db 0
07	
08	bubblesort: dec cx ; last element not compared
09	shl cx, 1 ; turn into byte count
10	
11	mainloop: mov si, 0 ; initialize array index to zero
12	mov byte [swap], 0 ; reset swap flag to no swaps
13	
14	innerloop: mov ax, [bx+si] ; load number in ax
15	cmp ax, [bx+si+2] ; compare with next number
16	jbe noswap ; no swap if already in order
17	
18	mov dx, [bx+si+2] ; load second element in dx
19	mov [bx+si], dx ; store first number in second
20	mov [bx+si+2], ax ; store second number in first
21	mov byte [swap], 1 ; flag that a swap has been done
22	
23	noswap: add si, 2 ; advance si to next index
24	cmp si, cx ; are we at last index
25	jne innerloop ; if not compare next two
26	
27	cmp byte [swap], 1 ; check if a swap has been done
28	je mainloop ; if yes make another pass
29	
30	ret ; go back to where we came from
31	
32	start: mov bx, data ; send start of array in bx
33	mov cx, 10 ; send count of elements in cx
34	call bubblesort ; call our subroutine
35	
36	mov ax, 0x4c00 ; terminate program
37	int 0x21
08-09	The routine has received the count of elements in CX. Since it makes one less comparison than the number of elements it decrements it. Then it multiplies it by two since this a word array and each element

14	takes two bytes. Left shifting has been used to multiply by two.
32-37	Base+index+offset addressing has been used. BX holds the start of array, SI the offset into it and an offset of 2 when the next element is to be read. BX can be directly changed but then a separate counter would be needed, as SI is directly compared with CX in our case.
	The code starting from the start label is our main program analogous to the main in the C language. BX and CX hold our parameters for the bubblesort subroutine and the CALL is made to invoke the subroutine.

Inside the debugger we observe the same unsigned data that we are so used to now. The number 0103 is passed via BX to the subroutine which is the start of our data and the number 000A via CX which is the number of elements in our data. If we step over the CALL instruction we see our data sorted in a single step and we are at the termination instructions. The processor has jumped to the bubblesort routine, executed it to completion, and returned back from it but the process was hidden due to the step over command. If however we trace into the CALL instruction, we land at the first instruction of our routine. At the end of the routine, when the RET instruction is executed, we immediately land back to our termination instructions, to be precise the instruction following the CALL.

Also observe that with the CALL instruction SP is decremented by two from FFFE to FFEC, and the stack windows shows 0150 at its top. As the RET is executed SP is recovered and the 0150 is also removed from the stack. Match it with the address of the instruction following the CALL which is 0150 as well. The 0150 removed from the stack by the RET instruction has been loaded into the IP register thereby resuming execution from address 0150. CALL placed where to return on the stack for the RET instruction. The stack is automatically used with the CALL and RET instructions. Stack will be explained in detail later, however the idea is that the one who is departing stores the address to return at a known place. This is the place using which CALL and RET coordinate. How this placed is actually used by the CALL and RET instructions will be described after the stack is discussed.

After emphasizing reusability so much, it is time for another example which uses the same bubblesort routine on two different arrays of different sizes.

Example 5.2	
01	; bubble sort subroutine called twice
02	[org 0x0100]
03	jmp start
04	
05	data: dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
06	data2: dw 328, 329, 898, 8923, 8293, 2345, 10, 877, 355, 98
07	dw 888, 533, 2000, 1020, 30, 200, 761, 167, 90, 5
08	swap: db 0
09	
10	bubblesort: dec cx ; last element not compared
11	shl cx, 1 ; turn into byte count
12	
13	mainloop: mov si, 0 ; initialize array index to zero
14	mov byte [swap], 0 ; reset swap flag to no swaps
15	
16	innerloop: mov ax, [bx+si] ; load number in ax
17	cmp ax, [bx+si+2] ; compare with next number
18	jbe noswap ; no swap if already in order
19	
20	mov dx, [bx+si+2] ; load second element in dx
21	mov [bx+si], dx ; store first number in second
22	mov [bx+si+2], ax ; store second number in first
23	mov byte [swap], 1 ; flag that a swap has been done
24	
25	noswap: add si, 2 ; advance si to next index

26		cmp si, cx ; are we at last index	
27		jne innerloop ; if not compare next two	
28			
29		cmp byte [swap], 1 ; check if a swap has been done	
30		je mainloop ; if yes make another pass	
31			
32		ret ; go back to where we came from	
33			
34	start:	mov bx, data ; send start of array in bx	
35		mov cx, 10 ; send count of elements in cx	
36		call bubblesort ; call our subroutine	
37			
38		mov bx, data2 ; send start of array in bx	
39		mov cx, 20 ; send count of elements in cx	
40		call bubblesort ; call our subroutine again	
41			
42		mov ax, 0x4c00 ; terminate program	
43		int 0x21	

05-07	There are two different data arrays declared. One of 10 elements and the other of 20 elements. The second array is declared on two lines, where the second line is continuation of the first. No additional label is needed since they are situated consecutively in memory.
34-40	The other change is in the main where the bubblesort subroutine is called twice, once on the first array and once on the second.

Inside the debugger observe that stepping over the first call, the first array is sorted and stepping over the second call the second array is sorted. If however we step in SP is decremented and the stack holds 0178 which is the address of the instruction following the call. The RET consumes that 0178 and restores SP. The next CALL places 0181 on the stack and SP is again decremented. The RET consumes this number and execution resumes from the instruction at 0181. This is the coordinated function of CALL and RET using the stack.

In both of the above examples, there is a shortcoming. The subroutine to sort the elements is destroying the registers AX, CX, DX, and SI. That means that the caller of this routine has to make sure that it does not hold any important data in these registers before calling this function, because after the call has returned the registers will be containing meaningless data for the caller. With a program containing thousands of subroutines expecting the caller to remember the set of modified registers for each subroutine is unrealistic and unreasonable. Also registers are limited in number, and restricting the caller on the use of register will make the caller's job very tough. This shortcoming will be removed using the very important system stack.

5.3. STACK

Stack is a data structure that behaves in a first in last out manner. It can contain many elements and there is only one way in and out of the container. When an element is inserted it sits on top of all other elements and when an element is removed the one sitting at top of all others is removed first. To visualize the structure consider a test tube and put some balls in it. The second ball will come above the first and the third will come above the second. When a ball is taken out only the one at the top can be removed. The operation of placing an element on top of the stack is called pushing the element and the operation of removing an element from the top of the stack is called popping the element. The last thing pushed is popped out first; the last in first out behavior.

We can peek at any ball inside the test tube but we cannot remove it without removing every ball on top of it. Similarly we can read any element from the stack but cannot remove it without removing everything above it. The stack operations of pushing and popping only work at the top of the

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

stack. This top of stack is contained in the SP register. The physical address of the stack is obtained by the SS:SP combination. The stack segment registers tells where the stack is located and the stack pointer marks the top of stack inside this segment.

Whenever an element is pushed on the stack SP is decremented by two as the 8088 stack works on word sized elements. Single bytes cannot be pushed or popped from the stack. Also it is a decrementing stack. Another possibility is an incrementing stack. A decrementing stack moves from higher addresses to lower addresses as elements are added in it while an incrementing stack moves from lower addresses to higher addresses as elements are added. There is no special reason or argument in favor of one or another, and more or less depends on the choice of the designers. Another processor 8051 by the same manufacturer has an incrementing stack while 8088 has a decrementing one.

Memory is like a shelf numbered as zero at the top and the maximum at the bottom. If a decrementing stack starts at shelf 5, the first item is placed in shelf 5, the next item is placed in shelf 4, the next in shelf 3 and so on. The operations of placing items on the stack and removing them from there are called push and pop. The push operation copies its operand on the stack, while the pop operation makes a copy from the top of the stack into its operand. When an item is pushed on a decrementing stack, the top of the stack is first decremented and the element is then copied into this space. With a pop the element at the top of the stack is copied into the pop operand and the top of stack is incremented afterwards.

The basic use of the stack is to save things and recover from there when needed. For example we discussed the shortcoming in our last example that it destroyed the caller's registers, and the callers are not supposed to remember which registers are destroyed by the thousand routines they use. Using the stack the subroutine can save the caller's value of the registers on the stack, and recover them from there before returning. Meanwhile the subroutine can freely use the registers. From the caller's point of view if the registers contain the same value before and after the call, it doesn't matter if the subroutine used them meanwhile.

Similarly during the CALL operation, the current value of the instruction pointer is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer. Execution therefore resumes from the destination of CALL. When the RET instruction is executed, it recovers the value of the instruction pointer from the stack. The next instruction executed is therefore the one following the CALL. Observe how playing with the instruction pointer affects the program flow.

There is a form of the RET instruction called "RET n" where n is a numeric argument. After performing the operation of RET, it further increments the stack pointer by this number, i.e. SP is first incremented by two and then by n. Its function will become clear when parameter passing is discussed.

Now we describe the operation of the stack in CALL and RET with an example. The top of stack stored in the stack pointer is initialized at 2000. The space above SP is considered empty and free. When the stack pointer is decremented by two, we took a word from the empty space and can use it for our purpose. The unit of stack operations is a word. Some instructions push multiple words; however byte pushes cannot be made. Now the value 017B is stored in the word reserved on the stack. The RET will copy this value in the instruction pointer and increment the stack pointer by two making it 2000 again, thereby reverting the operation of CALL.

This is how CALL and RET behave for near calls. There is also a far version of these functions when the target routine is in another segment. This version of CALL takes a segment offset pair just like the far jump instruction. The CALL will push both the segment and the offset on the stack in this case, followed by loading CS and IP with the values given in the instruction.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

The corresponding instruction RETF will pop the offset in the instruction pointer followed by popping the segment in the code segment register.

Apart from CALL and RET, the operations that use the stack are PUSH and POP. Two other operations that will be discussed later are INT and IRET. Regarding the stack, the operation of PUSH is similar to CALL however with a register other than the instruction pointer. For example “push ax” will push the current value of the AX register on the stack. The operation of PUSH is shown below.

```
SP ← SP - 2
[SP] ← AX
```

The operation of POP is the reverse of this. A copy of the element at the top of the stack is made in the operand, and the top of the stack is incremented afterwards. The operation of “pop ax” is shown below.

```
AX ← [SP]
SP ← SP + 2
```

Making corresponding PUSH and POP operations is the responsibility of the programmer. If “push ax” is followed by “pop dx” effectively copying the value of the AX register in the DX register, the processor won’t complain. Whether this sequence is logically correct or not should be ensured by the programmer. For example when PUSH and POP are used to save and restore registers from the stack, order must be correct so that the saved value of AX is reloaded in the AX register and not any other register. For this the order of POP operations need to be the reverse of the order of PUSH operations.

Now we consider another example that is similar to the previous examples, however the code to swap the two elements has been extracted into another subroutine, so that the formation of stack can be observed during nested subroutine calls.

Example 5.3	
01	; bubble sort subroutine using swap subroutine
02	[org 0x0100]
03	jmp start
04	
05	data: dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
06	data2: dw 328, 329, 898, 8923, 8293, 2345, 10, 877, 355, 98
07	dw 888, 533, 2000, 1020, 30, 200, 761, 167, 90, 5
08	swapflag: db 0
09	
10	swap: mov ax, [bx+si] ; load first number in ax
11	xchg ax, [bx+si+2] ; exchange with second number
12	mov [bx+si], ax ; store second number in first
13	ret ; go back to where we came from
14	
15	bubblesort: dec cx ; last element not compared
16	shl cx, 1 ; turn into byte count
17	
18	mainloop: mov si, 0 ; initialize array index to zero
19	mov byte [swapflag], 0 ; reset swap flag to no swaps
20	
21	innerloop: mov ax, [bx+si] ; load number in ax
22	cmp ax, [bx+si+2] ; compare with next number
23	jbe noswap ; no swap if already in order
24	
25	call swap ; swaps two elements
26	mov byte [swapflag], 1 ; flag that a swap has been done
27	
28	noswap: add si, 2 ; advance si to next index
29	cmp si, cx ; are we at last index
30	jne innerloop ; if not compare next two
31	
32	cmp byte [swapflag], 1 ; check if a swap has been done
33	je mainloop ; if yes make another pass
34	ret ; go back to where we came from
35	
36	start: mov bx, data ; send start of array in bx
37	mov cx, 10 ; send count of elements in cx

38	call bubblesort	; call our subroutine
39		
40	mov bx, data2	; send start of array in bx
41	mov cx, 20	; send count of elements in cx
42	call bubblesort	; call our subroutine again
43		
44	mov ax, 0x4c00	; terminate program
45	int 0x21	
11	A new instruction XCHG has been introduced. The instruction swaps its source and its destination operands however at most one of the operands could be in memory, so the other has to be loaded in a register. The instruction has reduced the code size by one instruction.	
13	The RET at the end of swap makes it a subroutine.	

Inside the debugger observe the use of stack by CALL and RET instructions, especially the nested CALL.

5.4. SAVING AND RESTORING REGISTERS

The subroutines we wrote till now have been destroying certain registers and our calling code has been carefully written to not use those registers. However this cannot be remembered for a good number of subroutines. Therefore our subroutines need to implement some mechanism of retaining the callers' value of any registers used.

The trick is to use the PUSH and POP operations and save the callers' value on the stack and recover it from there on return. Our swap subroutine destroyed the AX register while the bubblesort subroutine destroyed AX, CX, and SI. BX was not modified in the subroutine. It had the same value at entry and at exit; it was only used by the subroutine. Our next example improves on the previous version by saving and restoring any registers that it will modify using the PUSH and POP operations.

Example 5.4		
01	; bubble sort and swap subroutines saving and restoring registers	
02	[org 0x0100]	
03	jmp start	
04		
05	data:	dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
06	data2:	dw 328, 329, 898, 8923, 8293, 2345, 10, 877, 355, 98
07		dw 888, 533, 2000, 1020, 30, 200, 761, 167, 90, 5
08	swapflag:	db 0
09		
10	swap:	push ax ; save old value of ax
11		
12	mov ax, [bx+si]	; load first number in ax
13	xchg ax, [bx+si+2]	; exchange with second number
14	mov [bx+si], ax	; store second number in first
15		
16	pop ax	; restore old value of ax
17	ret	; go back to where we came from
18		
19	bubblesort:	push ax ; save old value of ax
20	push cx	; save old value of cx
21	push si	; save old value of si
22		
23	dec cx	; last element not compared
24	shl cx, 1	; turn into byte count
25		
26	mainloop:	mov si, 0 ; initialize array index to zero
27	mov byte [swapflag], 0	; reset swap flag to no swaps
28		
29	innerloop:	mov ax, [bx+si] ; load number in ax
30	cmp ax, [bx+si+2]	; compare with next number
31	jbe noswap	; no swap if already in order
32		

33		call swap	; swaps two elements
34		mov byte [swapflag], 1	; flag that a swap has been done
35			
36	noswap:	add si, 2	; advance si to next index
37		cmp si, cx	; are we at last index
38		jne innerloop	; if not compare next two
39			
40		cmp byte [swapflag], 1	; check if a swap has been done
41		je mainloop	; if yes make another pass
42			
43		pop si	; restore old value of si
44		pop cx	; restore old value of cx
45		pop ax	; restore old value of ax
46		ret	; go back to where we came from
47			
48	start:	mov bx, data	; send start of array in bx
49		mov cx, 10	; send count of elements in cx
50		call bubblesort	; call our subroutine
51			
52		mov bx, data2	; send start of array in bx
53		mov cx, 20	; send count of elements in cx
54		call bubblesort	; call our subroutine again
55			
56		mov ax, 0x4c00	; terminate program
57		int 0x21	
19-21	When multiple registers are pushed, order is very important. If AX, CX, and SI are pushed in this order, they must be popped in the reverse order of SI, CX, and AX. This is again because the stack behaves in a Last In First Out manner.		

Inside the debugger we can observe that the registers before and after the CALL operation are exactly identical. Effectively the caller can assume the registers are untouched. By tracing into the subroutines we can observe how their value is saved on the stack by the PUSH instructions and recovered from their before exit. Saving and restoring registers this way in subroutines is a standard way and must be followed.

PUSH

PUSH decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP. PUSH often is used to place parameters on the stack before calling a procedure; more generally, it is the basic means of storing temporary data on the stack.

POP

POP transfers the word at the current top of stack (pointed to by SP) to the destination operand and then increments SP by two to point to the new top of stack. POP can be used to move temporary variables from the stack to registers or memory.

Observe that the operand of PUSH is called a source operand since the data is moving to the stack from the operand, while the operand of POP is called destination since data is moving from the stack to the operand.

CALL

CALL activates an out-of-line procedure, saving information on the stack to permit a RET (return) instruction in the procedure to transfer control back to the instruction following the CALL. For an intra segment direct CALL, SP is decremented by two and IP is pushed onto the stack. The target procedure's relative displacement from the CALL instruction is then added to the instruction pointer. For an inter segment direct CALL, SP is decremented by two, and CS is pushed onto the stack. CS is replaced by the segment word contained in the instruction. SP again is decremented by two. IP is pushed onto the stack and replaced by the offset word in the instruction.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

The out-of-line procedure is the temporary division, the concept of roundabout that we discussed. Near calls are also called intra segment calls, while far calls are called inter-segment calls. There are also versions that are called indirect calls; however they will be discuss later when they are used.

RET

RET (Return) transfers control from a procedure back to the instruction following the CALL that activated the procedure. RET pops the word at the top of the stack (pointed to by register SP) into the instruction pointer and increments SP by two. If RETF (inter segment RET) is used the word at the top of the stack is popped into the IP register and SP is incremented by two. The word at the new top of stack is popped into the CS register, and SP is again incremented by two. If an optional pop value has been specified, RET adds that value to SP. This feature may be used to discard parameters pushed onto the stack before the execution of the CALL instruction.

5.5. PARAMETER PASSING THROUGH STACK

Due to the limited number of registers, parameter passing by registers is constrained in two ways. The maximum parameters a subroutine can receive are seven when all the general registers are used. Also, with the subroutines are themselves limited in their use of registers, and this limited increases when the subroutine has to make a nested call thereby using certain registers as its parameters. Due to this, parameter passing by registers is not expandable and generalizable. However this is the fastest mechanism available for passing parameters and is used where speed is important.

Considering stack as an alternate, we observe that whatever data is placed there, it stays there, and across function calls as well. For example the bubble sort subroutine needs an array address and the count of elements. If we place both of these on the stack, and call the subroutine afterwards, it will stay there. The subroutine is invoked with its return address on top of the stack and its parameters beneath it.

To access the arguments from the stack, the immediate idea that strikes is to pop them off the stack. And this is the only possibility using the given set of information. However the first thing popped off the stack would be the return address and not the arguments. This is because the arguments were first pushed on the stack and the subroutine was called afterwards. The arguments cannot be popped without first popping the return address. If a heaving thing falls on someone's leg, the heavy thing is removed first and the leg is not pulled out to reduce the damage. Same is the case with our parameters on which the return address has fallen.

To handle this using PUSH and POP, we must first pop the return address in a register, then pop the operands, and push the return address back on the stack so that RET will function normally. However so much effort doesn't seem to pay back the price. Processor designers should have provided a logical and neat way to perform this operation. They did provided a way and infact we will do this without introducing any new instruction.

Recall that the default segment association of the BP register is the stack segment and the reason for this association had been deferred for now. The reason is to peek inside the stack using the BP register and read the parameters without removing them and without touching the stack pointer. The stack pointer could not be used for this purpose, as it cannot be used in an effective address. It is automatically used as a pointer and cannot be explicitly used. Also the stack pointer is a dynamic pointer and sometimes changes without telling us in the background. It is just that whenever we touch it, it is where we expect it to be. The base pointer is provided as a replacement of the stack pointer so that we can peek inside the stack without modifying the structure of the stack.

When the bubble sort subroutine is called, the stack pointer is pointing to the return address. Two bytes below it is the second parameter and four bytes below is the first parameter. The stack pointer is a reference point to these parameters. If the value of SP is captured in BP, then the return address is located at [bp+0], the second parameter is at [bp+2], and the first parameter is at [bp+4]. This is because SP and BP both had the same value and they both defaulted to the same segment, the stack segment.

This copying of SP into BP is like taking a snapshot or like freezing the stack at that moment. Even if more pushes are made on the stack decrementing the stack pointer, our reference point will not change. The parameters will still be accessible at the same offsets from the base pointer. If however the stack pointer increments beyond the base pointer, the references will become invalid. The base pointer will act as the datum point to access our parameters. However we have destroyed the original value of BP in the process, and this will cause problems in nested calls where both the outer and the inner subroutines need to access their own parameters. The outer subroutine will have its base pointer destroyed after the call and will be unable to access its parameters.

To solve both of these problems, we reach at the standard way of accessing parameters on the stack. The first two instructions of any subroutines accessing its parameters from the stack are given below.

```
push bp
mov bp, sp
```

As a result our datum point has shifted by a word. Now the old value of BP will be contained in [bp] and the return address will be at [bp+2]. The second parameters will be [bp+4] while the first one will be at [bp+6]. We give an example of bubble sort subroutine using this standard way of argument passing through stack.

Example 5.5

```
01      ; bubble sort subroutine taking parameters from stack
02      [org 0x0100]
03      jmp start
04
05      data:      dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
06      data2:     dw 328, 329, 898, 8923, 8293, 2345, 10, 877, 355, 98
07      dw 888, 533, 2000, 1020, 30, 200, 761, 167, 90, 5
08      swapflag:  db 0
09
10      bubblesort: push bp          ; save old value of bp
11                  mov bp, sp      ; make bp our reference point
12                  push ax         ; save old value of ax
13                  push bx         ; save old value of bx
14                  push cx         ; save old value of cx
15                  push si         ; save old value of si
16
17                  mov bx, [bp+6]   ; load start of array in bx
18                  mov cx, [bp+4]   ; load count of elements in cx
19                  dec cx          ; last element not compared
20                  shl cx, 1        ; turn into byte count
21
22      mainloop:   mov si, 0        ; initialize array index to zero
23                  mov byte [swapflag], 0 ; reset swap flag to no swaps
24
25      innerloop:  mov ax, [bx+si]   ; load number in ax
26                  cmp ax, [bx+si+2] ; compare with next number
27                  jbe noswap       ; no swap if already in order
28
29                  xchg ax, [bx+si+2] ; exchange ax with second number
30                  mov [bx+si], ax   ; store second number in first
31                  mov byte [swapflag], 1 ; flag that a swap has been done
32
33      noswap:     add si, 2         ; advance si to next index
34                  cmp si, cx       ; are we at last index
35                  jne innerloop     ; if not compare next two
36
```

37		cmp	byte [swapflag], 1	; check if a swap has been done
38		je	mainloop	; if yes make another pass
39				
40		pop	si	; restore old value of si
41		pop	cx	; restore old value of cx
42		pop	bx	; restore old value of bx
43		pop	ax	; restore old value of ax
44		pop	bp	; restore old value of bp
45		ret	4	; go back and remove two params
46				
47	start:	mov	ax, data	
48		push	ax	; place start of array on stack
49		mov	ax, 10	
50		push	ax	; place element count on stack
51		call	bubblesort	; call our subroutine
52				
53		mov	ax, data2	
54		push	ax	; place start of array on stack
55		mov	ax, 20	
56		push	ax	; place element count on stack
57		call	bubblesort	; call our subroutine again
58				
59		mov	ax, 0x4c00	; terminate program
60		int	0x21	

11	The value of the stack pointer is captured in the base pointer. With further pushes SP will change but BP will not and therefore we will read parameters from bp+4 and bp+6.
45	The form of RET that takes an argument is used causing four to be added to SP after the return address has been popped in the instruction pointer. This will effectively discard the parameters that are still there on the stack.
47-50	We push the address of the array we want to sort followed by the count of elements. As immediate cannot be directly pushed in the 8088 architecture, we first load it in the AX register and then push the AX register on the stack.

Inside the debugger, concentrate on the operation of BP and the stack. The parameters are placed on the stack by the caller, the subroutine accesses them using the base pointer, and the special form of RET removes them without any extra instruction. The value of stack pointer of FFF6 is turned into FFFE by the RET instruction. This was the value in SP before any of the parameters was pushed.

Stack Clearing by Caller or Callee

Parameters pushed for a subroutine are a waste after the subroutine has returned. They have to be cleared from the stack. Either of the caller and the callee can take the responsibility of clearing them from there. If the callee has to clear the stack it cannot do this easily unless RET n exists. That is why most general processors have this instruction. Stack clearing by the caller needs an extra instruction on behalf of the caller after every call made to the subroutine, unnecessarily increasing instructions in the program. If there are thousand calls to a subroutine the code to clear the stack is repeated a thousand times. Therefore the prevalent convention in most high level languages is stack clearing by the callee; even though the other convention is still used in some languages.

If RET n is not available, stack clearing by the callee is a complicated process. It will have to save the return address in a register, then remove the parameters, and then place back the return address so that RET will function. When this instruction was introduced in processors, only then high level language designers switched to stack clearing by the callee. This is also exactly why RET n adds n to SP after performing the operation of RET. The other way around would be totally useless for our purpose. Consider the stack condition at the time of RET and this will become clear why this will be

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

useless. Also observe that RET n has discarded the arguments rather than popping them as they were no longer of any use either of the caller or the callee.

The strong argument in favour of callee cleared stacks is that the arguments were placed on the stack for the subroutine, the caller did not needed them for itself, so the subroutine is responsible for removing them. Removing the arguments is important as if the stack is not cleared or is partially cleared the stack will eventually become full, SP will reach 0, and thereafter wraparound producing unexpected results. This is called stack overflow. Therefore clearing anything placed on the stack is very important.

5.6. LOCAL VARIABLES

Another important role of the stack is in the creation of local variables that are only needed while the subroutine is in execution and not afterwards. They should not take permanent space like global variables. Local variables should be created when the subroutine is called and discarded afterwards. So that the space used by them can be reused for the local variables of another subroutine. They only have meaning inside the subroutine and no meaning outside it.

The most convenient place to store these variables is the stack. We need some special manipulation of the stack for this task. We need to produce a gap in the stack for our variables. This is explained with the help of the swapflag in the bubble sort example.

The swapflag we have declared as a word occupying space permanently is only needed by the bubble sort subroutine and should be a local variable. Actually the variable was introduced with the intent of making it a local variable at this time. The stack pointer will be decremented by an extra two bytes thereby producing a gap in which a word can reside. This gap will be used for our temporary, local, or automatic variable; however we name it. We can decrement it as much as we want producing the desired space, however the decrement must be by an even number, as the unit of stack operation is a word. In our case we needed just one word. Also the most convenient position for this gap is immediately after saving the value of SP in BP. So that the same base pointer can be used to access the local variables as well; this time using negative offsets. The standard way to start a subroutine which needs to access parameters and has local variables is as under.

```
push bp
mov  bp, sp
sub  sp, 2
```

The gap could have been created with a dummy push, but the subtraction makes it clear that the value pushed is not important and the gap will be used for our local variable. Also gap of any size can be created in a single instruction with subtraction. The parameters can still be accessed at bp+4 and bp+6 and the swapflag can be accessed at bp-2. The subtraction in SP was after taking the snapshot; therefore BP is above the parameters but below the local variables. The parameters are therefore accessed using positive offsets from BP and the local variables are accessed using negative offsets.

We modify the bubble sort subroutine to use a local variable to store the swap flag. The swap flag remembered whether a swap has been done in a particular iteration of bubble sort.

Example 5.6	
01	; bubble sort subroutine using a local variable
02	[org 0x0100]
03	jmp start
04	
05	data: dw 60, 55, 45, 50, 40, 35, 25, 30, 10, 0
06	data2: dw 328, 329, 898, 8923, 8293, 2345, 10, 877, 355, 98

07		dw	888, 533, 2000, 1020, 30, 200, 761, 167, 90, 5	
08				
09	bubblesort:	push bp		; save old value of bp
10		mov bp, sp		; make bp our reference point
11		sub sp, 2		; make two byte space on stack
12		push ax		; save old value of ax
13		push bx		; save old value of bx
14		push cx		; save old value of cx
15		push si		; save old value of si
16				
17		mov bx, [bp+6]		; load start of array in bx
18		mov cx, [bp+4]		; load count of elements in cx
19		dec cx		; last element not compared
20		shl cx, 1		; turn into byte count
21				
22	mainloop:	mov si, 0		; initialize array index to zero
23		mov word [bp-2], 0		; reset swap flag to no swaps
24				
25	innerloop:	mov ax, [bx+si]		; load number in ax
26		cmp ax, [bx+si+2]		; compare with next number
27		jbe noswap		; no swap if already in order
28				
29		xchg ax, [bx+si+2]		; exchange ax with second number
30		mov [bx+si], ax		; store second number in first
31		mov word [bp-2], 1		; flag that a swap has been done
32				
33	noswap:	add si, 2		; advance si to next index
34		cmp si, cx		; are we at last index
35		jne innerloop		; if not compare next two
36				
37		cmp word [bp-2], 1		; check if a swap has been done
38		je mainloop		; if yes make another pass
39				
40		pop si		; restore old value of si
41		pop cx		; restore old value of cx
42		pop bx		; restore old value of bx
43		pop ax		; restore old value of ax
44		mov sp, bp		; remove space created on stack
45		pop bp		; restore old value of bp
46		ret 4		; go back and remove two params
47				
48	start:	mov ax, data		
49		push ax		; place start of array on stack
50		mov ax, 10		
51		push ax		; place element count on stack
52		call bubblesort		; call our subroutine
53				
54		mov ax, data2		
55		push ax		; place start of array on stack
56		mov ax, 20		
57		push ax		; place element count on stack
58		call bubblesort		; call our subroutine again
59				
60		mov ax, 0x4c00		; terminate program
61		int 0x21		

- 11 A word gap has been created for swap flag. This is equivalent to a dummy push. The registers are pushed above this gap.
- 23 The swapflag is accessed with [bp-2]. The parameters are accessed in the same manner as the last examples.
- 44 We are removing the hole that we created. The hole is removed by restoring the value of SP that it had at the time of snapshot or at the value it had before the local variable was created. This can be replaced with “add sp, 2” however the one used in the code is preferred since it does not require to remember how much space for local variables was allocated in the start. After this operation SP points to the old value of BP from where we can proceed as usual.

We needed memory to store the swap flag. The fact that it is in the stack segment or the data segment doesn't bother us. This will just change the addressing scheme.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

EXERCISES

- Replace the following valid instruction with a single instruction that has the same effect. Don't consider the effect on flags.


```

push word L1
jmp L2
L1:

```
- Replace the following invalid instructions with a single instruction that has the same effect.
 - pop ip
 - mov ip, L5
 - sub sp, 2
mov [ss:sp], ax
 - mov ax, [ss:sp]
add sp, 2
 - add sp, 6
mov ip, [ss:sp-6]
- Write a recursive function to calculate the Fibonacci of a number. The number is passed as a parameter via the stack and the calculated Fibonacci number is returned in the AX register. A local variable should be used to store the return value from the first recursive call. Fibonacci function is defined as follows:


```

Fibonacci(0) = 0
Fibonacci(1) = 1
Fibonacci(n) = Fibonacci(n-1) + Fibonacci(n-2)

```
- Write the above Fibonacci function iteratively.
HINT: Use two registers to hold the current and the previous Fibonacci numbers in a loop.
- Write a function switch_stack meant to change the current stack and will be called as below. The function should destroy no registers.


```

push word [new_stack_segment]
push word [new_stack_offset]
call switch_stack

```
- Write a function "addtiset" that takes offset of a function and remembers this offset in an array that can hold a maximum of 8 offsets. It does nothing if there are already eight offsets in the set. Write another function "callset" that makes a call to all functions in the set one by one.
- Do the above exercise such that "callset" does not use a CALL or a JMP to invoke the functions.
HINT: Setup the stack appropriately such that the RET will execute the first function, its RET execute the next and so on till the last RET returns to the caller of "callset."
- Make an array of 0x80 bytes and treat it as one of 0x400 bits. Write a function myalloc that takes one argument, the number of bits. It finds that many consecutive zero bits in the array, makes them one, and returns in AX the index of the first bit. Write another function myfree that takes two arguments, index of a bit in the array, and the number of bits. It makes that many consecutive bits zero, whatever their previous values are, starting from the index in the first argument.
- [Circular Queue] Write functions to implement circular queues. Declare 16x32 words of data for 16 queues numbered from 0 to 15. Each queue has a front index, a rear index and 30 locations for data totaling to 32 words. Declare another word variable whose 16 bits correspond to the 16 queues and a 1 bit signals that the corresponding queue is used and a 0 bit signals that it is free. Write a function "qcreate" that returns a queue number after finding a free queue or -1 if it failed. Write a function "qdestroy" that marks the queue as free. Write two other functions "qadd" and "qremove" that

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

can add and remove items from the circular queue. The two functions return 0 if they failed and 1 otherwise.

10. [Linked List] Declare 1024 nodes of four bytes each. The first 2 bytes will be used for data and the next 2 bytes for storing the offset of another node. Also declare a word variable "firstfree" to store the offset of the first free node. Write the following five functions:
 - a. "init" chains all 1024 nodes into a list with offset of first node in firstfree, offset of the second node in the later two bytes of the first node and so on. The later two bytes of the last node contains zero.
 - b. "createlist" returns the offset of the node stored in firstfree through AX. It sets firstfree to the offset stored in the later two bytes of that node, and it sets the later two bytes of that node to zero.
 - c. "insertafter" takes two parameters, the offset of a node and a word data. It removes one node from freelist just like "createlist" and inserts it after the said node and updates the new node's data part.
 - d. "deleteafter" takes a node as its parameter and removes the node immediately after it in the linked list if there is one.
 - e. "deletelist" takes a node as its parameters and traverses the linked list starting at this node and removes all nodes from it and add them back to the free list.

Display Memory

The debugger gives a very close vision of the processor. That is why every program written till now was executed inside the debugger. Also the debugger is a very useful tool in assembly language program development, since many bugs only become visible when each instruction is independently monitored the way the debugger allows us to do. We will now be using the display screen in character mode, the way DOS uses this screen. The way we will access this screen is specific to the IBM PC.

6.1. ASCII CODES

The computer listens, sees, and speaks in numbers. Even a character is a number inside the computer. For example the keyboard is labeled with characters however when we press 'A', a specific number is transferred from the keyboard to the computer. Our program interprets that number as the character 'A'. When the same number comes on display, the Video Graphics Adapter (VGA) in our computer shows the shape of 'A'. Even the shape is stored in binary numbers with a one bit representing a pixel on the screen that is turned on and a zero bit representing a pixel that is not glowing. This example is considering a white on black display and no colors. This is the way a shape is drawn on the screen. The interpretation of 'A' is performed by the VGA card, while the monitor or CRT (cathode ray tube) only glows the pixels on and turns them off. The keyboard has a key labeled 'A' and pressing it the screen shows 'A' but all that happened inside was in numbers.

An 'A' on any computer and any operating system is an 'A' on every other computer and operating system. This is because a standard numeric representation of all commonly used characters has been developed. This is called the ASCII code, where ASCII stands for American Standard Code for Information Interchange. The name depicts that this is a code that allows the interchange of information; 'A' written on one computer will remain an 'A' on another. The ASCII table lists all defined characters and symbols and their standardized numbers. All ASCII based computers use the same code. There are few other standards like EBCDIC and gray codes, but ASCII has become the most prevalent standard and is used for Internet communication as well. It has become the de facto standard for global communication. The character mode displays of our computer use the ASCII standard. Some newer operating systems use a new standard Unicode but it is not relevant to us in the current discussion.

Standard ASCII has 128 characters with assigned numbers from 0 to 127. When IBM PC was introduced, they extended the standard ASCII and defined 128 more characters. Thus extending the total number of symbols from 128 to 256 numbered from 0 to 255 fitting in an 8-bit byte. The newer characters were used for line drawing, window corners, and some non-English characters. The need for these characters was never felt on teletype terminals, but with the advent of IBM PC and its full screen display, these semi-graphics characters were the need of the day. Keep in mind that at that time there was no graphics mode available.

The extended ASCII code is just a de facto industry standard but it is not defined by an organization like the standard ASCII. Printers, displays, and all other peripherals related to the IBM PC understand the ASCII code. If the

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

code for 'A' is sent to the printer, the printer will print the shape of 'A', if it is sent to the display, the VGA card will form the shape of 'A' on the CRT. If it is sent to another computer via the serial port, the other computer will understand that this is an 'A'.

The important thing to observe in the ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers (30-39). This helps in certain operations with ASCII, for example converting the case of characters by adding or subtracting 0x20 from it. It also helps in converting a digit into its ASCII representation by adding 0x30 to it.

6.2. DISPLAY MEMORY FORMATION

We will explore the working of the display with ASCII codes, since it is our immediately accessible hardware. When 0x40 is sent to the VGA card, it will turn pixels on and off in such a way that a visual representation of 'A' appears on the screen. It has no reality, just an interpretation. In later chapters we will program the VGA controller to display a new shape when the ASCII of 'A' is received by it.

The video device is seen by the computer as a memory area containing the ASCII codes that are currently displayed on the screen and a set of I/O ports controlling things like the resolution, the cursor height, and the cursor position. The VGA memory is seen by the computer just like its own memory. There is no difference; rather the computer doesn't differentiate, as it is accessible on the same bus as the system memory. Therefore if that appropriate block of the screen is cleared, the screen will be cleared. If the ASCII of 'A' is placed somewhere in that block, the shape of 'A' will appear on the screen at a corresponding place.

This correspondence must be defined as the memory is a single dimensional space while the screen is two dimensional having 80 rows and 25 columns. The memory is linearly mapped on this two dimensional space, just like a two dimensional is mapped in linear memory. There is one word per character in which a byte is needed for the ASCII code and the other byte is used for the character's attributes discussed later. Now the first 80 words will correspond to the first row of the screen and the next 80 words will correspond to the next row. By making the memory on the video controller accessible to the processor via the system bus, the processor is now in control of what is displayed on the screen.

The three important things that we discussed are.

- One screen location corresponds to a word in the video memory
- The video controller memory is accessible to the processor like its own memory.
- ASCII code of a character placed at a cell in the VGA memory will cause the corresponding ASCII shape to be displayed on the corresponding screen location.

Display Memory Base Address

The memory at which the video controller's memory is mapped must be a standard, so that the program can be written in a video card independent manner. Otherwise if different vendors map their video memory at different places in the address space, as was the problem in the start, writing software was a headache. BIOS vendors had a problem of dealing with various card vendors. The IBM PC text mode color display is now fixed so that system software can work uniformly. It was fixed at the physical memory location of B8000. The first byte at this location contains the ASCII for the character displayed at the top left of the video screen. Dropping the zero we can load the rest in a segment register to access the video memory. If we do something in this memory, the effect can be seen on the screen. For example we can

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

write a virus that makes any character we write drop to the bottom of the screen.

Attribute Byte

The second byte in the word designated for one screen location holds the foreground and background colors for the character. This is called its video attribute. So the pair of the ASCII code in one byte and the attribute in the second byte makes the word that corresponds to one location on the screen. The lower address contains the code while the higher one contains the attribute. The attribute byte as detailed below has the RGB for the foreground and the background. It has an intensity bit for the foreground color as well thus making 16 possible colors of the foreground and 8 possible colors for the background. When bit 7 is set the character keeps on blinking on the screen. This bit has some more interpretations like background intensity that has to be activated in the video controller through its I/O ports.

7	6	5	4	3	2	1	0

- 7 – Blinking of foreground character
- 6 – Red component of background color
- 5 – Green component of background color
- 4 – Blue component of background color
- 3 – Intensity component of foreground color
- 2 – Red component of foreground color
- 1 – Green component of foreground color
- 0 – Blue component of foreground color

Display Examples

Both DS and ES can be used to access the video memory. However we commonly keep DS for accessing our data, and load ES with the segment of video memory. Loading a segment register with an immediate operand is not allowed in the 8088 architecture. We therefore load the segment register via a general purpose register. Other methods are loading from a memory location and a combination of push and pop.

```
mov ax, 0xb800
mov es, ax
```

This operation has opened a window to the video memory. Now the following instruction will print an 'A' on the top left of the screen in white color on black background.

```
mov word [es:0], 0x0741
```

The segment override is used since ES is pointing to the video memory. Since the first word is written to, the character will appear at the top left of the screen. The 41 that goes in the lower byte is the ASCII code for 'A'. The 07 that goes in the higher byte is the attribute with I=0, R=1, G=1, B=1 for the foreground, meaning white color in low intensity and R=0, G=0, B=0 for the background meaning black color and the most significant bit cleared so that there is no blinking. Now consider the following instruction.

```
mov word [es:160], 0x1230
```

This is displayed 80 words after the start and there are 80 characters in one screen row. Therefore this is displayed on the first column of the second line. The ASCII code used is 30, which represents a '0' while the attribute byte is 12 meaning green color on black background.

We take our first example to clear the screen.

Example 6.1

01		; clear the screen	
02		[org 0x0100]	
03		mov ax, 0xb800	; load video base in ax
04		mov es, ax	; point es to video base
05		mov di, 0	; point di to top left column
06			
07	nextchar:	mov word [es:di], 0x0720	; clear next char on screen
08		add di, 2	; move to next screen location
09		cmp di, 4000	; has the whole screen cleared
10		jne nextchar	; if no clear next position
11			
12		mov ax, 0x4c00	; terminate program
13		int 0x21	

07		The code for space is 20 while 07 is the normal attribute of low intensity white on black with no blinking. Even to clear the screen or put a blank on a location there is a numeric code.
08		DI is incremented twice since each screen location corresponds to two byte in video memory.
09		DI is compared with $80 \times 25 \times 2 = 4000$. The last word location that corresponds to the screen is 3998.

Inside the debugger the operation of clearing the screen cannot be observed since the debugger overwrites whatever is displayed on the screen. Directly executing the COM file from the command prompt*, we can see that the screen is cleared. The command prompt that reappeared is printed after the termination of our application. This is the first application that can be directly executed to see some output on the screen.

6.3. HELLO WORLD IN ASSEMBLY LANGUAGE

To declare a character in assembly language, we store its ASCII code in a byte. The assembler provides us with another syntax that doesn't force us to remember the ASCII code. The assembler also provides a syntax that simplifies declaration of consecutive characters, usually called a string. The three ways used below are identical in their meaning.

```

db 0x61, 0x62, 0x63
db 'a', 'b', 'c'
db 'abc'

```

When characters are stored in any high level or low level language the actual thing stored in a byte is their ASCII code. The only thing the language helps in is a simplified declaration.

Traditionally the first program in higher level languages is to print "hello world" on the screen. However due to the highly granular nature of assembly language, we are only now able to write it in assembly language. In writing this program, we make a generic routine that can print any string on the screen.

Example 6.2			
01	; hello world in assembly		
02	[org 0x0100]		
03		jmp	start
04			
05	message:	db	'hello world' ; string to be printed
06	length:	dw	11 ; length of the string
07			
08	; subroutine to clear the screen		
09	clrscr:	push	es
10		push	ax
11		push	di

* Remember that if this example is run in a DOS window on some newer operating systems, a full screen DOS application must be run before this program so that screen access is enabled.

12		
13		mov ax, 0xb800
14		mov es, ax ; point es to video base
15		mov di, 0 ; point di to top left column
16		
17	nextloc:	mov word [es:di], 0x0720 ; clear next char on screen
18		add di, 2 ; move to next screen location
19		cmp di, 4000 ; has the whole screen cleared
20		jne nextloc ; if no clear next position
21		
22		pop di
23		pop ax
24		pop es
25		ret
26		
27		; subroutine to print a string at top left of screen
28		; takes address of string and its length as parameters
29	printstr:	push bp
30		mov bp, sp
31		push es
32		push ax
33		push cx
34		push si
35		push di
36		
37		mov ax, 0xb800
38		mov es, ax ; point es to video base
39		mov di, 0 ; point di to top left column
40		mov si, [bp+6] ; point si to string
41		mov cx, [bp+4] ; load length of string in cx
42		mov ah, 0x07 ; normal attribute fixed in al
43		
44	nextchar:	mov al, [si] ; load next char of string
45		mov [es:di], ax ; show this char on screen
46		add di, 2 ; move to next screen location
47		add si, 1 ; move to next char in string
48		loop nextchar ; repeat the operation cx times
49		
50		pop di
51		pop si
52		pop cx
53		pop ax
54		pop es
55		pop bp
56		ret 4
57		
58	start:	call clrscr ; call the clrscr subroutine
59		
60		mov ax, message
61		push ax ; push address of message
62		push word [length] ; push message length
63		call printstr ; call the printstr subroutine
64		
65		mov ax, 0x4c00 ; terminate program
66		int 0x21

05-06	The string definition syntax discussed above is used to declare a string "hello world" of 11 bytes and the length is stored in a separate variable.
09-25	The code to clear the screen from the last example is written in the form of a subroutine. Since the subroutine had no parameters, only modified registers are saved and restored from the stack.
29-35	The standard subroutine format with parameters received via stack and all registers saved and restored is used.
37-42	ES is initialized to point to the video memory via the AX register. Two pointer registers are used; SI to point to the string and DI to point to the top left location of the screen. CX is loaded with the length of the string. Normal attribute of low intensity white on black with no blinking is loaded in the AH register.
44-45	The next character from the string is loaded into AL. Now AH holds the attribute and AL the ASCII code of the character. This pair is

46-47	written on the video memory using DI with the segment override prefix for ES to access the video memory segment.
48	The string pointer is incremented by one while the video memory pointer is incremented by two since one char corresponds to a word on the screen.
50-56	The loop instruction used is equivalent to a combination of “dec cx” and “jnz nextchar.” The loop is executed CX times.
	The registers pushed on the stack are recovered in opposite order and the “ret 4” instruction removes the two parameters placed on the stack.
62	Memory can be directly pushed on the stack.

When the program is executed, screen is cleared and the greetings is displayed on the top left of the screen. This screen location and the attribute used were hard coded in the program and can also be made variable. Then we will be able to print anywhere on the screen.

6.4. NUMBER PRINTING IN ASSEMBLY

Another problem related to the display is printing numbers. Every high level language allows some simple way to print numbers on the screen. As we have seen, everything on the screen is a pair of ASCII code and its attribute and a number is a raw binary number and not a collection of ASCII codes. For example a 10 is stored as a 10 and not as the ASCII code of 1 followed by the ASCII code of 0. If this 10 is stored in a screen location, the output will be meaningless, as the character associate to ASCII code 10 will be shown on the screen. So there is a process that converts a number in its ASCII representation. This process works for any number in any base. We will discuss our examples with respect to the decimal base and later observe the effect of changing to different bases.

Number Printing Algorithm

The key idea is to divide the number by the base number, 10 in the case of decimal. The remainder can be from 0-9 and is the right most digit of the original number. The remaining digits fall in the quotient. The remainder can be easily converted into its ASCII equivalent and printed on the screen. The other digits can be printed in a similar manner by dividing the quotient again by 10 to separate the next digit and so on.

However the problem with this approach is that the first digit printed is the right most one. For example 253 will be printed as 352. The remainder after first division was 3, after second division was 5 and after the third division was 2. We have to somehow correct the order so that the actual number 253 is displayed, and the trick is to use the stack since the stack is a Last In First Out structure so if 3, 5, and 2 are pushed on it, 2, 5, and 3 will come out in this order. The steps of our algorithm are outlined below.

- Divide the number by base (10 in case of decimal)
- The remainder is its right most digit
- Convert the digit to its ASCII representation (Add 0x30 to the remainder in case of decimal)
- Save this digit on stack
- If the quotient is non-zero repeat the whole process to get the next digit, otherwise stop
- Pop digits one by one and print on screen left to right

DIV Instruction

The division used in the process is integer division and not floating point division. Integer division gives an integer quotient and an integer remainder. A division algorithm is now needed. Fortunately or unfortunately there is a

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

DIV instruction available in the 8088 processor. There are two forms of the DIV instruction. The first form divides a 32bit number in DX:AX by its 16bit operand and stores the 16bit quotient in AX and the 16bit remainder in DX. The second form divides a 16bit number in AX by its 8bit operand and stores the 8bit quotient in AL and the 8bit remainder in AH. For example “DIV BL” has an 8bit operand, so the implied dividend is 16bit and is stored in the AX register and “DIV BX” has a 16bit operand, so the implied dividend is 32bit and is therefore stored in the concatenation of the DX and AX registers. The higher word is stored in DX and the lower word in AX.

If a large number is divided by a very small number it is possible that the quotient is larger than the space provided for it in the implied destination. In this case an interrupt is automatically generated and the program is usually terminated as a result. This is called a divide overflow error; just like the calculator shows an –E– when the result cannot be displayed. This interrupt will be discussed later in the discussion of interrupts.

DIV (divide) performs an unsigned division of the accumulator (and its extension) by the source operand. If the source operand is a byte, it is divided into the two-byte dividend assumed to be in registers AL and AH. The byte quotient is returned in AL, and the byte remainder is returned in AH. If the source operand is a word, it is divided into the two-word dividend in registers AX and DX. The word quotient is returned in AX, and the word remainder is returned in DX. If the quotient exceeds the capacity of its destination register (FF for byte source, FFFF for word source), as when division by zero is attempted, a type 0 interrupt is generated, and the quotient and remainder are undefined.

Number Printing Example

The next example introduces a subroutine that can print a number received as its only argument at the top left of the screen using the algorithm just discussed.

Example 6.3	
001	; number printing algorithm
002	[org 0x0100]
003	jmp start
004	
005-022	;;;; COPY LINES 008-025 FROM EXAMPLE 6.2 (clrscr) ;;;;
023	
024	; subroutine to print a number at top left of screen
025	; takes the number to be printed as its parameter
026	printnum: push bp
027	mov bp, sp
028	push es
029	push ax
030	push bx
031	push cx
032	push dx
033	push di
034	
035	mov ax, 0xb800
036	mov es, ax ; point es to video base
037	mov ax, [bp+4] ; load number in ax
038	mov bx, 10 ; use base 10 for division
039	mov cx, 0 ; initialize count of digits
040	
041	nextdigit: mov dx, 0 ; zero upper half of dividend
042	div bx ; divide by 10
043	add dl, 0x30 ; convert digit into ascii value
044	push dx ; save ascii value on stack
045	inc cx ; increment count of values
046	cmp ax, 0 ; is the quotient zero
047	jnz nextdigit ; if no divide it again
048	
049	mov di, 0 ; point di to top left column
050	

051 052 053 054 055 056 057 058 059 060 061 062 063 064 065 066 067 068 069 070 071 072 073	<pre> nextpos: pop dx ; remove a digit from the stack mov dh, 0x07 ; use normal attribute mov [es:di], dx ; print char on screen add di, 2 ; move to next screen location loop nextpos ; repeat for all digits on stack pop di pop dx pop cx pop bx pop ax pop es pop bp ret 2 start: call clrscr ; call the clrscr subroutine mov ax, 4529 push ax ; place number on stack call printnum ; call the printnum subroutine mov ax, 0x4c00 ; terminate program int 0x21 </pre>
026-033	The registers are saved as an essential practice. The only parameter received is the number to be printed.
035-039	ES is initialized to video memory. AX holds the number to be printed. BX is the desired base, and can be loaded from a parameter. CX holds the number of digits pushed on the stack. This count is initialized to zero, incremented with every digit pushed and is used when the digits are popped one by one.
041-042	DX must be zeroed as our dividend is in AX and we want a 32bit division. After the division AX holds the quotient and DX holds the remainder. Actually the remainder is only in DL since the remainder can be from 0 to 9.
043-045	The remainder is converted into its ASCII representation and saved on the stack. The count of digits on the stack is incremented as well.
046-047	If the quotient is zero, all digits have been saved on the stack and if it is non-zero, we have to repeat the process to print the next digit.
049	DI is initialized to point to the top left of the screen, called the cursor home. If the screen location is to become a parameter, the value loaded in DI will change.
051-053	A digit is popped off the stack, the attribute byte is appended to it and it is displayed on the screen.
054-055	The next screen location is two bytes ahead so DI is incremented by two. The process is repeated CX times which holds the number of digits pushed on the stack.
057-064	We pop the registers pushed and “ret 2” to discard the only parameter on the stack.
066-070	The main program clears the screen and calls the printnum subroutine to print 4529 on the top left of the screen.

When the program is executed 4529 is printed on the top left of the screen. This algorithm is versatile in that the base number can be changed and the printing will be in the desired base. For example if “mov bx, 10” is changed to “mov bx, 2” the output will be in binary as 001000110110001. Similarly changing it to “mov bx, 8” outputs the number in octal as 10661. Printing it in hexadecimal is a bit tricky, as the ASCII codes for A-F do not consecutively start after the codes for 0-9. Inside the debugger observe the working of the algorithm is just as described in the above illustration. The digits are

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

separated one by one and saved on the stack. From bottom to top, the stack holds 0034, 0035, 0032, and 0039 after the first loop is completed. The next loop pops them one by one and routes them to the screen.

6.5. SCREEN LOCATION CALCULATION

Until now our algorithms used a fixed attribute and displayed at a fixed screen location. We will change that to use any position on the screen and any attribute. For mapping from the two dimensional coordinate system of the screen to the one dimensional memory, we need to multiply the row number by 80 since there are 80 columns per row and add the column number to it and again multiply by two since there are 2 bytes for each character.

For this purpose the multiplication routine written previously can be used. However we introduce an instruction of the 8088 microprocessor at this time that can multiply 8bit or 16bit numbers.

MUL Instruction

MUL (multiply) performs an unsigned multiplication of the source operand and the accumulator. If the source operand is a byte, then it is multiplied by register AL and the double-length result is returned in AH and AL. If the source operand is a word, then it is multiplied by register AX, and the double-length result is returned in registers DX and AX.

String Printing at Desired Location

We modify the string printing program to take the x-position, the y-position, and the attribute as parameters. The desired location on the screen can be calculated with the following formulae.

$$\text{location} = (\text{hypos} * 80 + \text{epos}) * 2$$

Example 6.4	
01	; hello world at desired screen location
02	[org 0x0100]
03	jmp start
04	
05	message: db 'hello world' ; string to be printed
06	length: dw 11 ; length of the string
07	
08-25	;;;; COPY LINES 008-025 FROM EXAMPLE 6.2 (clrscr) ;;;;
26	
27	; subroutine to print a string at top left of screen
28	; takes x position, y position, string attribute, address of string
29	; and its length as parameters
30	printstr: push bp
31	mov bp, sp
32	push es
33	push ax
34	push cx
35	push si
36	push di
37	
38	mov ax, 0xb800
39	mov es, ax ; point es to video base
40	mov al, 80 ; load al with columns per row
41	mull byte [bp+10] ; multiply with y position
42	add ax, [bp+12] ; add x position
43	shl ax, 1 ; turn into byte offset
44	mov di, ax ; point di to required location
45	mov si, [bp+6] ; point si to string
46	mov cx, [bp+4] ; load length of string in cx
47	mov ah, [bp+8] ; load attribute in ah
48	
49	nextchar: mov al, [si] ; load next char of string
50	mov [es:di], al ; show this char on screen
51	add di, 2 ; move to next screen location

52		add si, 1	; move to next char in string
53		loop nextchar	; repeat the operation cx times
54			
55		pop di	
56		pop si	
57		pop cx	
58		pop ax	
59		pop es	
60		pop bp	
61		ret 10	
62			
63	start:	call clrscr	; call the clrscr subroutine
64			
65		mov ax, 30	
66		push ax	; push x position
67		mov ax, 20	
68		push ax	; push y position
69		mov ax, 1	; blue on black attribute
70		push ax	; push attribute
71		mov ax, message	
72		push ax	; push address of message
73		push word [length]	; push message length
74		call printstr	; call the printstr subroutine
75			
76		mov ax, 0x4c00	; terminate program
77		int 0x21	

41	Push and pop operations always operate on words; however data can be read as a word or as a byte. For example we read the lower byte of the parameter y-position in this case.
43	Shifting is used for multiplication by two, which should always be the case when multiplication or division by a power of two is desired.
61	The subroutine had 5 parameters so “ret 10” is used.
65-74	The main program pushes 30 as x-position, 20 as y-position meaning 30th column on 20th row. It pushes 1 as the attribute meaning low intensity blue on black with no blinking.

When the program is executed hello world is displayed at the desired screen location in the desired color. The x-position, y-position, and attribute parameters can be changed and their effect be seen on the screen. The important difference in this example is the use of MUL instruction and the calculation of screen location given the x and y positions.

EXERCISES

1. Replace the following valid instruction with a single instruction that has the same effect. Don't consider the effect on flags.

```
dec cx
jnz L3
```
2. Write an infinite loop that shows two asterisks moving from right and left centers of the screen to the middle and then back. Use two empty nested loops with large counters to introduce some delay so that the movement is noticeable.
3. Write a function “printaddr” that takes two parameters, the segment and offset parts of an address, via the stack. The function should print the physical address corresponding to the segment offset pair passed at the top left of the screen. The address should be printed in hex and will therefore occupy exactly five columns. For example, passing 5600 and 7800 as parameters should result in 5D800 printed at the top left of the screen.
4. Write code that treats an array of 500 bytes as one of 4000 bits and for each blank position on the screen (i.e. space) sets the corresponding bit to zero and the rest to one.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

5. Write a function “drawrect” that takes four parameters via the stack. The parameters are top, left, bottom, and right in this order. The function should display a rectangle on the screen using the characters + - and |.

String Instructions

7.1. STRING PROCESSING

Till now very simple instructions of the 8088 microprocessor have been introduced. In this chapter we will discuss a bit more powerful instructions that can process blocks of data in one go. They are called block processing or string instructions. This is the appropriate place to discuss these instructions as we have just introduced a block of memory, which is the video memory. The vision of this memory for the processor is just a block of memory starting at a special address. For example the clear screen operation initializes this whole block to 0741.

There are just 5 block processing instructions in 8088. In the primitive form, the instructions themselves operate on a single cell of memory at one time. However a special prefix repeats the instruction in hardware called the REP prefix. The REP prefix allows these instructions to operate on a number of data elements in one instruction. This is not like a loop; rather this repetition is hard coded in the processor. The five instructions are STOS, LODS, CMPS, SCAS, and MOVS called store string, load string, compare string, scan string, and move string respectively. MOVS is the instruction that allows memory to memory moves, as was discussed in the exceptions to the memory to memory movement rules. String instructions are complex instruction in that they perform a number of tasks against one instruction. And with the REP prefix they perform the task of a complex loop in one instruction. This causes drastic speed improvements in operations on large blocks of memory. The reduction in code size and the improvement in speed are the two reasons why these instructions were introduced in the 8088 processor.

There are a number of common things in these instructions. Firstly they all work on a block of data. DI and SI are used to access memory. SI and DI are called source index and destination index because of string instructions. Whenever an instruction needs a memory source, DS:SI holds the pointer to it. An override is possible that can change the association from DS but the default is DS. Whenever a string instruction needs a memory destination, ES:DI holds the pointer to it. No override is possible in this case. Whenever a byte register is needed, AL holds the value. Whenever a word register is used AX holds the value. For example STOS stores a register in memory so AL or AX is the register used and ES:DI points to the destination. The LODS instruction loads from memory to register so the source is pointed to by DS:SI and the register used is AL or AX.

String instructions work on a block of data. A block has a start and an end. The instructions can work from the start towards the end and from the end towards the start. In fact they can work in both directions, and they must be allowed to work in both directions otherwise certain operations with overlapping blocks become impossible. This problem is discussed in detail later. The direction of movement is controlled with the Direction Flag (DF) in the flags register. If this flag is cleared the direction is from lower addresses towards higher addresses and if this flag is set the direction is from higher addresses to lower addresses. If DF is cleared, this is called the auto-increment mode of string instruction, and if DF is set, this is called the auto-decrement mode. There are two instructions to set and clear the direction flag.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

```

cld    ; clear direction flag
std    ; set direction flag

```

Every string instruction has two variants; a byte variant and a word variant. For example the two variants of STOS are STOSB and STOSW. Similarly the variants for the other string instructions are attained by appending a B or a W to the instruction name. The operation of each of the string instructions and each of the repetition prefixes is discussed below.

STOS

STOS transfers a byte or word from register AL or AX to the string element addressed by ES:DI and updates DI to point to the next location. STOS is often used to clear a block of memory or fill it with a constant.

The implied source will always be in AL or AX. If DF is clear, SI will be incremented by one or two depending of whether STOSB or STOSW is used. If DF is set SI will be decremented by one or two depending of whether STOSB or STOSW is used. If REP is used before this instruction, the process will be repeated CX times. CX is called the counter register because of the special treatment given to it in the LOOP and JCXZ instructions and the REP set of prefixes. So if REP is used with STOS the whole block of memory will be filled with a constant value. REP will always decrement CX like the LOOP instruction and this cannot be changed with the direction flag. It is also independent of whether the byte or the word variant is used. It always decrements by one; therefore CX has count of repetitions and not the count of bytes.

LODS

LODS transfers a byte or word from the source location DS:SI to AL or AX and updates SI to point to the next location. LODS is generally used in a loop and not with the REP prefix since the value previously loaded in the register is overwritten if the instruction is repeated and only the last value of the block remains in the register.

SCAS

SCAS compares a source byte or word in register AL or AX with the destination string element addressed by ES:DI and updates the flags. DI is updated to point to the next location. SCAS is often used to locate equality or in-equality in a string through the use of an appropriate prefix.

SCAS is a bit different from the other instructions. This is more like the CMP instruction in that it does subtraction of its operands. The prefixes REPE (repeat while equal) and REPNE (repeat while not equal) are used with this instruction. The instruction is used to locate a byte in AL in the block of memory. When the first equality or inequality is encountered; both have uses. For example this instruction can be used to search for a 0 in a null terminated string to calculate the length of the string. In this form REPNE will be used to repeat while the null is not there.

MOVS

MOVS transfers a byte or word from the source location DS:SI to the destination ES:DI and updates SI and DI to point to the next locations. MOVS is used to move a block of memory. The DF is important in the case of overlapping blocks. For example when the source and destination blocks overlap and the source is below the destination copy must be done upwards while if the destination is below the source copy must be done downwards. We cannot perform both these copy operations properly if the direction flag was not provided. If the source is below the destination and an upwards copy is used the source to be copied is destroyed. If however the copy is done downwards the portion of source destroyed is the one that has already been

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

copied. Therefore we need the control of the direction flag to handle this problem. This problem is further detailed in a later example.

CMPS

CMPS subtracts the source location DS:SI from the destination location ES:DI. Source and Destination are unaffected. SI and DI are updated accordingly. CMPS compares two blocks of memory for equality or inequality of the block. It subtracts byte by byte or word by word. If used with a REPE or a REPNE prefix is repeats as long as the blocks are same or as long as they are different. For example it can be used for find a substring. A substring is a string that is contained in another string. For example “has” is contained in “Mary has a little lamp.” Using CMPS we can do the operation of a complex loop in a single instruction. Only the REPE and REPNE prefixes are meaningful with this instruction.

REP Prefix

REP repeats the following string instruction CX times. The use of CX is implied with the REP prefix. The decrement in CX doesn't affect any flags and the jump is also independent of the flags, just like JCXZ.

REPE and REPNE Prefixes

REPE or REPZ repeat the following string instruction while the zero flag is set and REPNE or REPNZ repeat the following instruction while the zero flag is not set. REPE or REPNE are used with the SCAS or CMPS instructions. The other string instructions have nothing to do with the condition since they are performing no comparison. Also the initial state of flags before the string instruction does not affect the operation. The most complex operation of the string instruction is with these prefixes.

7.2. STOS EXAMPLE – CLEARING THE SCREEN

We take the example of clearing the screen and observe that how simple and fast this operation is with the string instructions. Even if there are three instructions in a loop they have to be fetched and decoded with every iteration and the time of three instructions is multiplied by the number of iterations of the loop. In the case of string instructions, many operations are short circuited. The instruction is fetched and decoded once and only the execution is repeated CX times. That is why string instructions are so efficient in their operation. The program to clear the screen places 0x0720 on the 2000 words on the screen.

Example 7.1	
001	; clear screen using string instructions
002	[org 0x0100]
003	jmp start
004	
005	; subroutine to clear the screen
006	clrscr: push es
007	push ax
008	push cx
009	push di
010	
011	mov ax, 0xb800
012	mov es, ax ; point es to video base
013	xor di, di ; point di to top left column
014	mov ax, 0x0720 ; space char in normal attribute
015	mov cx, 2000 ; number of screen locations
016	
017	cld ; auto increment mode
018	rep stosw ; clear the whole screen
019	
020	pop di

021		pop	cx		
022		pop	ax		
023		pop	es		
024		ret			
025					
026	start:	call	clrscr	; call clrscr subroutine	
027					
028		mov	ax, 0x4c00	; terminate program	
029		int	0x21		
013	A space efficient way to zero a 16bit register is to XOR it with itself. Remember that exclusive or results in a zero whenever the bits at the source and at the destination are same. This instruction takes just two bytes compared to “mov di, 0” which would take three. This is a standard way to zero a 16bit register.				

Inside the debugger the operation of the string instruction can be monitored. The trace into command can be used to monitor every repetition of the string instruction. However screen will not be cleared inside the debugger as the debugger overwrites its display on the screen so CX decrements with every iteration, DI increments by 2. The first access is made at B800:0000 and the second at B800:0002 and so on. A complex and inefficient loop is replaced with a fast and simple instruction that does the same operation many times faster.

7.3. LODS EXAMPLE – STRING PRINTING

The use of LODS with the REP prefix is not meaningful as only the last value loaded will remain in the register. It is normally used in a loop paired with a STOS instruction to do some block processing. We use LODS to pick the data, do the processing, and then use STOS to put it back or at some other place. For example in string printing, we will use LODS to read a character of the string, attach the attribute byte to it, and use STOS to write it on the video memory.

The following example will print the string using string instructions.

Example 7.2					
001		;	hello world	printing using string instructions	
002		[org	0x0100]		
003		jmp	start		
004					
005	message:	db	'hello world'	; string to be printed	
006	length:	dw	11	; length of string	
007					
008-027	;;;;; COPY LINES 005-024 FROM EXAMPLE 7.1 (clrscr) ;;;;;				
028					
029		;	subroutine to print a string		
030		;	takes the x position, y position, attribute, address of string and		
031		;	its length as parameters		
032	printstr:	push	bp		
033		mov	bp, sp		
034		push	es		
035		push	ax		
036		push	cx		
037		push	si		
038		push	di		
039					
040		mov	ax, 0xb800		
041		mov	es, ax	; point es to video base	
042		mov	al, 80	; load al with columns per row	
043		mul	byte [bp+10]	; multiply with y position	
044		add	ax, [bp+12]	; add x position	
045		shl	ax, 1	; turn into byte offset	
046		mov	di, ax	; point di to required location	
047		mov	si, [bp+6]	; point si to string	
048		mov	cx, [bp+4]	; load length of string in cx	
049		mov	ah, [bp+8]	; load attribute in ah	

050			
051		cld	; auto increment mode
052	nextchar:	lodsb	; load next char in al
053		stosw	; print char/attribute pair
054		loop nextchar	; repeat for the whole string
055			
056		pop di	
057		pop si	
058		pop cx	
059		pop ax	
060		pop es	
061		pop bp	
062		ret 10	
063			
064	start:	call clrscr	; call the clrscr subroutine
065			
066		mov ax, 30	
067		push ax	; push x position
068		mov ax, 20	
069		push ax	; push y position
070		mov ax, 1	; blue on black attribute
071		push ax	; push attribute
072		mov ax, message	
073		push ax	; push address of message
074		push word [length]	; push message length
075		call printstr	; call the printstr subroutine
076			
077		mov ax, 0x4c00	; terminate program
078		int 0x21	
051	Both operations are in auto increment mode.		
052-053	DS is automatically initialized to our segment. ES points to video memory. SI points to the address of our string. DI points to the screen location. AH holds the attribute. Whenever we read a character from the string in AL, the attribute byte is implicitly attached and the pair is present in AX. The same effect could not be achieved with a REP prefix as the REP will repeat LODS and then start repeating STOS, but we need to alternate them.		
054	CX holds the length of the string. Therefore LOOP repeats for each character of the string.		

Inside the debugger we observe how LODS and STOS alternate and CX is only used by the LOOP instruction. In the original code there were four instructions inside the loop; now there are only two. This is how string instructions help in reducing code size.

7.4. SCAS EXAMPLE – STRING LENGTH

Many higher level languages do not explicitly store string length; rather they use a null character, a character with an ASCII code of zero, to signal the end of a string. In assembly language programs, it is also easier to store a zero at the end of the string, instead of calculating the length of string, which is very difficult process for longer strings. So we delegate length calculation to the processor and modify our string printing subroutine to take a null terminated string and no length. We use SCASB with REPNE and a zero in AL to find a zero byte in the string. In CX we load the maximum possible size, which is 64K bytes. However actual strings will be much smaller. An important thing regarding SCAS and CMPS is that if they stop due to equality or inequality, the index registers have already incremented. Therefore when SCAS will stop DI would be pointing past the null character.

Example 7.3	
001	; hello world printing with a null terminated string
002	[org 0x0100]

003	jmp start	
004		
005	message: db 'hello world', 0	; null terminated string
006		
007-026	;;;; COPY LINES 005-024 FROM EXAMPLE 7.1 (clrscr) ;;;;	
027		
028		; subroutine to print a string
029		; takes the x position, y position, attribute, and address of a null
030		; terminated string as parameters
031	printstr: push bp	
032	mov bp, sp	
033	push es	
034	push ax	
035	push cx	
036	push si	
037	push di	
038		
039	push ds	
040	pop es	; load ds in es
041	mov di, [bp+4]	; point di to string
042	mov cx, 0xffff	; load maximum number in cx
043	xor al, al	; load a zero in al
044	repne scasb	; find zero in the string
045	mov ax, 0xffff	; load maximum number in ax
046	sub ax, cx	; find change in cx
047	dec ax	; exclude null from length
048	jz exit	; no printing if string is empty
049		
050	mov cx, ax	; load string length in cx
051	mov ax, 0xb800	
052	mov es, ax	; point es to video base
053	mov al, 80	; load al with columns per row
054	mul byte [bp+8]	; multiply with y position
055	add ax, [bp+10]	; add x position
056	shl ax, 1	; turn into byte offset
057	mov di, ax	; point di to required location
058	mov si, [bp+4]	; point si to string
059	mov ah, [bp+6]	; load attribute in ah
060		
061	cld	; auto increment mode
062	nextchar: lodsb	; load next char in al
063	stosw	; print char/attribute pair
064	loop nextchar	; repeat for the whole string
065		
066	exit: pop di	
067	pop si	
068	pop cx	
069	pop ax	
070	pop es	
071	pop bp	
072	ret 8	
073		
074	start: call clrscr	; call the clrscr subroutine
075		
076	mov ax, 30	
077	push ax	; push x position
078	mov ax, 20	
079	push ax	; push y position
080	mov ax, 1	; blue on black attribute
081	push ax	; push attribute
082	mov ax, message	
083	push ax	; push address of message
084	call printstr	; call the printstr subroutine
085		
086	mov ax, 0x4c00	; terminate program
	int 0x21	
039-040	Another way to load a segment register is to use a combination of push and pop. The processor doesn't match pushes and pops. ES is equalized to DS in this pair of instructions.	

Inside the debugger observe the working of the code for length calculation after SCASB has completed its operation.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

LES and LDS Instructions

Since the string instructions need their source and destination in the form of a segment offset pair, there are two special instructions that load a segment register and a general purpose register from two consecutive memory locations. LES loads ES while LDS loads DS. Both these instructions have two parameters, one is the general purpose register to be loaded and the other is the memory location from which to load these registers. The major application of these instructions is when a subroutine receives a segment offset pair as an argument and the pair is to be loaded in a segment and an offset register. According to Intel rules of significance the word at higher address is loaded in the segment register while the word at lower address is loaded in the offset register. As parameters segment should be pushed first so that it ends up at a higher address and the offset should be pushed afterwards. When loading the lower address will be given. For example “lds si, [bp+4]” will load SI from BP+4 and DS from BP+6.

7.5. LES AND LDS EXAMPLE

We modify the string length calculation subroutine to take the segment and offset of the string and use the LES instruction to load that segment offset pair in ES and DI.

Example 7.4	
001	; hello world printing with length calculation subroutine
002	[org 0x0100]
003	jmp start
004	
005	message: db 'hello world', 0 ; null terminated string
006	
007-026	;;;; COPY LINES 005-024 FROM EXAMPLE 7.1 (clrscr) ;;;;
027	
028	; subroutine to calculate the length of a string
029	; takes the segment and offset of a string as parameters
030	strlen: push bp
031	mov bp, sp
032	push es
033	push cx
034	push di
035	
036	les di, [bp+4] ; point es:di to string
037	mov cx, 0xffff ; load maximum number in cx
038	xor al, al ; load a zero in al
039	repne scasb ; find zero in the string
040	mov ax, 0xffff ; load maximum number in ax
041	sub ax, cx ; find change in cx
042	dec ax ; exclude null from length
043	
044	pop di
045	pop cx
046	pop es
047	pop bp
048	ret 4
049	
050	; subroutine to print a string
051	; takes the x position, y position, attribute, and address of a null
052	; terminated string as parameters
053	printstr: push bp
054	mov bp, sp
055	push es
056	push ax
057	push cx
058	push si
059	push di
060	
061	push ds ; push segment of string
062	mov ax, [bp+4]
063	push ax ; push offset of string
064	call strlen ; calculate string length

065		cmp	ax, 0	; is the string empty
066		jz	exit	; no printing if string is empty
067		mov	cx, ax	; save length in cx
068				
069		mov	ax, 0xb800	
070		mov	es, ax	; point es to video base
071		mov	al, 80	; load al with columns per row
072		mul	byte [bp+8]	; multiply with y position
073		add	ax, [bp+10]	; add x position
074		shl	ax, 1	; turn into byte offset
075		mov	di, ax	; point di to required location
076		mov	si, [bp+4]	; point si to string
077		mov	ah, [bp+6]	; load attribute in ah
078				
079		cld		; auto increment mode
080	nextchar:	lodsb		; load next char in al
081		stosw		; print char/attribute pair
082		loop	nextchar	; repeat for the whole string
083				
084	exit:	pop	di	
085		pop	si	
086		pop	cx	
087		pop	ax	
088		pop	es	
089		pop	bp	
090		ret	8	
091				
092	start:	call	clrscr	; call the clrscr subroutine
093				
094		mov	ax, 30	
095		push	ax	; push x position
096		mov	ax, 20	
097		push	ax	; push y position
098		mov	ax, 0x71	; blue on white attribute
099		push	ax	; push attribute
100		mov	ax, message	
101		push	ax	; push address of message
102		call	printstr	; call the printstr subroutine
103				
104		mov	ax, 0x4c00	; terminate program
105		int	0x21	

036	The LES instruction is used to load the DI register from BP+4 and the ES register from BP+6.
065	The convention to return a value from a subroutine is to use the AX register. That is why AX is not saved and restored in the subroutine.

Inside the debugger observe that the segment register is pushed followed by the offset. The higher address FFE6 contains the segment and the lower address FFE4 contains the offset. This is because we have a decremting stack. Then observe the loading of ES and DI from the stack.

7.6. MOVS EXAMPLE – SCREEN SCROLLING

MOVS has the two forms MOVSB and MOVSW. REP allows the instruction to be repeated CX times allowing blocks of memory to be copied. We will perform this copy of the video screen.

Scrolling is the process when all the lines on the screen move one or more lines towards the top of towards the bottom and the new line that appears on the top or the bottom is cleared. Scrolling is a process on which string movement is naturally applicable. REP with MOVS will utilize the full processor power to do the scrolling in minimum time.

In this example we want to scroll a variable number of lines given as argument. Therefore we have to calculate the source address, which is 160 times the number of lines to clear. The destination address is 0, which is the top left of the screen. The lines that scroll up are discarded so the source pointer is placed after them. An equal number of lines at the bottom are cleared. These lines have actually been copied above.

Example 7.5	
001	; scroll up the screen
002	[org 0x0100]
003	jmp start
004	
005	; subroutine to scroll up the screen
006	; take the number of lines to scroll as parameter
007	scrollup: push bp
008	mov bp,sp
009	push ax
010	push cx
011	push si
012	push di
013	push es
014	push ds
015	
016	mov ax, 80 ; load chars per row in ax
017	mul byte [bp+4] ; calculate source position
018	mov si, ax ; load source position in si
019	push si ; save position for later use
020	shl si, 1 ; convert to byte offset
021	mov cx, 2000 ; number of screen locations
022	sub cx, ax ; count of words to move
023	mov ax, 0xb800
024	mov es, ax ; point es to video base
025	mov ds, ax ; point ds to video base
026	xor di, di ; point di to top left column
027	cld ; set auto increment mode
028	rep movsw ; scroll up
029	mov ax, 0x0720 ; space in normal attribute
030	pop cx ; count of positions to clear
031	rep stosw ; clear the scrolled space
032	
033	pop ds
034	pop es
035	pop di
036	pop si
037	pop cx
038	pop ax
039	pop bp
040	ret 2
041	
042	start: mov ax,5
043	push ax ; push number of lines to scroll
044	call scrollup ; call the scroll up subroutine
045	
046	mov ax, 0x4c00 ; terminate program
047	int 0x21

The beauty of this example is that the two memory blocks are overlapping. If the source and destination in the above algorithm are swapped in an expectation to scroll down the result is strange. For example if 5 lines were to scroll down, the top five lines of the screen are repeated on the whole screen. This is where the use of the direction flag comes in.

When the source is five lines below the destination, the first five lines are copied on the first five lines of the destination. However the next five lines to be copied from the source have been destroyed in the process; because they were the same as the first five lines of the destination. The same is the problem with every set of five lines as the source is destroyed during the previous copy. In this situation we must go from bottom of the screen towards the top. Now the last five lines are copied to the last five lines of the destination. The next five lines are copied to next five lines of the destination destroying the last five lines of source; but now these lines are no longer needed and have been previously copied. Therefore the copy will be appropriately done in this case.

We give an example of scrolling down with this consideration. Now we have to calculate the end of the block instead of the start.

Example 7.6	
<pre> 001 ; scroll down the screen 002 [org 0x0100] 003 jmp start 004 005 ; subroutine to scrolls down the screen 006 ; take the number of lines to scroll as parameter 007 scrolldown: push bp 008 mov bp,sp 009 push ax 010 push cx 011 push si 012 push di 013 push es 014 push ds 015 016 mov ax, 80 ; load chars per row in ax 017 mul byte [bp+4] ; calculate source position 018 push ax ; save position for later use 019 shl ax, 1 ; convert to byte offset 020 mov si, 3998 ; last location on the screen 021 sub si, ax ; load source position in si 022 mov cx, 2000 ; number of screen locations 023 sub cx, ax ; count of words to move 024 mov ax, 0xb800 025 mov es, ax ; point es to video base 026 mov ds, ax ; point ds to video base 027 mov di, 3998 ; point di to lower right column 028 std ; set auto decrement mode 029 rep movsw ; scroll up 030 mov ax, 0x0720 ; space in normal attribute 031 pop cx ; count of positions to clear 032 rep stosw ; clear the scrolled space 033 034 pop ds 035 pop es 036 pop di 037 pop si 038 pop cx 039 pop ax 040 pop bp 041 ret 2 042 043 start: mov ax,5 044 push ax ; push number of lines to scroll 045 call scrolldown ; call scroll down subroutine 046 047 mov ax, 0x4c00 ; terminate program 048 int 0x21 </pre>	

7.7. CMPS EXAMPLE – STRING COMPARISON

For the last string instruction, we take string comparison as an example. The subroutine will take two segment offset pairs containing the address of the two null terminated strings. The subroutine will return 0 if the strings are different and 1 if they are same. The AX register will be used to hold the return value.

Example 7.7	
<pre> 001 ; comparing null terminated strings 002 [org 0x0100] 003 jmp start 004 005 msg1: db 'hello world', 0 006 msg2: db 'hello WORLD', 0 007 msg3: db 'hello world', 0 008 009-031 ;;;; COPY LINES 028-050 FROM EXAMPLE 7.4 (strlen) ;;;; 032 033 ; subroutine to compare two strings 034 ; takes segment and offset pairs of two strings to compare </pre>	

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

```

035 ; returns 1 in ax if they match and 0 other wise
036 strcmp: push bp
037          mov bp, sp
038          push cx
039          push si
040          push di
041          push es
042          push ds
043
044          lds si, [bp+4] ; point ds:si to first string
045          les di, [bp+8] ; point es:di to second string
046
047          push ds ; push segment of first string
048          push si ; push offset of first string
049          call strlen ; calculate string length
050          mov cx, ax ; save length in cx
051
052          push es ; push segment of second string
053          push di ; push offset of second string
054          call strlen ; calculate string length
055          cmp cx, ax ; compare length of both strings
056          jne exitfalse ; return 0 if they are unequal
057
058          mov ax, 1 ; store 1 in ax to be returned
059          repe cmpsb ; compare both strings
060          jcxz exitsimple ; are they successfully compared
061
062 exitfalse: mov ax, 0 ; store 0 to mark unequal
063
064 exitsimple: pop ds
065             pop es
066             pop di
067             pop si
068             pop cx
069             pop bp
070             ret 8
071
072 start: push ds ; push segment of first string
073        mov ax, msg1
074        push ax ; push offset of first string
075        push ds ; push segment of second string
076        mov ax, msg2
077        push ax ; push offset of second string
078        call strcmp ; call strcmp subroutine
079
080        push ds ; push segment of first string
081        mov ax, msg1
082        push ax ; push offset of first string
083        push ds ; push segment of third string
084        mov ax, msg3
085        push ax ; push offset of third string
086        call strcmp ; call strcmp subroutine
087
088        mov ax, 0x4c00 ; terminate program
089        int 0x21

```

005-007 Three strings are declared out of which two are equal and one is different.

044-045 LDS and LES are used to load the pointers to the two strings in DS:SI and ES:DI.

070 Since there are 4 parameters to the subroutine “ret 8” is used.

Inside the debugger we observe that REPE is shown as REP. This is because REP and REPE are represented with the same prefix byte. When used with STOS, LODS, and MOVS it functions as REP and when used with SCAS and CMPS it functions as REPE.

EXERCISES

1. Write code to find the byte in AL in the whole megabyte of memory such that each memory location is compared to AL only once.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

2. Write a far procedure to reverse an array of 64k words such that the first element becomes the last and the last becomes the first and so on. For example if the first word contained 0102h, this value is swapped with the last word. The next word is swapped with the second last word and so on. The routine will be passed two parameters through the stack; the segment and offset of the first element of the array.
3. Write a near procedure to copy a given area on the screen at the center of the screen without using a temporary array. The routine will be passed top, left, bottom, and right in that order through the stack. The parameters passed will always be within range the height will be odd and the width will be even so that it can be exactly centered.
4. Write code to find two segments in the whole memory that are exactly the same. In other words find two distinct values which if loaded in ES and DS then for every value of SI [DS:SI]=[ES:SI].
5. Write a function writechar that takes two parameters. The first parameter is the character to write and the second is the address of a memory area containing top, left, bottom, right, current row, current column, normal attribute, and cursor attribute in 8 consecutive bytes. These define a virtual window on the screen.
The function writes the passed character at (current row, current column) using the normal attribute. It then increments current column, If current column goes outside the window, it makes it zero and increments current row. If current row gets out of window, it scrolls the window one line up, and blanks out the new line in the window. In the end, it sets the attribute of the new (current row, current column) to cursor attribute.
6. Write a function "strcpy" that takes the address of two parameters via stack, the one pushed first is source and the second is the destination. The function should copy the source on the destination including the null character assuming that sufficient space is reserved starting at destination.

Software Interrupts

8.1. INTERRUPTS

Interrupts in reality are events that occurred outside the processor and the processor must be informed about them. Interrupts are asynchronous and unpredictable. Asynchronous means that the interrupts occur, independent of the working of the processor, i.e. independent of the instruction currently executing. Synchronous events are those that occur side by side with another activity. Interrupts must be asynchronous as they are generated by the external world which is unaware of the happenings inside the processor. True interrupts that occur in real time are asynchronous with the execution. Also it is unpredictable at which time an interrupt will come. The two concepts of being unpredictable and asynchronous are overlapping. Unpredictable means the time at which an interrupt will come cannot be predicted, while asynchronous means that the interrupt has nothing to do with the currently executing instruction and the current state of the processor.

The 8088 processor divides interrupts into two classes. Software interrupts and hardware interrupts. Hardware interrupts are the real interrupts generated by the external world as discussed above. Software interrupts on the contrary are not generated from outside the processor. They just provide an extended far call mechanism. Far all allows us to jump anywhere in the whole megabyte of memory. To return from the target we place both the segment and offset on the stack. Software interrupts show a similar behavior. It however pushes one more thing before both the segment and offset and that is the FLAGS register. Just like the far call loads new values in CS and IP, the interrupt call loads new values in CS, IP, and FLAGS. Therefore the only way to retain the value of original FLAGS register is to push and pop as part of interrupt call and return instructions. Pushing and popping inside the routine will not work as the routine started with an already tampered value.

The discussion of real time interrupts is deferred till the next chapter. They play the critical part in control applications where external hardware must be control and events and changes therein must be appropriately responded by the processor. To generate an interrupt the INT instruction is used. The routine that executes in response to an INT instruction is called the interrupt service routine (ISR) or the interrupt handler. Taking example from real time interrupts the routine to instruct an external hardware to close the valve of a boiler in response to an interrupt from the pressure sensor is an interrupt routine.

The software interrupt mechanism in 8088 uses vectored interrupts meaning that the address of the interrupt routine is not directly mentioned in an interrupt call, rather the address is lookup up from a table. 8088 provides a mechanism for mapping interrupts to interrupt handlers. Introducing a new entry in this mapping table is called hooking an interrupt.

Syntax of the INT instruction is very simple. It takes a single byte argument varying from 0-255. This is the interrupt number informing the processor, which interrupt is currently of interest. This number correlates to the interrupt handler routine by a routing or vectoring mechanism. A few interrupt numbers in the start are reserved and we generally do not use them. They are related to the processor working. For example INT 0 is the

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

divide by zero interrupt. A list of all reserved interrupts is given later. Such interrupts are programmed in the hardware to generate the designated interrupt when the specified condition arises. The remaining interrupts are provided by the processor for our use. Some of these were reserved by the IBM PC designers to interface user programs with system software like DOS and BIOS. This was the logical choice for them as interrupts provided a very flexible architecture. The remaining interrupts are totally free for use in user software.

The correlation process from the interrupt number to the interrupt handler uses a table called interrupt vector table. Its location is fixed to physical memory address zero. Each entry of the table is four bytes long containing the segment and offset of the interrupt routine for the corresponding interrupt number. The first two bytes in the entry contain the offset and the next two bytes contain the segment. The little endian rule of putting the more significant part (segment) at a higher address is seen here as well. Mathematically offset of the interrupt n will be at $nx4$ while the segment will be at $nx4+2$. One entry in this table is called a vector. If the vector is changed for interrupt 0 then INT 0 will take execution to the new handler whose address is now placed at those four bytes. INT 1 vector occupies location 4, 5, 6, and 7 and similarly vector for INT 2 occupies locations 8, 9, 10, and 11. As the table is located in RAM it can be changed anytime. Immediately after changing it the interrupt mapping is changed and now the interrupt will result in execution of the new routine. This indirection gives the mechanism extreme flexibility.

The operation of interrupt is same whether it is the result of an INT instruction (software interrupt) or it is generated by an external hardware which passes the interrupt number by a different mechanism. The currently executing instruction is completed, the current value of FLAGS is pushed on the stack, then the current code segment is pushed, then the offset of the next instruction is pushed. After this it automatically clears the trap flag and the interrupt flag to disallow further interrupts until the current routine finishes. After this it loads the word at $nx4$ in IP and the word at $nx4+2$ in CS if interrupt n was generated. As soon as these values are loaded in CS and IP execution goes to the start of the interrupt handler. When the handler finishes its work it uses the IRET instruction to return to the caller. IRET pops IP, then CS, and then FLAGS. The original value of IF and TF is restored which re-enables further interrupts. IF and TF will be discussed in detail in the discussion of real time interrupts. We have discussed three things till now.

1. The INT and IRET instruction format and syntax
2. The formation of IVT (interrupt vector table)
3. Operation of the processor when an interrupt is generated

Just as discussed in the subroutines chapter, the processor will not match interrupt calls to interrupt returns. If a RETF is used in the end of an ISR the processor will still return to the caller but the FLAGS will remain on the stack which will destroy the expectations of the caller with the stack. If we know what we are doing we may use such different combination of instructions. Generally we will use IRET to return from an interrupt routine. Apart from indirection the software interrupt mechanism is similar to CALL and RET. Indirection is the major difference.

The operation of INT can be written as:

- $sp \leftarrow sp+2$
- $[sp] \leftarrow flag$
- $sp \leftarrow sp+2$
- $if \leftarrow 0$
- $tf \leftarrow 0$
- $[sp] \leftarrow cs$
- $sp \leftarrow sp+2$
- $[sp] \leftarrow ip$

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

- $ip \leftarrow [0:N*4]$
- $cs \leftarrow [0:N*4+2]$

The operation of IRET can be written as:

- $ip \leftarrow [sp]$
- $sp \leftarrow sp-2$
- $cs \leftarrow [sp]$
- $sp \leftarrow sp-2$
- $flag \leftarrow [sp]$
- $sp \leftarrow sp-2$

The above is the microcode description of INT and IRET. To obey an assembly language instruction the processor breaks it down into small operations. By reading the microcode of an instruction its working can be completely understood.

The interrupt mechanism we have studied is an extended far call mechanism. It pushes FLAGS in addition to CS and IP and it loads CS and IP with a special mechanism of indirection. It is just like the table of contents that is located at a fixed position and allows going directly to chapter 3, to chapter 4 etc. If this association is changed in the table of contents the direction of the reader changes. For example if Chapter 2 starts at page 220 while 240 is written in the table of contents, the reader will go to page 240 and not 220. The table of contents entry is a vector to point to map the chapter number to page number. IVT has 256 chapters and the interrupt mechanism looks up the appropriate chapter number to reach the desired page to find the interrupt routine.

Another important similarity is that table of contents is always placed at the start of the book, a well known place. Its physical position is fixed. If some publishers put it at some place, others at another place, the reader will be unable to find the desired chapter. Similarly in 8088 the physical memory address zero is fixed for the IVT and it occupies exactly a kilobyte of memory as the $256 \times 4 = 1K$ where 256 is the number of possible interrupt vectors while the size of one vector is 4 bytes.

Interrupts introduce temporary breakage in the program flow, sometimes programmed (software interrupts) and un-programmed at other times (hardware interrupts). By hooking interrupts various system functionalities can be controlled. The interrupts reserved by the processor and having special functions in 8088 are listed below:

- INT 0, Division by zero
Meaning the quotient did not fit in the destination register. This is a bit different as this interrupt does not return to the next instruction, rather it returns to the same instruction that generated it, a DIV instruction of course. Here INT 0 is automatically generated by a DIV when a specific situation arises, there is no INT 0 instruction.
- INT 1, Trap, Single step Interrupt
This interrupt is used in debugging with the trap flag. If the trap flag is set the Single Step Interrupt is generated after every instruction. By hooking this interrupt a debugger can get control after every instruction and display the registers etc. 8088 was the first processor that has this ability to support debugging.
- INT 2, NMI-Non Maskable Interrupt
Real interrupts come from outside the processor. INT 0 is not real as it is generated from inside. For real interrupts there are two pins in the processor, the INT pin and the NMI pin. The processor can be directed to listen or not to listen to the INT pin. Consider a recording studio, when the recording is going on, doors are closed so that no interruption occurs, and when there is a break, the doors are opened so that if someone is waiting outside can come in. However if there is an urgency like fire outside then the door must be broken and the recording must not be catered for. For such situations is the NMI pin

which informs about fatal hardware failures in the system and is tied to interrupt 2. INT pin can be masked but NMI cannot be masked.

- INT 3, Debug Interrupt
The only special thing about this interrupt is that it has a single byte opcode and not a two byte combination where the second byte tells the interrupt number. This allows it to replace any instruction whatsoever. It is also used by the debugger and will be discussed in detail with the debugger working.
- INT 4, Arithmetic Overflow, change of sign bit
The overflow flag is set if the sign bit unexpectedly changes as a result of a mathematical or logical instruction. However the overflow flag signals a real overflow only if the numbers in question are treated as signed numbers. So this interrupt is not automatically generated but as a result of a special instruction INTO (interrupt on overflow) if the overflow flag is set. Otherwise the INTO instruction behaves like a NOP (no operation).

These are the five interrupts reserved by Intel and are generally not used in our operations.

8.2. HOOKING AN INTERRUPT

To hook an interrupt we change the vector corresponding to that interrupt. As soon as the interrupt vector changes, that interrupt will be routed to the new handler. Our first example is with the divide by zero interrupt. The normal system defined behavior in response to divide by zero is to display an error message and terminate the program. We will change it to display our own message.

Example 8.1

```

001      ; hooking divide by zero interrupt
002      [org 0x0100]
003      jmp start
004
005      message:      db  'You divided something by zero.', 0
006
007-029  ;;;; COPY LINES 028-050 FROM EXAMPLE 7.4 (strlen) ;;;;
030-049  ;;;; COPY LINES 005-024 FROM EXAMPLE 7.1 (clrscr) ;;;;
050-090  ;;;; COPY LINES 050-090 FROM EXAMPLE 7.4 (printstr) ;;;;
091
092      ; divide by zero interrupt handler
093      myisrfor0:    push ax                ; push all regs
094                   push bx
095                   push cx
096                   push dx
097                   push si
098                   push di
099                   push bp
100                   push ds
101                   push es
102
103                   push cs
104                   pop  ds                ; point ds to our data segment
105
106                   call clrscr            ; clear the screen
107                   mov  ax, 30
108                   push ax                ; push x position
109                   mov  ax, 20
110                   push ax                ; push y position
111                   mov  ax, 0x71          ; white on blue attribute
112                   push ax                ; push attribute
113                   mov  ax, message
114                   push ax                ; push offset of message
115                   call printstr          ; print message
116
117                   pop  es
118                   pop  ds
119                   pop  bp

```

120	pop di	
121	pop si	
123	pop dx	
124	pop cx	
125	pop bx	
126	pop ax	
127	iret	; return from interrupt
128		
129	; subroutine to generate a divide by zero interrupt	
130	genint0:	mov ax, 0x8432 ; load a big number in ax
131		mov bl, 2 ; use a very small divisor
132		div bl ; interrupt 0 will be generated
133		ret
134		
135	start:	xor ax, ax
136		mov es, ax ; load zero in es
137		mov word [es:0*4], myisrfor0 ; store offset at n*4
138		mov [es:0*4+2], cs ; store segment at n*4+2
139		call genint0 ; generate interrupt 0
140		
141		mov ax, 0x4c00 ; terminate program
142		int 0x21
93-101	We often push all registers in an interrupt service routine just to be sure that no unintentional modification to any register is made. Since any code may be interrupted an unintentional modification will be hard to debug	
103-104	Since interrupt can be called from anywhere we are not sure about the value in DS so we reset it to our code segment.	

When this program is executed our desired message will be shown instead of the default message and the computer will hang thereafter. The first thing to observe is that there is no INT 0 call anywhere in the code. INT 0 was invoked automatically by an internal mechanism of the processor as a result of the DIV instruction producing a result that cannot fit in the destination register. Just by changing the vector we have changed the response of the system to divide overflow situations.

However the system stuck instead of returning to the next instruction. This is because divide overflow is a special type of interrupt that returns to the same instruction instead of the next instruction. This is why the default handler forcefully terminates the program instead of returning. Now the IRET will take control back to the DIV instruction which will again generate the same interrupt. So the computer is stuck in an infinite loop.

8.3. BIOS AND DOS INTERRUPTS

In IBM PC there are certain interrupts designated for user programs to communicate with system software to access various standard services like access to the floppy drive, hard drive, vga, clock etc. If the programmer does not use these services he has to understand the hardware details like which particular controller is used and how it works. To avoid this and provide interoperability a software interface to basic hardware devices is provided except in very early computers. Since the manufacturer knows the hardware it burns the software to control its hardware in ROM. Such software is called firmware and access to this firmware is provided through specified interrupts.

This basic interface to the hardware is called BIOS (basic input output services). When the computer is switched on, BIOS gets the control at a specified address. The messages at boot time on the screen giving BIOS version, detecting different hardware are from this code. BIOS has the responsibility of testing the basic hardware including video, keyboard, floppy drive, hard drive etc and a special program to bootstrap. Bootstrap means to load OS from hard disk and from there OS takes control and proceeds to load its components and display a command prompt in the end. There are two

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

important programs; BIOS and OS. OS services are high level and build upon the BIOS services. BIOS services are very low level. A level further lower is only directly controlling the hardware. BIOS services provide a hardware independent layer above the hardware and OS services provide another higher level layer over the BIOS services. We have practiced direct hardware access with the video device directly without using BIOS or DOS. The layer of BIOS provides services like display a character, clear the screen, etc. All these layers are optional in that we can skip to whatever lower layer we want.

The most logical way to provide access to firmware is to use the interrupt mechanism. Specific services are provided at specific interrupts. CALL could also have been used but in that case every manufacturer would be required to place specific routines at specific addresses, which is not a flexible mechanism. Interrupts provide standard interrupt number for the caller and flexibility to place the interrupt routine anywhere in the memory for the manufacturer. Now for the programmer it is decided that video services will be provided at INT 10 but the actual address of the video services can and do vary on computers from different manufacturers. Any computer that is IBM compatible must make the video services accessible through INT 10. Similarly keyboard services are available at INT 16 and this is standard in every IBM compatible. Manufacturers place the code wherever they want and the services are exported through this interrupt.

BIOS exports its various services through different interrupts. Keyboard services are exported through INT 16, parallel port services through INT 17 and similarly others through different interrupts. DOS has a single entry point through INT 21 just like a pin hole camera, this single entry points leads to a number of DOS services. So how one interrupt provides a number of different services. A concept of service number is used here which is a defacto standard in providing multiple services through an interrupt. INT 10 is for video services and each of character printing service, screen clearing service, cursor movement service etc. has a service number associated to it. So we say INT 10 service 0 is used for this purpose and INT 10 service 1 is used for that purpose etc. Service numbers for different standard services are also fixed for every IBM compatible. The concept of exported services through interrupts is expanded with the service numbering scheme.

The service number is usually given in the AH register. Sometimes these 256 services seem less. For example DOS exports thousands of services. So will be often seen an extension to a level further with sub-services. For examples INT 10 character generator services are all provided through a single service number and the services are distinguished with a sub-service number.

The finally selected service would need some arguments for it to work. In interrupts arguments are usually not given through stack, rather registers are used. The BIOS and DOS specifications list which register contains which argument for a particular service of a particular interrupt.

We will touch some important BIOS and DOS services and not cover it completely neither is it possible to cover it in this space. A very comprehensive reference of interrupts is the Ralph Brown List. It is just a reference and not to be studied from end to end. All interrupts cannot be remembered and there is no need to remember them.

The service number is almost always in AH while the sub-service number is in AL or BL and sometimes in other registers. The documentation of the service we are using will list which register should hold what when the interrupt is invoked for that particular service.

Our first target using BIOS is video so let us proceed to our first program that uses INT 10 service 13 to print a string on the screen. BIOS will work even if the video memory is not at B8000 (a very old video card) since BIOS knows everything about the hardware and is hardware specific.

Example 8.2	
001	; print string using bios service
002	[org 0x0100]
003	jmp start
004	message: db 'Hello World'
005	
006	start: mov ah, 0x13 ; service 13 - print string
007	mov al, 1 ; subservice 01 - update cursor
008	mov bh, 0 ; output on page 0
009	mov bl, 7 ; normal attrib
010	mov dx, 0x0A03 ; row 10 column 3
011	mov cx, 11 ; length of string
012	push cs
013	pop es ; segment of string
014	mov bp, message ; offset of string
015	int 0x10 ; call BIOS video service
016	
017	mov ax, 0x4c00 ; terminate program
018	int 0x21
007	The sub-service are versions of printstring that update and do not update the cursor after printing the string etc.
008	Text video screen is in the form of pages which can be upto 32. At one time one page is visible which is by default the zeroth page unless we change it.

When we execute it the string is printed and the cursor is updated as well. With direct access to video memory we had no control over the cursor. To control cursor a different mechanism to access the hardware was needed.

Our next example uses the keyboard service to read a key. The combination of keyboard and video services is used in almost every program that we see and use. We will wait for four key presses; clear the screen after the first, and draw different strings after the next key presses and exiting after the last. We will use INT 16 service 1 for this purpose. This is a blocking service so it does not return until a key has been pressed. We also used the blinking attribute in this example.

Example 8.3

```

001      ; print string and keyboard wait using BIOS services
002      [org 0x100]
003      jmp start
004
005      msg1:      db 'hello world', 0
006      msg2:      db 'hello world again', 0
007      msg3:      db 'hello world again and again', 0
008
009-028      ;;;; COPY LINES 005-024 FROM EXAMPLE 7.1 (clrscr) ;;;;
029-069      ;;;; COPY LINES 050-090 FROM EXAMPLE 7.4 (printstr) ;;;;
070-092      ;;;; COPY LINES 028-050 FROM EXAMPLE 7.4 (strlen) ;;;;
093
094      start:      mov ah, 0x10      ; service 10 - vga attributes
095                  mov al, 03      ; subservice 3 - toggle blinking
096                  mov bl, 01      ; enable blinking bit
097                  int 0x10      ; call BIOS video service
098
099                  mov ah, 0      ; service 0 - get keystroke
100                  int 0x16      ; call BIOS keyboard service
101
102                  call clrscr      ; clear the screen
103
104                  mov ah, 0      ; service 0 - get keystroke
105                  int 0x16      ; call BIOS keyboard service
106
107                  mov ax, 0
108                  push ax      ; push x position
109                  mov ax, 0
110                  push ax      ; push y position
111                  mov ax, 1      ; blue on black
112                  push ax      ; push attribute
113                  mov ax, msg1
114                  push ax      ; push offset of string
115                  call printstr    ; print the string
116
117                  mov ah, 0      ; service 0 - get keystroke
118                  int 0x16      ; call BIOS keyboard service
119
120                  mov ax, 0
121                  push ax      ; push x position
122                  mov ax, 0
123                  push ax      ; push y position
124                  mov ax, 0x71    ; blue on white
125                  push ax      ; push attribute
126                  mov ax, msg2
127                  push ax      ; push offset of string
128                  call printstr    ; print the string
129
130                  mov ah, 0      ; service 0 - get keystroke
131                  int 0x16      ; call BIOS keyboard service
132
133                  mov ax, 0
134                  push ax      ; push x position
135                  mov ax, 0
136                  push ax      ; push y position
137                  mov ax, 0xF4    ; red on white blinking
138                  push ax      ; push attribute
139                  mov ax, msg3
140                  push ax      ; push offset of string
141                  call printstr    ; print the string
142
143                  mov ah, 0      ; service 0 - get keystroke
144                  int 0x16      ; call BIOS keyboard service
145
146                  mov ax, 0x4c00    ; terminate program
147                  int 0x21
148

```

099-100 This service has no parameters so only the service number is initialized in AH. This is the only service so there is no sub-service number as well. The ASCII code of the char pressed is returned in AL after this service.

Computer Architecture & Assembly Language Programming	Course Code: CS401
CS401@vu.edu.pk	VU

EXERCISES

1. Write a TSR that forces a program to exit when it tries to become a TSR using INT 21h/Service 31h by converting its call into INT 21h/Service 4Ch.
2. Write a function to clear the screen whose only parameter is always zero. The function is hooked at interrupt 80h and may also be called directly both as a near call and as a far call. The function should detect how it is called and return appropriately. It is provided that the direction flag will be set before the function is called.
3. Write a function that takes three parameters, the interrupt number (N) and the segment and offset of an interrupt handler XISR. The arguments are pushed in the order N, XISR's offset and XISR's segment. It is known that the first two instructions of XISR are PUSHF and CALL 0:0 followed by the rest of the interrupt handler. PUSHF instruction is of one byte and far call is of 5 bytes with the first byte being the op-code, the next two containing the target offset and the last two containing the target segment. The function should hook XISR at interrupt N and chain it to the interrupt handler previously hooked at N by manipulating the call 0:0 instruction placed near the start of XISR.
4. Write a TSR that provide the circular queue services via interrupt 0x80 using the code written in Exercise 5.XX. The interrupt procedure should call one of qcreate, qdestroy, qempty, qadd, qremove, and uninstall based on the value in AH. The uninstall function should restore the old interrupt 0x80 handler and remove the TSR from memory.

AL-JUNAID TECH INSTITUTE



Virtual University

CS401-ASSEMBLY LANGUAGE
(SOLVED MCQs)
FROM MIDTERM PAPERS
LECTURE (1-22)



BC190202640@vu.edu.pk
Junaidfazal08@gmail.com

For More Visit: vulmshelp.com

JUNAID MALIK
(0304-1659294)



AL-JUNAID TECH INSTITUTE

**PAID SERVICE
CS619 PROJECTS**

Availible training courses

- HTML
- JQUERY
- PHPMYSQL
- JAVASCRIPT
- BOOTSTRAPS
- NODE.JS
- REACT.JS
- CSS

LMS HANDLING

**PAID
ASSIGNMENTS , QUIZ & GDB**

**95% RESULTS
ALL LMS ACTIVITIES**



Contact Us :

+92 304 1659294

www.vulmshelp.com

junaidfazal08@gmail.com

AL-JUNAID TECH INSTITUTE

Question No:1

The first instruction of "COM" file must be at offset:

► 0x0010

► **0x0100 (Page 19)**

► 0x1000

► 0x0000

Question No:2

The execution of the instruction "mov word [ES: 0], 0x0741" will print character "A" on screen , background color of the screen will be

► **Black (Page 81)**

► White

► Red

► Blue

Question No:3

The iAPX888 architecture consists of _____ register.

► 12

► 14

► **16 (Page 15)**

► 18

Question No:3

The iAPX888 architecture consists of _____ register.

► 12

► 14

► **16 (Page 15)**

► 18

AL-JUNAID TECH INSTITUTE

Question No:5

One screen location corresponds to a

► Byte

► **Word (Page 80)**

► Double byte

➤ Double word

Question No:6

When an item is pushed on the decrementing stack, the top of the stack is

► **First decremented and then element copied on to the stack (Page 68)**

► First incremented and then element copied on to the stack

► Decrement after the element copied on to the stack

► Incremented after the element copied on to the stack

Question No:7

one Each screen location corresponds to a word, the lower byte of this word contains ____

► **The character code (Page 81)**

► The attribute byte

► The parameters

► The dimensions

Question No:8

if ax contains decimal -2 and BX contains decimal 2 then after the execution of instructions: CMP AX, BX, JA label

► **Jump will be taken**

► Zero flag will set

► ZF will contain value -4

► Jump will not be taken

Question No:9

AL-JUNAID TECH INSTITUTE

If D is "35" is shift to left 2 bits the new value

► 35

► 70

► 140

► 17

Question No:10

When two 16bit numbers are added the answer can be 17 bits long, this extra bit that won't fit in the target register is placed in the where it can be used and tested

► carry flag (Page 16)

► Parity Flag

► Auxiliary Carry

► Zero Flag

Question No: 11

Only instructions allow moving data from memory to memory.

► string (Page 29)

► word

► indirect

► stack

Question No:12

When a 16 bit number is divided by an 8 bit number, the quotient will be in

► AL (Page 85)

► AX

► AH

► DX

Question No:13

one Which bit of the attributes byte represents the red component of background color ?

► 3

AL-JUNAID TECH INSTITUTE

► 4

► 5

► 6 (Page 81)

Question No:14

| 0 | --> | 1 | 1 | 0 | 1 | 0 | 0 | 0 | --> | C | is a example of _____

► Shl

► sar

► Shr (Page 52)

► Sal

Question No:15

allow changing specific processor behaviors and are used to play with it.

► Special Instructions (Page 14)

► Data Movement Instructions

► Program Control Instructions

► Arithmetic and Logic Instructions

Question No:16

8088 is a 16bit processor with its accumulator and all registers of _____.

► 32 bits

► 6 bits

AL-JUNAID INSTITUTE OF GROUP

► 16 bits (Page 14)

► 64 bits

Question No:17

decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack

► PUSH (Page 71)

► POP

► CALL

AL-JUNAID TECH INSTITUTE

► RET

Question No: 18

In instruction ADC the operands can be

- Two register only
- Two register and one memory location
- **CF and two other operands (Page 57)**
- ZF and two other operands

Question No:19

After the execution of instruction "RET"

- **SP is incremented by 2 (Page 66)**
- SP is decremented by 2
- SP is incremented by 1
- SP is decremented by 1

Question No: 20

The extended ASCII has

- 64 characters
- **128 characters (Page 79)** AL-JUNAID INSTITUTE OF GROUP
- 256 characters
- 502 characters

Question No:21

The second byte in the word designated for screen location holds

o The dimension of the screen

- Character position on the screen
- **Character color on the screen (Page 81)**
- ASCII code of the character

Question No: 22

REP will always o Incremented CX by 1

- Incremented CX by 2
- **Decrement CX by 1 (Page 92)**
- Decrement CX by 2

Question No:23

AL-JUNAID TECH INSTITUTE

The routine that executes in response to an INT instruction is called

- **ISR (Page 103)**

- IRS
- ISP
- IRT

Question No:24

In the instruction "CMP AX,BX" the contents of

- AX are changed
- BX are changed
- CX are changed

- **Flag register are changed (Page 39)**

Question No:25

All the addressing mechanisms in iAPX86 return a number called _____ address

- **• Effective (Page 33)**

- faulty
- indirect
- direct

Question No:26

The execution of the instruction "mov word [ES: DI], 0x0720"

- **will clear next character on screen (Page 82)**

- will print "20" at top left of the screen
- will print "20" at top right of the screen
- will move DI at location 0720 on the screen

Question No: 27

"mov byte [num1],5" is _____ instruction.

- **legal (Page 30)**

- illegal
- stack based
- memory indirect

Question No:28

MOV instruction transfers a byte or word from which of the following source location.

- DS:DI

AL-JUNAID TECH INSTITUTE

- ES:SI
- ES:DI
- **DS:SI (Page 92)**

Question No:29

If AX contains FFFFh, then after execution of instruction "SAL ax, 3", the result will be

- -3
- +3
- -8
- +8

Question No:30

If the decimal number "35" is shifted by two bit to left, the new value will be

- 35
 - 70
 - 140 (00100011 = 35 , 10001100=140)
- o17

Question No: 31

While using STOSB, if DF=1 then

- The value of SI will be incremented by one
 - The value of SI will be incremented by two
 - The value of SI will be decremented by one
 - The value of SI will be decremented by two

Question No:32

After the execution of STOSW, the CX will be

- Decrement by 1
- **Decrement by 2 (Page 92)**
- Increment by 1
- Increment by 2

Question No: 33

The memory address always move from

- processor to memory
- memory to processor
- memory to peripheral

AL-JUNAID TECH INSTITUTE

- peripheral to processor

Question No:34

An offset alone is not complete without

• Segment (Page 34)

- code label
- index register
- data label

Question No:35

Code Segment is associated to _____ register by default.

• IP (Page 34)

- SS
- BP
- CX

Question No:36

After the execution of SAR instruction:

- ▶ MSB remain as it is
- ▶ MSB Will change

▶ MSB move to left (Page 52)

- ▶ No change will occur.

Question No:37

ASCII stands for _____.

The screen is two dimensional space having:

- ▶ 25 Rows and 25 Columns
- ▶ 25 Rows and 80 Columns
- ▶ 80 Rows and 80 Columns

▶ 80 Rows and 25 Columns (Page 80)

Question No:38

In the Rotate Right Operation every bit moves one position to right and the bit dropped from the right is inserted at the left and:

AL-JUNAID TECH INSTITUTE

► Dropped in CF (Page 53)

- moves to AL
- Don't go anywhere.

Question No:39

_____ of the following flags will be affected by MOVSB?

► DF

- ZF
- PF
- No effect on flags

Question No:40

The iAPX88 processor supports _____ modes of memory access.

- 5
- 6

► 7 (Page 35)

- 8

Question No:41

Stack is a _____ that behaves in a first in last out manner.

- Program

► data structure (Page 67)

- Heap
- None of the Given

Question No:42

The physical address of the stack is obtained by

- SS:SI combination

► SS:SP combination (Page 68)

- ES:BP combination

AL-JUNAID TECH INSTITUTE

► ES:SP combination

Question No:43

Foreground and background parameter will be

- 32bits
- 16bits
- **8bits**
- 4bits

Question No:44

The clear screen operation initialize whole block of memory

- 0741
- 0417
- 0714

► **0174, 0720 (Page 91)**

Question No:45

In STOSB instruction, when DF is Set, SI is

- Incremented by 1
- Incremented by 2

► **Decrement by 1**

- Decrement by 2

Question No:46

Assembly language is:

► **Low-level programming language**

- High-level programming language
- Also known as machine language
- Not considered closer to the computer

Question No:47

A 32 Bit processor has accumulator of -----

- 8 bit

AL-JUNAID TECH INSTITUTE

► 16 bit

► **32 bit (Page 12)**

► 64 bit

Question No:48

To transfer control back the RET instruction take

► 1 argument

► 1 argument

► **3 arguments (Page 72) (Not sure)**

► No arguments

Question No:49

RET is executed, it recovers the values from

► Register

► **Stack (Page 71)**

► Data segment

► Code segment

Question No:50

To convert any digit to its ASCII representation

► **Add 0x30 in the digit (Page 80)**

► Subtract 0x30 from the digit

► Add 0x61 in the digit

► Subtract 0x61 from the digit

Question No:51

The prevalent convention in most high level languages is stack clearing by the

► Caller

► **Callee (Page 74)**

AL-JUNAID TECH INSTITUTE

► RET

► Stack

Question No:52

After execution of JCXZ instruction CX will be changed with flag affect.

► CF

► OF

► DF

► None of Above (Page 43)

Question No:53

Execution of the instruction "mov word [ES : 0], 0x0741" will print

► "A" appear on the top left of screen (Page 81)

► "A" appear on the top right of screen

► "A" appear on the center of screen

► "A" appear on the bottom left of screen

Question No:54

if contains decimal -2 and BX contains decimal 2 then after the execution of instructions

CMP AX, BX JA label

► Jump will be taken

► Zero flag will set

► ZF will contain value -4

► Jump will not be taken

Question No:55

Which of the following options contain the set of instructions to open a window to the video memory?

► mov AX, 0xb008

mov ES, AX

► mov AX, 0xb800

mov ES, AX (Page 81) rep

► mov AX, 0x8b00

mov ES, AX

AL-JUNAID TECH INSTITUTE

► mov AX, 0x800b

mov ES, AX

Question No:56

If D is "35" is shift to left 2 bits the new value

► 35

► 70

► 140

► 17

Question No:57

Execution of the instruction "mov word [ES : 0], 0x1230" will print the character color will

► Green

► White

► Red

► Black

Question No:58

After the execution of SAR instruction

► The msb is replaced by a 0

► The msb is replaced by 1

► The msb retains its original value

► The msb is replaced by the value of CF (Page 52)

Question No:59

RETF will pop the offset in the

► BP

► IP (Page 69)

► SP

► SI

Question No:60

Far" jump is not position relative but is _____

AL-JUNAID TECH INSTITUTE

- ▶ memory dependent

▶ Absolute (Page 46)

- ▶ Temporary
- ▶ indirect

Question No:61

After the execution of instruction "RET 2"

- ▶ SP is incremented by 2
- ▶ SP is decremented by 2

▶ SP is incremented by 4 (Page 66)

- ▶ SP is decremented by 4

Question No:62

DIV instruction has

▶ Two forms (Page 85)

- ▶ Three forms
- ▶ Four forms
- ▶ Five forms

Question No:63

When the operand of DIV instruction is of 16 bits then implied dividend will be of

▶ 8 bits (Page 85)

- ▶ 16 bits
- ▶ 32 bits
- ▶ 64 bits

Question No:64

After the execution of MOVS instruction which of the following registers are updated:

- ▶ SI only
- ▶ DI only

▶ SI and DI only (Page 92)

AL-JUNAID TECH INSTITUTE

- ▶ SI, DI and BP only

Question No:65

In the word designated for one screen location, the higher address contains

- ▶ The character code

▶ The attribute byte (Page 81)

- ▶ The parameters
- ▶ The dimensions

Question No:66

If the address of memory location Num1 is 0117 and its content is 0005 then after execution of the instruction " mov bx, Num1" bx will contain

- ▶ 0005

▶ 0117 (Page 30)

- ▶ Num1
- ▶ 1701

Question No:67

In STOS instruction, the implied source will always be in

▶ AL or AX registers (Page 92)

- ▶ DL or DX registers
- ▶ BL or BX registers
- ▶ CL or CX registers

Question No:68

The shift logical right operation inserts

- A zero at right

➤ A zero at left (Page 52)

- A one at right
- A one at right

Question No:69

assembly the CX register is used normally as a _____ register.

AL-JUNAID TECH INSTITUTE

► source

► counter (Page 32)

► index

► pointer

Question No:70

Which is the unidirectional bus ?

(I) Control Bus

(II) Data Bus

(III) Address Bus

► I only

► II only

► III only (Page 9)

► I and II only

Question No:71

JC and JNC test the _____ flag.

► carry (Page 41)

► parity

► zero

► sign

Question No:72

After the execution of REP instruction CX will be decremented then which of the following flags will be affected?

► CF

► OF

► DF

► No flags will be affected (Page 93)

AL-JUNAID TECH INSTITUTE

Question No:73

which bit sets the character "blinking" on the screen?

► 5

► 6

► 7 (Page 7)

► 8

Question No:74

If we want to divide a signed number by 2, this operation can better be accomplished by

► SHR

► SAR (Page 52)

► SHL

► SAL

Question No:75

mov ax,5 has

► 1 operand

► 2 opeand (Page 25)

► 3 operand

► 4 opeand

Question No:76

REP will always

► Increment CX by 1

► Increment CX by 2

► Decrement CX by 1 (Page 92)

► Decrement CX by 2

Question No:77

Index registers are used to store _____

► Data

► Intermediate result

AL-JUNAID TECH INSTITUTE

► Address (Page 16)

- Both data and addresses

Question No:78

The bits of the _____ work independently and individually

- index register
- base register

► flags register (Page 12)

- accumulator

Question No:79

Which mathematical operation is dominant during the execution of SCAS instruction

- Division
- Multiplication
- Addition
- Subtraction

Question No:80

The instruction ADC has _____ Operand(s)

- 0
- 1
- 2

► 3 (Page 56)

Question No:81

In assembly the CX register is used normally as a _____ register.

- source

► counter (Page 92)

- index
- pointer

Question No:82

When a 32 bit number is divided by a 16 bit number, the quotient will be stored in

AL-JUNAID TECH INSTITUTE

► AX (Page 85)

- BX
- CX
- DX

Question No:83

In STOSB instruction SI is decremented or incremented by

- 4

► 1 (Page 92) SI and DI both are Index registers

- 2
- 3

Question No:84

CMPS instruction subtracts the source location to the destination location. Destination location always lies in

- DS:SI
- DS:DI
- ES:SI

► ES:DI (Page 93)

Question No:85

Regarding assembler, which statement is true:

► Assembler converts mnemonics to the corresponding OPCODE (Page 13)

- Assembler converts OPCODE to the corresponding mnemonics
- Assembler executes the assembly code all at once
- Assembler executes the assembly code step by step

Question No:86

If "BB" is the OPCODE of the instruction which states to "move a constant value to AX register", the hexadecimal representation (Using little Endian notation) of the instruction "Mov AX,336" ("150" in hexadecimal number system) will be:

- 0xBB0150

AL-JUNAID TECH INSTITUTE

▶ 0x5001BB

▶ 0x01BB50

▶ 0xBB5001 (Page 19)

Question No:87

one The maximum parameters a subroutine can receive (with the help of registers) are

▶ 6

▶ 7 (Page 72)

▶ 8

▶ 9

Question No:88

When a 16 bit number is divided by an 8 bit number, the dividend will be in

▶ AX (Page 85)

▶ BX

▶ CX

▶ DX

Question No:89

in Left-Shift-Operation the left most bit _____

▶ will drop

▶ will go into CF (Page 52)

▶ Will come to the right most

▶ will be always 1

Question No:90

Which of the following jumps makes intra segment control possible?

➤ Short

➤ **Far**

➤ Long

➤ Near

AL-JUNAID TECH INSTITUTE

Question No:91

There is an auto –increment mode when the ----- is -----.

- CX, not set
- ZF, not set
- DF, set
- **DF, not set**

Question No:92

REP allows the instruction to be ----- CX times.

- Reported
- Replaced
- **Repeated**
- repositioned

Question No:93

In ----- operation, a carry flag is inserted from the left moving every bit one position to its right. The right most bit is dropped into carry flag.

- **Rotate through carry (RCR)**
- Rotate Left (ROL)
- Rotate through carry left (RCL)
- Rotate Right(ROR)

Question No:94

ASCII stand for -----.

- American Standard Code for information interaction
- **American standard code for information interchange**

Question No:95

----- or ----- are taken if it last arithmetic operation has produced a number in its destination that has odd parity.

- JP, JPE
- JS, JNS
- JCX, JNS
- **JNP, JPO**

Question No:96

AL-JUNAID TECH INSTITUTE

Which of the following string instruction is generally used in loop instead of REP prefix?

- SCAS
- **LODS**
- CMPS
- STOS

Question No:97

mov, ax [num 1] is an example of ----- bit move.

- 8
- **16**
- 24
- 32

Question No:98

Which of the following used data segment by default?

- Base pointer
- **Base register**
- Stack pointer
- Instruction pointer

Question No:99

XOR can also be used a----- to invert selective bits.

- Destination operand
- Source operand
- **Masking operator**
- Masking bit

Question No:100

There are two----- from of MOVS instruction.

- 1
- **2**
- 3
- 4

Question No:101

AL-JUNAID TECH INSTITUTE

Registers are stored cells-----.

- Outside the processor
- Both outside and inside the processor
- **Inside the processor**
- Inside the memory

Question No:102

When the operand of DIV instruction is in 16-bits, then the implied dividend will be in.

- AX
- **DX:AX**
- ES:AX
- DS:BX

Question No:103

Shifting the integer 5 left by 1 bit result in----

- 5
- **10**
- 25
- 50

Question No:104

In which of the following addressing both registers are not constant?

- Index +offset
- **Base+ Index**
- Base +offset
- Base+ offset+ index

Question No:105

----- Transfers the word at top of stack (pointed to by Sp) to the destination operand and increment SP by two to the new top of stack.

- PUSH
- **POP**
- CALL
- RET n

Question No:106

In ----- operation, the output is 1 only if both input are 1.

AL-JUNAID TECH INSTITUTE

➤ **AND**

- OR
- NOT
- XOR

Question No:107

Which of the following is used for calculating the segment address of an interrupt of a service routine "n"?

➤ **$n*4+2$**

- $n*4-2$
- $n*4$
- $n*2+4$

Question No:108

Conditional jump can only be-----.

- Far
- **Short**
- Near
- long

Question No:109

Operand of POP is called ----- since data is moving from the stack to the operand.

- source
- **destination**
- increment
- decrement

Question No:110

How many are the functions of a register?

- 3
- **2**
- 1
- 4

Question No:111

AL-JUNAID TECH INSTITUTE

SACS compares a source in accumulator register with the ----- string element addressed by ES:DI and updates the flags.

- Source
- **destination**
- flags
- register

Question No:112

which of the following instruction is used for non- destructive AND operation?

- CMP
- NAND
- **TEST**
- ADC

Question No:113

Simple CMP instruction uses ----- operation.

- Addition
- Division
- **Subtraction**
- Multiplication

Question No:114

What does the given instruction do?

SHR DL, 1

- **Move right most bit in carry**
- Move bit in DL
- Shift and store the One bit in DL
- Shift and store the most bit in DL

Question No:115

----- Subtracts one from the operand.

- MEC
- Modulus
- **DEC**
- Shift

AL-JUNAID TECH INSTITUTE

Question No:116

Which of following is an illegal assembly instruction?

- mov{num1}, al
- mov ax, bx
- mov al, bl
- **mov { num1},{num2}**

Question No:117

In assembly language, the first executable instruction of the program should be placed at the offset_____.

- 0*1100
- 0*0010
- 0*0000
- **0*0100**

Question No:118

Cs and Is are the both _____ bit register.

- **16**
- 4
- 8
- 32

Question No:119

Which of the following is a special instruction?

- Add bx,0534
- **Cli**
- Mov ax,bx
- Cmp ax,0

Question No:120

Extended shifting algorithm consists of:

- 3 instructions
- 1 instructions
- **2 instructions**
- 4 instructions

AL-JUNAID TECH INSTITUTE

1. Stack is a data structure that behaves in a First in Last manner.

Out (Handsout searching Page 67)

2. We can convert a digit into its ASCII representation by adding to it.

0x30 (Page 80)

3. _____ decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack?

Or _____ function decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.

Push (page 71)

4. The _____ flag has a special role in debugging.

trap (Page 17)

5. iAPX88 stands for:

Intel Advanced Processor Extensions 88 (Page 14)

6. In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left. This bit is also copied into the carry flag.

Rotate Right (ROR) (Page 53)

7. Which of the following registers hold the address of data in

AL-JUNAID TECH INSTITUTE

Intel Boss?



AL-JUNAID TECH INSTITUTE

BX, BP, SI and DI

(page 30)

8. Which of the following is the general form of addressing?
Base + Index + Offset (page 35)
9. All mathematical and logical operations are performed on the:
Accumulator (page 12)
10. Which of the following register is used to hold the address of the next instruction to be executed?
Program Counter (page 13)
11. Which of the following is a program control instruction?
jne 0534 (page 14)
12. The basic function of register is to hold -----.
Oparands
13. A 16-bit processor has an accumulator of:
16 bit (Page 12)
14. REPE and REPNE prefixes are only meaningful with -----.
CMPS (page 93)
15. Stack is a data structure that behaves in ----- manner.
LIFO
Note: Stack two manner pr behave krta hy first FILO or second LIFO.Handsout mai LIFO Nhi hy magr iss ka correct answer LIFO hy.
16. Which of the following bit is dropped into the carry flag after the execution of "Shift Logical Right" operation?
Right most bit (page 52)
17. How logical errors are different from syntax error?

AL-JUNAID TECH INSTITUTE

Identifying syntax and logical errors is the responsibility of assembler and programmer respectively.

18. In ___ operation, the output is 1 only if both inputs are 1.
AND
19. How jmp is different from jnz?
jmp is an unconditional jump, however jnz is a conditional jump
20. In Shift Logical Left (SHL), the _____ bit is dropped into the carry flag.
most significant (page 52)
21. In STOS, the implied source operand always resides in -----.
AL or AX (page 92)
22. Which of the following is an illegal assembly instruction?
mov [num1], [num2] (page 30)
23. sub sp, 2
The above instruction is used to perform a _____ operation on stack.
PUSH (page 75, 76)
24. The maximum amount of memory accessible using 8085 processor is _____.
64 KB (page 15)
25. In direct addressing, the memory address given in the instruction is ____:
fixed (page 28)
26. What makes CMP different from SUB?

AL-JUNAID TECH INSTITUTE

SUB is a destructive subtraction while CMP is a non-destructive subtraction. Both affect the flags.

- 27. Which of the following are the basic bitwise logical operations?
AND, OR, XOR, and NOT (page 58)**
- 28. To convert the case of character, we add or subtract _____ from
it its ASCII code
0x20 (page 80)**
- 29. When the RET instruction is executed, it recovers the value of
the _____ from the stack.
instruction pointer (page 68)**
- 30. When the relative address stored with the instruction is in 16
bits, the jump is called a _____ jump.
Near (page 45)**
- 31. Registers are storage cells _____.
inside the processor (page 11)**
- 32. _____ subtracts one from the operand.
DEC (page 55)**
- 33. STI stands for?
Set the Interrupt flag**
- 34. RET instruction takes _____ arguments(s).
0 (page 64)**
- 35. A symbol associated to a point in the program is called a _____.
Or A symbol associated to an instruction in the program is known
as:
Label (page 25)**

AL-JUNAID TECH INSTITUTE

36. In 8088, there is a _____ stack.
decrementing (page 68)
37. CX register is mostly used as a _____ register.
Counter (page 32)
38. There are _____ byte(s) for each character on the screen.
2 (page 87)
39. Whenever an instruction needs a memory source,
holds the pointer to it.
DS:SI (page 91)
40. Keeping in view the downward compatible between the two
systems, the codes written for the intel iAPX88 are _____ on
the intel 386 processor.
valid (page 14)
41. In multiplication algorithm, We take the first digit of the
multiplier and multiply it with the _____.
multiplicand (page 51)
42. The ordering of a program's instructions is ensured by the ----
register.
Program Counter (page 13)
43. When an element is pushed on the stack, SP is decremented by
.....
two (page 68)
44. In iAPX88, _____ flag is specifically related to string
instructions.
direction (page 17)

AL-JUNAID TECH INSTITUTE

45. Which of following conditional jump is not dependant on any flag?

Or Which of the following jump does not depend on flags register?
JCXZ (page 43)

46. In ----- operation, a zero is inserted from the left and every bit moves one position to the right. The right most bit is dropped or down into the carry flag.

Shift Logical Right (SHR) (page 52)

47. In case of DIV BX instruction, the quotient is stored in ----- register.

AX (page 85)

48. Which of the following flags sets when a large number is subtracted from a smaller number?

CF (page 88)

49. Which of the following is the most illegal instruction?

Mov al, ax

50. In ADC instruction, there are-----.

two operands and the CF (page 57)

51. Call instruction takes a -----as an argument.

Label (page 64)

Or Call instruction takes ----- argument(s).

1 (page 64)

52. VGA stands for_____.

Video Graphics Adapter (page 79)

53. When the control is transferred to a subroutine?

After the arguments are pushed on the stack

AL-JUNAID TECH INSTITUTE

54. CS and IP are both-----bit registers.
16 (page 21)
55. A decrementing stack moves from ----- to -----addresses as elements are added in it.
higher, lower (page 68)
56. The most convenient place to store local variables is -----.
stack (page 75)
57. ----- can be used to check whether particular bits of a number are set or not.
AND (page 60)
58. Which of the following operations is used to set any specific bit in a binary number?
OR (page 59)
- b. Which of the following operations is used to clear any specific bit in a binary number?
AND (page 59)
59. Which bit of attribute byte represents the blue component of foreground color?
0 (page 81)
60. The segment offset pair is called a/an ----- address.
logical (page 22)
61. The extra bit produced as a result of an arithmetic operation that does not fit in the target register is stored in:
carry flag (page 16)
62. Cell width refers to the total number of bits in a memory cell while the total number of cells is called the-----.

AL-JUNAID TECH INSTITUTE

depth

(page 10)

- 63. Intel 8085 can access up to ----- of memory, whereas Intel 8088 can access up to----- of memory.**

64 KB, 1 MB

(page 15)

- 64. Registers are normally part of-----.**
CPU

- 65. The base pointer accesses local variables using ----- offsets.**
negative

(page 75)

- 66. ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers-----.**

30-39

(page 80)

- 67. Which of the following string instructions is generally used in a loop instead of REP prefix?**

LODS

(page 92)

- 68. The correct instruction to subtract with borrow is -----.**

SBB

(page 57)

- 69. mov [1234], ax is an example of----- addressing.**

direct

(page 35)

- 70. What the following piece of code does?**

Shl word [multiplicand], 1

Rcl word [multiplicand+2], 1

Extended Shift left

(page 58)

- 71. When the execution of Call instruction, the value of----- is decremented by 2.**

SP

(page 66)

AL-JUNAID TECH INSTITUTE

72. Which of the following bus is used to inform the memory that whether processor wants to read data or write data?

Control Bus (page 9)

73. In far jump, -----.

both offset and segment are given (page 46)

74. A value 0500 is stored in memory. If we transfer this value to a general-purpose register, then it will be shown as:

0005 (page 28)

75. In which of the following addressing, both registers are not constant?

Base + Index (page 36)

76. The number of ----- in a cell is called the cell width.

Bits (page 10)

77. In extended multiplication, we store the multiplicand in ----- bits, and the result is stored in ----- bits.

32, 32 (page 57)

78. Synchronization between the processor and the memory for the read and write operations is done by the-----.

Control bus (page 11)

79. iAPX88 Architecture consists of:

14 Registers (page 15)

80. Shift logical Left (SHL) moves all the bits one position to the --- and inserts a zero from the ----- --.

left, right (page 52)

AL-JUNAID TECH INSTITUTE

81. In MUL instruction, if the source operand is a byte, then it is multiplied with ----- register.
AL (page 87)
- b. In MUL instruction, if the source operand is a word, then it is multiplied with ----- register.
AX (page 87)
82. When the operand of DIV instruction is in 16 bits, then the implied dividend will be in -----.
DX:AX (page 85)
83. What does the PUSH operation do?
its copies the operand on the stack (page 68)
84. ----- transfers the word at the top of stack (pointed to by SP) to the destination operand and then increments SP by two.
POP (page 71)
85. Which of the following is an incorrect XOR operation?
XOR BX,AX (page 59)
86. This jump is taken if the last arithmetic operation changed the sign unexpectedly.
JO (page 43)
87. Which of the following jumps is taken if the last arithmetic operation did not changed the sign unexpectedly?
JNO (page 43)
88. Which would be the output of MOV AX, BL?
Assembler will declare that this is an illegal instruction (page 30)
89. ----- or ----- are taken if the last arithmetic operation has produced a number in the destination that has add parity.

AL-JUNAID TECH INSTITUTE

JNP, JNO

(page 43)

- 90. Group of bits processor uses to inform memory which element to read/write is collectively known as:**

Address bus

(page 9)

- 91. MOVS is used to move a block of memory.**

Move

(page 92)

- 92. Stack Pointer is ----- when the ret instruction is executed. incremented by 2**

(page 68)

- 93. ----- instruction directs the flow of program.**

Program Control

(page 14)

- 94. Which of the following instructions allows memory to memory movement of data?**

MOVS

(page 91)

- 95. By default, IP is associated with -----.**

CS

(page 34)

- 96. After the execution of Shift Arithmetic Right instruction, the most significant bit-----.**

retains its original value

(page 52)

- 97. In case of DIV BX instruction, the quotient is stored in ----- register.**

AX

(page 85)

- 98. Which part of B80500 (encoded instruction) is an opcode?**

B8

(page 25)

- 99. The operand of ----- is called the source operand because the data moves to the stack from the operand.**

AL-JUNAID TECH INSTITUTE

PUSH

(page 71)

100. Which of the following jump is taken if the last arithmetic operation has produced a number in its destination that has even parity?

JPE

(page 43)

101. When there is a 16 bit operand with the DIV instruction, then the implied dividend will be in -----bits.

32

(page 85)

102. EBCDIC, grey code and ASCII are the standards for ----- representation of characters.
alpha-numeric

103. In -----, a zero is inserted from the right and every bit moves one position to its left with the most significant bit dropped into the carry flag.

both SHL and SAL

(page 52)

104. Conditional jumps can only be -----.

Short

(page 46)

105. BP stands for-----pointer.

Base

(page 16)

106. In video memory, each screen location corresponds to----- bytes.

2

(page 82)

107. What makes CMP different from SUB?

SUB is a destructive subtraction while CMP is a non-destructive subtraction. Both affect the flags.

108. Which of the following registers is used as a counter?

AL-JUNAID TECH INSTITUTE

CX

(page 15)

109. The memory on video controller is accessible to the processor by ----- bus.

System

(page 80)

110. Which of the following addressing scheme has been used in the instruction MOV [BX], AX?

Based Register Indirect

(page 35)

111. The multiplier is stored in ---- bits in Extended Multiplication.

16

(page 57)

112. Which part of machine code tells the central processor to perform a certain task?

Operation code

113. Which of the following elements will come out first using the POP instruction if 6,3,5,8,4,7 and 9 were pushed on the stack? 9

114. During program execution, if any change occurs in AH or AL, is also reflected in -----.

AX

(page 15)

115. To modify the processor's behavior, ----- instructions are used.

special

(page 14)

116. Which of the following uses data segment by default?

Base register or BX

(page 35)

117. The operand of POP is called the ----- because the data is moving from the stack to the operand.

destination

(page 71)

AL-JUNAID TECH INSTITUTE

118. In assembly language, the first executable instruction of the program should be placed at the offset ----- --.

0x0100

(page 26)

119. Suppose, the current value of IP is 0x0129, and the relative address is 0x0012. What will be the new value of IP?

0x013B

120. In the instruction "mov word [es:0], 0x1230", 12 refers to ----- color on----- background.

green, blue

(page 81)

121. mov ax, [num1] is an example of----- bit move.

8

122. The physical memory address of 1234:5678 segment-offset pair is:

179B8

123. XOR can also be used as a ----- to invert selective bits.
masking operator

(page 60)

124. Call instruction fchanges the values of ----- and ----- registers.

IP, SP

(page 65)

125. There are ----- registers in iAPX88 architecture that can hold address of data.

4

(page 80)

126. What does the given instruction do?

SHR DL, 1

Move the right most bit into the carry flag (page 55)

AL-JUNAID TECH INSTITUTE

127. In which of the following terms, a character is stored in memory while writing assembly language program?
ASCII code (page 82)
128. Physical memory address is of:
20 bits (page 21)
129. In 8088, there is a----- stack.
decrementing (page 68)
130. Which of the following is also called intro-segment cell?
Near call (page 72)
131. Suppose $BX=0x0120$, $CS=0x1000$, and the memory under consideration is $[CS:BX+0x0880]$.
Which of the following will be the physical address?
109A0
132. Which of the following is the correct syntax for 'OR' operation in assembly language?
or ax, bx (page 59)
133. In assembly language, "DIV BL" instruction has an ----- bit operand.
8 (page 85)
134. Flags register is a special register in every architecture. It is also known as-----
program status word (page 12)
135. An important role of the stack is in the creation of ----- variables that are only needed while the subroutine is in execution and not afterwards.
Local (page 75)
136. Stack is:

AL-JUNAID TECH INSTITUTE

a data structure

(page 67)

137. Constant can never be used as-----.

destination

(page 25)

138. SP is associated (by default) with -----.

SS

(page 34)

139. Which bit of attribute byte represents the green component of foreground color?

1

(page 81)

140. BP stands for-----pointer.

Base

(page 32)

141. Shift logical Right (SHR) moves all the bits one position to the ----- and inserts a zero from the-----.

right, left

(page 52)

142. The execution of the instruction “mov word [ES :160], 0x1230” will print a character on the screen at the ----- column of the ----- row.

first, second

(page 81)

143. The stack of iAPX88 works on ----- sized elements.

word

(page 68)

144. MOVS is used to ----- a block of memory.

Move

(page 92)

145. Which bit of attribute byte represents the red component of foreground color?

2

(page 81)

AL-JUNAID TECH INSTITUTE

146. ----- can also be used as a masking operation to invert selective bits.

XOR

(page 60)

147. What is the content of stack pointer?

Address of the top element of the stack

148. Far jump is not position relative rather-----.

Absolute

(page 46)

149. In MUL instruction, if the source operand is a byte then the result is returned in ----- and-----.

AH, AL

(page 87)

150. Which of the following are the two variants of STOS instruction?

STOSB and STOSW

(page 92)

151. ----- cannot appear as an operand in any instruction, it is used for accessing instructions.

IP

152. Which of the following instructions allows code reusability in 8088?

CALL

(page 64)

153. Which bit of the attribute byte represents the red component of foreground color?

2

(page 81)

154. In 8051, there is an-----stack.

incrementing

(page 68)

155. Shift logical Right (SHR) operation copies the ----- bit into the carry flag.

right most

(page 52)

AL-JUNAID TECH INSTITUTE

156. Shift Arithmetic Right (SAR) shifts every bit one place to the right and places a copy of the ----- significant bit at the ----- significant place.

most, most (page 52)

157. There are ----- columns per row on the screen.

80 (page 87)

158. In XOR operation, the output is 1 if both inputs are -----, different (page 59)

159. Which of the following is the extension of object file?

.exe

160. The stack pointer marks the ----- of stack.

top (page 68)

161. Shifting the integer 5 left by 1 bit results in -----.

10

162. Program consists of ----- logical parts.

Two (page 20)

163. In ----- operation, a carry flag is inserted from the left moving every bit one position to its right. The right most bit is dropped in the carry flag.

Rotate Through Carry Right (RCR) (page 53)

164. Memory address always goes from -----

processor to memory

165. The unit of stack operations in iAPX88 is a -----

word (page 68)

166. 8088 is a processor with its accumulator and all registers of
16 bit, 16 bits (page 14)
167. Which of the following keywords is used to define two bytes in memory?
DW (page 25)
168. jump is absolute and not position relative.
Far (page 46)
169. Which of the following directive is used to reserve an eight bit space in memory?
db (page 25)
170. What makes db different from dw?
dw occupies 16 bits and db occupies 8 bits (page 25)
171. The total numbers of symbols in extended ASCII are
256
172. is used for permanent diversion.
Jump (page 63)
173. In MUL instruction, if the source operand is a word, then it is multiplied with ----- register.
AX (page 87)
174. The intent of the processor is carried to the memory for read/write operations through the.....
control bus (page 11)



CS-401 Assembly Language Update MCQS For Mid Term Solve By Vu Topper RM



80 To 100% Marks



وَتَعَزُّ مِنْ تَشَاءٍ وَتَذُلُّ مِنْ تَشَاءٍ



PROFESSIONAL ONLINE ACADEMY

We Offers

LMS Handling

Important Notes

Online Classes

Assignments

Quiz & GDB's

Projects

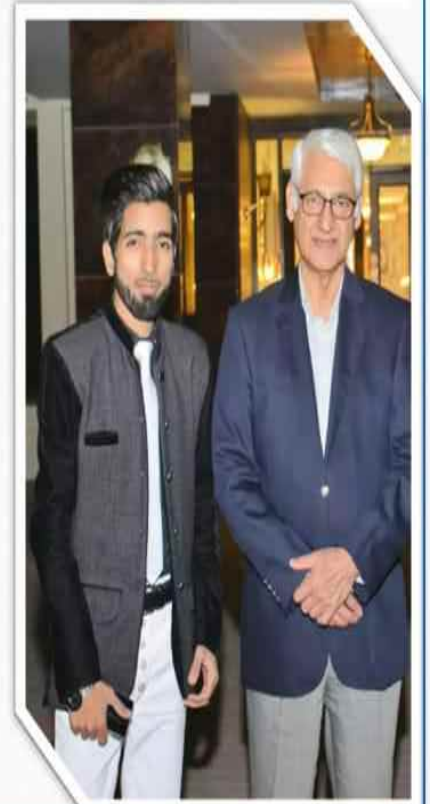
**NOTHING
Is
IMPOSSIBLE**

**Join Us
Now**

For More Info
Contact us at:

Rizwan Manzoor

☎ **0322-4021365**



بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:1**(Marks:1)****Vu-Topper RM**

The instruction for permanent diversion in Intel 8088 is _____.

- A. Rot
- B. Mov
- C. Call

D. Jump**Page 56****Question No:2****(Marks:1)****Vu-Topper RM**

There are _____ types of address wraparound.

A. 2**Page 29**

- B. 4
- C. 6
- D. 8

Question No:3**(Marks:1)****Vu-Topper RM**

The top of stack is contained in _____ register.

A. Bx**B. Sp****Page 63****C. Ax****D. Bp****Question No:4****(Marks:1)****Vu-Topper RM**

There are _____ registers in Intel 8088 that can hold the address of data.

A. 4**Page 22****B. 8****C. 10****D. 12****Question No:5****(Marks:1)****Vu-Topper RM**

"mov [bp], al" moves the one byte contents of the AL register to the address contained in the BP register in the current _____.

A. Data segment.**B. Stack segment.****Page 27**

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

- C. Code segment.
- D. Extra segment.

Question No:6

(Marks:1)

Vu-Topper RM

Which of the following is the number of operands for ADC (Add with Carry) instruction?

- A. 1
- B. 2**
- C. 3
- D. 4

Page 48

Question No:7

(Marks:1)

Vu-Topper RM

The jump is taken if the last arithmetic operation has changed the sign unexpectedly.

- A. JZN
- B. JZ
- C. JNO
- D. JO**

Page 35

Question No:8

(Marks:1)

Vu-Topper RM

Which of the following is shifted right in the 4-bit multiplication algorithm?

- A. Multiplicand
- B. Multiplier**
- C. RCL
- D. OLC

Page 46

Question No:9

(Marks:1)

Vu-Topper RM

Which of the following is an illegal instruction?

- A. Mov BX, 10
- B. MOV AX, BX**
- C. MOV AX, 65
- D. MOV AX, [BX+BP]

Page 22

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:10**(Marks:1)****Vu-Topper RM**

Extended Shifting Algorithm consists of:

A. 4 instructions

B. 2 instructions **Page 48**

C. 1 instruction

D. 5 instructions

Question No:11**(Marks:1)****Vu-Topper RM**

All the addressing mechanisms in iAPX88 return a number called the _____ address.

A. Faulty

B. Direct

C. Indirect

D. Effective **Page 25**

Question No:12**(Marks:1)****Vu-Topper RM**

Call instruction takes ----- argument(s).

A. 0

B. Page 64

C. 2

D. 3

Question No:13**(Marks:1)****Vu-Topper RM**

With the execution of CALL instruction, the value of _____ is decremented by 2.

A. SP

Page 66

B. BX

C. CX

D. DX

Question No:14**(Marks:1)****Vu-Topper RM**

Which bit of the attribute byte represents the red component of foreground color?

A. 2 **Page 73**

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

- B. 4
- C. 6
- D. 8

Question No:15

(Marks:1)

Vu-Topper RM

In MUL instruction, if the source operand is a byte, then the result is returned in ____ and

A. AH,AL **Page 79**

- B. AX,BL
- C. HL,AX
- D. AH,LX

Question No:16

(Marks:1)

Vu-Topper RM

Which of the following is also called intro-segment call?

A. Far call

B. Near call **Page 72**

- C. Short call
- D. Long call

Question No:17

(Marks:1)

Vu-Topper RM

Which bit of the attribute byte represents the Blinking of foreground character?

A. 7 **Page 73**

- B. 1
- C. 3
- D. 5

Question No:18

(Marks:1)

Vu-Topper RM

Which bit of the attribute byte represents the Red component of background color?

- A. 2
- B. 4
- C. 8

D. 6 **Page 7**

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:19**(Marks:1)****Vu-Topper RM**

Which bit of the attribute byte represents the Green component of background color?

A. 3

B. 5 **Page 73**

C. 9

D. 11

Question No:20**(Marks:1)****Vu-Topper RM**

Which bit of the attribute byte represents the Blue component of background color?

A. 4 **Page 73**

B. 6

C. 8

D. 10

Question No:21**(Marks:1)****Vu-Topper RM**

Which bit of the attribute byte represents the Intensity component of foreground color?

A. 3 **Page 73**

B. 4

C. 5

D. 6

Question No:22**(Marks:1)****Vu-Topper RM**

----- can be used to check whether particular bits of a number are set or not.

A. OR

B. AND **Page 52**

C. NOT

D. XOR

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:23

(Marks:1)

Vu-Topper RM

RET instruction takes _____ arguments(s).

A. Page 64

B. 1

C. 2

D. 3

Question No:24

(Marks:1)

Vu-Topper RM

When the RET instruction is executed, it recovers the value of the _____ from the stack.

A. Current instruction

B. Accumulator register

C. Previous instruction

D. Instruction pointer

Page 68

Question No:25

(Marks:1)

Vu-Topper RM

Stack is a data structure that behaves in a First in Last manner.

A. In

B. Add

C. Out Page 67

D. Posh

Question No:26

(Marks:1)

Vu-Topper RM

In _____ operation, the Most Significant Bit (MSB) is inserted from the right, causing every bit to move one position to the left. The MSB is also copied into the carry flag.

A. ROR

Page 45

B. RCL

C. CLR

D. LOR

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:27

(Marks:1)

Vu-Topper RM

We can convert a digit into its ASCII representation by adding to it.

- A. 0X05
- B. 0X10
- C. 0X20

D. 0x30 Page 80

Question No:28

(Marks:1)

Vu-Topper RM

_____ decrements SP (the stack pointer) by two, and then transfers a word from the source operand to the top of the stack.

- A. RET
- B. POP
- C. CALL

D. PUSH Page 71

Question No:29

(Marks:1)

Vu-Topper RM

In 8088, there is a _____ stack.

- A. Stagnant
- B. Incrementing

C. Decrementing Page 68

- D. Semi-automatic

Question No:30

(Marks:1)

Vu-Topper RM

There are _____ byte(s) for each character on the screen.

A. 2 Page 87

- B. 4
- C. 1
- D. 5

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:31

(Marks:1)

Vu-Topper RM

In case of DIV BX instruction, the quotient is stored in -----register.

A. AX **Page 85**

B. DX

C. AL

D. BL

Question No:32

(Marks:1)

Vu-Topper RM

Call instruction takes a ----- as an argument.

A. Label **Page 64**

B. source register

C. memory address

D. destination register

Question No:33

(Marks:1)

Vu-Topper RM

A decremting stack moves from ----- to -----addresses as elements are added in it.

A. lower, higher

B. higher, lower **Page 68**

C. positive, negative

D. negative, positive

Question No:34

(Marks:1)

Vu-Topper RM

Which of the following operations is used to set any specific bit in a binary number?

A. OR **Page 59**

B. AND

C. NOT

D. XOR

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:35

(Marks:1)

Vu-Topper RM

Which of the following operations is used to clear any specific bit in a binary number?

A. AND **Page 59**

B. XOR

C. NOT

D. NAND

Question No:36

(Marks:1)

Vu-Topper RM

Which of the following string instructions is generally used in a loop instead of REP prefix?

A. SCAS

B. CMPS

C. STOS

D. LODS **Page 92**

Question No:37

(Marks:1)

Vu-Topper RM

What the following piece of code does?

A. Shl word [multiplicand], 1

B. Rcl word [multiplicand+2], 1

C. Extended Shift left **Page 58**

Question No:38

(Marks:1)

Vu-Topper RM

In MUL instruction, if the source operand is a word, then it is multiplied with ----- register.

A. AX **Page 87**

B. AH

C. AL

D. BL

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:39

(Marks:1)

Vu-Topper RM

When the operand of DIV instruction is in 16 bits, then the implied dividend will be in _____.

A. AX

B. EX:AX

C. DX:AX **Page 85**

D. DX:BX

Question No:40

(Marks:1)

Vu-Topper RM

What does the PUSH operation do?

A. It increments the stack pointer

B. It increments the stack segment

C. Its copies the operand on the stack **Page 68**

D. It deletes the operand from the stack

Question No:41

(Marks:1)

Vu-Topper RM

Which of the following is an incorrect XOR operation?

A. XOR AX

B. XOR BX,AX **Page 59**

C. XOR AX, BX

D. XOR BYTE [VAR], 5

Question No:42

(Marks:1)

Vu-Topper RM

Stack Pointer is ----- when the ret instruction is executed.

A. divided by 2

B. multiplied by 2

C. decremented by 2

D. incremented by 2 **Page 68**

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:43

(Marks:1)

Vu-Topper RM

The operand of ----- is called the source operand because the data moves to the stack from the operand.

- A. RET
- B. CALL
- C. POP

D. PUSH

Page 71

Question No:44

(Marks:1)

Vu-Topper RM

The memory on video controller is accessible to the processor by the _____ bus.

- A. Cache
- B. Control

C. System

Page 80

D. Virtual memory

Question No:45

(Marks:1)

Vu-Topper RM

The operand of POP is called the ----- because the data is moving from the stack to the operand.

- A. Source
- B. Increment
- C. Decrement

D. Destination

Page 71

Question No:46

(Marks:1)

Vu-Topper RM

In a video memory, each screen location corresponds to _____

- A. One byte
- B. Two bytes
- C. Four bytes
- D. Eight bytes

Page 74

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:47

(Marks:1)

Vu-Topper RM

The iAP888 architecture consists of _____ register.

A. 12

B. 14 **Page 15**

C. 16

D. 10

Question No:48

(Marks:1)

Vu-Topper RM

The unit of stack operations in iAPX88 is a

A. Bit

B. Byte

C. Word **Page 68**

D. Double word

Question No:49

(Marks:1)

Vu-Topper RM

In XOR operation, the output is 1 if both inputs are _____.

A. Same

B. True

C. False

D. Different **Page 59**

Question No:50

(Marks:1)

Vu-Topper RM

Which of the following instructions allows code reusability in 8088?

A. RET

B. MOV

C. ADD

D. CALL **Page 64**

Question No:51

(Marks:1)

Vu-Topper RM

Which bit of the attribute byte represents the green component of

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

foreground color?

A. 0

B. 1

Page 73

C. 2

D. 3

Question No:52

(Marks:1)

Vu-Topper RM

Which of the following is also called intra-segment call?

A. Far call

B. Near call

Page 72

C. Short call

D. Long call

Question No:53

(Marks:1)

Vu-Topper RM

The stack of iAPX88 works on ----- sized elements.

A. Byte

B. Nibble

C. Word

Page 68

D. Paragraph

Question No:54

(Marks:1)

Vu-Topper RM

In 8051, there is an ----- stack.

A. Stagnant

B. Decrementing

C. Semi-automatic

D. Incrementing

Page 68

Question No:55

(Marks:1)

Vu-Topper RM

The maximum parameters a subroutine can receive are _____
when all the general purpose registers are used.

A. 1

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

B. 3

C. 5

D. 7 **Page 64**

Question No:56

(Marks:1)

Vu-Topper RM

Which bit is attribute but representing the blue component of foreground color

A. 0 **Page 73**

B. 1

C. 2

D. 3

Question No:57

(Marks:1)

Vu-Topper RM

How many characters are defined by the standard ASCII?

A. 32

B. 64

C. 132

D. 128 **Page 71**

Question No:58

(Marks:1)

Vu-Topper RM

Which of the following statement is used to clear the value of AX register,

A. NOT AX

B. XOR AX,0

C. XOR AX, AX

D. AND AX, AX

Question No:59

(Marks:1)

Vu-Topper RM

MUL instruction performs an unsigned multiplication of the source operand and the _____.

A. Base

B. Carry

C. Accumulator

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

D. Word at ES:DI

Question No:60

(Marks:1)

Vu-Topper RM

The clear screen operation initializes whole block of video memory to

A. 0417

B. 0714

C. 0721

D. 0174

Question No:61

(Marks:1)

Vu-Topper RM

Which of the following elements will come out first using the POP instruction if 6,3,5,8,4,7 and 9 were pushed on the stack?

A. 9

B. 8

C. 7

D. 6

Question No:62

(Marks:1)

Vu-Topper RM

To convert the case of a character, we add or subtract _____ from its ASCII code.

A. 0X20

B. 0X10

C. 0X41

D. 0X30

Question No:63

(Marks:1)

Vu-Topper RM

Scrolling is the process of moving one or more lines towards the top or bottom of the screen and this is _____.

A. Cleared

B. Add

C. Different

D. None of these

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:64

(Marks:1)

Vu-Topper RM

8051 by the same manufacturer has an_____ stack

A. Incrementing

B. Decrementing

C. Both

D. None of these

Question No:65

(Marks:1)

Vu-Topper RM

Mov [1234] ax is an example of — addressing.

A. Direct

B. Undirect

C. Pointer

D. Physical

Question No:66

(Marks:1)

Vu-Topper RM

How many are the functions of a register.

A. Four

B. Five

C. One

D. Two

Question No:67

(Marks:1)

Vu-Topper RM

B80500, B8 was the opcode and

A. 0500

B. 5000

C. 0055

D. 0505

Question No:68

(Marks:1)

Vu-Topper RM

This precise synchronization between the processor and the memory is the responsibility of the

A. Control bus

B. System bus

C. Logical bus

D. Digital bus

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:69

(Marks:1)

Vu-Topper RM

Which of the following is not a valid instruction in assembly language?

- A. MOV AX, 55
- B. MOV AX, BX
- C. MOV CS, 0xb800**
- D. MOV BX, AX

Question No:70

(Marks:1)

Vu-Topper RM

By default, BP is associated with _____.

- A. CS
- B. SS
- C. IP
- D. SP**

Question No:71

(Marks:1)

Vu-Topper RM

Execution of CALL instruction affects _____ number of bits in Flags register.

- A. 1**
- B. 2
- C. 3
- D. 4

Question No:72

(Marks:1)

Vu-Topper RM

If AND operation is performed between AX and BX, then how many AND operations will be performed in total?

- A. 2
- B. 8
- C. 16**
- D. 32

Question No:73

(Marks:1)

Vu-Topper RM

Which of the following cannot be pushed on the stack?

- A. 1 byte**
- B. 2 byte
- C. 3 byte

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

D. 4 byte

Question No:74

(Marks:1)

Vu-Topper RM

Call instruction changes the values of _____ and _____ registers.

A. IP,SP

B. IP,DX

C. IP,AX

D. IP,AL

Question No:75

(Marks:1)

Vu-Topper RM

Which combination will you prefer to obtain the physical address of the stack?

A. SP:SI

B. SS:SI

C. SP:DI

D. SS:SP

Question No:76

(Marks:1)

Vu-Topper RM

Which of the following instructions is used for non-destructive AND operation?

A. CMP

B. ADC

C. TEST

D. NAND

Question No:77

(Marks:1)

Vu-Topper RM

When the control is transferred to a subroutine?

A. When the return address is recovered

B. After the arguments are pushed on the stack

C. When the instruction pointer is not available

D. After the arguments are popped out from the stack

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:78

(Marks:1)

Vu-Topper RM

What is the content of stack pointer?

- A. Address of the next instruction
- B. Address of the current instruction
- C. Address of the top element of the stack**
- D. Address of the lower element of the stack

Question No:79

(Marks:1)

Vu-Topper RM

A 16-bit processor has an accumulator of:

- A. Bit Page 12**
- B. Byte
- C. Rate
- D. Value

Question No:80

(Marks:1)

Vu-Topper RM

The maximum amount of memory accessible using 8085 processor is

- A. 64 KB Page 15**
- B. 46 MB
- C. 66 KB
- D. 64 BK

Question No:81

(Marks:1)

Vu-Topper RM

VGA stands for _____.

Video Graphics Array Page 79

Question No:82

(Marks:1)

Vu-Topper RM

The _____ flag has a special role in debugging.

Trap Page 17

بري صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:83

(Marks:1)

Vu-Topper RM

iAPX88 stands for:

Intel Advanced Processor Extensions 88

Page 14

Question No:84

(Marks:1)

Vu-Topper RM

In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left. This bit is also copied into the carry flag.

Rotate Right (ROR)

Page 53

Question No:85

(Marks:1)

Vu-Topper RM

Which of the following registers hold the address of data in Intel Boss?

BX, BP, SI and DI

Page 30

Question No:86

(Marks:1)

Vu-Topper RM

Which of the following is the general form of addressing?

Base + Index + Offset

Page 35

Question No:87

(Marks:1)

Vu-Topper RM

All mathematical and logical operations are performed on the:

Accumulator

Page 12

Question No:88

(Marks:1)

Vu-Topper RM

Which of the following register is used to hold the address of the next instruction to be executed?

Program Counter

Page 13

Question No:89

(Marks:1)

Vu-Topper RM

REPE and REPNE prefixes are only meaningful with-----.

CMPS

Page 93

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:90

(Marks:1)

Vu-Topper RM

In Shift Logical Left (SHL), the _____ bit is dropped into the carry flag.

Most significant

Page 52

Question No:91

(Marks:1)

Vu-Topper RM

In STOS, the implied source operand always resides in -----.

AL or AX

Page 92

Question No:92

(Marks:1)

Vu-Topper RM

Which of the following is an illegal assembly instruction?

mov [num1], [num2]

Page 30

Question No:93

(Marks:1)

Vu-Topper RM

The above instruction is used to perform a _____ operation on stack.

PUSH

Page 76

Question No:94

(Marks:1)

Vu-Topper RM

In direct addressing, the memory address given in the instruction is

Fixed

Page 28

Question No:95

(Marks:1)

Vu-Topper RM

Which of the following are the basic bitwise logical operations?

AND, OR, XOR, and NOT

Page 58

Question No:96

(Marks:1)

Vu-Topper RM

To convert the case of character, we add or subtract _____ from its ASCII code.....

0x20

Page 80

بري صحبت سے تنہائی بہتر ہے اور نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:97

(Marks:1)

Vu-Topper RM

When the relative address stored with the instruction is in 16 bits, the jump is called a _____ jump.

Near Page 45

Question No:98

(Marks:1)

Vu-Topper RM

Registers are storage cells _____.

inside the processor Page 11

Question No:99

(Marks:1)

Vu-Topper RM

_____ subtracts one from the operand.

DEC Page 47

Question No:100

(Marks:1)

Vu-Topper RM

STI stands for?

Set the Interrupt flag

Question No:101

(Marks:1)

Vu-Topper RM

A symbol associated to a point in the program is called a ____.

Or A symbol associated to an instruction in the program is known as:

Label Page 25

Question No:102

(Marks:1)

Vu-Topper RM

CX register is mostly used as a _____ register.

Counter Page 32

Question No:103

(Marks:1)

Vu-Topper RM

Whenever an instruction needs a memory source, holds the pointer to it.

DS:SI Page 91

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:104

(Marks:1)

Vu-Topper RM

Keeping in view the downward compatible between the two systems, the codes written for the intel iAPX88 are _____ on the intel 386 processor.

Valid

Page 14

Question No:105

(Marks:1)

Vu-Topper RM

In multiplication algorithm, we take the first digit of the multiplier and multiply it with the _____.

Multiplicand Page 43

Question No:106

(Marks:1)

Vu-Topper RM

The ordering of a program's instructions is ensured by the ---register.

Program Counter Page 13

Question No:107

(Marks:1)

Vu-Topper RM

When an element is pushed on the stack, SP is decremented by.....

Two Page 68

Question No:108

(Marks:1)

Vu-Topper RM

In iAPX88, _____ flag is specifically related to string instructions.

Direction Page 17

Question No:109

(Marks:1)

Vu-Topper RM

Which of following conditional jump is not dependant on any flag?

RCL

JCXZ Page 43

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:110

(Marks:1)

Vu-Topper RM

In ----- operation, a zero is inserted from the left and every bit moves one position to the right. The right most bit is dropped or down into the carry flag.

Shift Logical Right (SHR)

Page 52

Question No:111

(Marks:1)

Vu-Topper RM

Which of the following flags sets when a large number is subtracted from a smaller number?

CF

Page 88

Question No:112

(Marks:1)

Vu-Topper RM

Stack two manner pr behave krta hy first FILO or second LIFO.

Handsout mai LIFO Nhi hy magr iss ka correct answer LIFO hy.

Which of the following bit is dropped into the carry flag after the execution of "Shift Logical Right" operation?

Right most bit

Page 52

Question No:113

(Marks:1)

Vu-Topper RM

In ADC instruction, there are -----.

Two operands and the CF

Page 57

Question No:114

(Marks:1)

Vu-Topper RM

The most convenient place to store local variables is -----.

Stack

Page 75

Question No:115

(Marks:1)

Vu-Topper RM

The segment offset pair is called a/an ----- address.

Logical

Page 22

برى صحبت سے تڻهائى بهتر هے اور تڻهائى سے نيك صحبت بهتر هے

For More Help Contact What's app 03224021365

Question No:116

(Marks:1)

Vu-Topper RM

The extra bit produced as a result of an arithmetic operation that does not fit in the target register is stored in:

Carry flag

Page 16

Question No:117

(Marks:1)

Vu-Topper RM

Cell width refers to the total number of bits in a memory cell while the total number of cells is called the_____.

Depth

Page 10

Question No:118

(Marks:1)

Vu-Topper RM

Intel 8085 can access up to ----- of memory, whereas Intel 8088 can access up to ----- of memory.

64 KB, 1 MB

Page 15

Question No:119

(Marks:1)

Vu-Topper RM

The base pointer accesses local variables using ----- offsets.

Negative

Page 75

Question No:120

(Marks:1)

Vu-Topper RM

ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers -----.

30-39

Page 80

Question No:121

(Marks:1)

Vu-Topper RM

The correct instruction to subtract with borrow is -----.

SBB

Page 57

Question No:122

(Marks:1)

Vu-Topper RM

mov [1234], ax is an example of ----- addressing.

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:123 **(Marks:1)** **Vu-Topper RM**

Which of the following bus is used to inform the memory that whether processor wants to read data or write data?

Control Bus **Page 9**

Question No:124 **(Marks:1)** **Vu-Topper RM**

In far jump,-----.

both offset and segment are given **Page 46**

Question No:125 **(Marks:1)** **Vu-Topper RM**

A value 0500 is stored in memory. If we transfer this value to a general-purpose register, then it will be shown as:

0005 **Page 28**

Question No:126 **(Marks:1)** **Vu-Topper RM**

In which of the following addressing, both registers are not constant?

Base + Index **Page 36**

Question No:127 **(Marks:1)** **Vu-Topper RM**

The number of ----- in a cell is called the cell width.

Bits **Page 10**

Question No:128 **(Marks:1)** **Vu-Topper RM**

In extended multiplication, we store the multiplicand in -----bits, and the result is stored in ----- bits.

32, 32 **Page 57**

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

Question No:129

(Marks:1)

Vu-Topper RM

Synchronization between the processor and the memory for the read and write operations is done by the -----.

Control bus Page 11

Question No:130

(Marks:1)

Vu-Topper RM

Shift logical Left (SHL) moves all the bits one position to the --and inserts a zero from the -----.

left, right Page 52

Question No:131

(Marks:1)

Vu-Topper RM

In MUL instruction, if the source operand is a byte, then it is multiplied with ----- register.

AH, AL Page 87

Question No:132

(Marks:1)

Vu-Topper RM

----- transfers the word at the top of stack (pointed to by SP) to the destination operand and then increments SP by two.

POP Page 71

Question No:133

(Marks:1)

Vu-Topper RM

This jump is taken if the last arithmetic operation changed the sign unexpectedly.

JO Page 43

Question No:134

(Marks:1)

Vu-Topper RM

Which of the following jumps is taken if the last arithmetic operation did not change the sign unexpectedly?

JNO Page 43

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:135

(Marks:1)

Vu-Topper RM

Which would be the output of MOV AX, BL?

Assembler will declare that this is an illegal instruction Page 30

Question No:136

(Marks:1)

Vu-Topper RM

_____ or _____ are taken if the last arithmetic operation has produced a number in its destination that has odd parity.

JNP, JPO Page 43

Question No:137

(Marks:1)

Vu-Topper RM

Group of bits processor uses to inform memory which element to read/write is collectively known as:

Address bus Page 9

Question No:138

(Marks:1)

Vu-Topper RM

MOVS is used to ----- a block of memory.

Move Page 92

Question No:139

(Marks:1)

Vu-Topper RM

----- instruction directs the flow of program.

Program Control Page 14

Question No:140

(Marks:1)

Vu-Topper RM

Which of the following instructions allows memory to memory movement of data?

MOVS Page 91

Question No:141

(Marks:1)

Vu-Topper RM

By default, IP is associated with -----.

CS Page 34

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:142

(Marks:1)

Vu-Topper RM

After the execution of Shift Arithmetic Right instruction, the most significant bit _____.

Retains its original value

Page 52

Question No:143

(Marks:1)

Vu-Topper RM

Which part of B80500 (encoded instruction) is an opcode?

B8

Page 25

Question No:144

(Marks:1)

Vu-Topper RM

Which of the following jump is taken if the last arithmetic operation has produced a number in its destination that has even parity?

JPE

Page 43

Question No:145

(Marks:1)

Vu-Topper RM

When there is a 16-bit operand with the DIV instruction, then the implied dividend will be in ----- bits.

32

Page 85

Question No:146

(Marks:1)

Vu-Topper RM

During program execution, if any change occurs in AH or AL, is also reflected in -----.

AX

Page 15

Question No:147

(Marks:1)

Vu-Topper RM

In -----, a zero is inserted from the right and every bit moves one position to its left with the most significant bit dropped into the carry flag.

Both SHL and SAL

Page 52

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:148

(Marks:1)

Vu-Topper RM

Conditional jumps can only be -----.

Short

Page 46

Question No:149

(Marks:1)

Vu-Topper RM

BP stands for ----- pointer.

Base

Page 16

Question No:150

(Marks:1)

Vu-Topper RM

In video memory, each screen location corresponds to ---

Bytes

Page 82

Question No:151

(Marks:1)

Vu-Topper RM

Which of the following registers is used as a counter?

CX

Page 15

Question No:152

(Marks:1)

Vu-Topper RM

Which of the following addressing scheme has been used in the instruction MOV [BX], AX?

Based Register Indirect

Page 35

Question No:153

(Marks:1)

Vu-Topper RM

The multiplier is stored in ---- bits in Extended Multiplication.

16

Page 57

Question No:154

(Marks:1)

Vu-Topper RM

To modify the processor's behavior, ----- instructions are used.

Special

Page 14

بري صحبت سے تھائی بہتر ہے اور تھائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:155

(Marks:1)

Vu-Topper RM

Which of the following uses data segment by default?

Base register or BX

Page 35

Question No:156

(Marks:1)

Vu-Topper RM

In assembly language, the first executable instruction of the program should be placed at the offset _____.

0x0100

Page 26

Question No:157

(Marks:1)

Vu-Topper RM

In the instruction “mov word [es:0], 0x1230”, 12 refers to -----color on - ----- background.

green, blue

Page 81

Question No:158

(Marks:1)

Vu-Topper RM

XOR can also be used as a ----- to invert selective bits. masking

Operator

Page 60

Question No:159

(Marks:1)

Vu-Topper RM

Call instruction fchanges the values of ----- and ----- registers.

IP, SP

Page 65

Question No:160

(Marks:1)

Vu-Topper RM

There are _____ registers in iAPX88 architecture that can hold address of data.

4

Page 22

Question No:161

(Marks:1)

Vu-Topper RM

What does the given instruction do?

SHR DL, 1

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:162

(Marks:1)

Vu-Topper RM

In which of the following terms, a character is stored in memory while writing assembly language program?

ASCII code

Page 82

Question No:163

(Marks:1)

Vu-Topper RM

Physical memory address is of:

20 bits

Page 21

Question No:164

(Marks:1)

Vu-Topper RM

Which of the following is the correct syntax for 'OR' operation in assembly language?

or ax, bx

Page 59

Question No:165

(Marks:1)

Vu-Topper RM

In assembly language, "DIV BL" instruction has an ----- bit operand.

8

Page 85

Question No:166

(Marks:1)

Vu-Topper RM

Flags register is a special register in every architecture. It is also known as -----.

program status word

Page 12

Question No:167

(Marks:1)

Vu-Topper RM

An important role of the stack is in the creation of ----variables that are only needed while the subroutine is in execution and not afterwards.

Local

Page 75

بری صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:168

(Marks:1)

Vu-Topper RM

Stack is:

A data structure

Page 67

Question No:169

(Marks:1)

Vu-Topper RM

Constant can never be used as -----.

Destination

Page 25

Question No:170

(Marks:1)

Vu-Topper RM

SP is associated (by default) with -----.

SS

Page 34

Question No:171

(Marks:1)

Vu-Topper RM

The execution of the instruction “mov word [ES :160], 0x1230” will print a character on the screen at the ----- column of the ----- row.

first, second

Page 81

Question No:172

(Marks:1)

Vu-Topper RM

----- can also be used as a masking operation to invert selective bits.

XOR

Page 60

Question No:173

(Marks:1)

Vu-Topper RM

The stack pointer marks the ----- of stack.

Top

Page 68

Question No:174

(Marks:1)

Vu-Topper RM

Far jump is not position relative rather -----.

Absolute

Page 46

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:175

(Marks:1)

Vu-Topper RM

Which of the following are the two variants of STOS instruction?

STOSB and STOSW Page 92

Question No:176

(Marks:1)

Vu-Topper RM

Shift logical Right (SHR) operation copies the ----- bit into the carry flag.

right most Page 52

Question No:177

(Marks:1)

Vu-Topper RM

Shift Arithmetic Right (SAR) shifts every bit one place to the right and places a copy of the ----- significant bit at the ----significant place.

most, most Page 52

Question No:178

(Marks:1)

Vu-Topper RM

There are ----- columns per row on the screen.

80 Page 87

Question No:179

(Marks:1)

Vu-Topper RM

Program consists of ----- logical parts.

Two Page 20

Question No:180

(Marks:1)

Vu-Topper RM

In ----- operation, a carry flag is inserted from the left moving every bit one position to its right. The right most bit is dropped in the carry flag.

Rotate Through Carry Right (RCR) Page 53

Question No:181

(Marks:1)

Vu-Topper RM

8088 is a processor with its accumulator and all registers of ...

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:182**(Marks:1)****Vu-Topper RM**

Which of the following keywords is used to define two bytes in memory?

DW Page 25**Question No:183****(Marks:1)****Vu-Topper RM**

..... jump is absolute and not position relative.

Far Page 46**Question No:184****(Marks:1)****Vu-Topper RM**

Which of the following directive is used to reserve an eight bits space in memory?

db Page 25**Question No:185****(Marks:1)****Vu-Topper RM**

What makes db different from dw?

dw occupies 16 bits and db occupies 8 bits Page 25**Question No:186****(Marks:1)****Vu-Topper RM**

..... is used for permanent diversion.

Jump Page 63**Question No:187****(Marks:1)****Vu-Topper RM**

The intent of the processor is carried to the memory for read/write operations through the.....

Control bus Page 11

بری صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:188

(Marks:1)

Vu-Topper RM

_____ containing the address of the next instruction to be executed.

Instruction pointer

Question No:189

(Marks:1)

Vu-Topper RM

How jmp is different from jnz?

jmp is an unconditional jump, however jnz is a conditional jump

Question No:190

(Marks:1)

Vu-Topper RM

There is an auto-increment mode when the _____ is _____.

DF, set

Question No:191

(Marks:1)

Vu-Topper RM

Unconditional jump _____.

Always transfer the control

Question No:192

(Marks:1)

Vu-Topper RM

num1: 75,50,77,82

Which of the following accesses the third element i.e. 77?

[num1+4]

Question No:193

(Marks:1)

Vu-Topper RM

In the opcode B80500, B8 was the opcode and 0500 was the operand stored immediately

Afterwards

Question No:194

(Marks:1)

Vu-Topper RM

In assembly language programming, "JNZ" is used to jump if the

_____ flag _____.

Zero, set value is 0 Google

بري صحبت سے تھائی بہتر ہے اور تھائی سے نيک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:195

(Marks:1)

Vu-Topper RM

Which of the following is used to clear the direction flag?

CLD

Question No:196

(Marks:1)

Vu-Topper RM

Which of the following has magnitude and sign?

Signed number

Question No:197

(Marks:1)

Vu-Topper RM

_____ is a special prefix that is used to repeat the block instructions.

REP

Question No:198

(Marks:1)

Vu-Topper RM

This jump is taken if the last arithmetic operation produced a negative number in its destination.

JS

Question No:199

(Marks:1)

Vu-Topper RM

Push and pop operations always operate on_____.

Words

Question No:200

(Marks:1)

Vu-Topper RM

Multiplying two 4-bit numbers result in a(an) _____ bit number.

8 bits

Question No:201

(Marks:1)

Vu-Topper RM

Suppose AX=5, BX=5, DX=0, CF=1, ZF=1 and AF=1. What will be the final value in AX register after the execution of ADC AX, BX?

10

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:202

(Marks:1)

Vu-Topper RM

Number of operands of ADC (add with carry) register are:

3

Question No:203

(Marks:1)

Vu-Topper RM

Which of the following is the renamed version of conditional jump JZ?

JE

Question No:204

(Marks:1)

Vu-Topper RM

If AX=00FF, then which of the following instruction can be used to change the value of AX to FF00?

AND AX, FF00

Question No:205

(Marks:1)

Vu-Topper RM

Conditional jump can only be _____.

Short

Question No:206

(Marks:1)

Vu-Topper RM

REP NZ repeats the following instruction while the _____.

ZF is not set

Question No:207

(Marks:1)

Vu-Topper RM

To declare a character in assembly language, we store its ASCII code in a _____.

Byte

Question No:208

(Marks:1)

Vu-Topper RM

REP with MOVS utilizes the _____ power of a processor to do scrolling in minimum time.

Full

بري صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:209

(Marks:1)

Vu-Topper RM

The basic function of register is to hold-----.

Oparands

Question No:210

(Marks:1)

Vu-Topper RM

Stack is a data structure that behaves in -----manner.

LIFO

Question No:211

(Marks:1)

Vu-Topper RM

In ___operation, the output is 1 only if both inputs are 1.

AND

Question No:212

(Marks:1)

Vu-Topper RM

Which of the following is the most illegal instruction?

Mov al, ax

Question No:213

(Marks:1)

Vu-Topper RM

Registers are normally part of -----.

CPU

Question No:214

(Marks:1)

Vu-Topper RM

When the execution of Call instruction, the value of ----- is

Decrementd by 2

Question No:215

(Marks:1)

Vu-Topper RM

EBCDIC, grey code and ASCII are the standards for -----

Representation of characters. alpha-numeric

بري صحبت سے تڻهائي بهتر هے اور تڻهائي سے نيڪ صحبت بهتر هے

For More Help Contact What's app 03224021365

Question No:216

(Marks:1)

Vu-Topper RM

Which part of machine code tells the central processor to perform a certain task?

Operation code

Question No:217

(Marks:1)

Vu-Topper RM

Suppose, the current value of IP is 0x0129, and the relative address is 0x0012. What will be the new value of IP?

0x013B

Question No:218

(Marks:1)

Vu-Topper RM

mov ax, [num1] is an example of ----- bit move.

8

Question No:219

(Marks:1)

Vu-Topper RM

The physical memory address of 1234:5678 segment-offset pair is:

179B8

Question No:220

(Marks:1)

Vu-Topper RM

Which of the following will be the physical address?

109A0

Question No:221

(Marks:1)

Vu-Topper RM

----- cannot appear as an operand in any instruction, it is used for accessing instructions.

IP

Question No:222

(Marks:1)

Vu-Topper RM

Which of the following is the extension of object file?

.exe

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:223

(Marks:1)

Vu-Topper RM

Shifting the integer 5 left by 1 bit results in -----.

10

Question No:224

(Marks:1)

Vu-Topper RM

Memory address always goes from

Processor to memory

Question No:225

(Marks:1)

Vu-Topper RM

The total numbers of symbols in extended ASCII are

256

Question No:226

(Marks:1)

Vu-Topper RM

SHL and SAL are same

True

Question No:227

(Marks:1)

Vu-Topper RM

Simple CMP instruction uses _____ operation

Subtraction

Question No:228

(Marks:1)

Vu-Topper RM

The segment off set pair is called a/an _____ address.

Logical

Question No:229

(Marks:1)

Vu-Topper RM

A/An _____ is an area of memory that holds all local variables and parameters.

Stack

Question No:230

(Marks:1)

Vu-Topper RM

-----is used for temporary diversion.

CALL

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:231

(Marks:1)

Vu-Topper RM

The swap flag can be stored in _____

A Register

Question No:232

(Marks:1)

Vu-Topper RM

The JUMP is taken if the last arithmetic has produced a positive number in its destination

JNP

Question No:233

(Marks:1)

Vu-Topper RM

Which of the following describes the purpose of MOVS instruction?

Move memory to memory

Question No:234

(Marks:1)

Vu-Topper RM

Which part of this (0000000B80500) encoded instruction is an offset?

0500

Question No:235

(Marks:1)

Vu-Topper RM

In the instruction “mov word [es:160], 0x1230”, 30 represents _____ character.

0

Question No:236

(Marks:1)

Vu-Topper RM

Multiplying two 4 bit numbers result in a _____ bit number.

8

Question No:237

(Marks:1)

Vu-Topper RM

In case of near jump, the relative address is stored in _____ bits.

16

Question No:238

(Marks:1)

Vu-Topper RM

_____ instructions have two parameters, one is the general purpose register to be loaded and the other is the memory location from which to load these registers.

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

LDS

Question No:239

(Marks:1)

Vu-Topper RM

_____ ports which interface the processor to the external world, including keyboards, mice, monitors, disc drives.

Input, output

Question No:240

(Marks:1)

Vu-Topper RM

In base+offset addressing, the value contained in the base register is add with offset to get _____.

Effective address

Question No:241

(Marks:1)

Vu-Topper RM

AX register can be divided into _____ and _____ bytes

Lower, higher

Question No:242

(Marks:1)

Vu-Topper RM

CLI stands for

Clear the interrupt flag

Question No:243

(Marks:1)

Vu-Topper RM

DW can store-----bit value in it.

16

Question No:244

(Marks:1)

Vu-Topper RM

When the stack pointer, points to the return address?

When the bubble sort subroutine is called

Question No:245

(Marks:1)

Vu-Topper RM

90 is the op-code of

Do nothing

بري صحبت سے تڻهائي بهتر هے اور تڻهائي سے نيك صحبت بهتر هے

For More Help Contact What's app 03224021365

Question No:246

(Marks:1)

Vu-Topper RM

When characters are stored in any high level or low level language, the actual thing stored in a byte is their _____.

ASCII code

Question No:247

(Marks:1)

Vu-Topper RM

We can convert any digit to-----by adding 0x30 in the digit.

ASCII

Question No:248

(Marks:1)

Vu-Topper RM

A complete _____ is called a pass over the array

Iteration

Question No:249

(Marks:1)

Vu-Topper RM

In _____ operation the carry flag is inserted from the right causing every bit to move one location to its left and the most significant bit occupying the carry flag.

Rotate Through Carry Left (RCL)

Question No:250

(Marks:1)

Vu-Topper RM

BH register is-----bit register.

8

Question No:251

(Marks:1)

Vu-Topper RM

In XOR operation the output is 1 if

Both inputs are different

Question No:252

(Marks:1)

Vu-Topper RM

Which of the following instruction is effectively same as to multiply the value of AX by 8?

SHL AX, 8

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:253

(Marks:1)

Vu-Topper RM

_____ interrupts are those which occur side by side with some other activity.

Synchronous

Question No:254

(Marks:1)

Vu-Topper RM

During CALL operation, the current value of the _____ is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer.

Instruction pointer

Question No:255

(Marks:1)

Vu-Topper RM

In SCAS Example, we use SCASB with _____ and a zero in AL register to find a zero byte in a string

REPNE

Question No:256

(Marks:1)

Vu-Topper RM

In interrupt vector table. Introducing a new entry in this mapping table is called _____ an interrupt.

Hooking

Question No:257

(Marks:1)

Vu-Topper RM

What does the following instruction do?

ADD AX. BX

Question No:258

(Marks:1)

Vu-Topper RM

The process through which the segment register can be explicitly specified as known as

Segment addressing

Question No:259

(Marks:1)

Vu-Topper RM

refers to the total number of bits in a memory cell.

Cell width

بري صحبت سے تنهائي بهتر ہے اور تنهائي سے نيك صحبت بهتر ہے

For More Help Contact What's app 03224021365

Question No:260

(Marks:1)

Vu-Topper RM

The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

Shift Arithmetic Right (SAR)

Question No:261

(Marks:1)

Vu-Topper RM

_____ and _____ cannot be used as 8bit register pairs like AX, BX, CX, and DX.

SI, DI

Question No:262

(Marks:1)

Vu-Topper RM

AX and BX both are 16-bit register, if we perform AND operation on these two registers, then how many AND operations will be performed?

16 And operation

Question No:263

(Marks:1)

Vu-Topper RM

CS and IP are both _____ bit registers.

16

Question No:264

(Marks:1)

Vu-Topper RM

In 8080, there is a _____ stack.

Decrementing

Question No:265

(Marks:1)

Vu-Topper RM

_____ movement of data is not allowed in assembly language.

Memory-to-Memory

Question No:266

(Marks:1)

Vu-Topper RM

Motorola follows _____

Big endian

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:267

(Marks:1)

Vu-Topper RM

When the first thing popped off from the stack, the stack would be the return “address” and not the _____

Argument

Question No:268

(Marks:1)

Vu-Topper RM

Which of the following is a Program Control Instruction?

cmp ax,0

Question No:269

(Marks:1)

Vu-Topper RM

Logical addressing is a mechanism to access ____ memory.

Physical memory

Question No:270

(Marks:1)

Vu-Topper RM

Twenty-bit register is formed by the combination of two ____ bit register.

Sixteen

Question No:271

(Marks:1)

Vu-Topper RM

Physical address calculating depends on

Effective address

Question No:272

(Marks:1)

Vu-Topper RM

_____ also known as source operand since the data is moving to stack from this operand.

PUSH

Question No:273

(Marks:1)

Vu-Topper RM

By default CS is associated with

IP

Question No:274

(Marks:1)

Vu-Topper RM

The stack pointer contains the address of the word that is currently on _

Top the stack

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:275

(Marks:1)

Vu-Topper RM

All addressing mechanisms in iAPX88 return a number called ____ address.

Effective

Question No:276

(Marks:1)

Vu-Topper RM

Which of the following is the interrupt number for NMI?

INT 3

Question No:277

(Marks:1)

Vu-Topper RM

Use of AND operation to make selective bits zero in its destination operand is known as ____.

Selective Bit Clearing

Question No:278

(Marks:1)

Vu-Topper RM

____ is used to store both the instructions to be executes by the microprocessor and the data to be used in the computation.

Microprocessor

Question No:279

(Marks:1)

Vu-Topper RM

DX play an important role in arithmetic

Addition

Question No:280

(Marks:1)

Vu-Topper RM

REPE or REPNE are used with the _____instructions

SCAS

Question No:281

(Marks:1)

Vu-Topper RM

Keywords used to define two bytes program

DW

Question No:282

(Marks:1)

Vu-Topper RM

The shift logical left operation is the exact_____ of shift logical right

Opposite

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:283

(Marks:1)

Vu-Topper RM

Sending the appropriate signal on the control bus to the memory is the responsibility of

Control Bus

Question No:284

(Marks:1)

Vu-Topper RM

A parallel port has ———- views

2

Question No:285

(Marks:1)

Vu-Topper RM

Which of the following is the interrupt number for debug interrupt

INT 3

Question No:286

(Marks:1)

Vu-Topper RM

Each entry of the interrupt vector table is of ____ bytes

4

Question No:287

(Marks:1)

Vu-Topper RM

If BL contains 000000101 then after a Single Right Shift, BL will contain

00000010

Question No:288

(Marks:1)

Vu-Topper RM

_____can be used to check whether particular bit of number are set or not

AND

Question No:289

(Marks:1)

Vu-Topper RM

The Stack of 8088 works on ____Sized element

Word

Question No:290

(Marks:1)

Vu-Topper RM

The interrupt call loads new values in segment

Flag

بري صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:291

(Marks:1)

Vu-Topper RM

Mov AX, 0XB800, Move ES, AX : this instruction points ES to

Video Base

Question No:292

(Marks:1)

Vu-Topper RM

When the operand of DIV instruction is of 16 bits then implied dividend will be of

Bits

Question No:293

(Marks:1)

Vu-Topper RM

DB-25 is a _____ Port Connector

Parallel

Question No:294

(Marks:1)

Vu-Topper RM

Flag register is a special register in every architecture ,, is as also known as

Program Status Word

Question No:295

(Marks:1)

Vu-Topper RM

Mov [1234].ax is an example of

direct addressing

Question No:296

(Marks:1)

Vu-Topper RM

Source operand always resided in

accumulator register

Question No:297

(Marks:1)

Vu-Topper RM

Always resided in accumulator register

source operand

Question No:298

(Marks:1)

Vu-Topper RM

INT instruction takes _____ argument varying from 0-255.

1 byte

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:299

(Marks:1)

Vu-Topper RM

Program consists of
logical parts

Question No:300

(Marks:1)

Vu-Topper RM

8088 provides a mechanism for mapping interrupts to interrupt handlers
is called h
hooking an interrupt

Question No:301

(Marks:1)

Vu-Topper RM

The segment, offset pair is called a
logical address

Question No:302

(Marks:1)

Vu-Topper RM

The local variables and the parameters are always stored in
stack segment

Question No:303

(Marks:1)

Vu-Topper RM

JNP and JPO is taken if the last arithmetic operation produced a number
in its destination that has
Odd parity

Question No:304

(Marks:1)

Vu-Topper RM

JP and JPE is taken if the last arithmetic operation produced a number
in its destination that has
Even parity

Question No:305

(Marks:1)

Vu-Topper RM

In which of the following addressing, the memory address is fixed and
is given in the instruction?
Direct

بري صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:306

(Marks:1)

Vu-Topper RM

-----pair of registers used to access memory

DI and SI

Question No:307

(Marks:1)

Vu-Topper RM

Copies the_____in the carry flag Shift Logical Right (SHR) ,

right most bit

Question No:308

(Marks:1)

Vu-Topper RM

REP with _____ will utilize the full processor power to do the scrolling in minimum time.

MOVS

Question No:309

(Marks:1)

Vu-Topper RM

The correlation process from the interrupt number to the interrupt handler uses a table called

interrupt vector table

Question No:310

(Marks:1)

Vu-Topper RM

POP is also known as

destination operand

Question No:311

(Marks:1)

Vu-Topper RM

The parallel port connector is a 25pin connector called

DB-25

Question No:312

(Marks:1)

Vu-Topper RM

The _____port connector is a 25pin connector called DB-25

Parallel

Question No:313

(Marks:1)

Vu-Topper RM

There are just-----block processing instructions in 8088.

5

بري صحبت سے تڻهائي بهتر هے اور تڻهائي سے نيك صحبت بهتر هے

For More Help Contact What's app 03224021365

Question No:314 (Marks:1)

Vu-Topper RM

Interrupts are _____and unpredictable

Asynchronous

Question No:315 (Marks:1)

Vu-Topper RM

Program Control Instructions

cmp ax, 0

Question No:316 (Marks:1)

Vu-Topper RM

A 32bit processor has an accumulator of

32 bits

Question No:317 (Marks:1)

Vu-Topper RM

Left shift on hexa-decimal number 9C40 ans is

0x13880

Question No:318 (Marks:1)

Vu-Topper RM

Each entry of the table is of _____bytes

Four

Question No:319 (Marks:1)

Vu-Topper RM

Video Graphics Adapter

VGA

Question No:320 (Marks:1)

Vu-Topper RM

-----can store 16 bits

DW

Question No:321 (Marks:1)

Vu-Topper RM

Another important role of the stack is in the creation of _____that are only needed while the subroutine is in execution and not afterwards.

local variables

بري صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:322

(Marks:1)

Vu-Topper RM

To multiply a number in register by 2 the number is
shift left one bit

Question No:323

(Marks:1)

Vu-Topper RM

Which assembly instruction is used to ADD data at address 1200 to bx register
add bx, [1200]

Question No:324

(Marks:1)

Vu-Topper RM

Chose the correct option from the following addressing modes , from which both register moves into the data segment
base+offset

Question No:325

(Marks:1)

Vu-Topper RM

_____is the part of microprocessor that ménages the execution of instruction
Control unit

Question No:326

(Marks:1)

Vu-Topper RM

In a comparison, if the both operands are same , the result of subtraction will be zero and the zero flag will be
Set

Question No:327

(Marks:1)

Vu-Topper RM

When SI and DI are used, we name the method
Indexed Addressing

Question No:328

(Marks:1)

Vu-Topper RM

Whenever we need access to a memory location whose address is not know until run time we use
index register

بري صحبت سے تتهائي بهتر هے اور تتهائي سے نيك صحبت بهتر هے

For More Help Contact What's app 03224021365

Question No:329

(Marks:1)

Vu-Topper RM

Interrupts are _____ event

Asynchronous

Question No:330

(Marks:1)

Vu-Topper RM

During program execution , if any change in AH or AL is reflected in _____ as Well

AX

Question No:331

(Marks:1)

Vu-Topper RM

Basic function of register is to

Hold operand

Question No:332

(Marks:1)

Vu-Topper RM

Which among the following is the pointer registers?

index pointer and decession pointer

Question No:333

(Marks:1)

Vu-Topper RM

mov [bx], ax moves the two bytes content of AX register to the address contained in BX register in the current

Data segment

Question No:334

(Marks:1)

Vu-Topper RM

If A is subtracted from B and the resulting answer is negative figure it means B is

Small number

Question No:335

(Marks:1)

Vu-Topper RM

_____ jump is not position relative but is absolute

Far

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:336

(Marks:1)

Vu-Topper RM

Which of the following bit that “Shift Logical Right” operation copies in the carry flag?

Left most bit

Question No:337

(Marks:1)

Vu-Topper RM

During the CALL operation, the current value of the _____ is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer.

Instruction pointer

Question No:338

(Marks:1)

Vu-Topper RM

If BL contains 00000101 then after a Single Right Shift, BL will contain;

0000011

Question No:339

(Marks:1)

Vu-Topper RM

Which of the following operation is used to clear any specific bit in binary number?

XOR

Question No:340

(Marks:1)

Vu-Topper RM

OUR computer screen is like a 2-D array having _____ rows and _____ columns.

25, 40

Question No:341

(Marks:1)

Vu-Topper RM

REPE repeats a string instruction while the _____

ZERO

بري صحبت سے تھائی بہتر ہے اور تھائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:342

(Marks:1)

Vu-Topper RM

REP allows the instruction to be repeated__times.

CX

Question No:343

(Marks:1)

Vu-Topper RM

EACH bit of the _____ register conveys a different meaning.

FLAGS

Question No:344

(Marks:1)

Vu-Topper RM

When SI or DI are used, we name the method ____ addressing.

Indexed

Question No:345

(Marks:1)

Vu-Topper RM

A value 0500 is stored in my memory. If we transfer this value to a general-Purpose register. Then it will be shown as

0500

Question No:346

(Marks:1)

Vu-Topper RM

Also observe that with the CALL instruction ____is decremented by two from FFFE to FFEC, and the stack windows shows 0150 at its top.

SP

Question No:347

(Marks:1)

Vu-Topper RM

Software interrupts on the contrary are not generated from outside the processor. They just provide an extended call mechanism. Far call allows us to jump anywhere in the whole megabyte of memory.

Far

Question No:348

(Marks:1)

Vu-Topper RM

In general the memory cell cannot be wider than the width of the

Data bus.

Question No:349

(Marks:1)

Vu-Topper RM

BP the default segment used is

SS

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:350

(Marks:1)

Vu-Topper RM

Code size reduction and improvement in speed were the two reasons for introducing block processing instructions in the _____ processor.

A. 8088

B. 8085

C. 8080

D. iAPX386

Question No:351

(Marks:1)

Vu-Topper RM

The first instruction of "COM" file must be at offset

A. 0x0010

B. 0x0100

C. 0x1000

D. 0x0000

Question No:352

(Marks:1)

Vu-Topper RM

One screen location corresponds to a

A. Byte

B. Word

C. Double type

D. Double word

Question No:353

(Marks:1)

Vu-Topper RM

When an item is pushed on the decrementing stack, the top of the stack is

A. First decremented and then element copied to the stack

B. First incremented and then element copied to the stack

C. decremented after the element copied to the stack

D. incremented after the element copied to the stack

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:354

(Marks:1)

Vu-Topper RM

Each screen location corresponds to a word, the lower byte of this word contains _____.

A. The character code

B. The attribute byte

C. the parameters

D. The dimensions

Question No:355

(Marks:1)

Vu-Topper RM

If ax contains decimal -2 and BX contains decimal 2 then after the execution of

Instruction: CMP AX, BX, JA label

A. Jump will be taken

B. Zero flag will set

C. 2F will contain value -4

D. Jump will not be taken

Question No:356

(Marks:1)

Vu-Topper RM

Only instructions allow moving data from memory to memory.

A. String

B. Word

C. Indirect

D. Stack

Question No:357

(Marks:1)

Vu-Topper RM

mov ax,5 has

A. 1 operand

B. 2 operand

C. 3 operand

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

Question No:358

(Marks:1)

Vu-Topper RM

The physical address of the stack is obtained by

- A. SS:SI combination
- B. SS:SP combination**
- C. ES:BP combination
- D. ES:SP combination

Question No:359

(Marks:1)

Vu-Topper RM

Index registers are used to store _____

- A. Data
- B. Intermediate result
- C. Address**
- D. Both data and addresses

Question No:360

(Marks:1)

Vu-Topper RM

When a 32 bit number is divided by a 16 bit number, the quotient is of

- A. 32 bits
- B. 16 bits**
- C. 8 bits

Question No:361

(Marks:1)

Vu-Topper RM

If the direction of the processing of a string is from higher addresses towards lower addresses, then

- A. ZF is cleared
- B. DF is cleared**
- C. ZF is set

Question No:362

(Marks:1)

Vu-Topper RM

In STOS instruction, the implied source will always be in

- A. AL or AX registers**

بری صحبت سے تنہائی بہتر ہے اور تنہائی سے نیک صحبت بہتر ہے

For More Help Contact What's app 03224021365

- B. DL or DX registers
- C. BL or BX registers
- D. CL or CX registers

Question No:363

(Marks:1)

Vu-Topper RM

When a 32 bit number is divided by a 16 bit number, the quotient will be store in

- A. AX**
- B. BX
- C. CX
- D. DX

Question No:364

(Marks:1)

Vu-Topper RM

“mov byte [num1], 5” is _____ instruction.

- A. Legal**
- B. Illegal
- C. Stack bases
- D. Memory indirect

Question No:365

(Marks:1)

Vu-Topper RM

To transfer control back the RET instruction take

- A. 1 argument
- B. 2 arguments
- C. 3 arguments**
- D. No arguments

Question No:366

(Marks:1)

Vu-Topper RM

The basic function of SCAS instruction is to

- A. Compare**
- B. Scan

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

- C. Sort
- D. Move data

Question No:367

(Marks:1)

Vu-Topper RM

The bits of the _____ work independently and individually.

- A. Index register
- B. Base register
- C. Flags register**
- D. Accumulator

Question No:368

(Marks:1)

Vu-Topper RM

To convert any digit to its ASCII representation

- A. Add 0x30 in the digit**
- B. Subtract 0x30 from the digit
- C. Add 0x61 in the digit
- D. Subtract 0x61 from the digit

Question No:369

(Marks:1)

Vu-Topper RM

JC and JNC test the _____ flag.

- A. Carry**
- B. Parity
- C. Zero
- D. Sign

Question No:370

(Marks:1)

Vu-Topper RM

After the execution of REP instruction CX will be decremented then which of the following flags will be affected?

- A. CF
- B. OF
- C. DF

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

D. No flags will be affected

Question No:371

(Marks:1)

Vu-Topper RM

_____ register holds the address of next instruction is to be executed

- A. Base pointer
- B. Code segment
- C. Source index

D. Program counter

Question No:372

(Marks:1)

Vu-Topper RM

The 8088 processor divides interrupts into _____ classes.

- A. One
- B. Two**
- C. Three
- D. Four

Question No:373

(Marks:1)

Vu-Topper RM

Which of the following directive used to reserve a 8 bit space in the memory holding data?

- A. Db**
- B. dw
- C. dd
- D. dq

**Visit My YouTube Channel
For More Important Notes
Channel Name = #VuTopperRM**

بري صحبت سے تنہائی بہتر ہے اور تنہائی سے نيك صحبت بہتر ہے

For More Help Contact What's app 03224021365

DOWNLOAD MIDTERM

PAST PAPERS BY WAQAR SIDDHU

More in PDF From

VU Answer

Get All Solutions.



The assembly instruction to put the value of AX on the stack is

Answer (Please select your correct option)

☐

PUSH AX

correct

☐

Mov Stack, AX

☐

ADD AX

☐

SEND AX

Made By: Waqar Siddhu



Which bit of the attributes byte represents the red component of foreground color

Answer (Please select your correct option)

☐ 5

☐ 4

☐ 3

☒ 2

correct

Made By: Waqar Siddhu



The clear screen operation initializes whole block of video memory to

Answer (Please select your correct option)

☐

0417

☐

0741

☐

0714

☐

0174

Made By: Waqar Siddhu



In the execution of the instruction "REP STOSB" STOSB will repeat itself

Answer (Please select your correct option)

☐

AX times

☐

BX times

☒

CX times

correct

☐

DX times

Made By: Waqar Siddhu



By executing which of the following instruction, the value in AX register becomes double of previously stored value ?

Answer (Please select your correct option)

☐

Shl AX, 1

correct

☐

Shr AX, 1

☐

NOT AX

☐

AND AX, 0xFFFF

Made By: Waqar Siddhu



In the statement "Num1: dw 5", 5 is stored in memory as (on Intel-based machines) :

Answer (Please select your correct option)

☐ 0005

☐ 0500

☐ 5

☐ 005

correct



In the instruction "CMP AX,BX" the contents of

Answer (Please select your correct option)

☐

AX are changed

☐

BX are changed

☐

CX are changed

☐

Flags register are changed

correct

Made By: Waqar Siddhu



The maximum parameters a subroutine can receive (with the help of registers) are

Answer (Please select your correct option)

☐ 6

☐ 7

correct

☐ 8

☐ 9

Made By: Waqar Siddhu



To convert any digit to its ASCII representation

Answer (Please select your correct option)

☐

Add 0x30 in the digit

correct

☐

Subtract 0x30 from the digit

☐

Add 0x61 in the digit

☐

Subtract 0x61 from the digit

Made By: Waqar Siddhu



When a very large number is divided by very small number so that the quotient is larger than the space provided, this is called

Answer (Please select your correct option)

☐

Divide logical error

☐

Divide overflow error

☐

Divide syntax error

☐

An illegal instruction

correct

Made By: Waqar Siddhu



In the primitive form, the block processing instructions operate on _____ of memory at one time.

Answer (Please select your correct option)

☐

Whole block

☐

A single cell

☐

A single bit

☐

Whole segment

correct

Made By: Waqar Siddhu



In string manipulation whenever an instruction needs a memory destination, which of the following will hold the pointer to it?

Answer (Please select your correct option)

☐ ES:DI

☐ ES:BP

☐ DS:BP

☐ DS:SI

correct

Made By: Waqar Siddhu



In A4FB:4872 Segment:offset pair the physical address is (both segment and offset are in hexadecimal):

Answer (Please select your correct option)

☐ A4FB0

☐ 04872

☒ A9822

correct

☐ A4872

Made By: Waqar Siddhu



Which bit of the Attribute Byte (counting from right to left, starting from 0) represents the BLUE component of foreground color?

Answer (Please select your correct option)

☐ 3

☐ 2

☐ 1

☒ 0

correct

Made By: Waqar Siddhu



Each screen location corresponds to _____ in a video memory.

Answer (Please select your correct option)

☐ One Byte

☐ Two Bytes

correct

☐ Four Bytes

☐ Eight Bytes



When a 16 bit number is divided by 8 bit number, the quotient will be stored in

Answer (Please select your correct option)

☐ AX

☐ AL

correct

☐ AH

☐ DX



The IAPX88 processor supports _____ memory addressing modes.

Answer (Please select your correct option)

☐ 5

☐ 6

☒ 7

correct

☐ 8

Made By: Waqar Siddhu



The operation of _____ is to subtract the source operand from the destination operand.

Answer (Please select your correct option)

☐

ADDX

☐

JNC

☒

CMP

correct

☐

JNZ

Made By: Waqar Siddhu



If the decimal number "35" is shifted by two bits to left, the new value will be _____

Answer (Please select your correct option)

☐ 35

☐ 70

☐ 140

☐ 17

correct



While using STOSW, if DF=0 then

Answer (Please select your correct option)

☐

The value of SI will be incremented by one

☐

The value of SI will be incremented by two

☐

The value of SI will be decremented by one

☐

The value of SI will be decremented by two

correct

Made By: Waqar Siddhu



If a rotate through carry right (RCR) instruction is applied on a 16 bit word, effectively there is

Answer (Please select your correct option)

☐

16-bits rotation

☐

1-bit rotation

☒

17-bits rotation

☐

8-bits rotation

correct

Made By: Waqar Siddhu



The near CALL instruction

Answer (Please select your correct option)

☐

push only segment on the stack

☐

push only offset on the stack

☐

push both segment and offset on the stack

☐

do not push any thing

correct

Made By: Waqar Siddhu



If AX=5, BX=5, CF and ZF are set, AF and DX contains zero then after the execution of instruction "ADC AX, BX", AX will contain the value:

Answer (Please select your correct option)

☐ 5

☐ 10

☒ 11

☐ 12

correct



For the execution of the instruction "DIV BL", the implied dividend will be stored in

Answer (Please select your correct option)

☐ AX

correct

☐ BX

☐ CX

☐ DX



MS DOS uses ____ display mode.

Answer (Please select your correct option)

☐ Character based

correct

☐ Graphics based

☐ Numeric based

☐ Console based



The execution of the instruction "mov word [ES : 160], 0x1230" will print character "0" on the screen , the color of the character "0" will be

Answer (Please select your correct option)

☐

White

☐

Red

☒

Green

correct

☐

Blue

Made By: Waqar Siddhu



In the primitive form, the block processing instructions operate on _____ of memory at one time.

Answer (Please select your correct option)

☐

Whole block

☐

A single cell

correct

☐

A single bit

☐

Whole segment

Made By: Waqar Siddhu



In STOSB instruction, SI is decremented or incremented by

Answer (Please select your correct option)

☐ 3

☐ 2

correct

☐ 1

☐ 4

Made By: Waqar Siddhu



CMPS instruction subtracts the source location to the destination location. Destination location always lies in

Answer (Please select your correct option)

☐ DS:SI

☐ DS:DI

☐ ES:SI

☐ ES:DI

correct

Made By: Waqar Siddhu

If BL contains 00000101 then after a Single Right Shift, BL will contain

Answer (Please select your correct option)

☐

00000011

☐

00000010

correct

☐

10000011

☐

10000010

Made By: Waqar Siddhu



Each screen location corresponds to _____ in a video memory.

Answer (Please select your correct option)

☐

One Byte

☐

Two Bytes

correct

☐

Four Bytes

☐

Eight Bytes

ASCII Stands for _____

Answer (Please select your correct option)

☐

American Standard Code for Information Interchange

correct

☐

American Scientific Code for International Interchange

☐

American Standard Code for Intelligence Interchange

☐

American Scientific Code for Information Interchange

Made By: Waqar Siddhu



What is the ASCII code for 'A'?

Answer (Please select your correct option)

☐ 0x41

correct

☐ 0x40

☐ 0x31

☐ 0x30

386 processor is _____ compatible with iAPX88

Answer (Please select your correct option)

☐

Upward

☐

Downward

☐

Cross

☐

Not

correct

Made By: Waqar Siddhu



If AL contains -15 then what will be the decimal value of AL after "SAR AL, 1" is executed

Answer (Please select your correct option)

☐

-7

correct

☐

7

☐

-8

☐

8



Inside the computer signed numbers are represented in _____ complement notation.

Answer (Please select your correct option)

☐ two's

correct

☐ one's

☐ zero's

☐ fractional



The _____ flag plays the role of borrow during the subtraction operation.

Answer (Please select your correct option)

☐ carry

correct

☐ parity

☐ sign

☐ zero



The foreground and background color parameters are set by

Answer (Please select your correct option)

☐

32 bits

☐

16 bits

☐

8 bits

☐

4 bits

Made By: Waqar Siddhu



Which bit of the Attribute Byte (counting from right to left, starting from 0) represents the GREEN component of background color ?

Answer (Please select your correct option)

☐ 6

☐ 3

☐ 4

☐ 5

correct

Made By: Waqar Siddhu



In string manipulation whenever an instruction needs a memory source, which of the following will hold the pointer to it?

Answer (Please select your correct option)

☐

ES:DI

correct

☐

ES:BP

☐

DS:BP

☐

DS:SI

Made By: Waqar Siddhu



ASCII stands for

Answer (Please select your correct option)

☐

American Standard Code for Information Interchange

corect

☐

American Standard Code for Internet Information

☐

American Standard Code for Interchanging Information

☐

All Standard Codes for Information Interchange

Made By: Waqar Siddhu



The extended ASCII has

Answer (Please select your correct option)



64 characters



128 characters



256 characters



502 characters

correct

Made By: Waqar Siddhu



To reserve 8-bits in memory ____ directive is used.

Answer (Please select your correct option)

☐ db

correct

☐ dw

☐ dn

☐ dd

Made By: Waqar Siddhu



When the RET instruction is executed, it recovers the value of instruction pointer from the

Answer (Please select your correct option)

Register



Stack



Data segment



Code segment



correct

Made By: Waqar Siddhu



The execution of the instruction “mov word [ES : DI], 0x0720”

Answer (Please select your correct option)



will clear next character on screen

correct



will print “20” at top left of the screen



will print “20” on top right of the screen



will move DI at location 0720 on the screen

Made By: Waqar Siddhu



which bit sets the character "blinking" on the screen?

Answer (Please select your correct option)

5



6



7



correct

8



Made By: Waqar Siddhu



To transfer control back, the RET instruction takes

Answer (Please select your correct option)

☐

2 arguments

☐

3 arguments

☐

4 arguments

☐

No arguments

correct

Made By: Waqar Siddhu



When a very large number is divided by very small number so that the quotient is larger than the space provided then



Answer (Please select your correct option)



Quotient will take an extra byte to be stored



Quotient will not be stored anywhere and instruction will be skipped



An interrupt will be generated and the program will be terminated



Quotient will be divided again until it fits the space provided



Each screen location corresponds to a word, the lower byte of this word contains ____

Answer (Please select your correct option)



The character code

correct



The attribute byte



The parameters



The dimensions

Made By: Waqar Siddhu



The execution of the instruction “mov word [ES : 160], 0x1230” will print character “0” on the screen , the color of the character “0” will be

Answer (Please select your correct option)



White



Red



Green

correct



Blue

Made By: Waqar Siddhu



In A4FB:4872 Segment:offset pair the physical address is (both segment and offset are in hexadecimal):

Answer (Please select your correct option)

☐ A4FB0

☐ 04872

☒ A9822

☐ A4872

correct

Made By: Waqar Siddhu



Which of the following flags will be affected by MOVSB?

Answer (Please select your correct option)

☐ DF

☐ ZF

☐ PF

☐ No effect on flags

correct

Made By: Waqar Siddhu



When the operand of DIV instruction is of 16 bits then implied dividend will be stored in _____

Answer (Please select your correct option)



The concatenation of DS and BX



AX register

correct



The concatenation of ES and AX



The concatenation of DX and AX

Made By: Waqar Siddhu



The execution of the instruction “mov word [ES : 160], 0x1230” will print character “0” on the screen , the Background color will be

Answer (Please select your correct option)

Black

☐

Red

☐

Green

☐

Blue

☐

correct

Made By: Waqar Siddhu



When a 16 bit number is divided by 8 bit number, the quotient will be stored in

Answer (Please select your correct option)

☐ AX

☐ AL

correct

☐ AH

☐ DX

Made By: Waqar Siddhu



The bits of the _____ work independently and individually.

Answer (Please select your correct option)

☐ Index register

☐ Base register

☐ Flags register

☐ Accumulator register

correct

Made By: Waqar Siddhu



All possible types of Jumps in assembly language are:

Answer (Please select your correct option)



short, far



short, near



short, near, far



short, near, far, long

correct

Made By: Waqar Siddhu



The first possible segment base value is _____.

Answer (Please select your correct option)

☐ 0000

correct

☐ 0001

☐ 0010

☐ 0100



Code Segment is associated to _____ register by default.

Answer (Please select your correct option)

IP



correct

SS



BP



CX



Made By: Waqar Siddhu



Which bit of the Attribute Byte (Counting from right to left, starting from 0) represents the GREEN component of foreground color?

Answer (Please select your correct option)

☐ 1

☐ 2

☐ 3

☐ 4

Made By: Waqar Siddhu



In instruction ADC the operands can be

Answer (Please select your correct option)

☐

Two registers only

☐

Two register and one memory location

☒

CF and two other operands

correct

☐

ZF and two other operands

Made By: Waqar Siddhu



The attribute byte has

Answer (Please select your correct option)

☐

32 possible background colors

☐

16 possible background colors

☒

8 possible background colors

☐

4 possible background colors

correct

Made By: Waqar Siddhu



Memory cell width refers to _____

Answer (Please select your correct option)

☐ Number of bits in a cell

correct

☐ The maximum value that can be stored in a cell including sign-bit

☐ The maximum value that can be stored in a cell excluding sign-bit

☐ Address of that cell



Which of the following is true about Interrupt Flag:

Answer (Please select your correct option)

☐ It tells whether processor can be interrupted from outside or not

correct

☐ It tells whether processor has interrupted or not

☐ It is cleared when an interrupt occurs from outside

☐ It is set when processor wants to generate an interrupt

Made By: Waqar Siddhu



When the operand of DIV instruction is of 16 bits then implied dividend will be of

Answer (Please select your correct option)

8 bits



correct

16 bits



32 bits



64 bits



Made By: Waqar Siddhu



If AX=5, BX=5, CF and ZF are set, AF and DX contains zero then after the execution of instruction “ ADC AX, BX” , AX will contain the value:

Answer (Please select your correct option)

5

☐

10

☐

11

☒

correct

12

☐

Made By: Waqar Siddhu



MS DOS uses ____ display mode.

Answer (Please select your correct option)

☐ Character based

correct

☐ Graphics based

☐ Numeric based

☐ Console based

Made By: Waqar Siddhu



Each screen location corresponds to a word, the lower byte of this word contains ____

Answer (Please select your correct option)

☐

The character code

correct

☐

The attribute byte

☐

The parameters

☐

The dimensions



Which of the following options contain the set of instructions to open a window to the video memory?

Answer (Please select your correct option)

☐ `mov AX, 0xb008`
`mov ES, AX`

☐ `mov AX, 0xb800`
`mov ES, AX`

☐ `mov AX, 0x8b00`
`mov ES, AX`

☐ `mov AX, 0x800b`
`mov ES, AX`

correct

Made By: Waqar Siddhu



In STOSW instruction, SI is decremented or incremented by

Answer (Please select your correct option)

☐ 1

☒ 2

correct

☐ 3

☐ 4

Made By: Waqar Siddhu



8088 processor divides interrupts into ____ classes.

Answer (Please select your correct option)

2

☐

correct

3

☐

4

☐

5

☐

Made By: Waqar Siddhu



MOVS instruction transfers a byte or word from which of the following source locations.

Answer (Please select your correct option)

☐

ES:DI

☐

DS:DI

☐

ES:SI

☐

DS:SI

correct

Made By: Waqar Siddhu



Which of the following is the pair of register used to access memory in string instructions?

Answer (Please select your correct option)

☐

DI and BP

☐

DI and SI

☐

SI and BP

☐

DS and SI

correct

Made By: Waqar Siddhu



When the instruction "CMP AX,BX" executes and after execution ZF is set, it indicates that

Answer (Please select your correct option)

☐

BX is greater

☒

Both AX and BX are equal

☐

ZF has no relation with CMP

☐

AX is greater

correct

Made By: Waqar Siddhu



Which Register(s) is/are changed during execution of the CALL instruction?

Answer (Please select your correct option)

☐

IP only

☐

IP and SP only

☐

SP only

☐

BP and IP only

correct

Made By: Waqar Siddhu



Which of the following is an illegal instruction ?

Answer (Please select your correct option)

☐

PUSH AX

☐

PUSH BX

☒

PUSH DL

correct

☐

PUSH DX

Made By: Waqar Siddhu



Relating to the stack in 8088 architecture which of the following statement is true ?

Answer (Please select your correct option)

☐

Single word can not be pushed on the stack

correct

☐

Single word can not be popped from the stack

☐

Single byte can not be popped from the stack

☐

Single quad can be popped from the stack

Made By: Waqar Siddhu



ASCII Stands for _____

Answer (Please select your correct option)

☐

American Standard Code for Information Interchange

correct

☐

American Scientific Code for International Interchange

☐

American Standard Code for Intelligence Interchange

☐

American Scientific Code for Information Interchange

Made By: Waqar Siddhu



In assembly the CX register is used normally as a _____ register.

Answer (Please select your correct option)



source



counter



index



pointer

correct

Made By: Waqar Siddhu



How many characters were defined by standard ASCII?

Answer (Please select your correct option)

64

☐

128

☒

correct

256

☐

512

☐

Made By: Waqar Siddhu

CS401 Quiz 1 2021

ORANGE MONKEY TEAM

Opening date 30 and Due date 31

1. The programmable interval Timer (PTI) has input frequency of _____.

1. 19318MHZ ..confirm

2. Which of the following IRQs is used for sound card or network card or the modem?

IRQ 5....confirm

3. Which of the following IRQ is used by the floppy disk drive?

IRQ 6....confrim

4. Which of the following IRQ is used by the parallel port

IRQ 7....confirm

5. Which of the following IRQ is derived by the keyboard when generates an interrupts when a key is passed or released

IRQ 1.....confirm

6. Which of the following IRQ is the cascading interrupt connected to the output of the second 8451 in the machine.

IRQ 2.....confirm

7. Which of the following IRQ is connected to serial port COM 2

IRQ 3.....confirm

8. Which of the following IRQ is connected to serial port COM 1?

IRQ 4....confirm

9. Which of the following instructions is used to read a character from the keyboard port?

in al, 0x60....confirm

ORANGE MONKEY TEAM

10. Which of the following instruction is the key left shift?

`cmp al, 0x2a.....confirm`

11. Which of the following instruction is no, try next comparison?

`jne nextcmp....confirm`

12. Which of the following instruction is used to point es to video memory?

`mov es, ax...confirm`

13. Which of the following instruction is used to send EOI to PIC?

`out 0x20, al.....confrim`

14. Which of the following instruction is used to point as to IVT base

`mov es, ax.....confirm`

15. Which of the following instruction is used to disable interrupts?

`Cli ....confirm`

16. Which of the following instruction is used to store offset at $n*4$?

`mov word [es:9*4], kbisr....confrim`

17. Which of the following instruction is used to store segment at $n*4+2$?

`mov [es:9*4+2], cs....confirm`

18. Which of the following instruction is used to enable interrupts?

`Sti.....confitm`

19. All register and stack of a program are saved in_____.

`Multitasking...confirm from net`

20. _____ is/are the port number(s) for parallel port

`378...confirm`

21. _____ is/are the port number(s) for keyboard

`60 to 64...confrim`

22. _____ is/are the port number(s) for PIC

20 and 21...confirm

23. Which of the following is the ACK pin in DB-25 connector?

10....confirm

24. IBM AT has __ Programmable Interrupt Controllers(PIC)

2.....confirm

25. Which of the following interrupts is of highest priority?

Timer interrupt.....confirm

24. Which of the following is the source register in OUT instruction?

AL or AX.....confrim

25. At the end of servicing an interrupt_____signal is used to inform the Programmable Interrupt Controller (PIC) about it.

EOI.....confirm

26. The space where all registers of a task are stored is called_____

Process Control Block(PCB)

27. INT 08 that is saving and restoring the registers is called_____and the whole event is called a _____

Schedule, context switch.....confirm

28. _____ is used for exporting keyboard service.

INT 16...confirm

29. _____ is used for exporting video services accessible

INT 10...confirm

30._____is used for parallel port services and similarly others through different interrupts.

INT 17...confirm

31. _____just like a pin hole camera

INT 21...confirm

32. Which of the following are the data pins used to take data from the processor to the devices connected through parallel port

2 to 9....confrim

33. Which of the following the ACK, is normally used by the printer to acknowledge the receipt of data and show the willingness to receive more data

10.... confrim

34. Which of the following are ground and must be connected to the external circuit ground to provide.

18 to 25....confrim

35. Which of the following stands for TSR?

Terminate and stay resident

36. The first parallel port LPT1 has ports designated from _ to _____.

378, 37A..confrim

37. In intel 8088, there are a total of _____ possible interrupt vectors in an interrupt Vector Table.

256....confrim

38. A parallel port has _____ views

2.....confirm

39. The parallel port connector is called_____.

DB-25....confrim

40. Which of the following is the highest priority interrupt?

INT 8.....confirm

41. iret returns on the basis of _____ and _____

CS , IP.....confirm

42. When two devices in a system want to use the same interrupt Request (IRQ) line, is referred as IRQ_____.

Conflict.....confirm

43. Which of the following interrupt request(IRQ) is derived by a timer device.

IRQ 0.....confrim

44. Which of the following interrupt request(IRQ) is derived by a keyboard.

IRQ 1....confrim

45. Which of the following interrupt request(IRQ) is connected to serial port COM 2.

IRQ 3...confirm

46. Changing and restoring the state of Central Processing Unit (CPU) is called

Context switching...confirm from net

47. Which of the following is used for exporting parallel port services

INT 17.....confirm

48. The parallel port connector is called _____.

DB-25....confrim

49. Port ____ is used to send an End Of interrupt(EOI) signal to the programmable Interrupt Controller(PIC) after an interrupt has ended?

0x20...confrim

50. Which of the following is the destination register in IN instruction?

AL or AX.....confrim

51. _____ is used in debugging along with the trap flag.

INT 1.....confirm

52. An End of interrupt (EOI) signal is sent by the

Programmable Interrupt controller.....confirm

53. Each of the bits of port _____ corresponds to one of the interrupt request (IRQ) lines

21.....confirm

54. The thread registration code initializes the Process Control Block (PCB) and adds it to the linked list. The _____ then gives it a turn.

Scheduler.....confirm

55. Which of the following is BIOS interrupt providing keyboard services?

Int 0x16...confirm

56. DOS has a single entry point to access all of its services through_____.

INT 21.....confirm

57. The output of programmable interval timer (PIT) is connected to the ____ line of programmable interrupt controller (PIC).

IRQ 0....confirm

58. Which of the following flags can be used in mathematical operations?

Carry flag....confirm

59. Programmable Interrupt Controller (PIC) has _____ input signal(s) and _____ output signal(s).

Eight, one.....confirm

60. Which of the following pins of DB-25 connector are ground?

18-25.....confirm

61. IF the _____ flag is set, then after every instruction, a type 1 interrupt will be generated

Automatically

Trap.....confirm

62. Which of the following interrupts plays the most significant part during single step debugging of a program?

INT 1....confirm

63. Which of the following flags is cleared with the help of stack?

Interrupt flag

64. Programmable interrupt controller has _____ ports 20 and 21.

2.....confrim

65. PCB stands for _____.

Process Control Block.....confirm

66. Which of the following is correct regarding the input frequency of Programmable interval Timer (PIT)?

It is fixed.....confirm

67. Which of the following is the order of pushing the contents on stack during the execution of INT instruction?

FLAGS CS and then IP....confirm

68. _____ is a special type of interrupt that returns to the same instruction instead of the next instruction

Divide overflow interrupt....confrim

69. Which of the following instruction is used to disable all interrupts during the execution of a program?

cli...confirm

70. The offset address of an interrupt service routine "n" at _____.

nx4.....confirm

71. The offset address of the segment will be at_____.

nx4+2.....confirm

72. In intel 8088 the interrupt vector table occupies _____ of memory

4 bytes

73. _____ is a special type of interrupt that returns to the same instruction instead of the next instruction.

Divide overflow interrupt.....confirm

74. Which of the following interrupts is used for saving and restoring the register?

INT 08.....confirm

75. Which of the following instruction is used to return from an interrupt service routine?

IRET.....confirm

76. Which of the following is the order of pushing the contents on stack during the execution of IRET pops instruction?

IP, CS, and then FLAGS.....confirm

77. Which of the following is the priority of interrupt Request(IRQ) 0?

Highest.....confirm

78. The number of _____ in a cell is called the cell width

Bits....confirm

79. iAPXBB architecture consists of

14 register....confirm

80. Synchronization between the processor and the memory for read and write operations is done by the_____.

Control bus....confirm

90. _____ transfer the word at the top of the stock (pointed to by SP) to the destination operand and increments SP by two.

POP....confirm

91. In extended multiplication, we store the multiplicand in _____ bits, and the result is stored in _____ bits

32, 32.....confirm

92. Suppose AX=5, BX=5, DX=0, CF=1, ZF=1, What will be the final value in AX register after the execution of ADC, AX, BX?

10....confirm

93. Shift logical left (SHL) moves of the bits one position to the _____ and inserts to zero from the _____.

Right, left...confirm

94. In MUL instruction, if the source operand is a byte, then it is multiplied with _____ register.

Al.....confirm

95. When the operand of DIV instruction is in 16 bits, then the implied dividend will be in

DX:AX.....confirm

96. We can convert a digit into its ASCII representation by adding _____ to it.

0x30.....confirm

97. Shifting the integer 5 left by 1 bit results in

10....confirm from net

98. Intel 8085 can access up to _____ of memory, whereas intel 8088 can access up to _____ of memory

64KB , 1MB

99. The ordering of a program's instruction is ensured by the _____ register

Program counter....confirm

100. When the RET instruction is executed, it recovers the value of the _____ from the stack.

Instruction pointer.....confrim

101. If AND operation is performed between AX and BX, then how many AND operations will be performed in total?

16.....confirm

102. Keeping in view the downward compatability between the two systems, the codes written for the intel 8088 are _____ on the intel 386 processor.

Valid...confirm

103. SP is associated (by default) with _____.

SS....confrim

104. IP is associated (by default) with _____.

CS....confirm

105. ASCII stands for _____.

American Standard Code for information interchange

106. _____ is used for permanent diversion.

Jump.....confrim

107. Which of the following uses data segment by default?

Base pointer....confirm

108. Which of the following is the number of operands for ADC (ADD with carry) instruction

3

109. Which of the following addressing scheme has been used in the instruction MOV[BX],AX?

Base register indirect.....confrom from net

110. Suppose AX=5, BX=5, DX=0, CF=1, ZF=1 and AF=1. What will be the final value in AX register after the execution of ADC AX, BX?

10... not sure

111. Suppose BX=0x0120, CS=0x1000, and the memory under consideration is [CX:BX+0x0880]. Which of the following will be the physical address?

11A00

112. _____ transfer the word at the top of the stack (pointed to by SP) to the destination operand, and increment SP by two

POP

113. In extended multiplication, we store the multiplied in _____ bits, and the result is stored in _____ bits.

32, 32.....confirm

114. The number of ____ in a cell is called the cell width?

Bits

115. iAPx88 architecture consist of

14 registers

116. Synchronization between the processor and the memory for read and write operation is done by the _____.

Control bus.....confirm

117. What does the PUSH operation do

It copies the operand on the stack.....confirm

118. Intel 8085 can access up to _ of memory, whereas intel 8088 can access up to ____ of memory.

64KB , 1MB ...confirm

119. The basic function of register is to hold _____.

Operand...confirm

120. Which of the following bus is used to inform the memory that whether process wants to read data or write data?

Control bus....confirm

121. In which of the following addressing, both register are not constant?

Base + index

122. mov [1234], ax is an example of _____addressing.

Direct....confirm

123. What the following piece of code does?

Shl word[multiplicand],1

Rcl word[multiplicand +2], 1

Extended shift left.....confirm

124. The correct instruction to subtract with borrow is _____

SBB...confirm

125. The correct instruction to subtract without borrow is _____.

SUB...confirm

126. Which of the following string instructions is generally used in a loop instead of REP prefix?

LODs...confirm

127. With the execution of Call instruction, the value of _____ is decremented by 2

SP...confirm

128. iAPX88 stands for _____.

Intel Advanced Processor Extension 88

129. In far jump ____.

Both offset and segment are given

130. A value of 0500 is stored in memory. If we transfer this value to a general-purpose register, then it will be shown as

0005....confirm

131. Which of the following operations is used to set any specific bit in a binary number?

AND....confirm from net

132. The multiplier is stored in _____ bits in Extended Multiplication

32...confirm

133. After the execution of Shift Arithmetic Right instruction, the most significant bit _____.

Is replaced by a 1

134. In multiplication algorithm, we take the first digit of the multiplier and multiply it with the _____.

Multiplicand....confirm

135. During program execution, if any change occurs in AH or AL is also reflected in _____.

AX.....confrim

136. Which of the following jump does not depend on flogs register?

JCXZ...confirm

137. RET instruction takes _____ argument(s)

0 ...confirm

138. Shift logical Right(SHR) moves all the bits one position to the _____ and inserts a zero from the _____.

Left, right....confrim

139. Shift logical LEft(SHL) moves all the bits one position to the _____ and inserts a zero from the _____.

Rigt, left.....confrim

139. All mathematical and logical operations are performed on the _____.

Accumulator....confrim

140. STI stands for?

Set the interrupt flag....confirm

141. What is the purpose of MOV AX, 0xFFFF?

Mov AX into 0xFFFF.....confirm

142. An important role of the stack is in the creation of _____ variables that are only needed while the subroutine is in execution and not afterwards.

Local...confirm

143. _____ decrements SP(the stack pointer) by two, and then transfers a word from the source operation the top of the stack

PUSH

144. Which of the following cannot be pushed on the stack?

1 byte..confirm

145. The maximum parameters a subroutine can receive are _____ when all the general purpose register are used

7.....confirm

146. The _____ flag has a special role in debugging.

Trap...confirm

147. In the instruction mov word [es:0] 0x1230, 12 refers to ____ color on ____ background

Green, blue.....confirm

148. In the instruction mov word [es:0] 0x0741, 12 refers to ____ color on ____ background

White, black....confirm

149. In the instruction mov word [es:0] 0xb800, 12 refers to ____ color on ____ background

White,black...confirm

150. Which of the following is a program control instruction?

Jne 0534....confirm

151. Which of the following instruction allows memory to memory movement of data?

MOVS....confirm

152. _____ can also be used as a masking operation to invert selective bits

XOR...confirm

153. Code size reduction and improvement in speed were the two reasons for introducing block processing instructions in the _____ processor

8088....confrim

154. In assembly language, the first executable instruction of the program should be placed at the offset.

0x0100....confrim

155. BH register is a (an) _____ bit register.

8...confirm

156. Which of the following instructions allows code reusability in 8088?

Call...confrim

157. In ADC instruction, there are _____.

Two operands and the CF....confriim

158. $\frac{1}{4}$ can be represented by _____ as a fractional binary number.

0.01...confirm

159. _____ subtracts one from the operand

DEC

160. Stack is a data structure that behaves in _____ manner.

Both LIFO and FIFO...confirm

161. How logical errors are different from syntax error?

Identifying both types of errors is the responsibility of assembler....confrim

162. A symbol associated to a point in the program is called a_____.

Label...confirm

163. IN 8088, there is a _____ stack.

Decrementing

164. CX register is mostly used as _____ register.

Counter...confirm

165. There are__byte(S) for each character on the screen.

1...confirm

166. During program execution if any change occurs in AH or AL is also reflected in _____.

AX...confirm

167. Whenever on instruction needs a memory source _____ holds the pointer to it.

DS:SI..confirm

168. How jmp is different from jnz?

Jnz permanently diverts the program flow, but jmp does not..confirm

169. What makes CMP different from SUB?

CMP is a non destructive subtraction that does not affect any flag. However SUB is a destructive subtraction

170. The ordering of a program's instructions is ensured by the _____ register.

Program counter...confirm

171. When an element is pushed on the stack, Sp is decremented by _____.

Two..confirm

172. Which of the following bus is used to inform the memory that whether processor wants to read data or write data.

Control bus..confirm

173. In _____ operation a zero is inserted from the left and every bit moves one position to the right, the right most bit is allocated into the carry flag.

Shift logical right....confirm

174. In iAPX88 _____ flag is specially related to the string instruction.

Direction...confirm

175. To convert the case of a character, we add or subtract _____ from the ASCII code.

0x20...confirm from net

176. Which of the following flags sets when a larger number is subtracted from a smaller number.

CF...confirm

177. Which of the following is an illegal assembly instruction?

Mov [num1],[num2]....confirm

178. mov ax,[num1] is an example of _bit move

8.

179. Call instruction takes a _____ as an argument

Label....confirm

180. Which of the following is the most illegal instruction?

Mov ax, [num1]

181 When the control is transferred to a subroutine?

After the arguments are pushed on the stack..confirm

182. CS and CP are both _____ bit register

16...confirm

183. A decrementing stack moves from _____ to _____ addresses as elements are added in it.

Higher, lower...confirm

184. The most convenient place to store local variables is _____.

Stack...confirm

185. ____ can be used to check whether particular bits of a number are set or not.

AND...confirm

186. The segment offset pair is called a/an _____ address.

Logical...confirm

7 – Blinking of foreground character

6 – Red component of background color

5 – Green component of background color

4 – Blue component of background color

3 – Intensity component of foreground color

2 – Red component of foreground color

1 – Green component of foreground color

0 – Blue component of foreground color

187. The _____ flag has a special role in debugging.

Trap...confirm

188. The extra bit produced as a result of an arithmetic operation that does not fit in the target register is stored in.

Carry flag..confrim

189. Cell width refers to the total number of bits in a memory cell while the total number of cells is called the _____.

Cell width...confirm

190. Register are normally part of_____.

CPU...confirm

191. When the relative address stored with the instruction is in 16 bits as in the last example the jump is called a _____ jump.

Near..confrim

192. If the offset is stored in a single byte as in 75F2 with the opcode 75 and operand F2, the jump is called a _____ jump.

Short...confirm

193. How jump is different from jnz?

Jump is an unconditional jump, however jnz is a conditional jump...confirm

194. A 16-bit processor has an accumulator of _____.

32 bits...confirm

195. REPE and REPNE prefixes are only meaningful with _____.

CMPS or SCAS...both are confirm

196. Which of the following bit is dropped into the carry flag after the execution of 'Shift Logical Right' operation

Right most bit...confirm

197. In _____ operation, the carry flag is inserted from the right causing every bit to move one location to its left. The dropped bit from the left goes into the carry flag.

Rotate through the Carry left...confirm

198. In _____ operation, the output is 1 only if both inputs are 1

AND...confirm

199. In Shift Logical Left (SHL), the _____ bit is dropped into the carry flag.

Most significant..confirm

200. In STOS, the implied source operand always resides in _____.

AL,AX.....confirm

201. The maximum amount of memory accessible using 8085 processor is _____.

64KB

202. sub sp,2. The above instruction is used to perform a _____ operation on stack.

PUSH..confirm

203. In direct addressing the memory address given in the instruction is _____.

Fixed....confirm

204. Which of the following are the basic bitwise operations?

AND, OR, XOR and NOT....confirm

205. Which of the following is also called intra-segment call?

Far call....confrim

206. A/An ____ is an area of memory that holds all local variables and parameters.

Stack...confirm

207. How many are the functions of a register?

Four...confirm

208. The shift logical right operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag.

True...confirm

209. The shift logical left zero bit is inserted from the right and every bit moves one position to its left with the most significant bit dropping into the carry flag.

Shift Logical Left (SHL) / Shift Arithmetic Left (SAL)

210. Shift arithmetic left is just another name for shift logical left.

True

211. A signed number holds the sign in its most significant bit. If this bit was one a logical right shifting will change the sign of this number because of insertion of a zero from the left.

Shift arithmetic Right(SAR)...confirm

212. The operation of shift arithmetic right is therefore to shift every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

True..confirm

213. In the rotate right operation every bit moves one position to the right and the bit dropped from the right is inserted at the left. This bit is also copied into the carry flag.

Rotate Right (ROR)...true confirm

214. In the operation of rotate left instruction, the most significant bit is copied to the carry flag and is inserted from the right, causing every bit to move one position to the left

Rotate Left(ROL)...confirm

215. In the rotate through carry right instruction, the carry flag is inserted from the left, every bit moves one position to the right, and the right most bit is dropped in the carry flag.

Rotate Through Carry Right(RCR)...confirm

216. The exact opposite of rotate through carry right instruction is the rotate through carry left instruction. In its operation the carry flag is inserted from the right causing every bit to move one location to its left and the most significant bit occupying the carry flag

Rotate THorough Carry Left...confirm

217. Register are storage cells _____.

Inside the processor...confirm

218. Which of the following is the meaning of partial pivoting while employing the row transformations?

Making the largest element as pivot.....confrim

219. In case of DIV BX instruction, the quotient is stored in _____ register

AX...confirm

220. The base pointer accesses local variables using _____ offset

Negative....confrim

221. The important thing to observe in the ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers (30-39).

True

222. Which of the following registers hold the address of data in intel 8088?

BX BP ST and ES...confirm from net

223. Which of the following is the general form of addressing?

Base+index+offset...confirm

224. All mathematical and logical operations are performed on its.

Arithmetic Logic Unit(ALU)....confrim from net

225. Call instruction changes the values of _____ and _____ registers.

IP,SP

226. Which of the following keywords is used to define two bytes in memory?

DW....confrim

227. The stack pointer marks the _____ of stack.

Top...confirm

228. The extra bit produced as a result of an arithmetic operation that does not fit in the target register is stored in.

Carry flag

229. Conditional jump can only be _____.

Short...confirm from net

230. Which of the following is the correct syntax for 'OR operation in assembly language?

Or ax,bx and or bye[mem] 5...confirm

231. . Which of the following is the correct syntax for 'AND operation in assembly language?

And ax, bx and 'and byte [mem] ,5..confrim

232. . Which of the following is the correct syntax for 'XOR operation in assembly language?

Xor ax,bx and xor byte [mem], 5...confirm

233. . Which of the following is the correct syntax for 'NOT operation in assembly language?

Not ax, and not byte[mem]

234. _____ jump is absolute and not position relative.

Far....confirm

235. Constant can never be used as _____.

Destination...confirm

236. Call instruction takes _____ argumetns

1...confirm

237. Program consists of ____ logical parts

Two....confrim

238. All the addressing mechanism in iApX88 return a number called the _____ address

Effective....confirm

239. the physical memory address of 1234:5678 segment-offset pair is

179B8

240. Which of the following gives the more logical view of the storage medium?

DOS

241.If AX=00ff then which of the following instruction can be used to change the value of AX to FF00.

AND AX,FF00...confrim

1. An 8 x 16 font is stored in _____ bytes.

16

2. In DOS input buffer, the number of characters actually read on return is stored in _____ byte.

2nd

3. Which of the following interrupts is Non maskable interrupt?

INT 2

4. In STOSW instruction, when DF is clear, DI is _____

Incremented by 2

5. Which bit of the attributes byte represents the red component of foreground color?

2

6. SP is associated with _____, by default.

SS

7. When an element is pushed on the stack SP is decremented by _____.

2

8. The other directive is "define word" or "dw" with the same syntax as "db" but reserving a whole word of _____ bits instead of a byte.

16

9. The 8088 processor divides interrupts into _____ classes.

Four

10. Physical memory address is of

32 Bits

11. Which of the following string instruction is generally used in a loop instead of REP prefix?

LODS

12. "mov [bp], al" moves the one byte contents of the AL register to the address contained in BP register in the current _____.

Data Segment

13. Number of operands of ADC (add with carry) register are:

3

14. There are _____ registers in iAPX88 architecture that can hold address of data.

4

15. In branching mechanism, the assembler

Finds the Branch offset and replaces the Branch target with it

16. The purpose of MOVS instruction is:

Move register to register

17. ASCII table is the contiguous arrangement of the upercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers _____.

31-41

18. In a comparison, if the both operands are same. The result of subtraction will be zero and the zero flag will be _____.

Set

19. Which of the following is a non-destructive AND operation?

OR

20. mov [1234], ax is an example of

base+index addressing

21. DX plays an important role in arithmetic _____.

multiplication

22. Which of the following Move generates error in .com file?

Fetal move

23. Which of the following register is used to hold address of the next instruction to be executed?

Program counter

24. _____ instruction makes the code reusable.

CALL

25. How many characters were defined by standard ASCII?

128

26. The top of stack is contained in _____ register.

BP

27. Sending the appropriate signals on the control bus to the memory is the responsibility of _____.

Processor

28. The execution of the instruction "mov word [ES : 160], 0x1230" will print a character on the screen at:

first column of second row

29. In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag?

ROR(Rotate right)

30. CX register is mostly used as a

counter register

31. If AX=5, BX=5, CF and ZF are set, AF and DX contains zero then after the execution of instruction "ADC AX, BX", AX will contain the value _____.

11

32. MOV [BX+SI+300], AX is a _____ addressing mode instruction.

Base + Index

33. In general, width of a memory cell cannot be greater than the width of _____.

I/O Bus

34. All mathematical and logical operations are performed on the _____.

Arithmetic Logic Unit

35. _____ is a special instruction that load a segment register and a general purpose register from a memory locations.

CALL

36. Group of bits processor uses to inform memory which element to read/write is collectively known as

Control bus

37. Which of the following operation is used to clear any specific bit in a binary number?

AND

38. Which part of this (000000000000 B80500) encoded instruction is an offset?

0500

39. SCAS compares a source byte or word in register AL or AX with the _____ string element addressed by ES: DI and updates the flags.

Destination...confirm

40. What does the given instruction do?
SHR DL, 1

Shift and store the One bit in DL

41. _____ is set only when the last mathematical or logical operation produces a zero in its destination.

Zero flag

42. The extension of assembly language file is

.asm

43. Stack is a data structure that behaves in a first in last _____ manner.

Out

44. The instruction, MOV AX, 0005H belongs to the address mode of _____ .

Immediate

45. This jump is taken if the last arithmetic operation produced a positive number in its destination.

JNS

46. BH register is a _____ bit register.

8

47. The jump command that does not depend on FLAG register is

JCXZ

48. _____ is the basic means of sorting temporary data on the stack.

PUSH

49. Which of the following addressing scheme has been used in the instruction MOV [BX], AX?

Base Register Indirect

50. _____ and _____ are taken if the last arithmetic operation produced a number in its destination that has odd parity.

JNP, JPO

51. In case of 32-bit processor, the size of an accumulator register will be _____ bits.

32

52. The interrupt call loads new values in _____ segment.

CS

53. The jump is taken if the last arithmetic operation did not changed the sign unexpectedly.

JNO

54. This jump is taken if the last arithmetic operation generated a carry or required a borrow. After a CMP it is taken if the unsigned destination is smaller than the unsigned source.

JC JB JNAE

55. This jump is taken if the last arithmetic operation did not generated a carry or required a borrow. After a CMP it is taken if the unsigned destination is larger or equal to the unsigned source.

JNC, JNB, JAE

56. This jump is taken if the last arithmetic operation produced a zero in its destination. After a CMP it is taken if both operands were equal.

JE ,JZ

57. This jump is taken if the last arithmetic operation did not produce a zero in its destination. After a CMP it is taken if both operands were different.

JNE ,JNZ

58. This jump is taken after a CMP if the unsigned destination is larger than the unsigned source.

JA, JNBE

59. This jump is taken after a CMP if the unsigned destination is smaller than or equal to the unsigned source.

JNA, JBE

60. This jump is taken after a CMP if the signed destination is smaller than the signed source.

JL ,JNGE

61. This jump is taken after a CMP if the signed destination is larger than or equal to the signed source.

JNL ,JGE

62. This jump is taken after a CMP if the signed destination is larger than the signed source.

JG,JNLE

63. This jump is taken after a CMP if the signed destination is smaller than or equal to the signed source.

JNG ,JLE

64. This jump is taken if the last arithmetic operation did not change the sign unexpectedly.

JNO

65. This jump is taken if the last arithmetic operation produced a negative number in its destination.

JS

66. This jump is taken if the last arithmetic operation produced a positive number in its destination.

JNS

67. This jump is taken if the last arithmetic operation produced a number in its destination that has even parity.

JP JPE

68. This jump is taken if the last arithmetic operation produced a number in its destination that has odd parity.

JNP JPO

69. This jump is taken if the CX register is zero.

JCXZ

70. The jump is taken if the last arithmetic operation changed the sign unexpectedly.

JO....confirm

71. All the addressing modes return the number after calculation, this number is known as _____ ?

Effective Address

72. _____ is a temporary storage places inside the processor.

Register

73. There are just _____ block processing instructions in 8088.

5

74. MUL (multiply) instruction performs an unsigned multiplication of the source operand and the _____ .

Accumulator

75. When a 32 bit number is divided by a 16 bit number, the quotient is of _____.

16 bits

76. In the instruction MOV AX, 5 the number of operands are _____.

1

77. In DOS input buffer, number of characters actually read on return is stored in _____.

2nd byte

78. Which of the following is used to terminate a string in assembly language?

\$

79. In multitasking, which of the following interrupts is used as scheduler during context switching?

INT 8

80. We can set the current file position in DOS using service number _____.

0x42

81. In the context of video services, if we want to set cursor position which of the following will be the value of AH?

AH = 02h

82. Which of the following registers hold the page number for using the write string service of INT 10?

BH

83. _____ is the highest priority interrupt in interrupt controller.

IRQ 0

84. The 'program segment prefix' for com files is of size:

64 bytes

85. "INT 13-BIOS disk services" generally uses which register to return the 'error code'?

AH

86. Which of the following services of INT 10 is used for writing the string?

0x13

87. Which of the following INT 21 service is used to read character from standard input?

01

88. Far jump is not position relative but is absolute. Both segment and offset must be given to a far jump. The previous two jumps were used to jump within a segment.

True....confirm

mov [1234], ax is an example of _____ addressing.

Direct

_____ jump is absolute and not position relative.

FAR

In Extended Multiplication, we store the Multiplication in _____ bits , and the result is stored in _____ bits

32 , 32...confirm

SHL and SAL are same

True

Simple CMP instruction uses _____ operation

Subtraction

**The jump is taken if the last arithmetic operation changed the sign unexpectedly.
JO**

The Execution of the instruction “mov word [ES:160]” will print a character on the screen at the _____

First, second

The segment offset pair is called a/an _____ address.

Logical

If AX=00FF, then which of the following instruction can be used to change the value of AX to FF00

AND AX,FF00

Total NUMBER of reserved interrupts on the intel 8088 are _____

8

A/An ____ is an area of memory that holds all local variables and parameters.

Stack

_____ is used for temporary diversion.

CALL

The swap flag can be stored in _____

A Register

The JUMP is taken if the last arithmetic has produced a positive number in its destination

JNP

OUR computer screen is like a 2-D array having _____ rows and _____ columns.

25, 40

The correlation process from the interrupt a number to the interrupt handler uses a table is called.

Interrupt Vector table

The 8088 processor divides interrupts into _____ classes.

Two

_____ function decrements SP (the STACK pointer) by, Two, and then transfer a word from the source operand to the top of the stack now pointed to by SP.

PUSH

REPE repeats a string instruction while the _____

ZERO

Which of the following is the extension of object file?

.exe

Code size reduction and improvement in speed were the two reasons for introducing block processing instruction in the _____ processor.

8088

REP allows the instruction to be repeated _____ times.

CX

EACH bit of the _____ register conveys a different meaning.

FLAGS

Scrolling is the process of the moving one or more lines towards this top or bottom of the screen, and the new line that appears on the top or bottom is _____

cleared

The most convenient place to store local variables is _____

Stack

When SI or DI are used, we name the method _____ addressing.

indexed

A value 0500 is stored in my memory. If we transfer this value to a general-Purpose register. Then it will be shown as _____

0500

In iAPX88, when an element is popped from the stack .SP is _____ by 2

DECREMENTED

Which of the following is used to clear the direction flag?

cld

In _____ operation, a zero is inserted from the left and every bit moves one position to the right. The right most bit is dropped into the carry flag.

shift logical right operation

iAPX88 consist of _____ register

14

Also observe that with the CALL instruction _____ is decremented by two from FFFE to FFEC, and the stack windows shows 0150 at its top.

SP

For example the clear screen operation initializes this whole block to

0/0720.

Software interrupts on the contrary are not generated from outside the processor. They just provide an extended call mechanism. Far call allows us to jump anywhere in the whole megabyte of memory.

Far

The _____ and DPL have the same meaning as in data and code descriptors.

P

The maximum parameters a subroutine can receive are _____ when all the general registers are used.

Seven

In general the memory cell cannot be wider than the width of the

data bus.

BP the default segment used is

SS.

IN 8051 by the same manufacturer has an_____ stack

Incrementing

Mov [1234] ax is an example of — addressing.

Direct

The important thing to observe in the ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers

(30-39).

How many are the functions of a register.

Two

B80500, B8 was the opcode and

0500

In the opcode B80500, B8 was the opcode and 0500 was the operand stored immediately

afterwards

Scrolling is the process when all the lines on the screen move one or more lines towards the top or towards the bottom and the new line that appears on the top or the bottom is

cleared.

This precise synchronization between the processor and the memory is the responsibility of the

control bus.

In the _____the carry flag is inserted from the left, every bit moves one position to the right, and the right most bit is dropped in the carry flag

rotate through carry right instruction(RCR),

the interrupt call loads new values in CS, IP, and

FLAGS.

SCAS compares a source byte or word in register AL or AX with the _____ string element addressed by ES:DI and updates the flags.

Destination

How logical errors are different from syntax error _____ is generally used in a loop instead the REP.

Identifying syntax and logical errors is responsibility of assembler and programmer respectively

LODS

The instructions for permanent diversion in intel 8088 is

jmp

In _____ a zero is inserted from the right, and every bit moves one position to its left , the most significant bit drops into the carry flag.

_____ subtracts one from its single operand.

DEC

Which of the following formulaes calculates the desired location on the screen?

$$\text{location} = (\text{hypos} * 80 + \text{epos}) * 2$$

Stack is a data structure that behaves in a first in last _____ manner.

Out

8088 is a _____ processor with its accumulator and all registers of

16 bits

REPE or REPNE are used with the _____ instructions.

SCAS

_____ can be used to check whether particular bits of a number are set or not.

AND

Which of the following flags sets when a larger number is subtracted from a smaller number?

0F

When an element is pushed on the stack, SP is decremented by _____

2

Which of the following operations is used to clear any specific bit in a binary number?

AND

“mov [bx], ax” moves the two byte contents of the AX register to the address contained in the BX register in the current

data segment.

OR operation in assembly

“or ax, bx”

AX and BX. There are _____ AND operations as a result; one for every bit of AX

16

operand of POP is called _____ since data is moving from the stack to the operand.

Destination

8088 is a _____ bit processor with its accumulator and all registers of

16

Whenever an element is pushed on the stack SP is decremented by

Two

The_____ has a special role in debugging

trap flag

The convention to return a value from a subroutine is to use

the AX register

The iAPX88 architecture consists

of 14 registers..

the CALL instruction _____is decremented by two

SP

The P and DPL have the same meaning as in data and code descriptors.

. The maximum parameters a subroutine can receive are _____when all the general registers are used.

Seven

memory cell cannot be wider than the width of

the data bus.

BP is attached to SS by default

In 8051, there is an _____stack

Incrementing

in the ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers

(30-39).

multiply two 32bit numbers and store the answer in a 64bit location.

In the opcode B80500, B8 was the opcode and 0500 was the operand stored immediately afterwards

0500

The first instruction of “COM” file must be at offset

- 0x0010
- **0x0100**
- 0x1000
- 0x0000

The iAP888 architecture consists of _____ register.

- 12
- 14
- **16**
- 18

One screen location corresponds to a

- Byte
- **Word**
- Double type
- Double word

When an item is pushed on the decrementing stack, the top of the stack is

- **First decremented and then element copied to the stack**
- First incremented and then element copied to the stack
- decremented after the element copied to the stack
- incremented after the element copied to the stack

Each screen location corresponds to a word, the lower byte of this word contains _____.

- **The character code**
- The attribute byte
- the parameters
- The dimensions

If ax contains decimal -2 and BX contains decimal 2 then after the execution of

Instruction: CMP AX, BX, JA label

- Jump will be taken
- Zero flag will set

- 2F will contain value -4
- **Jump will not be taken**

Only instructions allow moving data from memory to memory.

- **String**
- Word
- Indirect
- Stack

In a video memory, each screen location corresponds to _____

- One byte
- **Two bytes**
- Four bytes
- Eight bytes

mov ax,5 has

- 1 operand
- **2 operand**
- 3 operand
- 4 operand

The physical address of the stack is obtained by

- SS:SI combination
- **SS:SP combination**
- ES:BP combination
- ES:SP combination

Index registers are used to store _____

- Data
- Intermediate result
- **Address**
- Both data and addresses

When a 32 bit number is divided by a 16 bit number, the quotient is of

- 32 bits
- **16 bits**
- 8 bits
- 4 bits

If the direction of the processing of a string is from higher addresses towards lower addresses then

- ZF is cleared
- **DF is cleared**
- ZF is set

• DF is set
The instruction ADC has _____ Operand(s)

- 0
- 1
- 2
- 3

Which bit of the attribute byte represents the red component of background color?

- 3
- 4
- 5
- 6

In STOS instruction, the implied source will always be in

- **AL or AX registers**
- DL or DX registers
- BL or BX registers
- CL or CX registers

When a 32 bit number is divided by a 16 bit number, the quotient will be store in

- **AX**
- BX
- CX
- DX

“mov byte [num1], 5” is _____ instruction.

- **Legal**
- Illegal
- Stack bases
- Memory indirect

To transfer control back the RET instruction take

- 1 argument
- 2 arguments
- **3 arguments**
- No arguments

The maximum parameters a subroutine can receive (with the help of registers) are

- 6
- 7
- 8
- 9

The basic function of SCAS instruction is to

- **Compare**
- Scan
- Sort
- Move data

The bits of the _____ work independently and individually.

- Index register
- Base register
- **Flags register**
- Accumulator

To convert any digit to its ASCII representation

- **Add 0x30 in the digit**
- Subtract 0x30 from the digit
- Add 0x61 in the digit
- Subtract 0x61 from the digit

Each screen contains location corresponds to a word, the lower byte of this word contains _____

- **The character code**
- The attribute byte
- The parameters
- The dimensions

JC and JNC test the _____ flag.

- **Carry**
- Parity
- Zero
- Sign

After the execution of REP instruction CX will be decremented then which of the following flags will be affected?

- CF
- OF
- DF
- **No flags will be affected**

_____ register holds the address of next instruction is to be executed

- Base pointer
- Code segment
- Source index
- **Program counter**

The clear screen operation initializes whole block of video memory to

- 0417

- 0714
- **0721**
- 0174

The 8088 processor divides interrupts into _____ classes.

- One
- **Two**
- Three
- Four

Which of the following directive used to reserve a 8 bit space in the memory holding data?

- **Db**
- dw
- dd
- dq

1. The base pointer accesses local variables using _____ offsets.
Negative

2. Which of the following describes the purpose of MOVS instruction?

Move memory to memory

3. Which part of this (00000000B80500) encoded instruction is an offset?

0500

4. Stack is a data structure that behaves a first in last _____ manner. Out

5. In the instruction “mov word [es:160], 0x1230”, 30 represents _____ character. 0

6. Multiplying two 4 bit numbers result in a _____ bit number.

8

7. In case of near jump, the relative address is stored in _____ bits.

16

8. _____ instructions have two parameters, one is the general purpose register to be loaded and the other is the memory location from which to load these registers.

LDS

9. Physical memory address is of

20 bit

10. _____ ports which interface the processor to the external world, including keyboards, mice, monitors, disc drives.

Input, output

11. In base+offset addressing, the value contained in the base register is add with offset to get _____.

Effective address

12. In 8051, there is an _____ stack. Incrementing

13. AX register can be divided into _____ and _____ bytes

Lower, higher

14. CLI stands for

Clear the interrupt flag

15. When a 32 bit number is divided by a 16 bit number, the remainder is of

16 bits

16. MUL instruction performs an unsigned multiplication of _____ with the source operand.

Accumulator

17. DW can store _____ bit value in it.

16

18. When the stack pointer, points to the return address? When the bubble sort subroutine is called

19. IAPX88 stands for _____.

Intel Advanced Processor Extensions 88

20. 90 is the op-code of Do nothing

21. When characters are stored in any high level or low level language, the actual thing stored in a byte is their _____.

ASCII code

22. We can convert any digit to _____ by adding 0x30 in the digit.

ASCII

23. A complete _____ is called a pass over the array

Iteration

24. Which of the following is a non-destructive AND operation? Test

25. In _____ operation the carry flag is inserted from the right causing every bit to move one location to its left and the most significant bit occupying the carry flag. Rotate Through Carry Left (RCL)

26. ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers _____.

30-39

27. _____ can also be used as a masking operation to invert selective bits.

XOR

28. BH register is a _____ bit register.

8

29. Which of the following is the renamed version of conditional jump JZ?

JE

30. SP is associated (by default) with _____.

SS

31. The maximum amount of memory accessible using 8085 processor is _____.

64 KB

32. In XOR operation the output is 1 if

Both inputs are different

33. The clear screen operation initializes whole block of video memory to:

0720

34. The 8088 processor divides interrupts into _____ classes.

Two

35. Which of the following directive is used to reserve a 8 bit space in the memory for holding data?

db

36. All mathematical and logical operations are performed on the _____

Accumulator

37. _____ jump is not position relative but is absolute

Far

38. Which of the following bit that “Shift Logical Right” operation copies in the carry flag?

Right most bit

39. Which of the following register is used to hold address of the next instruction to be executed?

Program counter

40. Group of bits processor uses to inform memory which element to read/write is collectively known as

Address bus

41. _____ containing the address of the next instruction to be executed. Instruction pointer (IP)

42. To convert the case of a character, we add or subtract _____ from its ASCII code.

0x20

43. Which of the following instruction is effectively same as to multiply the value of AX by 8?

SHL AX, 8

MUL AX,3

44. _____ interrupts are those which occur side by side with some other activity.

Synchronous

45. During CALL operation, the current value of the _____ is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer.

Instruction pointer

46. In SCAS Example, we use SCASB with _____ and a zero in AL register to find a zero byte in a string

REPNE

47. In interrupt vector table. Introducing a new entry in this mapping table is called _____ an interrupt.

Hooking

48. What does the following instruction do? ADD AX. BX

Add both registers and load value into ax register

49. The process through which the segment register can be explicitly specified as known as

Segment addressing

50. REPE and REPNE prefixes are only meaningful with _____.

CMPS

51. _____ refers to the total number of bits in a memory cell.

Cell width

52. The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

Shift Arithmetic Right (SAR)

53. _____ and _____ cannot be used as 8bit register pairs like AX, BX, CX, and DX.

SI, DI

54. AX and BX both are 16-bit register, if we perform AND operation on these two registers, then how many AND operations will be performed?

16 And operation

55. 8085 can access up to _____ of memory, whereas 8088 can access up to _____ of memory.

64Kb, 1Mb

56. CS and IP are both _____ bit registers.

16

57. In 8080, there is a _____ stack.

Decrementing

58. An important role of the stack is in the creation of ____ variables that are only needed while the subroutine is in execution and not afterward.

Local

59. ____ movement of data is not allowed in assembly language.

Memory-to-Memory

60. With the execution of CALL instruction, the value of ____ is decremented by 2.

SP

61. In interrupt vector table, introducing a new entry in the mapping table is called ____ an interrupt.

Hocking

62. Which of the following is the most illegal instruction? Mov al, ax

63 Motorola follows ____

Big endian

64. Which of the following instruction allows code reusability in 8088?

CALL

65. When the first thing popped off from the stack, the stack would be the return “address” and not the ____

Argument

66. ____ decrements SP (the stack pointer) by two and then transfers a word from source operand to the top of stack now pointed to by SP,

PUSH

67. Which of the following is a Program Control Instruction?

cmp ax,0

68. Logical addressing is a mechanism to access ____ memory.

Physical memory

69. In assembly language “JN2” is used to Jump if the zero flag is not set

70. In segmented memory model, the size of one window is restricted to ____.

64 KB

71. Twenty-bit register is formed by the combination of two ____ bit register.

Sixteen

72. MOV[BX+SI+300],AX is a ____ addressing mode instruction.

Base + index + offset

73. Physical address calculating depends on

Effective address

74. There are ____ registers in iAPX88 architecture that can hold address of data.

4

75. ____ also known as source operand since the data is moving to stack from this operand.

PUSH

76. By default CS is associated with

IP

77. The stack pointer contains the address of the word that is currently on ____.

Top the stack

78. If AX=00FF, then which of the following instruction can be used to change the value of AX to FFOO

ANDAX, FFOO

79. All addressing mechanisms in iAPX88 return a number called ____ address.

Effective

80. In 8088 processor, interrupts are divided into the following classes. Software Interrupts, Hardware Interrupts

81. Which of the following is the interrupt number for NMI? INT 3

82. here are ____ registers in IAPX88 architecture that can hold address of data. 4

83. Use of AND operation to make selective bits zero in its destination operand is known as ____.

Selective Bit Clearing

84. Standard ASCII has ____ characters? 128

85. ____ is used to store both the instructions to be executes by the microprocessor and the data to be used in the computation.

Microprocessor

86. Number of operands of ADC (add with carry) register are: 3

87. DX play an important role in arithmetic

Addition.

88.Stack is a

Data Structure

89. REPE or REPNE are used with the _____instructions

SCAS

90. _____Instruction have two parameters, one is the general purpose register to be loaded and other is the memory location from which to load these registers

LDS

91. Keywords used to define two bytes program

DW

92. The shift logical left operation is the exact_____ of shift logical right

Opposite

94. Sending the appropriate signal on the control bus to the memory is the responsibility of Control Bus

95. A parallel port has _____ views

2

96. The mechanism used to drop carry for making the calculated address valid is known is address wraparound

97. In _____ a zero is inserted form right and every bit moves one position to its left wth most significant but dropping in to carry flag

Both SHL and SAL

98. The reduction in code size and the improvement in speed are the two reasons why block processing instruction were introduced in the _____ Processor

8080

99. Mov ax, [NUM1] is a _____ bit move instruction.

16

100. Which of the following is the interrupt number for debug interrupt

INT 3

101. Each entry of the interrupt vector table is of ____ bytes

4

102. If BL contains 000000101 then after a Singe Right Shift, BL will contain

00000010

103. _____ can be used to check weather particular bit of number are set or not

AND

104. When the relative address stored with the instruction is in 16 bit , the jump is called a _____ jump

Near

105. The Stack of 8088 works on ____ Sized element

Word

105. The interrupt call loads new values in segment

Flag

106. Mov AX, 0XB800, Move ES, AX : this instruction points ES to

Video Base

107. When the operant of DIV instruction is of 16 bits then implied dividend will be of

Bits

8

108. Which bit is attribute but representing the blue component of foreground color

0

109. When the operand of DIV instruction is of 16-bits then implied dividend will be stored in

AX Register

110. Constant can never be used as

destination

111. DB-25 is a _____ Port Connector

Parallel

112. Flag register is a special register in every architecture ,, is as also known as

Program Status Word

113. BP stands for

base pointer

114. Intel follows

little endian

115. Mov [1234].ax is an example of

direct addressing

1. 116. **OR** is used to clear any specific bit in a binary number

117. In general the memory cell cannot be wider than the width of the

data bus.

118. Source operand always resided in

accumulator register

119. _____always resided in accumulator register

source operand

120. INT instruction takes _____argument varying from 0-255.

1 byte

122. Program consists of

logical parts

123. 8088 provides a mechanism for mapping interrupts to interrupt handlers is called h
hooking an interrupt.

124. The routine that executes in response to an INT instruction is called the _____or_____
interrupt service routine (ISR) , the interrupt handler.

125. The push operation copies its operand on the stack , while the____operation makes a copy
from the top of the stack into its operand.

pop

126. ROR : in the rotate right operation every bit moves one position to the right and the bit
dropped from the right is inserted at the left and also copied into the carry flag.

127. The segment, offset pair is called a

logical address

128. the **local variables** and the **parameters** are always stored in

stack segment

129. SCAS compares a source byte or word in register AL or AX with the destination string
element addressed by ES:DI and updates the flags.

130. JNP and JPO is taken if the last arithmetic operation produced a number in its destination that has

odd parity

130. JP and JPE is taken if the last arithmetic operation produced a number in its destination that has

even parity.

131. There are two forms of the DIV instruction.

132. Unconditional jump

always transfer the control

133. The group of bits that the processor uses to inform the memory about which element to read or write is collectively known as the

address bus.

134. ADC has three operands

135. In direct addressing the memory address given in the instruction is

fixed

136. In which of the following addressing, the memory address is fixed and is given in the instruction?

Direct

137. _____ pair of registers used to access memory

DI and SI

138. Total number of cells is called the

depth

139. _____ copies the _____ in the carry flag

Shift Logical Right (SHR) , right most bit

140. REP with _____ will utilize the full processor power to do the scrolling in minimum time.

MOVS

141. The correlation process from the interrupt number to the interrupt handler uses a table called

interrupt vector table

142. POP is also known as

destination operand

143. The **parallel** port connector is a 25pin connector called

DB-25

144. The _____ port connector is a 25pin connector called DB-25

parallel

145. There are just _____ block processing instructions in 8088.

5

146. Interrupts are _____ and unpredictable

asynchronous

147. _____ instruction allows code reusability in 8088

CALL

148. Program Control Instructions

cmp ax, 0

149. In MULTIPLICATION ALGORITHM ,We take the first digit of the multiplier and multiply it with the

multiplicand

150. _____jump is taken if the last arithmetic operation changed the sign unexpectedly.

JO

151. _____is a special instructions

CLI

152. the interrupt call loads new values in

CS

153. A 32bit processor has an accumulator of

32 bits.

154. Left shift on hexa-decimal number 9C40 ans is

0x13880

154. Each entry of the table is of _____bytes

four

155. Video Graphics Adapter

VGA.

156. The instruction “mov [bp], al” moves the one byte content of the AL register to the address contained in the BP register in the current

stack segment.

158. STI stands for

Set Interrupt Flag

159. Flags register is a special register in every architecture, it is also known as

program status word

160. A special register exists in every processor called

the **program counter or the instruction pointer**

161. `mov word [es:160], 0x1230` meaning

green color on blue background.

162. _____ can store 16 bits

DW

163. two variants of STOS are _____ and _____ STOSB, STOSW

164. Another important role of the stack is in the creation of _____ that are only needed while the subroutine is in execution and not afterwards.

local variables

165. In Far jump

both offset and segment are given

167. to multiply a number in register by 2 the number is

shift left one bit

168. In case of downward compatible mechanism, the codes written for 8080 are _____ for 386 processor

Valid

169. Interrupt hooking is the mechanism that is used for mapping interrupt to interrupt handler

170. Which of the following statement is used to clear the value of AX register, xor ax,

0

171. Which assembly instruction is used to ADD data at address 1200 to bx register

add bx, [1200]

172. Chose the correct option from the following addressing modes , from which both register moves into the data segment

base+offset

173. _____operation , a carry flag is inserted from left moving every bit one position to the right, with the right most bit is dropped in carry flag

RCR

175. After the execution of SAR

instruction most significant bit retains its original value

175. _____ is the part of microprocessor that ménages the execution of instruction

Control unit

176. In a comparison, if the both operands are same , the result of subtraction will be zero and the zero flag will be

set

177. When SI and DI are used, we name the method

Indexed Addressing

178. Which combination will you prefer to obtain the physical address of the stack

SS:SP

179. Whenever we need access to a memory location whose address is not known until run time we use

index register

180. Interrupts are _____ event

asynchronous

181. During program execution, if any change in AH or AL is reflected in _____ as well

AX

182. Basic function of register is to

hold operand

183. Which among the following is the pointer registers?

index pointer and decession pointer

184. `mov [bx], ax` moves the two bytes content of AX register to the address contained in BX register in the current

data segment

185. in string manipulation the instruction to clear the direction flag is

CLD

186. If A is subtracted from B and the resulting answer is negative figure it means B is

small number

187. in ____ operation the output is 1 only if both inputs are 1 ?

AND

188. The interrupt call loads new values in CS, IP, and

FLAGS.



CS401 QUIZ 2, 2022

Quiz-2 will be launched on March 02, 2021 that will remain open for 72 hours only. The syllabus for the quiz is Lecture No. 19-39. Lesson 7 to 15

ORANGE MONKEY TEAM

1. There are _____ forms of MOVS instruction.

Two.....confirm

2. The program segment prefix for com files is of _____ size.

256 bytes

3. Which of the following instructions, selects the memory address space?

Mov confirm

4. Which of the following instructions, selects the peripheral address space?

IN.....confirm

5. Which of the following instructions to read and write from the peripheral address space?

IN and OUT.....confirm

6. INT instruction takes a _____ byte(s) argument varying from 0-255.

Single(1).....confirm

7. Which of the following interrupt requests (IRQ) is derived by the keyboard when generates an interrupts when a key is pressed or released.

IRQ 1....confirm

8. Which of the following interrupt requests (IRQ) is the cascading interrupt connected to the output of the second 8451 in the machine?

IRQ 2....confirm

9. Which of the following interrupt requests (IRQ) is connected to serial port COM 2?

IRQ 3.....confirm

10. A FAT32 file system directory entry in DOS consist of how many bytes ?

32....confirm

11. Which of the following interrupt requests (IRQ) is connected to serial port COM 1?

IRQ 4....confirm

12. Which of the following interrupt requests (IRQ) is used by the sound card or network card.

IRQ 5.....confirm

13. Which of the following interrupt requests (IRQ) is used by the floppy disk drive ?

IRQ 6....confirm

14. Which of the following interrupt requests (IRQ) is used by the parallel port?

IRQ 7 confirm

15. _____ register is used for storing the base and limit of GDT.

GDTR.....confirm

16. The first entry of the GDT must always be zero. It is called the _____.

Null descriptor....confirm

17. In intel 8088 there are a total of _____ possible interrupt vectors in an interrupt Vector Table.

256....confirm

18. Which register is generally used to specify the service number of an interrupt?

Ax.....confirm from net

19. In the context of video services, if we want to write string then which of the value will be placed in AH?

13h....confirm

20. In 9pin DB 9 connector, which pin is assigned to Signal Ground?

5....confirm

21. In 9pin DB 9 connector, which pin is assigned to Data terminal?

4....confirm

22. In 9pin DB 9 connector, which pin is assigned to transmitted data?

3.....confirm

23. In 9pin DB 9 connector, which pin is assigned to Received data?

2.....confirm

24. In 9pin DB 9 connector, which pin is assigned to Carrier Detect?

1.....confirm

25. In 9pin DB 9 connector, which pin is assigned to Data Set?

6.....confirm

26. In 9pin DB 9 connector, which pin is assigned to request to send (RTS)?

7.....confirm

27. In 9pin DB 9 connector, which pin is assigned to Clear to send (CTS)?

8.....confirm

28. In 9pin DB 9 connector, which pin is assigned to Ring indicator(RI)?

9.....confirm

29. The first parallel port LPT1 has ports designated from _____ to _____.
378 to 37 A....confirm

30. A parallel port has _____ views

2.....confirm

31. COM2 is connected with

IRQ3....confirm

32. Which of the following service of INT 21 is used to write a string on standard output?

09h.....confirm

33. Which of the following register hold the page number for using the write string service of INT 10?

BH....confirm

34. New devices are allowed to work with the existing operating system with the help of _____.

Device drivers...confirm

35. Which of the following register hold the cursor start and options for using the write string service of INT 10?

CH.....confirm

36. Which of the following register hold the number of lines by which to scroll up for using the write string service of INT 10?

AL....confirm

37. Which of the following register hold the character to display for using the write string service of INT 10?

AL.....confirm

38. Which of the following register hold write mode for using the write string service of INT 10?

AL...confirm

39. The _____ and DPL have the same meaning as in data and code descriptors.

P.....confirm

40. The routine which executes as a result of INT instruction is called _____.

Interrupt service routine.....confirm

41. To declare a character in assembly language, we store its ASCII code in a _____.

Byte....confirm

42. When two devices in a system want to use the same interrupt request (IRQ) line, is referred as IRQ _____.

Conflict....confirm

43. Serial port services are provided by the _____.

BIOS INT 14....confirm

44. During multitasking _____ is used to get control from the program without letting the program know about it.

IRQ 0....confirm

45. Which of the following is BIOS interrupt providing keyboard services?

INT 0x16.....confirm

46. REPNZ repeats the following instruction while the _____.

ZF is not set....confirm

47. Which of the following is the source register in OUT instruction?

AL or AX....confirm

48. At the end of servicing an interrupt _____ signal is used to inform the Programmable interrupt Controller (PIC) about it.

EOI...confirm

49. INT 0x13 service 0x03 is used to _____.

Write disk sector...confirm

50. INT 0x13 service 0x00 is used to _____.

RESET disk system....confirm

51. INT 0x13 service 0x02 is used to _____.

READ sector into memory

52. INT 0x13 service 0x08 is used to _____.

GET drive parameters....confirm

53. INT 0x21 service 0x01 is used to _____.

READ character from standard input, with echo.....confirm

54. INT 0x21 service 0x09 is used to _____.

Write string to standard output...confirm

55. INT 0x21 service 0x0A is used to _____.

Buffered input....confirm

56. In programmable interrupt Controller (PIC) which of the following ports is used to selectively enable or disable interrupts?

21....confirm

57. In multitasking, which of the following interrupts is used scheduler?

INT 08....confirm

58. Which of the following interrupts is used for saving and restoring the registers?

INT 08....confirm

59. Scrolling is the process of moving one or more lines towards the top or bottom of the screen and this is _____.

Cleared.....confirm

60. Software interrupts are called by extended _____ call mechanism

Far

61. The root directory of floppy contains _____ fixed entries

512....confirm

62. Programmable interrupt controller has _____ ports.

Two(20, 21)...confirm

63. The space where all registers of a task are stored is called _____.

Process control block(PCB).....confirm

64. In the context of video services, if we want to scroll the window up which of the following will be the value of AH?

AH=06h....confirm

65. In the context of video services, if we want to scroll the window down which of the following will be the value of AH?

AH=07h....confirm

66. In the context of video services, if we want to set text-mode cursor shape which of the following will be the value of AH?

AH=01h....confirm

67. In the context of video services, if we want to set cursor position which of the following will be the value of AH?

AH=02h....confirm

68. In the context of video services, if we want to write character and attribute at cursor position which of the following will be the value of AH?

AH=09h....confirm

69. In the context of video services, if we want to write character only at cursor position which of the following will be the value of AH?

AH=0Ah....confirm

70. In the context of video services, if we want to write string which of the following will be the value of AH?

AH=13h....confirm

71. INT instruction takes a _____ byte(s) argument varying from 0-255.

1....confirm

72. Which of the following is the number of pin(s) outside the processor for generating interrupts by the external hardware?

1.....confirm

73. Each entry of the interrupt vector table is of _____ byte(s).

4....confirm

74. Which of the following is the priority of interrupt request (IRQ) 0?

Highest...confirm

75. The interrupt call loads new values in CS, IP and _____.

FLAgS....confirm

76. REP allows the instruction to be repeated _____ times.

CX ...confirm

77. Which of the following interrupt Requests (IRQ) is derived by a timer device?

IRQ 0 confirm

78. An END of interrupt (EOI) signal is sent by the _____

Programmable interrupt controller.....confirm

79. Which of the following INT21 service is used to read character from standard input?

AH= 01h.....confirm

80. Which of the following INT21 service is used to write string to standard output?

AH=09h.....confirm

81. Which of the following INT21 service is used to buffered Input?

AH=0Ah....confirm

82. Which of the following instructions, select the peripheral address space?

IN.....confirm

83. Software interrupts are called by extended _____ call mechanism

Far....confirm

84. Which of the following is the interrupt number for debug interrupt?

Int 3.....confirm

85. In intel 8088, the interrupt vector table occupies _____ of memory

4 bytes....confirm

86. The register size of 386 architecture is _____.

32 bit.....confirm

87. Which of the following instructions is used to disable all interrupts during the execution of a program?

cli.....confirm

88. In case of COM file, first command line parameter is stored at _____ offset of program segment prefix?

0x80....confirm

89. _____ is a special type of interrupt that returns to the same instruction instead of the next instruction.

Divide overflow interrupt.....confirm

90. Which of the following is the order of pushing the contents on stack during the execution of INT instruction?

CS, IP and flags.....confirm

91. The output of Programmable interval Timer (PIT) is connected to the _____ line of programmable interrupt controller (PIC).

IRQ 0....confirm

92. Which of the following formulae is used for calculating the segment address of an interrupt service routine “n”?

$N*4+2$ confirm

93. The thread registration code initializes the Process Control Block (PCB) and adds it to the linked list. The _____, then gives it a return.

Scheduler.....confirm

94. Interrupts are _____ and _____

Asynchronous , unpredictable.....confirm

95. The execution of the instruction `mov word {ES:160} 0x1230` will print a character on screen at the _____ column of the _____ row.

First, second....confirm

96. Which of the following formulae calculates the desired location on the screen?

$\text{Location} = (\text{rowno} * 80 + \text{column}) * 2$

97. The clear screen operation initializes the whole block of video memory to _____.

0x0720.....confirm

98. In programmable interrupt controller, which of the following ports is the control port?

Port 20....confirm

99. The convention to return a value from a subroutine is to use the _____ register.

AX.....confirm

100. The parallel port connector is called _____
DB-25.....confirm

101. Which of the following is the number of pin(s) outside the processor for generating interrupts by the external hardware?

1.....confirm

102. Which of the following is the time interval between two timer ticks?

55ms.....confirm

103. Which of the following IRQs is used for sound card or network card ?

IRQ 5.....confirm

104. _____ is used for exporting keyboard services.

INT 11....confirm

105. If the _____ flag is set, then after every instruction, a type 1 interrupt will be generated automatically.

Trap....confirm

106. On executing INT 0x21 service 0x3D, if file is successfully opened then

ZF will contain 0....confirm from net

107. On executing INT 0x21 service 0x3D, if file can't be opened then

ZF will contain 1....confirm from net

108. BPB stands for_____.

Bios parameter block.....confirm

109. Which of the following is the destination register in IN instruction?

AL or AX....confirm

110. Interrupts are _____ events

Synchronous

111. Which of the following is the interrupt number for NON-maskable interrupt(NMI)?

INT 2....confirm

112. The correlation process from the interrupt number to the interrupt handler uses a table called _____.

Interrupt vector table....confirm

113. IN DOS, if file name is more than 8-characters then which flags of attribute byte are set.

System flag and volume label flag

114. While using INT 10 for writing graphic pixel on screen, which of the following registers hold the information about pixel color?

AL....confirm

115. Which of the following services of INT 10 is used to get the video font information?

1130h....confirm

116. In case of COM file, maximum length of parameters passes through command line can be _____.

255 bytes

117. Programmable interrupt Controller (PIC) has _____ input signal(s) and _____ output signal(s).

Eight, one....confirm

118. Which of the following is used to clear the direction flag?

Cld.....confirm

119. Which of the following is used to set direction flag?
Std.....confirm
120. Which of the following is the ACK pin in DB-25 connector?
10.....confirm
121. BIOS video services are broadly classified into _____ categories.
2.....confirm
122. In multitasking environment, which of the following structure is used to maintain the order of active process control blocks (PCB)?
Register.....confirm
123. _____ is a special instruction that loads a general purpose register, and a segment register from two consecutive memory locations.
LES.....confirm
124. Programmable interrupt Controller (PIC) has _____ input signal(s) and _____ output signal(s).
Eight, one....confirm
125. Which of the following interrupts plays the most significant part during single step debugging of a program?
INT 1.....confirm
126. Iret returns on the basis of _____ and _____.
Cs,IP.....confirm
127. Total number of reserved interrupts in Intel 8088 are
255.....confirm
128. The 8088 processor divides interrupts into _____ classes.
Two...confirm
129. As compared to _____, _____ provides more cooked services.
BIOS, DOS.....confirm
130. The IDTR is a _____ bit register similar in structure to the GDTR

48.....confirm

131. INT13-BIOS disk services generally uses which register to return the error flag?

CF.....confirm

132. DOS has a single entry point to access all of its services through _____.

INT 21.....confirm

133. Which of the following flags can be used in mathematical operations?

Carry flag....confirm

134. What is the purpose of the following two instructions? MOV AX, 0XB800
MOV EX,AX

POINTS ES to video base.....confirm

135. Which of the following instruction is used to return from an interrupt service routine?

Iret....confrim

136. Which of the following pins of DB-25 connector are ground?

18-25.....confirm

137. INT13-BIOS disk services generally uses which register to return the error code?

AH.....confirm

138. IBM AT has _____ Programmable interrupt Controllers (PIC)

2....confirm

139. WHICH OF the following interrupts is used to maintain the system time?

INT 8.....confirm

140. Which of the following flags is cleared with the help of stack?

Direction flag.....confirm

141. The register size of 386 architecture is _____.

32 bit.....confirm

142. After the execution of STOSB and STOSW the CX register will be _____
by _____.

Decrement by 1.....confirm

143. Which of the following is the highest priority interrupt?

INT 08.....confirm

144. There is an auto-increment mode when the _____ is _____.

DF, set.....confirm

145. Which of the following services of INT 10 is used for writing the string?

0x13.....confirm

146. Changing and restoring the state of Central Processing Unit (CPU) is called.

Multitasking

147. Which of the following stands for TSR?

Terminate and stay resident....confirm

148. After the execution of STOSB and STOSW, the CX register will be _____
by _____.

Decrement by 1....confirm

149. While writing graphics pixel on the screen, which registers hold the value for setting cursor position?

CX, DX...confirm from net

150. Which of the following BIOS INT provides serial port services.

INT 14.....confirm

151. Which of the following stands for TSR?

Terminate and stay resident.....confirm

152. PCB stands for _____.

Process control block....confirm

153. The bits are sent one by one in a specially formatted fashion on the _____.

Serial port...confirm

154. In assembly language _____ is used to load a character in AL.

Lodsb.....confirm

155. Which of the following services of INT 10 is used to write graphic pixel on the screen?

0x0C.....confirm

156. COM1 is accessible via ports _____ while COM2 is accessible via _____.

358-3FF, 258-2FF.....confirm

157. Port ____ is used to send an END of interrupt (EOI) signal to the programmable interrupt Controller (PIC) after an interrupt has ended.

0x20.....confirm

158. _____ is a special prefix that is used to repeat the block instructions.

REP....confirm

159. COM1 is connected with

IRQ4....confirm

160. BIOS video services are exported via interrupt.

INT 10.....confirm

161. In the context of video services, if we want to set the cursor at top left corner, which of the values will be stored in DH and DL respectively?

00.00.....confirm

162. REPE repeats a string instruction while the _____ flag is set.

Zero...confirm

163. Device drivers can be divided into _____ major categories.

2.....confirm

164. A 32-bit address register can access up to ____ of memory

4GB....confirm

165. _____ is used in debugging along with the trap flag.

INT 1.....confirm

166. Push and pop operations always operate on _____.

Words.....confirm

167. Which of the following instructions is used to read a character from the keyboard port?

In al, 0x60.....confirm

168. REP with MOVS utilizes the _____ power of a processor to do scrolling in minimum time.

Full....confirm

169. The programmable interval timer (PIT) has input frequency of _____.

1.19318 MHZ.....confirm



CS401 Grand Quiz Solution File

The clear screen operation initializes whole block of video memory to:

0720

The 8088 processor divides interrupts into ____ classes.

Two

Which of the following directive is used to reserve a 8 bit space in the memory for holding data?

db

____ jump is not position relative but is absolute

Far

Which of the following bit that "Shift Logical Right" operation copies in the carry flag?

Left most bit

All mathematical and logical operations are performed on the _____

Accumulator

90 is the op-code of

do nothing

XOR can also be used as a _____ to invert selective bits.

Masking position

_____ can also be used as a masking operation to invert selective bits.

XOR

Which of the following register is used to hold address of the next instruction to be executed?

Program counter

Group of bits processor uses to inform memory which element to read/write is collectively known as

Address bus

_____ containing the address of the next instruction to be executed.

Instruction Pointer (IP)

To convert the case of a character, we add or subtract _____ from its ASCII code.

0x20

_____ interrupts are those which occur side by side with some other activity.

Synchronous

During the CALL operation, the current value of the _____ is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer.

Instruction pointer

Which of the following instruction is effectively same as to multiply the value of AX by 8?

MUL AX, 3

Which of the following is a non-destructive AND operation?

Test

In SCAS Example, We use SCASB with _____ and a zero in AL register to find a zero byte in a string.

REPNE

What does the following instruction do?

ADD AX, BX

Add both registers and load value into ax register

In interrupt vector table, introducing a new entry in this mapping table is called _____ an interrupt.

Hooking

_____ refers to the total number of bits in a memory cell.

Cell width

REPE and REPNE prefixes are only meaningful with _____.

CMPS

_____ and _____ cannot be used as 8bit register pairs like AX, BX, CX, and DX.

SI, DI

The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

Shift Arithmetic Right (SAR)

AX and BX both are 16-bit register, if we perform AND operation on these two registers, then how many AND operations will be performed?

16 And operation

_____ Instructions direct the flow of program.

Special

Program consists of..... Logical parts.

Ans: two

INT instruction takes a Byte(s) argument varying from 0255.

Ans: single (1)

8088 is a processor with its accumulator and all register of
Of.....

Ans: 16 bits, 16 bits.

If BL contains 00000101 then after a Single Right Shift, BL will contain;

Ans: 0000011

Which of the following operation is used to clear any specific bit in binary number?

Ans: XOR

8085 can access up to..... of memory, whereas 8088 can access up to.....

Ans : 64kb, 1Mb

Whenever an instruction needs a memory source, ... Holds the pointer to it.

Ans: DS:SI

Total numbers of reserved interrupts by Intel processor are

Ans: 256

The stack 8088 works on sized elements.

Ans: Word

Which one of the following is an illegal instruction?

MOV AX,BX

MOV AX,65

MOV ax,[bx+bp]

MOV BX,10

The iAPX88 processor supports modes of memory access.

Ans: 7

What is the content of stack pointer?

Ans: Address of the top element of the stack

13) iAPX88 Architecture consists of:

Ans: 14 Registers

Simple CMP instruction uses operation.

Ans: Subtraction

Which of the following instructions is used for non-destructive AND operation?

Ans: Test

An important role of stack is in the creation of..... variables that are only needed while the subroutine is in execution and not afterwards.

Ans: Local variables.

In 8088 processor, interrupts are divided into following classes.

Ans: Software interrupts and hardware interrupts

Mov [1234],ax is an example of:

Ans: Direct Addressing

Traditionally all mathematical and logical operations are performed by.....

Ans: Accumulator Register

If we use source index register and destination index register to access the memory known as:

Ans: Index format

Intel follows

Ans: Little endian

BP stands for.....?

Ans: Base Pointer

..... jump is not position relative but is absolute.

Ans: Far jump

Motrola follows

Ans: Big endian

What does the following instruction do

Ans: Add both registers and load value into ax register

All the addressing mechanism in iAPX88 return a number called..... Address.

Ans: Effective

The other directive is "define word" or "dw" with the same syntax

"db" but reserving a whole word of.....bits instead of a byte.

Ans: 16 bits

What does the following instruction do?

Mov ax, 0xFFFF

Ans: Store 0xFFFF into AX

Which of the following register is used to hold address of next instruction to be executed?

Ans: Program Counter

The reduction in code size and the improvement in speed are the two reasons why block processing instructions were introduced in the

..... processor.

Ans: 8088

MOV AL, [NUM1] is a.....bit move instruction

Ans: 8

The interrupt call loads new values in CS,IP and

Ans: Flags

Number of operands of ADC (add with carry) registers are:

Ans: Three (3)

In every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag?

Rotate Right (ROR)

In interrupt vector table, introducing a new entry in this mapping table is called An interrupt.

Ans: Hooking

In case of short jump, the offset is stored in.....

Ans: 1 byte

In segmented memory model, the size of one window is restricted to.....

Ans: 64kb

What does the following instruction do?

MOV [NUM4], AX

Ans: Store ax value in num4

Registers are storage cell :

Ans: Inside the processor

A symbol associated to a point in the program is called a.....

Ans: Label

There are Registers in lapx88 that can hold address of data.

Ans: 4

What does the given instruction do?

TEST BX, [MULTIPLIER]

Ans: Test the right most bit

AX and BX both are 16-bits register, if we perform AND operation on these two registers, then how many AND operations will performed?

Ans: 16 AND operation

"Far" jump is not position relative but is:

Ans: Absolute

The execution of the instruction "mov word[ES:160],0x1230" will print a character on the screen at:

Ans: First column of second row

How many characters were defined by standard ASCII? **Ans: 128**

Unconditional jump:

|

Ans: If the condition is true

BP by default associated with :

Ans: SP

After the execution of SAR instruction

Ans: The msb is replaced by the value of CF

There are Registers in intel 8088 that can hold address of data.

Ans: 4

In SCAS example, we use SCASB with and a zero in AL register to find a zero byte in a string.

Ans: REPNE

What does the following instruction do?

MOV AX,0x4c00

INT0x21

Ans: Terminate the program

..... instructions direct the flow of program.

Ans: program control

The routine which executes as a result of INT instruction is called.....

Ans: Interrupt Service Routine (ISR)

A carry if generated and it dropped without being stored anywhere in the flag then this phenomenon is called:

Ans: Address wraparound

When the subroutine is called?

Ans: when the arguments are pushed on the stack

Which of the following is also called intra-segment call?

Ans: Near calls

In assembly language, the first executable instruction of code should be placed at this offset.....

Ans: 0100

Which of the following string instruction is generally used in a loop instead REP prefix?

Ans: LODS

LES instruction loads..... Register.

Ans: ES

The maximum parameters a subroutine can receive are when all the general registers are used.

Ans: Seven (7)

REPE or REPNE are used with the Instructions.

Ans: SCAS

Which of the following is a special instruction?

Ans: cli; Clear the interrupt flag

..... Can be used to check whether particular bits of a number are set or not.

Ans: AND

XOR can be used as a..... to invert selective bit.

Ans: Masking operators

Stack is :

Ans: a data structure

Twenty-bit register is formed by the combination of two..... bit register.

Ans: 16

STI stands for?

Ans: set the interrupt flag

In Far jump.....

Ans: both offset and segment are given

Magnitude and sign are present in:

Ans: Signed number

REP with ____ will utilize the full processor power to do the scrolling in minimum time

MOVS

Number of operands of ADC (add with carry) register are:

3

Which of the following is a non-destructive AND operation?

MOV AL, [NUM1] is a ____ bit move instruction

8

The Jump command that does not depend on FLAG register is

JNE

Which of the following is a Program Control Instruction?

cmp ax,0

Program consists of ____ logical parts.

Two

8085 is ____ bit microprocessor, whereas 8088 is ____ bit microprocessor.

8 and 16

Extended Shifting Algorithm consists of:

2 instructions

_____ is a temporary storage place inside the processor.

Register

Simple CMP instruction uses ____ operation.

Subtraction

What does the following instruction do?

MOV [NUM4], AX

Store ax value in num4

To access the arguments from the stack, the immediate idea that strikes is to ____ them off the stack.

Pop

_____ function decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.

PUSH

If BL contains 00000101 then after a Single Right Shift, BL will contain

00000011

_____ Instructions direct the flow of program.

Program Control

A parallel port has ____ views.

2

Which of the following operation is used to set any specific bit in a binary number?

XOR

By default, Segment used by the instruction pointer is _____.

Stack Segment

After the execution of STOSWB, the CX will be:

Incremented by 2

DW can store _____ bit value in it.

16

In a comparison, if the both operands are same. The result of subtraction will be zero and the zero flag will be _____.

Set

The other directive is "define word" or "dw" with the same syntax as "db" but reserving a whole word of __ bits instead of a byte.

16

Which of the following are the two variants of STOS instruction?

STOSB and STOSW

Magnitude and sign are present in?

Signed number

With the execution of CALL instruction, the value of _____ is decremented by 2.

SP

When an element is pushed on the stack SP is decremented by _____ as the 8088 stack works on word sized elements.

Two

Which of the following statement is used to clear the value of AX register?

XOR AX, 0

_____ is part of microprocessor that manages the execution of instructions.

Arithmetic/Logic Unit

In assembly language, ISR stands for _____.

Interrupt service routine

The extension of assembly language file is

.asm

The stack of 8088 works on _____ sized elements.

word

A complete _____ is called a pass over the array

iteration

The keyword used for the conditional jumps is _____.

SHORT

In the instruction "mov word [es:160], 0x1230", 30 represents ____ character.

0

The purpose of MOVS instruction is:

Move memory to memory

MOV [BX+SI+300], AX is a _____ addressing mode instruction.

Base + Index + Offset

When the relative address stored with the instruction is in 16-bits, the jump is called a _____ jump.

Near

Logical addressing is a mechanism to access _____ memory.

Physical memory

How stack data structure behaves?

first in last out

MOV AX, 0XB800

MOV ES, AX ; This instruction points ES to_____.

video base

All the addressing modes return the number after calculation, this number is known as_____?

Effective Address

CS and IP are both _____ bit registers.

16

Stack is a data structure that behaves in _____ manner.

LIFO

Which of the following Move generates error in .com file?

Illegal move (not sure)

Memory address always go from _____.

Processor to memory

MUL (multiply) Instruction performs an unsigned multiplication of the source operand and the _____.

Accumulator

The interrupt call loads new values in_____ segment.

CS

Which part of machine code tells the central processor to perform a certain task.

CALL operation (not sure) operand code

Unconditional jump

Transfers the control if the condition is true

Which part of this (000000000000 B80500) encoded instruction is an offset?

0500

Which of the following is the renamed version of conditional jump JZ?

JE

Cell width refers to the total number of bits in a memory cell while the total number of cells is called the_____.

Cell width

Which of the following string instruction is generally used in a loop instead of REP prefix?

LODS

When a large number is subtracted from a smaller number, a borrow is needed; in this case which flag will be set

CF

In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag ?

ROR

The keyword used for the conditional jumps is_____.

SHORT

We can convert any digit to ____ by adding 0x30 in the digit.

ASCII

If AX=5, BX=5, CF and ZF are set, AF and DX contains zero then after the execution of instruction "ADC AX, BX", AX will contain the value _____.

10

In string instructions, the mode is called auto-increment mode when:

DF is set

The ASCII code for digit 8 is _____.

0x38

In a video memory, each screen location corresponds to ____ bytes.

2

Which combination will you prefer to obtain the physical address of the stack?

SS:SP

POP operand is also known as:

Source operand

In case of a downward compatible mechanism, the codes written for 8088 are _____ for 386 processor.

Valid

The routine that executes in response to an INT instruction is called?

ISR

The correlation process from the interrupt number to the interrupt handler uses a table called _____.

Interrupt Vector Table

Which assembly instruction is used to ADD data at address 1200 to bx register?

```
add  bx,[1200]
```

Which among the following is the pointer registers?

Stack pointer and index pointer

_____ Instructions have two parameters, one is the general purpose register to be loaded and the other is the memory location from which to load these registers.

LDS

When the first thing popped off from the stack, the stack would be the return "address" and not the_____.

Argument

The top of stack is contained in _____ register.

SP

In case of 32-bit processor, the size of an accumulator register will be _____ bits.

32

8088 is a _____ processor with its accumulator and all registers of _____.

16 bit, 16 bits

What does the given instruction do?

TEST BX, [MULTIPLIER]

Test the right most bit

Which of the following shift operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag ?

SHR

The iAPX88 processor supports _____ modes of memory access.

7

In XOR operation the output is 1 if

both inputs are same

DB-25 is a _____ port connector.

parallel

SCAS compares a source byte or word in register AL or AX with the _____ string element addressed by ES: DI and updates the flags.

Destination

The execution of the instruction "mov word [ES : 160], 0x1230" will print a character on the screen at:

First column of second row

Intel follows _____.

Little endian

_____ instruction makes the code reusable.

CALL

When the RET instruction is executed, it recovers the value of the _____ from the stack.

Instruction pointer

The maximum parameters a subroutine can receive
are_____ when all the general registers are used.

7

CLI stands for?

Clear the interrupt flag

_____ can be used to check whether particular bits of a number are set or not.

AND

Flags register is a special register in every architecture, it is also known as _____

Program Status Word

VGA stands for

Video Graphics Adapter

Interrupts are _____ events.

Synchronous

_____ is a special instruction that load a segment register and a general purpose register from a memory locations.

SCAS

What operation does the piece of code perform?

Shl word [multiplicand], 1

Rcl word [multiplicand+2], 1

Shift multiplicands left

Instruction Pointer holds the address of the

Next instruction to be executed

Stack is a data structure that behaves in a first in last _____ manner.

Out

Question # 1 of 10 Total Marks: 1

When two devices in the system want to use the same IRQ line then what will happen? Select correct option: **An IRQ Conflict** An IRQ Crash

An IRQ Collision

An IRQ Blockage

Question # 2 of 10 Total Marks: 1

Hard disk MBR(Master Boot Record) is of size_____.

Select _____ correct

option: **446 bytes**

350 bytes

512 bytes

256 bytes

Question # 3 of 10 Total Marks: 1

Which of the following IRQs is connected to serial port COM 2?

Select correct option:

IRQ 0

IRQ 1

IRQ 2

IRQ 3

Question # 4 of 10 Total Marks: 1 The first sector on hard disk contains the Select correct option: Hard disk size

Partition table

Data size

Sector size

Question # 5 of 10 Total Marks: 1

In programmable interrupt controller which of the following ports is referred as a control port?

Select correct option:

19

20

21

22

Question # 6 of 10 Total Marks: 1

Which of the following IRQs is used by the parallel port?

Select correct option:

IRQ 4

IRQ 5

IRQ 6

IRQ 7

Question # 7 of 10 Total Marks: 1

The programmable interval timer (PIT) has input frequency of Select correct option:

1.193MHZ

2.193MHZ

3.193MHZ

4.193MHZ

Question # 8 of 10 Total Marks: 1

CX register mostly use a Select correct option: **Counter register**

Flag register

Base register

Desination register

Question # 9 of 10 Total Marks: 1

The input frequency of the programmable interval timer (PIT) is Select correct option:

Fixed

Depends on processor clock

Variable

Depends on hardware attached

Question # 10 of 10 Total Marks: 1

The thread registration code initializes the PCB and adds it to the linked list so that the _____ will give it a turn.

Select correct option:

Assembler

Scheduler

Linker

Debugger

Question # 1 of 10 Total Marks: 1

INT13 --BIOS disk services" generally uses which register to return the 'error code' ? Select correct option:

CF

DL

AH

AL

Question # 2 of 10 Total Marks: 1

Operating system Organize data in the form of Select correct option: Folder

Batch file

File

None of the above

Question # 3 of 10 Total Marks: 1

___ decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.

Select correct option:

PUSH

POP

CALL

MOV

Question # 4 of 10 Total Marks: 1

Which of the following interrupts is Non maskable interrupt?

Select correct option:

INT 0

INT 1

INT 2

INT 3

Question # 5 of 10 Total Marks: 1

The maximum parameters a subroutine can receive are_____ when all the general registers are used.

Select correct option:

6

7

5

4

Question # 6 of 10 Total Marks: 1

When the operand of DIV instruction is of 16-bits then implied dividend will be stored in_____

Select correct option:

AX register

The concatenation of DX and AX

The concatenation of ES and AX

The concatenation of DS and BX

Question # 7 of 10 Total Marks: 1

COM2 is connected with

Select correct option:

IRQ 2

IRQ **3**

IRQ 4

IRQ 5

Question # 8 of 10 Total Marks: 1

The parallel port connector is called?

Select correct option:

BD-24

BD-25 **DB-25**

DB-24

Question # 9 of 10 Total Marks: 1

The instruction to call any software interrupt is
Select correct option:

GO INT interrupt_number

Call interrupt_number

INT interrupt_number

Call INT interrupt_number

Question # 10 of 10 Total Marks: 1

The INT 0x13 service 0x03 is use to
Select correct option: Get drive
parameter

Reset disk sector

Write disk sector

Read disk sector

Question # 1 of 10 Total Marks: 1

Data bus is

Select correct option: Uni-
directional

Bi-directional

Non-directional

None of the given

Question # 2 of 10 Total Marks: 1

PUSH increments SP (the stack pointer) by two and then transfers a word from the source
operand to the top of stack now pointed to by SP.

Select correct option:

True

False

Question # 3 of 10 Total Marks: 1

Peripheral address space is selected when which of the following instructions is given to the processor?

Select correct option:

MOV

DEC

IN

ADD

Question # 4 of 10 Total Marks: 1

Creation of threads can be Select correct option:

Static

Dynamic

Both

None of the above

Question # 5 of 10 Total Marks: 1

Which of the following IRQs is used by the parallel port?

Select correct option:

IRQ 4

5

6

7

Question # 6 of 10 Total Marks: 1

Priority of IRQ 0 interrupt is

Select correct option: **Highest** low

medium

None of the above

Question # 7 of 10 Total Marks: 1

The number of pins in a parallel port connector are?

Select correct option:

20

25

30

35

Question # 8 of 10 Total Marks: 1 The interrupt call loads new values in CS, IP and Select correct option:

DS

SS

FLAG Bookmar
k

Question # 9 of 10 Total Marks: 1 All the registers and stacks are saved in Select correct option:

Multitasking

Multi Processing

Function Call

BIOS

Question # 10 of 10 Total Marks: 1

In 9 pin DB connector ,which pin is assigned to TD.

Select correct option:

1

2

3

4

1. Assembly language is not a low level language.

a. True

b. False

2. In case of COM File first command parameter is stored at ____ offset of program segment prefix.

a. 0x80 (Not Confirm)

- b. 0x82
- c. 0x84
- d. 0x86

3. Address always goes from

a. Processor to memory

b. Memory to processor

- c. Memory to memory
- d. None of the above

4. The source register in OUT is

a. AL or AX

- b. BL or BX
- c. CL or CX
- d. DL or DX

5. By default CS is associated with

- a. SS
- b. BP
- c. CX

d. IP

6. Which of the following pins of parallel port are grounded

a. 10-18

b. 18-25

- c. 25-32
- d. 32-39

7. In the instruction `mov word [es:160], 0x1230`, 30 represents the character

- a. A
- b. B

c. 0

d. 1

8. On executing 0x21 0x3D, if file cant be opened then

a. CF will contain 1

- b. CF will contain 0
- c. ZF will contain 1
- d. ZF will contain 0

9. Which of the following IRQ is cascading interrupt

- a. IRQ 0
- b. IRQ 1

c. IRQ 2

d. IRQ 3

10. The execution of instruction mov word [es:160], 0x1230, will print a character on the screen at

a. First column of second row

- b. Second column of first row
- c. Second column of second row
- d. First column of third row

Question No: 1 (Marks: 1) - Please choose one To transfer control back the RET instruction take

- **1 argument** · 1 argument
- 3 arguments
- No arguments

Question No: 2 (Marks: 1) - Please choose one

In STOSB instruction SI is decremented or incremented by

- 4
- 1**
- 2
- 3

Question No: 3 (Marks: 1) - Please choose one

CMPS instruction subtracts the source location _____ from the destination location.

DS:SI DS:DI

ES:SI

ES:DI

Question No: 4

(Marks: 1) - Please choose one

Regarding assembler, which statement is true:

Assembler converts mnemonics to the corresponding OP CODE

Assembler converts OP CODE to the corresponding mnemonics

Assembler executes the assembly code all at once

Assembler executes the assembly code step by step

Question No: 5

(Marks: 1) - Please choose one

If "BB" is the OP CODE of the instruction which states to "move a constant value to AX register", the hexadecimal representation (Using little Endian notation) of the instruction "Mov AX,336" ("150" in hexadecimal number system) will be:

• **0xBB0150**

• 0x5001BB

• 0x01BB50

• 0xBB5001

Question No: 6

(Marks: 1) - Please choose one

In the instruction MOV AX, 5 the number of operands are

• **1**

• 2

• 3

• 4

Question No: 7

(Marks: 1) - Please choose one

The maximum parameters a subroutine can receive (with the help of registers) are

• 6

• **7**

• 8

• 9

Question No: 8

(Marks: 1) - Please choose one

In assembly the CX register is used normally as a _____register.

- source
- **counter**
- index
- pointer

Question No: 9

(Marks: 1) - Please choose one

All the addressing mechanisms in iAPX 8 8 return a number called _____ address .

- **effective**
- faulty
- indirect
- direct

Question No: 10

(Marks: 1) - Please choose one

When a 16 bit number is divided by an 8 bit number, the dividend will be in

- **AX**
- BX
- CX
- DX

Question No: 11

(Marks: 1) - Please choose one

in Left-Shift-Operation the left most bit _____

- **will drop**
- will go into CF
- Will come to the right most
- will be always 1

Question No: 12

(Marks: 1) - Please choose one

Suppose the decimal number "35" after shifting its binary two bits to left, the new value becomes _____

- 35

- 70
- **140**
- 17

Question No: 13

(Marks: 1) - Please choose one

When divide overflow occurs processor will be interrupted this type of interrupt is called

- Hardware interrupt
- Software interrupt
- **Processor exception**
- Logical interrupts

Question No: 14

(Marks: 1) - Please choose one

Which mathematical operation is dominant during the execution of SCAS instruction

- Division
- Multiplication
- Addition
- **Subtraction**

Question No: 15

(Marks: 1) - Please choose one

After the execution of REP instruction CX will be decremented then which of the following flags will be affected?

- CF ·
OF
- **DF**
- No flags will be affected

Question No: 16

(Marks: 1) - Please choose one

_____ is one of the reasons due to which string instructions are used in 8088

- Efficiency and accuracy
- Reduction in code size and accuracy
- **Reduction in code size and speed**
- Reduction in code size and efficiency

Question No: 1 (Marks: 1) - Please choose one

The physical address of the stack is obtained by

- ▶ SS:SI combination
- ▶ **SS:SP combination**
- ▶ ES:BP combination
- ▶ ES:SP combination

Question No: 2 (Marks: 1) - Please choose one

After the execution of instruction "RET "

- ▶ **SP is incremented by 2**
- ▶ SP is decremented by 2
- ▶ SP is incremented by 1
- ▶ SP is decremented by 1

Question No: 3 (Marks: 1) - Please choose one

The second byte in the word designated for one screen location holds

- ▶ The dimensions of the screen
- ▶ Character position on the screen
- ▶ **Character color on the screen**
- ▶ ASCII code of the character

Question No: 4 (Marks: 1) - Please choose one

REP will always

- ▶ Increment CX by 1
- ▶ Increment CX by 2
- ▶ **Decrement CX by 1**
- ▶ Decrement CX by 2

Question No: 5 (Marks: 1) - Please choose one

The basic function of SCAS instruction is to

- ▶ **Compare**
- ▶ Scan

- ▶ Sort
- ▶ Move data

Question No: 6 (Marks: 1) -
Please choose one Index registers are used to store _____ ▶ Data

- ▶ Intermediate result
- ▶ **Address**
- ▶ Both data and addresses

Question No: 7 (Marks: 1) - Please choose one

The bits of the _____ work independently and individually ▶ index register

- ▶ base register
- ▶ **flags register**
- ▶ accumulator

Question No: 8 (Marks: 1) -
Please choose one To convert any digit to its ASCII representation

- ▶ **Add 0x30 in the digit**
- ▶ Subtract 0x30 from the digit
- ▶ Add 0x61 in the digit
- ▶ Subtract 0x61 from the digit

Question No: 9 (Marks: 1) - Please choose one

When a 32 bit number is divided by a 16 bit number, the quotient is of _____ ▶ 32 bits

- ▶ 16 bits
- ▶ 8 bits
- ▶ **4 bits**

Question No: 10 (Marks: 1) - Please choose one

When a 16 bit number is divided by an 8 bit number, the quotient will be in ▶ AX

▶ **AL**

▶ AH ▶
DX

Question No: 11 (Marks: 1) - Please choose one

Which mathematical operation is dominant during the execution of SCAS instruction ▶ Division

▶ Multiplication

▶ Addition

▶ **Subtraction**

Question No: 12 (Marks: 1) - Please choose one

If AX contains decimal -2 and BX contains decimal 2 then after the execution of instructions:

CMP AX, BX JA label

▶ Jump will be taken

▶ **Zero flag will set**

▶ ZF will contain value -4

▶ Jump will not be taken

Question No: 13 (Marks: 1) - Please choose one

The execution of the instruction "mov word [ES : 160], 0x1230" will print a character "0" on the screen at

▶ Second column of first row

▶ **First column of second row**

▶ Second column of second row

▶ First column of third row

Question No: 14 (Marks: 1) - Please choose one

If the direction of the processing of a string is from higher addresses towards lower addresses then

▶ ZF is cleared

- ▶ **DF is cleared**
- ▶ ZF is set
- ▶ DF is set

Question No: 15 (Marks: 1) - Please choose one

The instruction ADC has_____ Operand(s)

- ▶ 0
- ▶ 1
- ▶ 2
- ▶ **3**

Question No: 16 (Marks: 1) - Please choose one

Which bit of the attributes byte represents the red component of background color ?

- ▶ **3**
- ▶ 4
- ▶ 5
- ▶ 6

Q=1:

Which bit of attributes byte represents the blue component of foreground color? **0**

- 1
- 2
- 3

Q=2:

The clear screen operation initializes the whole block of video memory to:

- 0417
- 0714
- 0741
- **0720**

Q=3:

When the operand of DIV instruction is of 16 bit then implied dividend will be of

- 64-bit
- **32-bits**
- 16-bits
- 8--bits

Q=4

Which of the following is the pair of register used to access memory in string instruction:

- DI and BP
- SI and BP
- **DI and SI**
- DS and Si

Q=5

A fat32 file system directory entry in DOS consist of how many bytes?

- 16
- 24
- **32**
- 64

Q=6:

Which register is generally used to specify the services number of an interrupt?

DX

AX

BX

CX

.....

Q=7:

In 9 pin db 9 connector ,which pin is assigned to RD(received data)

- 1
- **2**
- 3
- 4

Q=8

In case of COM file, maximum length of parameters passed through command line can be.....

- 63 bytes
- 127bytes
- 255 bytes
- 511 bytes

Q=9

We can access the DOS service using;

- **Int 0x21**
- Int 0x13
- Int 0x 10
- Int 0x 08

Q=10

In 9 pin 9 connector,which pin is assigned to signal ground

- 3
- 4
- **5**
- 6

Q=11:

BPB stands for

- Basic parameter block
- Bios precise block
- Basic precise block
- **Bios parameter block**

Q=12

Int 13-bios disk service "generally uses which register to return the error flag?

- CF ·
- DL

- **AH**
- AL

Q=13:

The first sector on the hard disk contains the

- Hard disk size
- **Partition table**
- Data size
- Sector size

Q=14

Operating system organize data in the form of

- Folder
- Batch file
- **File**
- None of above

.....

Q=15

In 9 pin db 9 connector, which pin is assigned to TD(transmitted data)

- 1
- 2
- **3**
- 4

Q=16"

Device derive can be divided into -----major categories.

- 5
- 4
- 3
- 2

1. BL contains 5 decimal then after right shift , BL will become

- 3
- **2.5**
- 5
- 10

2. 8 * 16 font is stored in _____ bytes.

- 3
- 4
- 8
- **16**

3. In DOS input buffer , number of characters actually read on return is stored in

- First byte
- **Second byte**
- Third byte
- Fourth byte

4. IRQ 0 has priority

- Low
- High
- **Highest**

- Medium

5. Thread registration code initialize PCB and add to linked list so that ____ will give itturn.

- Assembler

- Linker

- **Scheduler**

- Debugger

6. Traditional calling conventions are in ____ number

- 1

- 2

- 3

- 4

7. VESA VEB 2.0 is standard for

- **High Resolution Mode**

- Low Resolution Mode

- Very High Resolution Mode

- Medium Resolution Mode

8. To clear direction flag which instruction is used

- **Cld**

- Clrd

- Cl df

- Clr df

9. In STOSW instruction , When DI is cleared , SI is

- Incremented by 1

- **Incremented by 2**

- Decrement by 1

- Decrement by 2

10. Interrupt that is used in debugging with help of trap flag is

- INT 0

- **INT 1**

- INT 2

- INT 3

11. INT for arithmetic overflow is

- INT 1
- INT 2
- INT 3
- **INT 4**

12. IRQ referred as

- **Eight Input signals**
- One Input signal
- Eight Output signals
- One output signal

13. IRQ for keyboard is 1

14. IRQ for sound card is 5

15. IRQ for floppy disk is 6

16. IRQ with highest priority is

- Keyboard IRQ
- **Timer IRQ**
- Sound Card
- Floppy Disk

17. Pin for parallel port ground is

- 10-18
- **18-25**
- 25-32
- 32-39

18. The physical address of Interrupt Descriptor Table (IDT) is stored in

- GDTR
- **IDTR**
- IVT
- IDTT

19. Execution of "RET 2" results in?

20. CX register is· *Count register*
- Data register
 - Index register
 - Base register
21. OUT instruction uses AX as source register.
22. IN DB-9 connector the Data Set ready pin is at
- 5
 - *6*
 - 7
 - 8
23. If two devices uses same IRQ then there is
- IRQ collision
 - *IRQ conflict*
 - IRQ drop
24. VESA organizes 16 bit color for every pixel in ratio
- 5:5:5
 - *5:6:5*
 - 6:5:6
 - 5:6:7
25. Division by zero is done by which interrupt.*Interrupt 0.*

.....

Question No: 1 (Marks: 1) - Please choose one

After the execution of SAR instruction

- ▶ *The msb is replaced by a 0*
- ▶ The msb is replaced by 1
- ▶ The msb retains its original value
- ▶ The msb is replaced by the value of CF

Question No: 2 (Marks: 1) - Please choose one

RETf will pop the offset in the

- ▶ BP
- ▶ **IP**
- ▶ SP
- ▶ SI

Question No: 3 (Marks: 1) - Please choose one

The routine that executes in response to an INT instruction is called

- ▶ **ISR**
- ▶ IRS
- ▶ ISP
- ▶ IRT

Question No: 4 (Marks: 1) - Please choose one

The first instruction of "COM" file must be at offset:

- ▶ 0x0010
- ▶ **0x0100**
- ▶ 0x1000
- ▶ 0x0000

Question No: 5 (Marks: 1) - Please choose one

"Far" jump is not position relative but is _____

- ▶ memory dependent
- ▶ **Absolute**
- ▶ temporary
- ▶ indirect

Question No: 6 (Marks: 1) - Please choose one

Only _____ instructions allow moving data from memory to memory.

- ▶ **string**
- ▶ word
- ▶ indirect
- ▶ stack

Question No: 7 (Marks: 1) - Please choose one

After the execution of instruction "RET 2"

- ▶ **SP is incremented by 2**
- ▶ SP is decremented by 2
- ▶ SP is incremented by 4
- ▶ SP is decremented by 4

Question No: 8 (Marks: 1) - Please choose one

DIV instruction has

- ▶ **Two forms**
- ▶ Three forms
- ▶ Four forms
- ▶ Five forms

Question No: 9 (Marks: 1) - Please choose one

When the operand of DIV instruction is of 16 bits then implied dividend will be of

- ▶ 8 bits
- ▶ 16 bits
- ▶ **32 bits**
- ▶ 64 bits

Question No: 10 (Marks: 1) - Please choose one

After the execution of MOVS instruction which of the following registers are updated ► SI only

- DI only
- **SI and DI only**
- SI, DI and BP only

Question No: 11 (Marks: 1) - Please choose one

In 8088 architecture, whenever an element is pushed on the stack

- SP is decremented by 1
- **SP is decremented by 2** ► SP is decremented by 3
- SP is decremented by 4

Question No: 12 (Marks: 1) - Please choose one

When a very large number is divided by very small number so that the quotient is larger than the space provided, this is called

- Divide logical error
- **Divide overflow error**
- Divide syntax error
- An illegal instruction

Question No: 13 (Marks: 1) - Please choose one

In the word designated for one screen location, the higher address contains

- **The character code**
- The attribute byte
- The parameters
- The dimensions

Question No: 14 (Marks: 1) - Please choose one

Which of the following options contain the set of instructions to open a window to the video memory?

► mov AX, 0xb008 mov ES, AX

► **mov AX, 0xb800 mov ES, AX**

► mov AX, 0x8b00 mov ES, AX

► mov AX, 0x800b mov ES, AX

Question No: 15 (Marks: 1) - Please choose one

In a video memory, each screen location corresponds to

► One byte

► **Two bytes** ►
Four bytes

► Eight bytes

Question No: 16 (Marks: 1) - Please choose one

The execution of the instruction "mov word [ES : 0], 0x0741" will print character "A" on screen , background color of the screen will be

► **Black**

► White

► Red

► Blue

Question No: 1 __ (Marks: 1) - Please choose one Which of the following is not true about registers?

· Their operation is very much like memory

· Intermediate results may also be stored in registers. · They are also called scratch pad

ram · **None of given options.**

Question No: 2 __ (Marks: 1) - Please choose one

move [bp], al moves the one byte content of the AL register to the address contained in BP register in the current

- Stack segment
- **Code segment**
- Data segment
- Extra segment

Question No: 3 (Marks: 1) - Please choose one

In a rotate through carry right (RCR) instruction applied on a 16 bit word Effectively there is

- 16 bits rotation
- 1 bit rotation
- 17 bits rotation
- 8 bits rotation

Question No: 4__ (Marks: 1) - Please

choose one The 8088 stack works on

- **Word sized elements**
- Byte sized elements
- Double sized element
- Nibble sized element

Question No: 5 (Marks: 1) - Please choose one

An 8 x 16 font is stored in.....Bytes

- 2
- 4
- 8
- **16**

Question No: 6 (Marks: 1) - Please

INT 10 is used for.....services.

- RAM
- Disk
- **BIOS video**
- DOS video

Question No: 7 __ (Marks: 1) - Please choose one

Priority of IRQ 0 interrupt is

- medium
- high
- **highest**
- low

Question No: 8 __ (Marks: 1) - Please choose one

Threads can have function calls, parameters and _____variables.

- global
- **local**
- legal
- illegal

Question No: 9 __ (Marks: 1) - Please choose one How many prevalent calling conventions do.....exist

- 1
- **2**
- 3
- 4

VERY IMPORTANT

Question No: 10 (Marks: 1) - Please choose one In 9pin DB 9 DSR is assigned on pin number

- 4
- 5
- **6**
- 7

Question No: 11

(Marks: 1) - Please

choose one In 9pin DB 9 CTS is assigned on pin number

- 6
- 7
- **8**
- 9

Question No: 12__ (Marks: 1) - Please choose one

In 9pin DB 9 CD is assigned on pin number

- **1**
- 2
- 3
- 4

Question No: 13__ (Marks: 1) - Please choose one

In 9pin DB 9 RD is assigned on pin number

- 1
- **2**
- 3
- 4

Question No: 14 __ (Marks: 1) - Please choose one

in device attribute word which of the following bit decides whether it is a character device or a block device

- Bit 12 Bit 13
- Bit 14
- **Bit 15**

Question No: 15__ (Marks: 1) - Please choose one

Video services are classified into _____broad categories

- **2**
- 3
- 4
- 5

Question No: 16 (Marks: 1) - Please choose

one In STOSB instruction, when DF is clear, SI is

- **Incremented by 1**

- Incremented by 2
- Decrement by 1
- Decrement by 2

Question No: 17 (Marks: 1) - Please choose one The process of sending signals back and forth is called

- Activity
- Hand-shaking
- **Interruption**
- Time clicking

Question No: 18 (Marks: 1) - Please choose one which of the following is a special type of interrupt that returns to the same instruction instead of the next instruction

- **Divide overflow interrupt**
- Debug interrupt
- Arithmetic overflow interrupt
- Change of sign interrupt

Question No: 19 __ (Marks: 1) - Please choose one Which of the following IRQs is derived by a timer device?

- **IRQ 0**
- IRQ 1
- IRQ 2
- IRQ 3

Question No: 20 __ (Marks: 1) - Please choose one

Which of the following interrupts is used for Arithmetic overflow

- INT 1
- INT 2
- INT 3
- **INT 4**

Question No: 21 __ (Marks: 1) - Please choose one

Which of the following IRQs is connected to serial port COM 2?

- IRQ 0
- IRQ 1
- IRQ 2
- **IRQ 3**

Question No: 22 __ (Marks: 1) -

Please choose one

An End of Interrupt (EOI) signal is sent by

- **Handler**
- Processor
- IRQ
- PIC

Question No: 23 __ (Marks: 1) - Please

choose one The source registers in OUT is

- **AL or AX**
- BL or BX
- CL or CX
- DL or DX

Question No: 24 (Marks: 1) - Please choose one

In programmable interrupt controller which of the following ports is used for selectively enabling or disabling interrupts

- 19
- 20
- **21**
- 22

Question No: 25 (Marks: 1) - Please choose one The number of pins in a parallel port connector are?

- **25**
- 30
- 35
- 45

Question No: 26 (Marks: 1) - Please choose one

Which of the following pins of a parallel port connector are grounded?

- 10-18
- 18-25
- 25-32
- 32-39

Question No: 27 __ (Marks: 1) - Please choose one

Suppose a decimal number 35 when its binary is shifted to write two places the new number will become

- 35
- 70
- 140
- 17

Question No: 28 __ (Marks: 1) - Please choose one

A 32bit address register can access upto of memory so memory access has increased a lot.

- 2GB
- 4GB
- 6GB
- 8GB

Question No: 29 __ (Marks: 1) - Please choose one

In NASM an imported symbol is declared with the while and exported symbol is declared with the

- Global directive, External directive · External directive, Global directive
- Home Directive, Foreign Directive

- Foreign Directive, Home Directive

Question No: 30 (Marks: 1) - Please choose one Single step interrupt is

Question No: 1 (Marks: 1)

- Please choose one

Sun SPARC Processor has a fixed _____ instruction size.

- 16bit
- **32bit**
- 64bit
- 20bit

Question No: 2 (Marks: 1)

- Please choose one

When the subprogram finishes, the _____ retrieves the return address from the stack and transfers control to that location.

- **RET instruction**
- CALL instruction
- POP instruction
- Jump instruction

Question No: 3 (Marks: 1)

- Please choose one

A 32 bit address register can access upto _____ of memory.

- 1 GB
- 6 GB
- **4 GB**
- 2 GB

Question No: 4 (Marks: 1)

- Please choose one

The value of a segment register when the processor is running under protected mode is called

- **segment descriptor**
- segment selector
- global descriptor table
- protected register

Question No: 5 (Marks: 1)

- Please choose one

FS and GS are two _____ in protected mode.

- **segment registers**
- segment selectors
- stack pointers
- register pointers

Question No: 6 (Marks: 1)

- Please choose one

IRQ 0 interrupt have _____ priority

- low
- medium
- **highest**
- lowest

Question No: 7 (Marks: 1)

- Please choose one

IDT stands for _____.

- interrupt descriptor table
- individual descriptor table
- inline data table

· **interrupt descriptor table**

Question No: 8 (Marks: 1)

- Please choose one

Every bit of line status in serial port conveys _____ information.

· **different**

· same

· partial

· full

Question No: 9 (Marks: 1)

- Please choose one

There are total _____ bytes in a standard floppy disk.

· 1444k

· **1440k**

· 1280k

· 2480k

Question No: 10 (Marks: 1)

- Please choose one

An 8x16 font is stored in _____ bytes.

· 8

· **16**

· 4

· 20

=====

. Serial Port is also accessible via *I/O* ports , **COM 1** is accessible via ports 3F8-3FF while **COM 2** is accessible via 2F8 -2FF.

The first register at 3F8 is the *Transmitter* holding register if written to and the receiver *buffer* register if read from.

Other register of our interest include 3F9 whose *Bit 0* must be set to enable received data available interrupt and *Bit 1* must be set to enable transmitter holding register empty interrupt.

(Transmitter, COM 1, I/O ports, COM2. bit 0, Buffer, 3FA)

=====

Question # 1

There are three busses to communicate the processor and memory named as _____

- 1) : address bus.,data bus and data bus.
- 2) : addressing bus.,data bus and data bus.
- 3) : address bus.,datamove bus and data bus.
- 4) : **address bus.,data bus and control bus..**

Question # 2

The address bus is unidirectional and address always travels from processor to memory. 1) : **TRUE**

2) : FALSE

Question # 3

Data bus is bidirectional because_____

- 1) : To way
- 2) : Data moves from both, processor to memory and memory to processor,
- 3) : **Data moves from both, processor to memory and memory to data Bus,**
- 4) : None of the Given

Correct Option : 3 From : Lecture 1

Question # 4

Control bus_____ 1) :
is Not Important.

- 2) : is Important .
- 3) : **bidirectional.**
- 4) : unidirectional .

Correct Option : 3 From : Lecture 1

Question # 5

A memory cell is an n-bit location to store data, normally _____also called a byte

- 1) : 4-bit
- 2) : **8-bit**
- 3) : 6-bit
- 4) : 80-bit

Correct Option : 2 From : Lecture 1

Question # 6

The number of bits in a cell is called the cell width._____ define the memory completely.

- 1) : **Cell width and number of cells,**
- 2) : cell number and width of the cells,
- 3) : width
- 4) : Height

Correct Option : 1 From : Lecture 1

Question # 7

for memory we define two dimensions. The first dimension defines how many _____bits are there in a single memory cell.

- 1) : **parallel**
- 2) : Vertical
- 3) : long
- 4) : short

Correct Option : 1 From : Lecture 1

Question # 8

_____ operation requires the same size of data bus and memory cell width. 1)
: Normal

- 2) : **Best and simplest**
- 3) : first
- 4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 9

Control bus is only the mechanism. The responsibility of sending the appropriate signals on the control bus to the memory is of the_____.

- 1) : Data Bus
- 2) : **processor**
- 3) : Address Bus

4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 10

In "total: dw 0 " Opcode total is a _____ 1) :
Literal

2) : Variable

3) : **Label**

4) : Starting point

Correct Option : 3 From : Lecture 10

Question # 11

| 0 | --> | 1 | 1 | 0 | 1 | 0 | 0 | 0 | --> | C | is a example of _____

1) : Shl

2) : sar

3) : **Shr**

4) : Sal

Correct Option : 3 From : Lecture 10

Question # 12

| C | <-- | 1 | 1 | 0 | 1 | 0 | 0 | 0 | <-- | 0 | is a example of _____

1) : **Shl**

2) : sar

3) : Shr

4) : Sal

Correct Option : 1 From : Lecture 10

Question # 13

ADC has _____ operands.

1) : two

2) : **three**

3) : Five

4) : Zero

Correct Option : 2 From : Lecture 10

Question # 14

The basic purpose of a computer is to perform operations, and operations need _____.

1) : order

2) : nothing

3) : **operands**

4) : bit

Correct Option : 3 From : Lecture 2

Question # 15

Registers are like a scratch pad ram inside the processor and their operation is very much like normal_____.

1) : Number

2) : ooperations

3) : **memory cells**

4) : None of the Given

Correct Option : 3 From : Lecture 2

Question # 16

There is a central register in every processor called the _____ and The word size of a processor is defined by the width of its_____.

1) : **accumulator,accumulator**

2) : data bus,accumulator

3) : accumulator, Address Bus

4) : accumulator,memory

Correct Option : 1 From : Lecture 2

Question # 17

_____does not hold data but holds the address of data

1) : Pointer, Segment, or Base Register

2) : **Pointer, Index, or Base Register**

3) : General Registers

4) : Instruction Pointer

Correct Option : 2 From : Lecture 2

Question # 18

"The program counter holds the address of the next instruction to be _____"

1) : **executed.**

2) : called

3) : deleted

4) : copy

Correct Option : 1 From : Lecture 2

Question # 19

There are ____ types of "instruction groups"

1) : **4** 2)
: 5

3) : 3

4) : 2

Correct Option : 1 From : Lecture 2

Question # 20

These instructions are used to move data from one place to another.

1) : **TRUE** 2)
: FALSE

3) :

4) :

Correct Option : 1 From : Lecture 2

Question # 21

"mov" instruction is related to the _____ *****.

1) : Arithmetic and Logic Instructions

2) : **Data Movement Instructions**

3) : Program Control Instructions

4) : Special Instructions

Correct Option : 2 From : Lecture 2

Question # 22

_____allow changing specific processor behaviors and are used to play with it.

1) : **Special Instructions**

2) : Data Movement Instructions

3) : Program Control Instructions

4) : Arithmetic and Logic Instructions

Correct Option : 1 From : Lecture 2

Question # 23

8088 is a 16bit processor with its accumulator and all registers of _____.

1) : 32 bits

2) : 6 bits

3) : **16 bits**

4) : 64 bits

Correct Option : 3 From : Lecture 2

Question # 24

The _____ of a processor means the organization and functionalities of the registers it contains and the instructions that are valid on the processor.

- 1) : Manufactures
- 2) : **architecture**
- 3) : Deal
- 4) : None of the Given

Correct Option : 2 From : Lecture 2

Question # 25

Intel IAPX88 Architecture is _____

- 1) : **More then 25 old**
- 2) : New
- 3) : Not Good
- 4) : None of the Given

Correct Option : 1 From : Lecture 2

Question # 26

The iAPX88 architecture consists of _____ registers.

- 1) : 13
- 2) : 12
- 3) : 9
- 4) : **14**

Correct Option : 4 From : Lecture 3

Question # 27

General Registers are _____

- 1) : **AX, BX, CX, and DX**
- 2) : XA, BX, CX, and DX
- 3) : SS, SI and DI
- 4) : 3

Correct Option : 1 From : Lecture 3

Question # 28

AX means we are referring to the extended 16bit "A" register. Its upper and lower byte are separately accessible as _____.

- 1) : **AH and AL**

- 2) : A Lower and A Upper
- 3) : AL, AU
- 4) : AX

Correct Option : 1 From : Lecture 3

Question # 29

AX is General purpose Register where A stands for_____.

- 1) : Acadmic
- 2) : Ado
- 3) : Architecture
- 4) : **Accumulator**

Correct Option : 4 From : Lecture 3

Question # 30

The B of BX stands for _____because of its role in memory addressing. 1) :
Busy

- 2) : **Base**
- 3) : Better
- 4) : None of the Given

Correct Option : 2 From : Lecture 3

Question # 31

The D of DX stands for Destination as it acts as the destination in _____.

- 1) : **I/O operations**
- 2) : operations
- 3) : memory cells
- 4) : Memory I/O operations

Correct Option : 1 From : Lecture 3

Question # 32

The C of CX stands for Counter as there are certain instructions that work with an automatic count in the _____.

- 1) : DI register
- 2) : BX register
- 3) : **CX register**
- 4) : DX register

Correct Option : 3 From : Lecture 3

Question # 33

_____ are the index registers of the Intel architecture which hold address of data and used in memory access.

- 1) : SI and SS
- 2) : PI and DI
- 3) : SI and IP
- 4) : **SI and DI**

Correct Option : 4 From : Lecture 3

Question # 34

In Intel IAPX88 architecture _____ is the special register containing the address of the next instruction to be executed.

- 1) : AX
- 2) : PI
- 3) : **IP**
- 4) : SI

Correct Option : 3 From : Lecture 3

Question # 35

SP is a memory pointer and is used indirectly by a set of _____.

- 1) : **instructions**
- 2) : Pointers
- 3) : Indexes
- 4) : Variables

Correct Option : 1 From : Lecture 3

Question # 36

_____ is also a memory pointer containing the address in a special area of memory called the stack.

- 1) : SP
- 2) : **BP**
- 3) : PB
- 4) : AC

Correct Option : 2 From : Lecture 3

Question # 37

_____ is bit wise significant and accordingly each bit is named separately. 1) :

AX

- 2) : FS
- 3) : IP
- 4) : **Flags Register**

Correct Option : 4 From : Lecture 3

Question # 38

When two 16bit numbers are added the answer can be 17 bits long, this extra bit that won't fit in the target register is placed in the _____where it can be used and tested 1) : **carry flag**

2) : Parity Flag

3) : Auxiliary Carry

4) : Zero Flag

Correct Option : 1 From : Lecture 3

Question # 39

Program is an ordered set of instructions for the processor.

1) : **TRUE** 2)

: FALSE

3) :

4) :

Correct Option : 1 From : Lecture 3

Question # 40

For Intel Architecture "operation destination, source" is way of writing things.

1) : **TRUE** 2)

: FALSE

3) :

4) :

Correct Option : 1 From : Lecture 3

Question # 41

Operation code " add ax, bx " _____.

1) : Add the bx to ax and change the bx

2) : Add the ax to bx and change the ax

3) : **Add the bx to ax and change the ax**

4) : Add the bx to ax and change nothing

Correct Option : 3 From : Lecture 3

Question # 42

The maximum memory iAPX88 can access is_____.

1) : **1MB**

2) : 2MB

3) : 3MB

4) : 128MB

Correct Option : 1 From : Lecture 4

Question # 43

The maximum memory iAPX88 can access is 1MB which can be accessed with _____.

1) : 18 bits

2) : **20 bits**

3) : 16 bits

4) : 2 bits

Correct Option : 2 From : Lecture 4

Question # 44

_____ address of 1DED0 where the opcode B80500 is placed.

1) : **physical memory**

2) : memory

3) : effective

4) : None of the Given

Correct Option : 1 From : Lecture 4

Question # 45

16 bit of Segment and Offset Addresses can be converted to 20bit Address i.e

Segment Address with lower four bits zero + Offset Address with _____ four bits zero = 20bit Physical Address

1) : Middle

2) : lower

3) : Top

4) : **upper**

Correct Option : 4 From : Lecture 4

Question # 46

When adding two 20bit Addresses a carry if generated is dropped without being stored anywhere and the phenomenon is called address_____.

1) : **wraparound**

2) : mode

3) : ping

4) : error

Correct Option : 1 From : Lecture 4

Question # 47 segments can only be defined a 16byte boundaries called _____ boundaries.

- 1) : **segment**
- 2) : paragraph
- 3) : Cell
- 4) : RAM

Correct Option : 1 From : Lecture 4

Question # 48

in a Program CS, DS, SS, and ES all had the same value in them. This is called _____.

- 1) : equal memory
- 2) : **overlapping segments**
- 3) : segments hidding
- 4) : overlapping SI

Correct Option : 2 From : Lecture 4

Question # 49

"db num1" size of the memory is _____

- 1) : **1byte**
- 2) : 4bit
- 3) : 16bit
- 4) : 2byte

Correct Option : 1 From : Lecture 5

Question # 50

" 1-----[org 0x0100]

2-----mov ax, [num1] ; load first number in ax

3-----mov bx, [num2] ; load second number in bx

4-----add ax, bx _____

5-----int 0x21

6-----

7-----num1: dw 5 8-----num2:
dw 10

Comments for the 4 are :

- 1) : No comments Will be
- 2) : ; accumulate sum in add
- 3) : ; **accumulate sum in ax**
- 4) : ; accumulate sum in Bx

Correct Option : 3 From : Lecture 5

Question # 51

In " mov ax, bx " is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : Direct
- 4) : **Register**

Correct Option : 4 From : Lecture 5

Question # 52

In "mov ax, [bx] " is _____ Addressing Modes

- 1) : **Based Register Indirect**
- 2) : Indirect
- 3) : Base Indirect
- 4) : Immediate

Correct Option : 1 From : Lecture 5

Question # 53

In "mov ax, 5 " is _____ Addressing Modes

- 1) : **Immediate**
- 2) : Indirect
- 3) : Indirect
- 4) : Register

Correct Option : 1 From : Lecture 6

Question # 54

In " mov ax, [num1+bx] " is _____ ADDRESSING

- 1) : OFFSET+ Indirect
- 2) : Register + Direct
- 3) : Indirect + Reference
- 4) : **BASEd REGISTER + OFFSET**

Correct Option : 4 From : Lecture 7

Question # 55

"base + offset addressing " gives This number which came as the result of addition is called the _____. 1) : Address

2) : mode

3) : **effective address**

4) : Physical Address

Correct Option : 3 From : Lecture 7

Question # 56

"mov ax, [cs:bx]" associates _____ for this one instruction

1) : CS with BX

2) : **BX with**

CS 3) : BX with

AX

4) : None of the Given

Correct Option : 2 From : Lecture 7

Question # 57

For example

BX=0100

DS=FFF0

And Opcode are;

move [bx+0x0100], Ax

now what is the effective memory address;

1) : 0020

2) : **0200**

3) : 0300

4) : 0x02

Correct Option : 2 From : Lecture 7

Question # 58

For example

BX=0100

DS=FFF0

And Opcode are;

move [bx+0x0100], Ax

now what is the physical memory address;

- 1) : 0020
- 2) : **0x0100**
- 3) : 0x10100
- 4) : 0x100100

Correct Option : 2 From : Lecture 7

Question # 59

In " mov [1234], al " is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : **Direct**
- 4) : Register

Correct Option : 3 From : Lecture 8

Question # 60

In " mov [SI], AX " is _____ Addressing Modes.

- 1) : Basef Register Indirect
- 2) : Indirect
- 3) : **Indexed Register Indirect**
- 4) : Immediate

Correct Option : 3 From : Lecture 8

Question # 61

In " mov ax, [bx - Si] " is _____ ADDRESSING

- 1) : Basef Register Indirect
- 2) : Indirect
- 3) : Direct
- 4) : **illegal**

Correct Option : 4 From : Lecture 8

Question # 62

In " mov ax, [BL] " there is error i.e. _____

- 1) : Address must be 16bit
- 2) : Address must be 8bit
- 3) : Address must be 4bit
- 4) : **8 bit to 16 bit move illegal**

Correct Option : 4 From : Lecture 8

Question # 63

In " mov ax, [SI+DI] " there is error i.e. _____

- 1) : **Two indexes can't use as Memory Address**
- 2) : index can't use as Memory Address
- 3) : I don't Know
- 4) : None of the Given

Correct Option : 1 From : Lecture 8

Question # 64

In JNE and JNZ there is difference for only _____;

- 1) : **Programmer or Logic**
- 2) : Assembler
- 3) : Debugger
- 4) : IAPX88

Correct Option : 1 From : Lecture 9

Question # 65

JMP is Instruction that on executing take jump regardless of the state of all flags is called_____

- 1) : Jump
- 2) : Conditional jump
- 3) : **Unconditional jump**
- 4) : Stay

Correct Option : 3 From : Lecture 9

Question # 66

When result of the source subtraction from the destination is zero, zero flag is set i.e. ZF=1 its mean that;

- 1) : **DEST = SRC**
- 2) : DEST != SRC
- 3) : DEST < SRC
- 4) : DEST > SRC

Correct Option : 1 From : Lecture 9

Question # 67

When an unsigned source is subtracted from an unsigned destination and the destination is smaller, borrow is needed which sets the _____.

- 1) : carry flag i.e CF = 0
- 2) : **carry flag i.e CF = 1**
- 3) : Carry Flag + ZF=1
- 4) : None of the Given

Correct Option : 2 From : Lecture 9

Question # 68

In the case of unassigned source and destination when subtracting and in the result ZF =1
OR

CR=1 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST ? USRC
- 4) : **DEST < SRC**

Correct Option : 3 From : Lecture 9

Question # 69

In the case of unassigned source and destination when subtracting and in the result ZF =0

AND CR=0 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST < USRC
- 4) : **UDEST > USRC**

Correct Option : 4 From : Lecture 9

Question # 70

In the case of unassigned source and destination when subtracting and in the result
CR=0 then _____ 1) : DEST = SRC

- 2) : DEST != SRC
- 3) : UDEST < USRC
- 4) : **UDEST ? USRC**

Correct Option : 4 From : Lecture 9

Question # 71

_____This jump is taken if the last arithmetic operation produced a zero in its destination.

After a CMP it is taken if both operands were equal.

- 1) : **Jump if zero(JZ)/Jump if equal(JE)**
- 2) : Jump if equal(JE)
- 3) : Jump if zero(JZ)
- 4) : No Jump fot This

Correct Option : 1 From : Lecture 9

Question # 72

_____ This jump is taken after a CMP if the unsigned source is smaller than or equal to the unsigned destination.

- 1) : JBE(Jump if not below or equal)
- 2) : **JNA(Jump if not above)/JBE(Jump if not below or equal)**
- 3) : JNA(Jump if not above)
- 4) : No Jump for This

Correct Option : 2 From : Lecture 9

Question # 73

Numbers of any size can be added using a proper combination of _____.

- 1) : **ADD and ADC**
- 2) : ABD and ADC3) :
ADC and ADC
- 4) : None of the Given

Correct Option : 1 From : Lecture 11

Question # 74

Like addition with carry there is an instruction to subtract with borrows called _____.

- 1) : SwB
- 2) : **SBB**

- 3) : SBC
- 4) : SBBC

Correct Option : 2 From : Lecture 11

Question # 75

if "and ax, bx" instruction is given, There are _____ operations as a result

- 1) : **16**
- AND** 2) : 17
- AND

- 3) : 32 AND
- 4) : 8 AND

Correct Option : 1 From : Lecture 12

1. Assembly language is not a low level language.

a. True

b. **False**

2. In case of COM File first command parameter is stored at _____ offset of program segment prefix.

a. **0x80 (Not Confirm)**

b. 0x82

c. 0x84

d. 0x86

3. Address always goes from

a. Processor to meory

b. **Memory to processor**

c. Memory to memory

d. None of the above

4. The source register in OUT is

a. **AL or AX**

b. BL or BX

c. CL or CX

d. DL or DX

5. By default CS is associated with

a. SS

b. BP

c. CX

d. **IP**

6. Which of the following pins of parallel port are grounded

a. 10-18

b. **18-25**

c. 25-32

d. 32-39

7. In the instruction `mov word [es:160], 0x1230, 30` represents the character a. A

b. B

c. **0**

d. 1

8. On executing `0x21 0x3D`, if file cant be opened then

- a. **CF will contain 1**
- b. CF will contain 0
- c. ZF will contain 1
- d. ZF will contain 0

9. Which of the following IRQ is cascading interrupt

- a. IRQ 0
- b. IRQ 1
- c. **IRQ 2**
- d. IRQ 3

10. The execution of instruction mov word [es:160], 0x1230, will print a character on the screen at

- a. **First column of second row**
- b. Second column of first row
- c. Second column of second row
- d. First column of third row

1)))SHR and SAL are same?

.True (correct)

.False

2)))mov ax,0 will set ZF flag

.True

.False

3)))In 9 pin DB connector ,which pic is assigned to TD.

. 1

. 2

. 3(correct)

. 4

4)))Lower 16 bits of EAX are labeled as

. **AX(correct)**

. BX

.EAX

.none of above

5))) which is the special prefix used for repeating a block

.rep(correct)

.repeat

.repb

.repe

6)) JA can not after cmp if unsigned destination is greater than source

.true

.false

Q=1 Conditional jump can only:

1. Far
2. **short**
3. near
4. all of the given

q=2:

Address is always go from:

1. Processor to memory
2. **Memory to processor**
3. Memory to memory
4. None of given

Q=3;

Programmable interrupt controllers have two ports 20 and 21.....port 20 is a control port while port 21 is

1. The interrupt make register
2. **Interrupt port**
3. Output port
4. Input port

Q=4:

In the instruction "move word[es:160],0x1230 represent the charechter.....

1..... A

2.....

B

3. 0

4. 1

Q=5:

The 8088 processor divides interrupts into how many classes?

1. 2

2. 3

3.

4.

4. 5

Q=6:

Which of the following is the pair of register used to access memory in string instruction?

1. DI and BP

2. SI and BP

3. **DI and SI**

4. DS and SI

Q=7:

In case of COM file, first command line parameter is stored at offset of program segment prefix'

1. **0x80** 2. 0x8

2

3. 0x84

4. 0x86

Q=8:

The INT 0x13 service 0x03 is use to ...

1. Read disk sector

2. **Write disk sector**

3. Reset disk sector

4. Get drive parameters

Q=9:

After the execution of STOSWB, the CX will be.....

1. Incremented by 1

2. **Incremented by 2**
3. Decremented by 1
4. Decremented by 2

Q=10

The execution of the instruction "mov word [ES:160],0x1230" will print a character on the screen at:

1. **First column of second row**
2. Second column of first row
3. Second column of second row
4. First column of third row

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

CS401 QUIZ 1 SOLVED

Provide by VU Answer

1. In programmable interrupt controller which of the following ports is referred as interruptmask register?

- a) 19
- b) 20
- c) 21**
- d) 22

2. ____ is the highest priority interrupt in interrupt controller

- a) IRQ 0**
- b) IRQ 1
- c) IRQ 2
- d) IRQ 3

3. IRET returns on the basis of ____ and ____

- a) CS, IP**
- b) DS, IP
- c) CS, SS

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

d) IP, SP

4. If ___ is set, the after every instruction a type 1 interrupt will be automatically generated.

a) Parity flag

b) Trap flag

c) Carry flag

d) Overflow flag

5. The interrupt handler uses ___ instruction to return back to the caller.

a) IRET

b) RETI

c) INTR

d) RET

6. Which of the following interrupt is used for Arithmetic overflow?

a) INT 1

b) INT 2

c) INT 3

d) INT 4

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

7. During the execution of INT instruction, some contents are pushed on to the stack, the order of pushing then is

- a) CS, IP and then FLAGS register
- b) IP, CS and then FLAGS register
- c) FLAGS register, CS and then IP**
- d) FLAGS register, IP and the CS

8. Each of the bits at port _corresponding to one of the IRQ lines.

- a) 18
- b) 20
- c) 21**
- b) 19

9. Which of the following pins of the parallel port connector are grounded? a. a) 10-18

- b) 18-25**
- c) 25-32
- d) 32-39

10. Which of the following interrupts plays the most significant part during step debugging of a program?

- a) INT 0

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

b) INT 1

c) INT 2

d) INT 3

11. Which of the following IRQs is connected to serial port COM 2?

a) IRQ 0

b) IRQ 1

c) IRQ 2

d) IRQ 3

12. Which of the following IRQs is connected to serial port COM 1?

a) IRQ 4

b) IRQ 5

c) IRQ 6

d) IRQ 7

13. Which of the following IRQs is used by the parallel port?

a) IRQ 4

b) IRQ 5

c) IRQ 6

d) IRQ 7

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

14. In programmable interrupt controller which of the following ports is used to selectively enabling or disabling interrupts?

- a) 19
- b) 20
- c) 21**
- d) 22

15. The space where all the registers of task are stored is called ____

- a) Control block
- b) Process control block**
- c) Stack
- d) Memory

16. Which of the following is the destination register in IN instruction?

- a) AL or AX**
- b) BL or BX
- c) CL or CX
- d) DL or DX

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

17. In multitasking environment, which of the following structure is used to maintain the ordering of active PCBs?

- a) Array
- b) Register
- c) Linked List**
- d) Stack

18. In multitasking which of the following interrupts is used as scheduler during context switching?

- a) INT 21
- b) INT 16
- c) INT 13
- d) INT 8**

19. Which of the following are required for thread entry?

- a) CS and DS
- b) CS and IP**
- c) IP and general-purpose registers
- d) SS and SP

20. The time interval between two timer tick is?

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

- a) 40 ms
- b) 45 ms
- c) 50 ms
- d) 55 ms**

21. Which of the following flag can be used for mathematical operations?

- a) Direction Flag
- b) Carry Flag**
- c) Trap Flag
- d) Interrupt Flag

22. In multi-tasking, the process of saving and restoring to values of registers from a process control block (PCB) is called ____

- a) Context switching**
- b) Context saving
- c) Context restoring
- d) Code switching

23. When two devices in a system want to use the same IRQ line, is referred as:

- a) IRQ Collision
- b) IRQ Conflict**

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

- c) IRQ Crash
- d) IRQ Blockage

24. An End of Interrupt (EOI) signal is sent by

- a) Handler**
- b) Processor
- c) IRQ
- d) PIC

26. Threads can have function calls, parameters and _____ variables.

- a) Global
- b) Local**
- c) Legal
- d) Illegal

27. Which of the following arranges jobs in a sequence in order to be executed?

- a) Process control block**
- b) Arranger
- c) Control unit
- d) Scheduler

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

28. Which of the following flags cannot be cleared using an assembly instruction?

- a) Trap flag
- b) Interrupt flag
- c) Direction flag
- d) Carry flag**

29. Which of the following is the highest priority interrupt?

- a) INT3
- b) INT2
- c) INT1
- d) INT0**

30. At the end of servicing an interrupt, which of the following is used to inform the PIC that it is completed?

- a) RET
- b) EOI**
- c) IRET
- d) RET N

31. Which of the following interrupts is used for maintaining the system time?

- a) INT 0

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

b) INT 1

c) INT 8

d) INT 10

32. Which of the following is used for exporting parallel port services?

a) INT 17

b) INT 16

c) INT 15

d) INT 8

33. In programmable interrupt controller which of the following ports is referred as a control port?

a) 19

b) 20

c) 21

d) 22

34. ___is/are the port number(s) for parallel port.

a) 20 and 21

b) 60 to 64

c) 380

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

d) 378

35. The number of pins in a parallel port connector are?

a) 20

b) 25

c) 30

d) 35

36. The offset address of an interrupt n will be at

a) n

b) nx2

c) nx3

d) nx4

37. Programmable interrupt controller has two ports (20 and 21). Port 20 is the control port while port 21 is_.

a) The interrupt mask register

b) Interrupt port

c) Output port

d) Input port

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

40. Which of the following instruction selects memory address space?

a) MOV

b) DEC

c) IN

d) ADD

41. The thread registration code initializes the PCB and adds it to the linked list so that the ___ will give it a turn.

a) Assembler

b) Scheduler

c) Linker

d) Debugger

42. Which of the following instruction is used for disabling all the interrupts during a program execution?

a) Cli

b) Sti

c) Reti

d) Iret

43. Programmable interrupt controller (PIC) has

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

- a) One input signal and eight output signals
- b) One input signal and one output signal
- c) Eight input signals and one output signals**
- d) Eight input signals and eight output signals

44. In 8088 processor, there can be total possible entries in an interrupt vector table.

- a) 256**
- b) 64
- c) 128
- d) 512

45. Each thread can have their own

- a) Execution area
- b) Stack**
- c) Memory
- d) Array

46. The parallel port connector is called?

- a) DB-25**
- b) BD-25
- c) DB-24

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

d) BD-24

47. In 8088, the physical memory address for Interrupt Vector Table is fixed and the table occupies exactly ____ of memory.

a) 1 KB

b) 1 MB

c) 64 KB

d) 4 Bytes

48. Which of the following interrupts is non maskable interrupt?

a) INT 0

b) INT 1

c) INT 2

d) INT 3

49. Using OUT instruction on parallel data port results into a signal of ____ for every 1 bit.

a) 0 V

b) 1 V

c) 5 V

d) 10 V

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

50. PCB stands for?

a) Process Control Block

b) Process Cleaning Block

c) Programmable Counter Block

d) Programs Control Block

51. Which of the following port number is used to send an end of interrupt (EOI) signal to the PIC after an interrupt is ended?

a) 0x16

b) 0x20

c) 0x60

d) 0x378

52. Which of the following instruction is used for reading a char from keyboard?

a) out al, 0x60

b) in al, 0x60

c) out dx, al

d) out dx, 0x378

53. In PIC, which of the following port is used for selectively enabling or disabling interrupts?

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

a) 19

b) 20

c) 21

d) 22

54. DOS has single entry point to access all its services through interrupt__

a) INT 21

b) INT 16

c) INT 2

d) INT 10

55. Which of the following pins of parallel port are used to carry the data from processor to the device connected through parallel port?

a) Pin 2 to pin 9

b) Pin 10

c) Pin 4

d) Pin 18 to pin 25

56. The input frequency of the programmable interval timer (PIT) is

a) Fixed

b) Depends on processor block

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

- c) Variable
- d) Depends on hardware attached

60. All registers of task are stored in space called ____

- a) Process storage block
- b) Process control block**
- c) Process terminal
- d) Swapping space

61. Which of the following IRQ's is derived by a key board

- a) IRQ 0
- b) IRQ 1**
- c) IRQ 2
- d) IRQ 3

62. Changing and restoring the state of Control processing unit (CPU) is referred as

- a) Thrashing
- b) Multitasking**
- c) Context Switching
- d) Multithreading

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

63. For the external hardware to generate an interrupt there are how many pins outside the processor?

a) 1

b) 2

c) 3

d) 4

64. Which of the following instruction selects peripheral address space?

a) MOV

b) DEC

c) IN

d) ADD

65. Which of the following is the most commonly used port with printer?

a) Serial Port

b) USB port

c) Parallel Port

d) DVI Port

66. ___ allows us to execute one instruction at a time rather than the whole program at once

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

a) INT 0

b) INT 1

c) INT 2

d) INT 3

67. Which of the following is used for exporting keyboard services?

a) INT 16

b) INT 11

c) INT 12

d) INT 8

68. TSR stands for

a) Terminate and store routines

b) Transmits and save resources

c) Terminate and Stay resident

d) Truncate and start recursively

69. Which of the following is the ACK pin in DB-25 Connector

a) 10

b) 11

c) 12

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

d) 13

70. Which of the following IRQs is a cascading interrupt?

a) IRQ 0

b) IRQ 1

c) IRQ 2

d) IRQ 3

71. In interrupt masking, we use__to get control from the program without letting the program know about it.

a) IRQ 0

b) IRQ 1

c) IRQ 5

d) IRQ 7

72. The output of programmable timer interval is connected to which line of the PIC?

a) IRQ 0

b) IRQ 1

c) IRQ 2

d) IRQ 3

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

73. Which of the following IRQs is used for Floppy disk drive?

- a) IRQ 4
- b) IRQ 5
- c) IRQ 6**
- d) IRQ 7

74. In multitasking, each PCB requires__bytes to store the values of registers.

- a) 14
- b) 28**
- c) 32
- d) 16

75. Which of the following IRQs is used for sound card or network card?

- a) IRQ 4
- b) IRQ 5**
- c) IRQ 6
- d) IRQ 7

76. Programmable interrupt controller has____ports.

- a) 1
- b) 2**
- c) 3

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

d) 4

77. Which of the following bits of parallel control port enables the IRQ 7 triggering by the ACKpin?

a) Bit 1

b) Bit 2

c) Bit 3

d) Bit 4

78. Which of the following IRQs is derived by a timer device?

a) IRQ 0

b) IRQ 1

c) IRQ 2

d) IRQ 3

79. ___ is a special type of interrupt that returns to the same instruction instead of the next instruction

a) Divide overflow

b) Debug interrupt

c) Arithmetic overflow interrupt

d) Change the sign interrupt

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

80. All the registers & stack are saved in

a) Multitasking

b) Multi-Processing

c) Function call

d) BIOS

81. Which of the following is the source register in OUT instruction?

a) AL or AX

b) BL or BX

c) CL or CX

d) DL or DX

82. Which of the following is the destination register in IN instruction?

a) AL or AX

b) BL or BX

c) CL or CX

d) DL or DX

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

VISIT ANY MORE FREELY SOLUTIONS

VUAnswer.com

Note:

If you found any mistake in quiz correct on it.

Get more solved quizzes, assignment, past paper and vu solutions free to visit our website **VUAnswer.com**.

For help and suggestions contact us freely.

REGARD SARIM

WHATSAPP [+923162965677](https://www.whatsapp.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q)

VU Answer
your right path

REMEMBER IN YOUR PRECIOUS PRAYERS

SUBSCRIBE OUR CHANNEL

<https://www.youtube.com/channel/UCvFS8fAv4bvZsOmWuWMIg3Q>

CS401 MCQs Collection BY SONU MUGHAL

CS401- Computer Architecture and Assembly
Language

- ☐ 1 argument
- ☐ 1 argument
- ☐ 3 arguments
- ☒ No arguments

Question No: 2 (M - 1)

In STOSB instruction SI is decremented or incremented by

4

1

2

3

Question No: 3 (M - 1)

CMPS instruction subtracts the source location to the destination location.

Destination location always lies in

DS:SI

DS:DI

ES:SI

ES:DI

Question No: 4 (M - 1) .

Regarding assembler, which statement is true:

Assembler converts mnemonics to the corresponding OPCODE

Assembler converts OPCODE to the corresponding mnemonics.

Assembler executes the assembly code all at once

Assembler executes the assembly code step by step

Question No: 5 (M - 1) .

If "BB" is the OPCODE of the instruction which states to "move a constant value to AX register", the hexadecimal representation (Using little Endian notation) of the instruction "Mov AX,336" ("150" in hexadecimal number system) will be:

0xBB0150

0x5001BB

0x01BB50

0xBB5001

Question No: 6

(M - 1) .

In the instruction MOV AX, 5 the number of operands are

1

2

3

4

Question No: 7

(M - 1) .

The maximum parameters a subroutine can receive (with the help of registers) are

6

7

8

9

Question No: 8

(M - 1) .

In assembly the CX register is used normally as a _____ register.

source

counter

index

pointer

Question No: 9

(M - 1) .

All the addressing mechanisms in iAPX 8 8 return a number called _____ address .

effective

faulty

indirect

direct

Question No: 10 (M - 1) .

When a 16 bit number is divided by an 8 bit number, the dividend will be in

AX

BX

CX

DX

Question No: 11 (M - 1) .

in Left-Shift-Operation the left most bit _____

will drop

will go into CF

Will come to the right most

will be always 1

Question No: 12 (M - 1) .

Suppose the decimal number "35" after shifting its binary two bits to left, the new value becomes _____

35

70

140

17

Question No: 13

(M - 1) .

When divide overflow occurs processor will be interrupted this type of interrupt is called

Hardware interrupt

Software interrupt

Processor exception

Logical interrupts

Question No: 14

(M - 1) .

Which mathematical operation is dominant during the execution of SCAS instruction

Division

Multiplication

Addition

Subtraction

Question No: 15

(M - 1) .

After the execution of REP instruction CX will be decremented then which of the following flags will be affected?

CF

OF

DF

No flags will be affected

Question No: 16 (M - 1)

_____ is one of the reasons due to which string instructions are used in 8088

Efficiency and accuracy

Reduction in code size and accuracy

Reduction in code size and speed

Reduction in code size and efficiency

Question No: 17 (M - 1)

Write any two control instructions.

Question No: 18 (M - 1)

RET instruction take how many arguments

Question No: 19 (M - 2)

Explain the function of rotate right (ROR) instruction

Question No: 20 (M - 2)

Describe the PUSH function

Question No: 21 (M - 3)

Write down the names of four segment registers?

Question No: 22 (M - 3)

For what purpose "INT 4" is reserved?

Question No: 23 (M - 5)

Given that $[BX+0x0100]$ $BX=0x0100$

$DS=0xFFFF0$

Calculate the physical address

Question No: 1 (M - 1) :

Th

the physical address of the stack is obtained by

► **SS:SP combination**

► SS:SI combination

► **SS:SP combination**

► ES:BP combination

► ES:SP combination

Question No: 2 (M - 1)

Aft

After the execution of instruction "RET "

► **SP is incremented by 2**

► **SP is incremented by 2**

► SP is decremented by 2

► SP is incremented by 1

► SP is decremented by 1

Question No: 3 (M - 1)

Th

The second byte in the word designated for one screen location holds

► **Character color on the screen**

► The dimensions of the screen

► Character position on the screen

► **Character color on the screen**

- ▶ ASCII code of the character

Question No: 4 (M - 1) :

RE

P will always

- ▶ **Decrement CX by 1**
- ▶ Increment CX by 1
- ▶ Increment CX by 2
- ▶ **Decrement CX by 1**
- ▶ Decrement CX by 2

Question No: 5 (M - 1) :

Th

e basic function of SCAS instruction is to

- ▶ **Compare**
- ▶ **Compare**
- ▶ Scan
- ▶ Sort
- ▶ Move data

Question No: 6 (M - 1) :

Ind

ex registers are used to store _____

- ▶ **Address**
- ▶ Data
- ▶ Intermediate result

- ▶ **Address**
- ▶ Both data and addresses

Question No: 7 (M - 1) .

e bits of the _____ work independently and individually

Th

- ▶ **flags register**
- ▶ index register
- ▶ base register
- ▶ **flags register**
- ▶ accumulator

Question No: 8 (M - 1) .

convert any digit to its ASCII representation

To

- ▶ **Add 0x30 in the digit**
- ▶ **Add 0x30 in the digit**
- ▶ Subtract 0x30 from the digit
- ▶ Add 0x61 in the digit
- ▶ Subtract 0x61 from the digit

Question No: 9 (M - 1) .

hen a 32 bit number is divided by a 16 bit number, the quotient is of

W

- ▶ **4 bits**
- ▶ 32 bits
- ▶ 16 bits
- ▶ 8 bits
- ▶ **4 bits**

Question No: 10 (M - 1) .

W

When a 16 bit number is divided by an 8 bit number, the quotient will be in

► **AL**

- AX
- **AL**
- AH
- DX

Question No: 11 (M - 1) .

W

Which mathematical operation is dominant during the execution of SCAS instruction

- **Division**
- **Division**
- Multiplication
- Addition
- Subtraction

Question No: 12 (M - 1) .

If

AX contains decimal -2 and BX contains decimal 2 then after the execution of instructions:

`CMP AX, BX`

`JA label`

- **Zero flag will set**
- Jump will be taken
- **Zero flag will set**
- ZF will contain value -4
- Jump will not be taken

Question No: 13 (M - 1)

The execution of the instruction "mov word [ES : 160], 0x1230" will print a character "0" on the screen at

- ▶ First column of second row
- ▶ Second column of first row
- ▶ First column of second row
- ▶ Second column of second row
- ▶ First column of third row

Question No: 14 (M - 1)

If the direction of the processing of a string is from higher addresses towards lower addresses then

- ▶ DF is cleared
- ▶ ZF is cleared
- ▶ DF is cleared
- ▶ ZF is set
- ▶ DF is set

Question No: 15 (M - 1)

The instruction ADC has _____ Operand(s)

- ▶ 3
- ▶ 0
- ▶ 1
- ▶ 2
- ▶ 3

Question No: 16 (M - 1)

Which bit of the attributes byte represents the red component of background color ?

- ▶ 3

- ▶ 3
- ▶ 4
- ▶ 5
- ▶ 6

Question No: 17 (M - 2)

W

What is difference between SHR and SAR instructions?

SHR

The SHR inserts a zero from the left and moves every bit one position to the right and copy the rightmost bit in the carry flag.

SAR

The SAR shift every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket. The sign bit is retained in this operation.

Question No: 18 (M - 2)

Fo

For what purpose "INT 1" is reserved ?

Question No: 19 (M - 2)

De

Define implied operand?

It is always in a particular register say the accumulator. It needs to not be mentioned in the instruction.

Q=1:

Which bit of attributes byte represents the blue component of foreground color?

- ☐ 0
- ☐ 1
- ☐ 2
- ☐ 3

Q=2:

The clear screen operation initializes the whole block of video memory to:

- ☐ 0417
- ☐ 0714
- ☐ 0741
- ☐ 017

Q=3:

When the operand of DIV instruction is of 16 bit then implied dividend will be of

- ☐ 64-bit
- ☐ 32-bits
- ☐ 16-bits
- ☐ 8--bits

Q=4

Which of the following is the pair of register used to access memory instring instruction:

- ☐ DI and BP
- ☐ SI and BP
- ☐ DI and SI
- ☐ DS and Si

Q=5

A fat32 file system directory entry in DOS consist of how many bytes?

- ☐ 16
- ☐ 24
- ☐ 32
- ☐ 64

Q=6:

Which register is generally used to specify the services number of an interrupt?

DX

AX

BX

CX

.....

Q=7:

In 9 pin db 9 connector ,which pin is assigned to RD(received data)

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4

Q=8

In case of COM file, maximum length of parameters passed through command line can be.....

- ☐ 63 bytes
- ☐ 127bytes
- ☐ 255 bytes
- ☐ 511 bytes

Q=9

We can access the DOS service using;

- ☐ Int 0x21
- ☐ Int 0x13
- ☐ Int 0x 10
- ☐ Int 0x 08

Q=10

In 9 pin 9 connector,which pin is assigned to signal ground

- ☐ 3
- ☐ 4
- ☐ 5
- ☐ 6

Q=11:

BPB stands for

- ☐ Basic parameter block
- ☐ Bios precise block
- ☐ Basic precise block
- ☐ Bios parameter block

Q=12

Int 13-bios disk service “generally uses which register to return the error flag?

- ☐ CF
- ☐ DL
- ☐ AH
- ☐ AL

Q=13:

The first sector on the hard disk contains the

- ☐ Hard disk size
- ☐ Partition table
- ☐ Data size
- ☐ Sector size

Q=14

Operating system organize data in the form of

- ☐ Folder
- ☐ Batch file
- ☐ File

- ☐ None of above

.....

Q=15

In 9 pin db 9 connector, which pin is assigned to TD(transmitted data)

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4

Q=16"

Device derive can be divided into -----major categories.

- ☐ 5
- ☐ 4
- ☐ 3
- ☐ 2

1. BL contains 5 decimal then after right shift , BL will become

- ☐ 3
- ☐ 2.5
- ☐ 5
- ☐ 10

2. 8 * 16 font is stored in _____ bytes.

- ☐ 3
- ☐ 4
- ☐ 8
- ☐ 16

3. In DOS input buffer , number of characters actually read on return is stored in

- ☐ First byte
- ☒ **Second byte**
- ☐ Third byte
- ☐ Fourth byte

4. IRQ 0 has priority

- ☐ Low
- ☐ High
- ☒ **Highest**
- ☐ Medium

5. Thread registration code initialize PCB and add to linked list so that _____ will give it turn.

- ☐ Assembler
- ☐ Linker
- ☒ **Scheduler**
- ☐ Debugger

6. Traditional calling conventions are in _____ number

- ☐ 1
- ☒ **2**
- ☐ 3
- ☐ 4

7. VESA VEB 2.0 is standard for

- ☒ **High Resolution Mode**
- ☐ Low Resolution Mode
- ☐ Very High Resolution Mode
- ☐ Medium Resolution Mode

8. To clear direction flag which instruction is used

- ☒ **Cld**

- ☐ ☐ ☐ ☐ Clrd
- ☐ ☐ ☐ ☐ Cl df
- ☐ ☐ ☐ ☐ Clr df

9. In STOSW instruction , When DI is cleared , SI is

- ☐ ☐ ☐ ☐ Incremented by 1
- ☐ ☐ ☐ ☐ **Incremented by 2**
- ☐ ☐ ☐ ☐ Decremented by 1
- ☐ ☐ ☐ ☐ Decremented by 2

10. Interrupt that is used in debugging with help of trap flag is

- ☐ ☐ ☐ ☐ INT 0
- ☐ ☐ ☐ ☐ **INT 1**
- ☐ ☐ ☐ ☐ INT 2
- ☐ ☐ ☐ ☐ INT 3

11. INT for arithmetic overflow is

- ☐ ☐ ☐ ☐ INT 1
- ☐ ☐ ☐ ☐ INT 2
- ☐ ☐ ☐ ☐ INT 3
- ☐ ☐ ☐ ☐ **INT 4**

12. IRQ referred as

- ☐ ☐ ☐ ☐ **Eight Input signals**
- ☐ ☐ ☐ ☐ One Input signal
- ☐ ☐ ☐ ☐ Eight Output signals
- ☐ ☐ ☐ ☐ One output signal

13. IRQ for keyboard is 1

14. IRQ for sound card is 5

15. IRQ for floppy disk is 6

16. IRQ with highest priority is

- ☐ Keyboard IRQ
- ☒ Timer IRQ
- ☐ Sound Card
- ☐ Floppy Disk

17. Pin for parallel port ground is

- ☐ 10-18
- ☒ 18-25
- ☐ 25-32
- ☐ 32-39

18. The physical address of Interrupt Descriptor Table (IDT) is stored in

- ☐ GDTR
- ☒ IDTR
- ☐ IVT
- ☐ IDTT

19. Execution of "RET 2" results in?

20. CX register is

- ☒ Count register
- ☐ Data register
- ☐ Index register
- ☐ Base register

21. OUT instruction uses AX as source register.

22. IN DB-9 connector the Data Set ready pin is at

- ☐ 5
- ☒ 6

- ☐ ☐ ☐ ☐ 7
- ☐ ☐ ☐ ☐ 8

23. If two devices uses same IRQ then there is

- ☐ ☐ ☐ ☐ IRQ collision
- ☐ ☐ ☐ ☐ IRQ conflict
- ☐ ☐ ☐ ☐ IRQ drop

24. VESA organizes 16 bit color for every pixel in ratio

- ☐ ☐ ☐ ☐ 5:5:5
- ☐ ☐ ☐ ☐ 5:6:5
- ☐ ☐ ☐ ☐ 6:5:6
- ☐ ☐ ☐ ☐ 5:6:7

25. Division by zero is done by which interrupt.

Interrupt 0.

Question No: 1 (M - 1) :

er the execution of SAR instruction

Aft

▶ The msb is replaced by a 0

▶ The msb is replaced by 1

▶ The msb retains its original value

▶ The msb is replaced by the value of CF

Question No: 2 (M - 1) :

RE

TF will pop the offset in the

- ▶ BP
- ▶ IP
- ▶ SP
- ▶ SI

Question No: 3 (M - 1) .

Th

e routine that executes in response to an INT instruction is called

- ▶ ISR
- ▶ IRS
- ▶ ISP
- ▶ IRT

Question No: 4 (M - 1) .

Th

e first instruction of "COM" file must be at offset:

- ▶ 0x0010
- ▶ 0x0100
- ▶ 0x1000
- ▶ 0x0000

Question No: 5 (M - 1) .

"F

ar" jump is not position relative but is _____

- ▶ memory dependent
- ▶ Absolute
- ▶ temporary
- ▶ indirect

Question No: 6 (M - 1)

On

ly _____ instructions allow moving data from memory to memory.

► string

- word
- indirect
- stack

Question No: 7 (M - 1)

Aft

er the execution of instruction "RET 2"

► SP is incremented by 2

- SP is decremented by 2
- SP is incremented by 4
- SP is decremented by 4

Question No: 8 (M - 1)

DI

V instruction has

► Two forms

- Three forms
- Four forms
- Five forms

Question No: 9 (M - 1)

W

When the operand of DIV instruction is of 16 bits then implied dividend will be of

- ▶ 8 bits
- ▶ 16 bits
- ▶ **32 bits**
- ▶ 64 bits

Question No: 10 (M - 1)

Aft

After the execution of MOVS instruction which of the following registers are updated

- ▶ SI only
- ▶ DI only
- ▶ **SI and DI only**
- ▶ SI, DI and BP only

Question No: 11 (M - 1)

In

In 8088 architecture, whenever an element is pushed on the stack

- ▶ SP is decremented by 1
- ▶ **SP is decremented by 2**
- ▶ SP is decremented by 3
- ▶ SP is decremented by 4

Question No: 12 (M - 1)

W

When a very large number is divided by a very small number so that the quotient is larger than the space provided, this is called

- ▶ Divide logical error
- ▶ **Divide overflow error**

- ▶ Divide syntax error
- ▶ An illegal instruction

Question No: 13 (M - 1) :

In the word designated for one screen location, the higher address contains

▶ The character code

- ▶ The attribute byte
- ▶ The parameters
- ▶ The dimensions

Question No: 14 (M - 1) :

Which of the following options contain the set of instructions to open a window to the video memory?

- ▶ `mov AX, 0xb008`

`mov ES, AX`

▶ mov AX, 0xb800

mov ES, AX

- ▶ `mov AX, 0x8b00`

`mov ES, AX`

- ▶ `mov AX, 0x800b`

`mov ES, AX`

Question No: 15 (M - 1) :

In a video memory, each screen location corresponds to

- ▶ One byte
- ▶ Two bytes**
- ▶ Four bytes
- ▶ Eight bytes

Question No: 16 (M - 1) :

the execution of the instruction "mov word [ES : 0], 0x0741" will print character "A" on screen , background color of the screen will be

► **Black**

► White

► Red

► Blue

Question No: 1 ____ (M - 1) .

Which of the following is not true about registers?

1. Their operation is very much like memory
2. Intermediate results may also be stored in registers.
3. They are also called scratch pad ram
4. None of given options.

Question No: 2 ____ (M - 1) .

move [bp], al moves the one byte content of the AL register to the address contained in BP register in the current

1. Stack segment
2. Code segment
3. Data segment
4. Extra segment

Question No: 3 (M - 1) .

In a rotate through carry right (RCR) instruction applied on a 16 bit word
Effectively there is

1. 16 bits rotation
2. 1 bit rotation
3. 17 bits rotation
4. 8 bits rotation

Question No: 4__ (M - 1) - Please

choose one The 8088 stack works on

1. Word sized elements
2. Byte sized elements
3. Double sized element
4. Nibble sized element

Question No: 5 (M - 1) - Please

choose one

An 8 x 16 font is stored in.....Bytes

1. 2
2. 4
3. 8
4. 16

Question No: 6 (M - 1) - Please

INT 10 is used for.....services.

1. RAM
2. Disk
3. BIOS video

4. DOS video

Question No: 7 __ (M - 1) :

Priority of IRQ 0 interrupt is

1. medium
2. high
3. highest
4. low

Question No: 8 __ (M - 1) :

Threads can have function calls, parameters and _____ variables.

1. global
2. local
3. legal
4. illegal

Question No: 9 __ (M - 1) - Please choose

one How many prevalent calling conventions do.....exist

1. 1
2. 2
3. 3
4. 4

Question No: 10 (M - 1) - Please choose

one In 9pin DB 9 DSR is assigned on pin number

1. 4
2. 5
3. 6
4. 7

Question No: 11

(M - 1) - Please

choose one In 9pin DB 9 CTS is assigned on pin number

1.	6
2.	7
3.	8
4.	9

Question No: 12__ (M - 1) .

In 9pin DB 9 CD is assigned on pin number

1.	1
2.	2
3.	3
4.	4

Question No: 13__ (M - 1) :

In 9pin DB 9 RD is assigned on pin number

- [illegible]

Question No: 14 ____ (M - 1) .

in device attribute word which of the following bit decides whether it is a character

1. device or a block device
2. Bit 12 Bit 13

3. Bit 14
4. Bit 15

Question No: 15__ (M - 1) :

Video services are classified into _____ broad categories

- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 5

Question No: 16 (M - 1) - Please choose

one In STOSB instruction, when DF is clear, SI is

1. Incremented by 1
2. Incremented by 2
3. Decrement by 1
4. Decrement by 2

Question No: 17 (M - 1) . The

process of sending signals back and forth is called

1. Activity
2. Hand-shaking
3. Interruption
4. Time clicking

Question No: 18 (M - 1) :

which of the following is a special type of interrupt that returns to the same instruction instead of the next instruction

1. Divide overflow interrupt
2. Debug interrupt
3. Arithmetic overflow interrupt
4. Change of sign interrupt

Question No: 19 __ (M - 1) .

Which of the following IRQs is derived by a timer device?

1. IRQ 0
2. IRQ 1
3. IRQ 2
4. IRQ 3

Question No: 20 __ (M - 1) .

Which of the following interrupts is used for Arithmetic overflow

1. INT 1
2. INT 2
3. INT 3
4. INT 4

Question No: 21 __ (M - 1) .

Which of the following IRQs is connected to serial port COM 2?

1. IRQ 0
2. IRQ 1
3. IRQ 2
4. IRQ 3

Question No: 22 __ (M - 1) - Please

choose one

An End of Interrupt (EOI) signal is sent by

1. Handler
2. Processor
3. IRQ
4. PIC

Question No: 23 (M - 1) .

The source registers in OUT is

1. AL or AX
2. BL or BX
3. CL or CX
4. DL or DX

Question No: 24 (M - 1) .

In programmable interrupt controller which of the following ports is used for selectively enabling or disabling interrupts

1. 19
2. 20
3. 21
4. 22

Question No: 25 (M - 1) .

The number of pins in a parallel port connector are?

1. 25
2. 30
3. 35

Question No: 26 (M - 1) .

Which of the following pins of a parallel port connector are grounded?

1. 10-18
2. 18-25
3. 25-32
4. 32-39

Question No: 27 __ (M - 1) .

Suppose a decimal number 35 when its binary is shifted to write two places the new number will become

1. 35
2. 70
3. 140
4. 17

Question No: 28 __ (M - 1) .

A 32bit address register can access upto of memory so memory access has increased a lot.

1. 2GB
2. 4GB
3. 6GB
4. 8GB

Question No: 29 __ (M - 1) .

In NASM an imported symbol is declared with the while and exported symbol is declared with the

1. Global directive, External directive
2. External directive, Global directive
3. Home Directive, Foreign Directive
4. Foreign Directive, Home Directive

Question No: 30 (M - 1) - Please choose

one Single step interrupt is

1. Hardware interrupt
2. Like divide by zero interrupt

3. Like divide by 1 interrupt
4. Software interrupt

Question No: 31 __ (M - 1)

Which services are gained bi INT 0x16

Solution:

Hardware interrupt

Like divide by zero interrupt

Like divide by 1 interrupt

Software interrupt

Question No: 32 (M - 1

Give the name of any one VESA servic

- ☐ Hardware interrupt
- ☐ Like divide by zero interrupt
- ☐ Like divide by 1 interrupt
- ☐ Software interrupt

Question No: 33 (M - 2)

INT 14 - SERIAL - READ CHARACTER FROM PORT

By using above port what do AH,AL and DX shows here?

- ☐ Hardware interrupt
- ☐ Like divide by zero interrupt
- ☐ Like divide by 1 interrupt
- ☐ Software interrupt

Question No: 34 (M - 2)

What do these instructions do ? write your answer in single line.

mov cx, 0xffff

loop \$

- ☐ Hardware interrupt
- ☐ Like divide by zero interrupt
- ☐ Like divide by 1 interrupt

- Question No: 35 (M - 3)**
Define the protected mode
Solution:

- Question No: 36 (M - 3)**
Write a program in assembly language to disable keyboard interrupt using PIC mask register
Hint: Only five instructions are needed
Solution:

- Question No: 37 (M - 3)**
Read the following passage carefully and fill the blanks with proper words.
Note: Don't rewrite the passage just write the words in same order.
 "BIOS sees the disks as a combination of sectors, tracks, and....., as a raw storage device without concern to whether it is reading a file or directory.
 provides the simplest and most powerful interface to the storage medium. However this raw storage is meaningless to the user who needs to store his files and organize them into..... . "
Solution:

- Question No: 1 (M - 1)**

1

Sun SPARC Processor has a fixed _____ instruction size.

1. 16bit
2. 32bit
3. 64bit
4. 20bit

Question No: 2 (M - 1)

:-

When the subprogram finishes, the _____ retrieves the return address from the stack and transfers control to that location.

1. RET instruction
2. CALL instruction
3. POP instruction
4. Jump instruction

Question No: 3 (M - 1)

:-

A 32 bit address register can access upto _____ of memory.

- ☐ 1 GB
- ☐ 6 GB
- ☐ 4 GB
- ☐ 2 GB

Question No: 4 (M - 1)

:-

The value of a segment register when the processor is running under protected mode is called

1. segment descriptor
2. segment selector
3. global descriptor table
4. protected register

Question No: 5 (M - 1)

⋮

FS and GS are two _____ in protected mode.

1. segment registers
2. segment selectors
3. stack pointers
4. register pointers

Question No: 6 (M - 1)

⋮

IRQ 0 interrupt have _____ priority

1. low
2. medium
3. highest
4. lowest

Question No: 7 (M - 1)

⋮

IDT stands for _____.

1. interrupt descriptor table
2. individual descriptor table
3. inline data table
4. interrupt descriptor table

Question No: 8 (M - 1)

:-

Every bit of line status in serial port conveys _____ information.

1. different
2. same
3. partial
4. full

Question No: 9 (M - 1)

:-

There are total _____ bytes in a standard floppy disk.

1. 1444k
2. 1440k
3. 1280k
4. 2480k

Question No: 10 (M - 1)

:-

An 8x16 font is stored in _____ bytes.

- ☐ 8
- ☐ 16
- ☐ 4
- ☐ 20

=====

. Serial Port is also accessible via I/O ports , COM 1 is accessible via ports 3F8-3FF while COM 2 is accessible via 2F8 -2FF.

The first register at 3F8 is the Transmitter holding register if written to and the receiver buffer register if read from.

Other register of our interest include 3F9 whose Bit 0 must be set to enable received data available interrupt and Bit 1 must be set to enable transmitter holding register empty interrupt.

(Transmitter, COM 1, I/O ports , COM2. bit 0 , Buffer , 3FA)

=====

Question # 1

There are three busses to communicate the processor and memory named as _____

- 1) : address bus.,data bus and data bus.
- 2) : addressing bus.,data bus and data bus.
- 3) : address bus.,datamove bus and data bus.
- 4) : address bus.,data bus and control bus..

Correct Option : 4 From : Lecture 1

Question # 2

The address bus is unidirectional and address always travels from processor to memory.

- 1) : TRUE
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 1

Question # 3

Data bus is bidirectional because _____

- 1) : To way
- 2) : Data moves from both, processor to memory and memory to processor,
- 3) : Data moves from both, processor to memory and memory to data Bus,
- 4) : None of the Given

Correct Option : 3 From : Lecture 1

Question # 4

Control bus _____

- 1) : is Not Important.
- 2) : is Important .
- 3) : bidirectional.
- 4) : unidirectional .

Correct Option : 3 From : Lecture 1

Question # 5

A memory cell is an n-bit location to store data, normally _____ also called a byte

- 1) : 4-bit
- 2) : 8-bit
- 3) : 6-bit
- 4) : 80-bit

Correct Option : 2 From : Lecture 1

Question # 6

The number of bits in a cell is called the cell width. _____ define the memory completely.

- 1) : Cell width and number of cells,
- 2) : cell number and width of the cells,
- 3) : width
- 4) : Height

Correct Option : 1 From : Lecture 1

Question # 7

for memory we define two dimensions. The first dimension defines how many _____ bits are there in a single memory cell.

- 1) : parallel
- 2) : Vertical

3) : long

4) : short

Correct Option : 1 From : Lecture 1

Question # 8

_____ operation requires the same size of data bus and memory cell width.

1) : Normal

2) : Best and simplest

3) : first

4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 9

Control bus is only the mechanism. The responsibility of sending the appropriate signals on the control bus to the memory is of the _____.

1) : Data Bus

2) : processor

3) : Address Bus

4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 10

In "total: dw 0" Opcode total is a _____

1) : Literal

2) : Variable

3) : Label

4) : Starting point

Correct Option : 3 From : Lecture 10

Question # 11

| 0 | --> | 1 | 1 | 0 | 1 | 0 | 0 | 0 | --> | C | is a example of _____

1) : Shl

2) : sar

3) : Shr

4) : Sal

Correct Option : 3 From : Lecture 10

Question # 12

| C | <-- | 1 | 1 | 0 | 1 | 0 | 0 | 0 | <-- | 0 | is a example of _____

1) : Shl

2) : sar

3) : Shr

4) : Sal

Correct Option : 1 From : Lecture 10

Question # 13

ADC has _____ operands.

1) : two

2) : three

3) : Five

4) : Zero

Correct Option : 2 From : Lecture 10

Question # 14

The basic purpose of a computer is to perform operations, and operations need _____.

1) : order

- 2) : nothing
- 3) : operands
- 4) : bit

Correct Option : 3 From : Lecture 2

Question # 15

Registers are like a scratch pad ram inside the processor and their operation is very much like normal _____.

- 1) : Number
- 2) : operations
- 3) : memory cells
- 4) : None of the Given

Correct Option : 3 From : Lecture 2

Question # 16

There is a central register in every processor called the _____ and The word size of a processor is defined by the width of its _____.

- 1) : accumulator, accumulator
- 2) : data bus, accumulator
- 3) : accumulator, Address Bus
- 4) : accumulator, memory

Correct Option : 1 From : Lecture 2

Question # 17

_____ does not hold data but holds the address of data

- 1) : Pointer, Segment, or Base Register
- 2) : Pointer, Index, or Base Register
- 3) : General Registers
- 4) : Instruction Pointer

Correct Option : 2 From : Lecture 2

Question # 18

"The program counter holds the address of the next instruction to be _____"

- 1) : executed.
- 2) : called
- 3) : deleted
- 4) : copy

Correct Option : 1 From : Lecture 2

Question # 19

There are _____ types of "instruction groups"

- 1) : 4
- 2) : 5
- 3) : 3
- 4) : 2

Correct Option : 1 From : Lecture 2

Question # 20

These instructions are used to move data from one place to another.

- 1) : TRUE
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 2

Question # 21

“mov” instruction is related to the _____ *****.

- 1) : Arithmetic and Logic Instructions
- 2) : Data Movement Instructions
- 3) : Program Control Instructions
- 4) : Special Instructions

Correct Option : 2 From : Lecture 2

Question # 22

_____ allow changing specific processor behaviors and are used to play with it.

- 1) : Special Instructions
- 2) : Data Movement Instructions
- 3) : Program Control Instructions
- 4) : Arithmetic and Logic Instructions

Correct Option : 1 From : Lecture 2

Question # 23

8088 is a 16bit processor with its accumulator and all registers of _____.

- 1) : 32 bits
- 2) : 6 bits
- 3) : 16 bits
- 4) : 64 bits

Correct Option : 3 From : Lecture 2

Question # 24

The _____ of a processor means the organization and functionalities of the registers it contains and the instructions that are valid on the processor.

- 1) : Manufactures
- 2) : architecture
- 3) : Deal
- 4) : None of the Given

Correct Option : 2 From : Lecture 2

Question # 25

Intel IAPX88 Architecture is _____.

- 1) : More then 25 old
- 2) : New
- 3) : Not Good
- 4) : None of the Given

Correct Option : 1 From : Lecture 2

Question # 26

The iAPX88 architecture consists of _____ registers.

- 1) : 13
- 2) : 12
- 3) : 9
- 4) : 14

Correct Option : 4 From : Lecture 3

Question # 27

General Registers are _____.

- 1) : AX, BX, CX, and DX
- 2) : XA, BX, CX, and DX
- 3) : SS, SI and DI
- 4) : 3

Correct Option : 1 From : Lecture 3

Question # 28

AX means we are referring to the extended 16bit "A" register. Its upper and lower byte are separately accessible as _____.

- 1) : AH and AL
- 2) : A Lower and A Upper
- 3) : AL, AU
- 4) : AX

Correct Option : 1 From : Lecture 3

Question # 29

AX is General purpose Register where A stands for _____.

- 1) : Acadmic
- 2) : Ado
- 3) : Architecture
- 4) : Accumulator

Correct Option : 4 From : Lecture 3

Question # 30

The B of BX stands for _____ because of its role in memory addressing.

- 1) : Busy
- 2) : Base
- 3) : Better
- 4) : None of the Given

Correct Option : 2 From : Lecture 3

Question # 31

The D of DX stands for Destination as it acts as the destination in _____.

- 1) : I/O operations
- 2) : operations
- 3) : memory cells
- 4) : Memory I/O operations

Correct Option : 1 From : Lecture 3

Question # 32

The C of CX stands for Counter as there are certain instructions that work with an automatic count in the _____.

- 1) : DI register
- 2) : BX register
- 3) : CX register
- 4) : DX register

Correct Option : 3 From : Lecture 3

Question # 33

_____ are the index registers of the Intel architecture which hold address of data and used in memory access.

- 1) : SI and SS
- 2) : PI and DI
- 3) : SI and IP
- 4) : SI and DI

Correct Option : 4 From : Lecture 3

Question # 34

In Intel IAPX88 architecture _____ is the special register containing the address of the next instruction to be executed.

- 1) : AX
- 2) : PI

3) : IP

4) : SI

Correct Option : 3 From : Lecture 3

Question # 35

SP is a memory pointer and is used indirectly by a set of _____.

1) : instructions

2) : Pointers

3) : Indexes

4) : Variables

Correct Option : 1 From : Lecture 3

Question # 36

_____ is also a memory pointer containing the address in a special area of memory called the stack.

1) : SP

2) : BP

3) : PB

4) : AC

Correct Option : 2 From : Lecture 3

Question # 37

_____ is bit wise significant and accordingly each bit is named separately.

1) : AX

2) : FS

3) : IP

4) : Flags Register

Correct Option : 4 From : Lecture 3

Question # 38

When two 16bit numbers are added the answer can be 17 bits long, this extra bit that won't fit in the target register is placed in the _____ where it can be used and tested

1) : carry flag

2) : Parity Flag

3) : Auxiliary Carry

4) : Zero Flag

Correct Option : 1 From : Lecture 3

Question # 39

Program is an ordered set of instructions for the processor.

1) : TRUE

2) : FALSE

3) :

4) :

Correct Option : 1 From : Lecture 3

Question # 40

For Intel Architecture "operation destination, source" is way of writing things.

1) : TRUE

2) : FALSE

3) :

4) :

Correct Option : 1 From : Lecture 3

Question # 41

Operation code " add ax, bx " _____.

- 1) : Add the bx to ax and change the bx
 - 2) : Add the ax to bx and change the ax
 - 3) : Add the bx to ax and change the ax
 - 4) : Add the bx to ax and change nothing
- Correct Option : 3 From : Lecture 3

Question # 42

The maximum memory iAPX88 can access is_____.

- 1) : 1MB
- 2) : 2MB
- 3) : 3MB
- 4) : 128MB

Correct Option : 1 From : Lecture 4

Question # 43

The maximum memory iAPX88 can access is 1MB which can be accessed with_____.

- 1) : 18 bits
- 2) : 20 bits
- 3) : 16 bits
- 4) : 2 bits

Correct Option : 2 From : Lecture 4

Question # 44

_____address of 1DED0 where the opcode B80500 is placed.

- 1) : physical memory
- 2) : memory
- 3) : effective
- 4) : None of the Given

Correct Option : 1 From : Lecture 4

Question # 45

16 bit of Segment and Offset Addresses can be converted to 20bit Address i.e

Segment Address with lower four bits zero + Offset Address with _____ four bits zero = 20bit Physical Address

- 1) : Middle
- 2) : lower
- 3) : Top
- 4) : upper

Correct Option : 4 From : Lecture 4

Question # 46

When adding two 20bit Addresses a carry if generated is dropped without being stored anywhere and the phenomenon is called address_____.

- 1) : wraparound
- 2) : mode
- 3) : ping
- 4) : error

Correct Option : 1 From : Lecture 4

Question # 47

segments can only be defined a 16byte boundaries called _____ boundaries.

- 1) : segment
- 2) : paragraph
- 3) : Cell
- 4) : RAM

Correct Option : 1 From : Lecture 4

Question # 48

in a Program CS, DS, SS, and ES all had the same value in them. This is called _____.

- 1) : equal memory
- 2) : overlapping segments
- 3) : segments hidding
- 4) : overlapping SI

Correct Option : 2 From : Lecture 4

Question # 49

"db num1" size of the memory is _____

- 1) : 1byte
- 2) : 4bit
- 3) : 16bit
- 4) : 2byte

Correct Option : 1 From : Lecture 5

Question # 50

" 1-----[org 0x0100]
2-----mov ax, [num1] ; load first number in ax
3-----mov bx, [num2] ; load second number in bx
4-----add ax, bx _____
5-----int 0x21
6-----
7-----num1: dw 5
8-----num2: dw 10

Comments for the 4 are :

- 1) : No comments Will be
- 2) : ; accumulate sum in add
- 3) : ; accumulate sum in ax
- 4) : ; accumulate sum in Bx

Correct Option : 3 From : Lecture 5

Question # 51

In "mov ax, bx" is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : Direct
- 4) : Register

Correct Option : 4 From : Lecture 5

Question # 52

In "mov ax, [bx]" is _____ Addressing Modes

- 1) : Based Register Indirect
- 2) : Indirect
- 3) : Base Indirect
- 4) : Immediate

Correct Option : 1 From : Lecture 5

Question # 53

In "mov ax, 5" is _____ Addressing Modes

- 1) : Immediate
- 2) : Indirect

- 3) : Indirect
- 4) : Register

Correct Option : 1 From : Lecture 6

Question # 54

In " mov ax, [num1+bx] " is _____ ADDRESSING

- 1) : OFFSET+ Indirect
- 2) : Register + Direct
- 3) : Indirect + Reference
- 4) : BASEd REGISTER + OFFSET

Correct Option : 4 From : Lecture 7

Question # 55

"base + offset addressing " gives This number which came as the result of addition is called the _____.

- 1) : Address
- 2) : mode
- 3) : effective address
- 4) : Physical Address

Correct Option : 3 From : Lecture 7

Question # 56

"mov ax, [cs:bx]" associates _____ for this one instruction

- 1) : CS with BX
- 2) : BX with CS
- 3) : BX with AX
- 4) : None of the Given

Correct Option : 2 From : Lecture 7

Question # 57

For example

BX=0100

DS=FFF0

And Opcode are;

move [bx+0x0100], Ax

now what is the effective memory address;

- 1) : 0020
- 2) : 0200
- 3) : 0300
- 4) : 0x02

Correct Option : 2 From : Lecture 7

Question # 58

For example

BX=0100

DS=FFF0

And Opcode are;

move [bx+0x0100], Ax

now what is the physical memory address;

- 1) : 0020
- 2) : 0x0100
- 3) : 0x10100
- 4) : 0x100100

Correct Option : 2 From : Lecture 7

Question # 59

In "mov [1234], al" is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : Direct
- 4) : Register

Correct Option : 3 From : Lecture 8

Question # 60

In "mov [SI], AX" is _____ Addressing Modes.

- 1) : Base Register Indirect
- 2) : Indirect
- 3) : Indexed Register Indirect
- 4) : Immediate

Correct Option : 3 From : Lecture 8

Question # 61

In "mov ax, [bx - Si]" is _____ ADDRESSING

- 1) : Base Register Indirect
- 2) : Indirect
- 3) : Direct
- 4) : illegal

Correct Option : 4 From : Lecture 8

Question # 62

In "mov ax, [BL]" there is error i.e. _____

- 1) : Address must be 16bit
- 2) : Address must be 8bit
- 3) : Address must be 4bit
- 4) : 8 bit to 16 bit move illegal

Correct Option : 4 From : Lecture 8

Question # 63

In "mov ax, [SI+DI]" there is error i.e. _____

- 1) : Two indexes can't use as Memory Address
- 2) : index can't use as Memory Address
- 3) : I don't Know
- 4) : None of the Given

Correct Option : 1 From : Lecture 8

Question # 64

In JNE and JNZ there is difference for only _____;

- 1) : Programmer or Logic
- 2) : Assembler
- 3) : Debugger
- 4) : IAPX88

Correct Option : 1 From : Lecture 9

Question # 65

JMP is Instruction that on executing take jump regardless of the state of all flags is called _____

- 1) : Jump
- 2) : Conditional jump
- 3) : Unconditional jump
- 4) : Stay

Correct Option : 3 From : Lecture 9

Question # 66

When result of the source subtraction from the destination is zero, zero flag is set i.e. ZF=1
its mean that;

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : DEST < SRC
- 4) : DEST > SRC

Correct Option : 1 From : Lecture 9

Question # 67

When an unsigned source is subtracted from an unsigned destination and the destination is smaller, borrow is needed which sets the _____.

- 1) : carry flag i.e CF = 0
- 2) : carry flag i.e CF = 1
- 3) : Carry Flag + ZF=1
- 4) : None of the Given

Correct Option : 2 From : Lecture 9

Question # 68

In the case of unassigned source and destination when subtracting and in the result ZF =1 OR CR=1 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST ? USRC
- 4) : DEST > SRC

Correct Option : 3 From : Lecture 9

Question # 69

In the case of unassigned source and destination when subtracting and in the result ZF =0 AND CR=0 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST < USRC
- 4) : UDEST > USRC

Correct Option : 4 From : Lecture 9

Question # 70

In the case of unassigned source and destination when subtracting and in the result CR=0 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST < USRC
- 4) : UDEST ? USRC

Correct Option : 4 From : Lecture 9

Question # 71

_____ This jump is taken if the last arithmetic operation produced a zero in its destination. After a CMP it is taken if both operands were equal.

- 1) : Jump if zero(JZ)/Jump if equal(JE)
- 2) : Jump if equal(JE)
- 3) : Jump if zero(JZ)
- 4) : No Jump fot This

Correct Option : 1 From : Lecture 9

Question # 72

_____ This jump is taken after a CMP if the unsigned source is smaller than or equal to the unsigned destination.

- 1) : JBE(Jump if not below or equal)
- 2) : JNA(Jump if not above)/JBE(Jump if not below or equal)
- 3) : JNA(Jump if not above)
- 4) : No Jump fot This

Correct Option : 2 From : Lecture 9

Question # 1

Numbers of any size can be added using a proper combination of _____.

- 1) : ADD and ADC
- 2) : ABD and ADC
- 3) : ADC and ADC
- 4) : None of the Given

Correct Option : 1 From : Lecture 11

Question # 2

Like addition with carry there is an instruction to subtract with borrows called _____.

- 1) : SwB
- 2) : SBB
- 3) : SBC
- 4) : SBBC

Correct Option : 2 From : Lecture 11

Question # 3

if "and ax, bx" instruction is given, There are _____ operations as a result

- 1) : 16 AND
- 2) : 17 AND
- 3) : 32 AND
- 4) : 8 AND

Correct Option : 1 From : Lecture 12

Question # 4

_____ can be used to check whether particular bits of a number are set or not.

- 1) : AND
- 2) : OR
- 3) : XOR
- 4) : NOT

Correct Option : 1 From : Lecture 12

Question # 5

_____ can also be used as a masking operation to invert selective bits.

- 1) : AND
- 2) : OR
- 3) : XOR
- 4) : NOT

Correct Option : 3 From : Lecture 12

Question # 6

Masking Operations are Selective Bit _____

- 1) : Clearing, XOR, Inversion and Testing
- 2) : Clearing, Setting, Inversion and Testing
- 3) : Clearing, XOR, AND and Testing
- 4) : None of the Given

Correct Option : 2 From : Lecture 12

Question # 7

The _____ instruction allows temporary diversion and therefore reusability of code.

- 1) : CALL

- 2) : RET
- 3) : AND
- 4) : XOR

Correct Option : 1 From : Lecture 13

Question # 8

CALL takes a label as _____ and execution starts from that label,

- 1) : argument
- 2) : Lable
- 3) : TXt
- 4) : Register

Correct Option : 1 From : Lecture 13

Question # 9

When the _____ instruction is encountered and it takes execution back to the instruction following the CALL.

- 1) : CALL
- 2) : RET
- 3) : AND
- 4) : XOR

Correct Option : 2 From : Lecture 13

Question # 10

_____ Both the instructions are commonly used as a pair, however technically they are independent in their operation.

- 1) : RET and ADC
- 2) : Cal and SSb
- 3) : CALL and RET
- 4) : ADC and SSB

Correct Option : 3 From : Lecture 13

Question # 11

The CALL mechanism breaks the thread of execution and does not change registers, except _____.

- 1) : SI
- 2) : IP
- 3) : DI
- 4) : SP

Correct Option : 2 From : Lecture 13

Question # 12

Stack is a _____ that behaves in a first in last out manner.

- 1) : Program
- 2) : data structure
- 3) : Heap
- 4) : None of the Given

Correct Option : 2 From : Lecture 14

Question # 13

If _____ is not available, stack clearing by the callee is a complicated process.

- 1) : CALL
- 2) : SBB
- 3) : RET n
- 4) : None of the Given

Correct Option : 3 From : Lecture 14

Question # 14

When the stack will eventually become full, SP will reach 0, and thereafter wraparound producing unexpected results. This is called stack _____

- 1) : Overflow
- 2) : Leakage
- 3) : Error
- 4) : Pointer

Correct Option : 1 From : Lecture 14

Question # 15

The pop operation makes a copy from the top of the stack into its _____.

- 1) : Register
- 2) : operand
- 3) : RET n
- 4) : Pointer

Correct Option : 2 From : Lecture 14

Question # 16

_____ decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack

- 1) : PUSH
- 2) : POP
- 3) : CALL
- 4) : RET

Correct Option : 1 From : Lecture 14

Question # 17

POP transfers the word at the current top of stack (pointed to by SP) to the destination operand and then _____ SP by two to point to the new top of stack.

- 1) : increments
- 2) : decrements
- 3) : ++
- 4) : --

Correct Option : 1 From : Lecture 14

Question # 18

The trick is to use the _____ and _____ operations and save the callers' value on the stack and recover it from there on return.

- 1) : POP, ADC
- 2) : CALL, RET
- 3) : CALL, RET n
- 4) : PUSH, POP

Correct Option : 4 From : Lecture 14

Question # 19

To access the arguments from the stack, the immediate idea that strikes is to _____ them off the stack.

- 1) : PUSH
- 2) : POP
- 3) : CALL
- 4) : Register

Correct Option : 2 From : Lecture 15

Question # 20

push bp
we are _____

- 1) : sending bp copy to stack
 - 2) : making bp copy from stack
 - 3) : pushing bp on the stack
 - 4) : doing nothing
- Correct Option : 3 From : Lecture 15

Question # 21

Local Variables means variables that are used within the _____

- 1) : Subroutine
- 2) : Program
- 3) : CALL
- 4) : Label

Correct Option : 1 From : Lecture 15

Question # 22

Standard ASCII has 128 characters with assigned numbers from _____.

- 1) : 1to 129
- 2) : 0 to 127
- 3) : 0 to 128
- 4) : None of the Given

Correct Option : 2 From : Lecture 16

Question # 23

When _____ is sent to the VGA card, it will turn pixels on and off in such a way that a visual representation of 'A' appears on the screen.

- 1) : 0x60
- 2) : 0x90
- 3) : 0x30
- 4) : 0x40

Correct Option : 4 From : Lecture 16

Question # 24

Which bit is refer to the Blinking of foreground character

- 1) : 6
- 2) : 7
- 3) : 5
- 4) : 3

Correct Option : 2 From : Lecture 16

Question # 25

Which bit is refer to the Intensity component of foreground color

- 1) : 4
- 2) : 5
- 3) : 3
- 4) : 7

Correct Option : 3 From : Lecture 16

Question # 26

Which bit is refer to the Green component of background color

- 1) : 1
- 2) : 5
- 3) : 3
- 4) : 7

Correct Option : 2 From : Lecture 16

Question # 27

Which bit is refer to the Green component of foreground color

- 1) : 1
- 2) : 5
- 3) : 3
- 4) : 7

Correct Option : 1 From : Lecture 16

Question # 28

String can be indicate bye given

- 1) : db 0x61, 0x61, 0x63
- 2) : db 'a', 'b', 'c'
- 3) : db 'abc'
- 4) : All of the above

Correct Option : 4 From : Lecture 16

Question # 29

The first form divides a 32bit number in DX:AX by its 16bit operand and stores the _____ quotient in AX

- 1) : 16bit
- 2) : 17bit
- 3) : 32bit
- 4) : 64bit

Correct Option : 1 From : Lecture 17

Question # 30

The _____ (division) used in the process is integer division and not floating point division.

- 1) : DIV instruction
- 2) : ADC instruction
- 3) : SSB instruction
- 4) : DIVI instruction

Correct Option : 1 From : Lecture 17

Question # 31

_____ (multiply) performs an unsigned multiplication of the source operand and the accumulator.

- 1) : Multi
- 2) : DIV
- 3) : MUL
- 4) : Move

Correct Option : 3 From : Lecture 18

Question # 32

The desired location on the screen can be calculated with the following formulae.

- 1) : location = (hypos * 80 + SP) * 3
- 2) : location = (hypos * 80 + slocation) * 2
- 3) : location = (hypos * 80 + epos) * 2
- 4) : None of the Given

Correct Option : 3 From : Lecture 18

Question # 33

To play with string there are 5 instructions that are _____

- 1) : STOS, LODS, CMPS, SCAS, and MOVS
- 2) : MUL, DIV, ADD, ADC and MOVE
- 3) : SSB, ADD, CMPS, ADC, and MOVS
- 4) : None of the Given

Correct Option : 1 From : Lecture 18

Question # 34

_____ transfers a byte or word from register AL or AX to the string element addressed by ES:DI and updates DI to point to the next location.

- 1) : LODS
- 2) : STOS
- 3) : SCAS
- 4) : MOVE

Correct Option : 2 From : Lecture 18

Question # 35

_____ transfers a byte or word from the source location DS:SI to AL or AX and updates SI to point to the next location.

- 1) : LODS
- 2) : STOS
- 3) : SCAS
- 4) : MOVE

Correct Option : 1 From : Lecture 18

Question # 36

_____ compares a source byte or word in register AL or AX with the destination string element addressed by ES: DI and updates the flags.

- 1) : LODS
- 2) : STOS
- 3) : SCAS
- 4) : MOVE

Correct Option : 3 From : Lecture 18

Question # 37

_____ repeat the following string instruction while the zero flag is set and REPNE or REPNZ repeat the following instruction while the zero flag is not set.

- 1) : REP or REPZ
- 2) : REPE or REPZ
- 3) : REPE or RPZ
- 4) : RPE or REPZ

Correct Option : 2 From : Lecture 18

Question # 38

LES loads _____

- 1) : ES
- 2) : DS
- 3) : PS
- 4) : LS

Correct Option : 1 From : Lecture 20

Question # 39

LDS loads _____.

- 1) : ES
- 2) : DS
- 3) : PS
- 4) : LS

Correct Option : 2 From : Lecture 20

Question # 40

REP allows the instruction to be repeated _____ times allowing blocks of memory to be copied.

- 1) : DX
- 2) : CX
- 3) : BX
- 4) : AX

Correct Option : 2 From : Lecture 20

Question # 41

_____ pops IP, then CS, and then FLAGS.

- 1) : Ret n
- 2) : REZA
- 3) : REPE
- 4) : IRET

Correct Option : 4 From : Lecture 21

Question # 42

_____, Trap, Single step Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 2 From : Lecture 21

Question # 43

_____, NMI-Non Maskable Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 3 From : Lecture 21

Question # 44

To hook an interrupt we change the _____ corresponding to that interrupt.

- 1) : SX
- 2) : vector
- 3) : AX
- 4) : BX

Correct Option : 2 From : Lecture 22

Question # 1

_____ pops IP, then CS, and then FLAGS.

- 1) : Ret n
- 2) : REZA
- 3) : REPE
- 4) : IRET

Correct Option : 4 From : Lecture 21

Question # 2

_____, Trap, Single step Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 2 From : Lecture 21

Question # 3

_____, NMI-Non Maskable Interrupt

- 1) : INT 0

- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 3 From : Lecture 21

Question # 4

To hook an interrupt we change the _____ corresponding to that interrupt.

- 1) : SX
- 2) : vector
- 3) : AX
- 4) : BX

Question # 1

There are three busses to communicate the processor and memory named as _____

- 1) : address bus.,data bus and data bus.
- 2) : addressing bus.,data bus and data bus.
- 3) : address bus.,datamove bus and data bus.
- 4) : address bus.,data bus and control bus..

Correct Option : 4 From : Lecture 1

Question # 2

The address bus is unidirectional and address always travels from processor to memory.

- 1) : TRUE
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 1

Question # 3

Data bus is bidirectional because _____

- 1) : To way
- 2) : Data moves from both, processor to memory and memory to processor,

3) : Data moves from both, processor to memory and memory to data Bus,
4) : None of the Given
Correct Option : 3 From : Lecture 1

Question # 4

Control bus _____

- 1) : is Not Important.
- 2) : is Important .
- 3) : bidirectional.
- 4) : unidirectional .

Correct Option : 3 From : Lecture 1

Question # 5

A memory cell is an n-bit location to store data, normally _____ also called a byte

- 1) : 4-bit
- 2) : 8-bit
- 3) : 6-bit
- 4) : 80-bit

Correct Option : 2 From : Lecture 1

Question # 6

The number of bits in a cell is called the cell width. _____
define the memory completely.

- 1) : Cell width and number of cells,
- 2) : cell number and width of the cells,
- 3) : width
- 4) : Height

Correct Option : 1 From : Lecture 1

Question # 7

for memory we define two dimensions. The first dimension defines how many _____ bits are there in a single memory cell.

- 1) : parallel
- 2) : Vertical
- 3) : long
- 4) : short

Correct Option : 1 From : Lecture 1

Question # 8

_____ operation requires the same size of data bus and memory cell width.

- 1) : Normal
- 2) : Best and simplest
- 3) : first
- 4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 9

Control bus is only the mechanism. The responsibility of sending the appropriate signals on the control bus to the memory is of the _____.

- 1) : Data Bus
- 2) : processor
- 3) : Address Bus
- 4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 10

In "total: dw 0" Opcode total is a _____

- 1) : Literal
- 2) : Variable
- 3) : Label
- 4) : Starting point

Correct Option : 3 From : Lecture 10

Question # 11

| 0 | --> | 1 | 1 | 0 | 1 | 0 | 0 | 0 | --> | C | is a example of _____

- 1) : Shl
- 2) : sar
- 3) : Shr
- 4) : Sal

Correct Option : 3 From : Lecture 10

Question # 12

| C | <-- | 1 | 1 | 0 | 1 | 0 | 0 | 0 | <-- | 0 | is a example of _____

- 1) : Shl
- 2) : sar
- 3) : Shr
- 4) : Sal

Correct Option : 1 From : Lecture 10

Question # 13

ADC has _____ operands.

- 1) : two
- 2) : three
- 3) : Five
- 4) : Zero

Correct Option : 2 From : Lecture 10

Question # 14

The basic purpose of a computer is to perform operations, and operations need _____.

- 1) : order
- 2) : nothing
- 3) : operands
- 4) : bit

Correct Option : 3 From : Lecture 2

Question # 15

Registers are like a scratch pad ram inside the processor and their operation is very much like normal _____.

- 1) : Number
- 2) : operations
- 3) : memory cells
- 4) : None of the Given

Correct Option : 3 From : Lecture 2

Question # 16

There is a central register in every processor called the _____ and The word size of a processor is defined by the width of its _____.

- 1) : accumulator, accumulator
- 2) : data bus, accumulator
- 3) : accumulator, Address Bus
- 4) : accumulator, memory

Correct Option : 1 From : Lecture 2

Question # 17

_____ does not hold data but holds the address of data

- 1) : Pointer, Segment, or Base Register
- 2) : Pointer, Index, or Base Register
- 3) : General Registers
- 4) : Instruction Pointer

Correct Option : 2 From : Lecture 2

Question # 18

"The program counter holds the address of the next instruction to be _____"

- 1) : executed.
- 2) : called
- 3) : deleted
- 4) : copy

Correct Option : 1 From : Lecture 2

Question # 19

There are _____ types of "instruction groups"

- 1) : 4
- 2) : 5
- 3) : 3
- 4) : 2

Correct Option : 1 From : Lecture 2

Question # 20

These instructions are used to move data from one place to another.

- 1) : TRUE
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 2

Question # 21

"mov" instruction is related to the _____ Group.

- 1) : Arithmetic and Logic Instructions
- 2) : Data Movement Instructions
- 3) : Program Control Instructions
- 4) : Special Instructions

Correct Option : 2 From : Lecture 2

Question # 22

_____ allow changing specific processor behaviors and are used to play with it.

- 1) : Special Instructions
- 2) : Data Movement Instructions
- 3) : Program Control Instructions
- 4) : Arithmetic and Logic Instructions

Correct Option : 1 From : Lecture 2

Question # 23

8088 is a 16bit processor with its accumulator and all registers of _____.

- 1) : 32 bits
- 2) : 6 bits
- 3) : 16 bits
- 4) : 64 bits

Correct Option : 3 From : Lecture 2

Question # 24

The _____ of a processor means the organization and functionalities of the registers it contains and the instructions that are valid on the processor.

- 1) : Manufactures
- 2) : architecture
- 3) : Deal
- 4) : None of the Given

Correct Option : 2 From : Lecture 2

Question # 25

Intel IAPX88 Architecture is _____

- 1) : More then 25 old
- 2) : New
- 3) : Not Good
- 4) : None of the Given

Correct Option : 1 From : Lecture 2

Question # 26

The iAPX88 architecture consists of _____ registers.

- 1) : 13
- 2) : 12
- 3) : 9
- 4) : 14

Correct Option : 4 From : Lecture 3

Question # 27

General Registers are _____

- 1) : AX, BX, CX, and DX
- 2) : XA, BX, CX, and DX
- 3) : SS, SI and DI
- 4) : 3

Correct Option : 1 From : Lecture 3

Question # 28

AX means we are referring to the extended 16bit "A" register. Its upper and lower byte are separately accessible as _____.

- 1) : AH and AL
- 2) : A Lower and A Upper
- 3) : AL, AU
- 4) : AX

Correct Option : 1 From : Lecture 3

Question # 29

AX is General purpose Register where A stands for _____.

- 1) : Acadmic
- 2) : Ado
- 3) : Architecture
- 4) : Accumulator

Correct Option : 4 From : Lecture 3

Question # 30

The B of BX stands for _____ because of its role in memory addressing.

- 1) : Busy
- 2) : Base
- 3) : Better
- 4) : None of the Given

Correct Option : 2 From : Lecture 3

Question # 31

The D of DX stands for Destination as it acts as the destination in _____.

- 1) : I/O operations
- 2) : operations
- 3) : memory cells
- 4) : Memory I/O operations

Correct Option : 1 From : Lecture 3

Question # 32

The C of CX stands for Counter as there are certain instructions that work with an automatic count in the _____.

- 1) : DI register
- 2) : BX register
- 3) : CX register
- 4) : DX register

Correct Option : 3 From : Lecture 3

Question # 33

_____ are the index registers of the Intel architecture which hold address of data and used in memory access.

- 1) : SI and SS
- 2) : PI and DI
- 3) : SI and IP
- 4) : SI and DI

Correct Option : 4 From : Lecture 3

Question # 34

In Intel IAPX88 architecture _____ is the special register containing the address of the next instruction to be executed.

- 1) : AX
- 2) : PI
- 3) : IP
- 4) : SI

Correct Option : 3 From : Lecture 3

Question # 35

SP is a memory pointer and is used indirectly by a set of _____.

- 1) : instructions
- 2) : Pointers
- 3) : Indexes
- 4) : Variables

Correct Option : 1 From : Lecture 3

Question # 36

_____ is also a memory pointer containing the address in a special area of memory called the stack.

- 1) : SP
- 2) : BP
- 3) : PB
- 4) : AC

Correct Option : 2 From : Lecture 3

Question # 37

_____ is bit wise significant and accordingly each bit is named

separately.

- 1) : AX
- 2) : FS
- 3) : IP
- 4) : Flags Register

Correct Option : 4 From : Lecture 3

Question # 38

When two 16bit numbers are added the answer can be 17 bits long, this extra bit that won't fit in the target register is placed in the _____ where it can be used and tested

- 1) : carry flag
- 2) : Parity Flag
- 3) : Auxiliary Carry
- 4) : Zero Flag

Correct Option : 1 From : Lecture 3

Question # 39

Program is an ordered set of instructions for the processor.

- 1) : TRUE
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 3

Question # 40

For Intel Architecture "operation destination, source" is way of writing things.

- 1) : TRUE
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 3

Question # 41

Operation code " add ax, bx " _____.

- 1) : Add the bx to ax and change the bx
- 2) : Add the ax to bx and change the ax
- 3) : Add the bx to ax and change the ax
- 4) : Add the bx to ax and change nothing

Correct Option : 3 From : Lecture 3

Question # 42

The maximum memory iAPX88 can access is _____.

- 1) : 1MB
- 2) : 2MB
- 3) : 3MB
- 4) : 128MB

Correct Option : 1 From : Lecture 4

Question # 43

The maximum memory iAPX88 can access is 1MB which can be accessed with _____.

- 1) : 18 bits
- 2) : 20 bits
- 3) : 16 bits
- 4) : 2 bits

Correct Option : 2 From : Lecture 4

Question # 44

_____ address of 1DED0 where the opcode B80500 is placed.

- 1) : physical memory
- 2) : memory
- 3) : efective
- 4) : None of the Given

Correct Option : 1 From : Lecture 4

Question # 45

16 bit of Segment and Offset Addresses can be converted to 20bit Address i.e Segment Address with lower four bits zero + Offset Address with _____ four bits zero = 20bit Physical Address

- 1) : Middle
- 2) : lower
- 3) : Top
- 4) : upper

Correct Option : 4 From : Lecture 4

Question # 46

When adding two 20bit Addresses a carry if generated is dropped without being stored anywhere and the phenomenon is called address_____.

- 1) : wraparound
- 2) : mode
- 3) : ping
- 4) : error

Correct Option : 1 From : Lecture 4

Question # 47

segments can only be defined a 16byte boundaries called _____ boundaries.

- 1) : segment
- 2) : paragraph
- 3) : Cell
- 4) : RAM

Correct Option : 1 From : Lecture 4

Question # 48

in a Program CS, DS, SS, and ES all had the same value in them. This is called _____.

- 1) : equel memory
- 2) : overlapping segments
- 3) : segments hiding
- 4) : overlapping SI

Correct Option : 2 From : Lecture 4

Question # 49

"db num1" size of the memory is _____

- 1) : 1byte
- 2) : 4bit
- 3) : 16bit
- 4) : 2byte

Correct Option : 1 From : Lecture 5

Question # 50

" 1-----[org 0x0100]

2-----mov ax, [num1] ; load first number in ax

3-----mov bx, [num2] ; load second number in bx

4-----add ax, bx _____
5-----int 0x21
6-----
7-----num1: dw 5
8-----num2: dw 10

Comments for the 4 are :

- 1) : No comments Will be
- 2) : ; accumulate sum in add
- 3) : ; accumulate sum in ax
- 4) : ; accumulate sum in Bx

Correct Option : 3 From : Lecture 5

Question # 51

In " mov ax, bx " is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : Direct
- 4) : Register

Correct Option : 4 From : Lecture 5

Question # 52

In "mov ax, [bx] " is _____ Addressing Modes

- 1) : Based Register Indirect
- 2) : Indirect
- 3) : Base Indirect
- 4) : Immediate

Correct Option : 1 From : Lecture 5

Question # 53

In "mov ax, 5 " is _____ Addressing Modes

- 1) : Immediate
- 2) : Indirect
- 3) : Indirect
- 4) : Register

Correct Option : 1 From : Lecture 6

Question # 54

In " mov ax, [num1+bx] " is _____ ADDRESSING

- 1) : OFFSET+ Indirect
- 2) : Register + Direct
- 3) : Indirect + Reference
- 4) : BASEd REGISTER + OFFSET

Correct Option : 4 From : Lecture 7

Question # 55

"base + offset addressing " gives This number which came as the result of addition is called the _____.

- 1) : Address
- 2) : mode
- 3) : effective address
- 4) : Physical Address

Correct Option : 3 From : Lecture 7

Question # 56

"mov ax, [cs:bx]" associates _____ for this one instruction

- 1) : CS with BX
- 2) : BX with CS
- 3) : BX with AX
- 4) : None of the Given

Correct Option : 2 From : Lecture 7

Question # 57

For example

BX=0100

DS=FFFF0

And Opcode are;

move [bx+0x0100], Ax

now what is the effective memory address;

- 1) : 0020
- 2) : 0200
- 3) : 0300
- 4) : 0x02

Correct Option : 2 From : Lecture 7

Question # 58

For example

BX=0100

DS=FFFF0

And Opcode are;

move [bx+0x0100], Ax

now what is the physical memory address;

- 1) : 0020
- 2) : 0x0100
- 3) : 0x10100
- 4) : 0x100100

Correct Option : 2 From : Lecture 7

Question # 59

In "mov [1234], al" is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : Direct
- 4) : Register

Correct Option : 3 From : Lecture 8

Question # 60

In "mov [SI], AX" is _____ Addressing Modes.

- 1) : Base Register Indirect
- 2) : Indirect
- 3) : Indexed Register Indirect
- 4) : Immediate

Correct Option : 3 From : Lecture 8

Question # 61

In "mov ax, [bx - Si]" is _____ ADDRESSING

- 1) : Base Register Indirect
- 2) : Indirect
- 3) : Direct
- 4) : illegal

Correct Option : 4 From : Lecture 8

Question # 62

In " mov ax, [BL] " there is error i.e. _____

- 1) : Address must be 16bit
- 2) : Address must be 8bit
- 3) : Address must be 4bit
- 4) : 8 bit to 16 bit move illegal

Correct Option : 4 From : Lecture 8

Question # 63

In " mov ax, [SI+DI] " there is error i.e. _____

- 1) : Two indexes can't use as Memory Address
- 2) : index can't use as Memory Address
- 3) : I don't Know
- 4) : None of the Given

Correct Option : 1 From : Lecture 8

Question # 64

In JNE and JNZ there is difference for only _____;

- 1) : Programmer or Logic
- 2) : Assembler
- 3) : Debugger
- 4) : IAPX88

Correct Option : 1 From : Lecture 9

Question # 65

JMP is Instruction that on executing take jump regardless of the state of all flags is called _____

- 1) : Jump
- 2) : Conditional jump
- 3) : Unconditional jump
- 4) : Stay

Correct Option : 3 From : Lecture 9

Question # 66

When result of the source subtraction from the destination is zero, zero flag is set i.e. ZF=1 its mean that;

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : DEST < SRC
- 4) : DEST > SRC

Correct Option : 1 From : Lecture 9

Question # 67

When an unsigned source is subtracted from an unsigned destination and the destination is smaller, borrow is needed which sets the _____.

- 1) : carry flag i.e CF = 0
- 2) : carry flag i.e CF = 1
- 3) : Carry Flag + ZF=1
- 4) : None of the Given

Correct Option : 2 From : Lecture 9

Question # 68

In the case of unassigned source and destination when subtracting and in the result ZF =1 OR CR=1 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST ? USRC

4) : DEST > SRC

Correct Option : 3 From : Lecture 9

Question # 69

In the case of unassigned source and destination when subtracting and in the result ZF =0 AND CR=0 then _____

1) : DEST = SRC

2) : DEST != SRC

3) : UDEST < USRC

4) : UDEST > USRC

Correct Option : 4 From : Lecture 9

Question # 70

In the case of unassigned source and destination when subtracting and in the result CR=0 then _____

1) : DEST = SRC

2) : DEST != SRC

3) : UDEST < USRC

4) : UDEST ? USRC

Correct Option : 4 From : Lecture 9

Question # 71

_____ This jump is taken if the last arithmetic operation produced a zero in its destination. After a CMP it is taken if both operands were equal.

1) : Jump if zero(JZ)/Jump if equal(JE)

2) : Jump if equal(JE)

3) : Jump if zero(JZ)

4) : No Jump fot This

Correct Option : 1 From : Lecture 9

Question # 72

_____ This jump is taken after a CMP if the unsigned source is smaller than or equal to the unsigned destination.

1) : JBE(Jump if not below or equal)

2) : JNA(Jump if not above)/JBE(Jump if not below or equal)

3) : JNA(Jump if not above)

4) : No Jump fot This

Correct Option : 2 From : Lecture 9

Question # 73

Numbers of any size can be added using a proper combination of _____.

1) : ADD and ADC

2) : ABD and ADC

3) : ADC and ADC

4) : None of the Given

Correct Option : 1 From : Lecture 11

Question # 74

Like addition with carry there is an instruction to subtract with borrows called _____.

1) : SwB

2) : SBB

3) : SBC

4) : SBBC

Correct Option : 2 From : Lecture 11

Question # 75

if "and ax, bx" instruction is given, There are _____ operations as a result

- 1) : 16 AND
- 2) : 17 AND
- 3) : 32 AND
- 4) : 8 AND

Correct Option : 1 From : Lecture 12

Question # 76

_____ can be used to check whether particular bits of a number are set or not.

- 1) : AND
- 2) : OR
- 3) : XOR
- 4) : NOT

Correct Option : 1 From : Lecture 12

Question # 77

_____ can also be used as a masking operation to invert selective bits.

- 1) : AND
- 2) : OR
- 3) : XOR
- 4) : NOT

Correct Option : 3 From : Lecture 12

Question # 78

Masking Operations are Selective Bit _____

- 1) : Clearing, XOR, Inversion and Testing
- 2) : Clearing, Setting, Inversion and Testing
- 3) : Clearing, XOR, AND and Testing
- 4) : None of the Given

Correct Option : 2 From : Lecture 12

Question # 79

The _____ instruction allows temporary diversion and therefore reusability of code.

- 1) : CALL
- 2) : RET
- 3) : AND
- 4) : XOR

Correct Option : 1 From : Lecture 13

Question # 80

CALL takes a label as _____ and execution starts from that label,

- 1) : argument
- 2) : Lable
- 3) : TXt
- 4) : Register

Correct Option : 1 From : Lecture 13

Question # 81

When the _____ instruction is encountered and it takes execution back to the instruction following the CALL.

- 1) : CALL
- 2) : RET
- 3) : AND
- 4) : XOR

Correct Option : 2 From : Lecture 13

Question # 82

_____ Both the instructions are commonly used as a pair, however technically they are independent in their operation.

- 1) : RET and ADC
- 2) : Cal and SSb
- 3) : CALL and RET
- 4) : ADC and SSB

Correct Option : 3 From : Lecture 13

Question # 83

The CALL mechanism breaks the thread of execution and does not change registers, except _____.

- 1) : SI
- 2) : IP
- 3) : DI
- 4) : SP

Correct Option : 2 From : Lecture 13

Question # 84

Stack is a _____ that behaves in a first in last out manner.

- 1) : Program
- 2) : data structure
- 3) : Heap
- 4) : None of the Given

Correct Option : 2 From : Lecture 14

Question # 85

If _____ is not available, stack clearing by the callee is a complicated process.

- 1) : CALL
- 2) : SBB
- 3) : RET n
- 4) : None of the Given

Correct Option : 3 From : Lecture 14

Question # 86

When the stack will eventually become full, SP will reach 0, and thereafter wraparound producing unexpected results. This is called stack _____.

- 1) : Overflow
- 2) : Leakage
- 3) : Error
- 4) : Pointer

Correct Option : 1 From : Lecture 14

Question # 87

The pop operation makes a copy from the top of the stack into its _____.

- 1) : Register
- 2) : operand
- 3) : RET n
- 4) : Pointer

Correct Option : 2 From : Lecture 14

Question # 88

_____ decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack

- 1) : PUSH
- 2) : POP
- 3) : CALL
- 4) : RET

Correct Option : 1 From : Lecture 14

Question # 89

POP transfers the word at the current top of stack (pointed to by SP) to the destination operand and then _____ SP by two to point to the new top of stack.

- 1) : increments
- 2) : decrements
- 3) : ++
- 4) : --

Correct Option : 1 From : Lecture 14

Question # 90

The trick is to use the _____ and _____ operations and save the callers' value on the stack and recover it from there on return.

- 1) : POP, ADC
- 2) : CALL, RET
- 3) : CALL, RET n
- 4) : PUSH, POP

Correct Option : 4 From : Lecture 14

Question # 91

To access the arguments from the stack, the immediate idea that strikes is to _____ them off the stack.

- 1) : PUSH
- 2) : POP
- 3) : CALL
- 4) : Rregister

Correct Option : 2 From : Lecture 15

Question # 92

push bp

we are _____

- 1) : sending bp copy to stack
- 2) : making bp copy from stack
- 3) : pushing bp on the stack
- 4) : doing nothing

Correct Option : 3 From : Lecture 15

Question # 93

Local Variables means variables that are used within the _____

- 1) : Subroutine
- 2) : Program
- 3) : CALL
- 4) : Label

Correct Option : 1 From : Lecture 15

Question # 94

Standard ASCII has 128 characters with assigned numbers from _____.

- 1) : 1 to 129
- 2) : 0 to 127
- 3) : 0 to 128
- 4) : None of the Given

Correct Option : 2 From : Lecture 16

Question # 95

When _____ is sent to the VGA card, it will turn pixels on and off in such a way that a visual representation of 'A' appears on the screen.

- 1) : 0x60
- 2) : 0x90
- 3) : 0x30
- 4) : 0x40

Correct Option : 4 From : Lecture 16

Question # 96

Which bit is refer to the Blinking of foreground character

- 1) : 6
- 2) : 7
- 3) : 5
- 4) : 3

Correct Option : 2 From : Lecture 16

Question # 97

Which bit is refer to the Intensity component of foreground color

- 1) : 4
- 2) : 5
- 3) : 3
- 4) : 7

Correct Option : 3 From : Lecture 16

Question # 98

Which bit is refer to the Green component of background color

- 1) : 1
- 2) : 5
- 3) : 3
- 4) : 7

Correct Option : 2 From : Lecture 16

Question # 99

Which bit is refer to the Green component of foreground color

- 1) : 1
- 2) : 5
- 3) : 3
- 4) : 7

Correct Option : 1 From : Lecture 16

Question # 100

String can be indicate bye given

- 1) : db 0x61, 0x61, 0x63
- 2) : db 'a', 'b', 'c'
- 3) : db 'abc'
- 4) : All of the above

Correct Option : 4 From : Lecture 16

Question # 101

The first form divides a 32bit number in DX:AX by its 16bit operand and stores the _____ quotient in AX

- 1) : 16bit
- 2) : 17bit
- 3) : 32bit

4) : 64bit

Correct Option : 1 From : Lecture 17

Question # 102

The _____ (division) used in the process is integer division and not floating point division.

- 1) : DIV instruction
- 2) : ADC instruction
- 3) : SSB instruction
- 4) : DIVI instruction

Correct Option : 1 From : Lecture 17

Question # 103

_____ (multiply) performs an unsigned multiplication of the source operand and the accumulator.

- 1) : Multi
- 2) : DIV
- 3) : MUL
- 4) : Move

Correct Option : 3 From : Lecture 18

Question # 104

The desired location on the screen can be calculated with the following formulae.

- 1) : $\text{location} = (\text{hypos} * 80 + \text{SP}) * 3$
- 2) : $\text{location} = (\text{hypos} * 80 + \text{slocation}) * 2$
- 3) : $\text{location} = (\text{hypos} * 80 + \text{epos}) * 2$
- 4) : None of the Given

Correct Option : 3 From : Lecture 18

Question # 105

To play with string there are 5 instructions that are _____

- 1) : STOS, LODS, CMPS, SCAS, and MOVS
- 2) : MUL, DIV, ADD, ADC and MOVE
- 3) : SSB, ADD, CMPS, ADC, and MOVS
- 4) : None of the Given

Correct Option : 1 From : Lecture 18

Question # 106

_____ transfers a byte or word from register AL or AX to the string element addressed by ES:DI and updates DI to point to the next location.

- 1) : LODS
- 2) : STOS
- 3) : SCAS
- 4) : MOVE

Correct Option : 2 From : Lecture 18

Question # 107

_____ transfers a byte or word from the source location DS:SI to AL or AX and updates SI to point to the next location.

- 1) : LODS
- 2) : STOS
- 3) : SCAS
- 4) : MOVE

Correct Option : 1 From : Lecture 18

Question # 108

_____ compares a source byte or word in register AL or AX with the destination string element addressed by ES: DI and updates the flags.

- 1) : LODS
- 2) : STOS
- 3) : SCAS
- 4) : MOVE

Correct Option : 3 From : Lecture 18

Question # 109

_____ repeat the following string instruction while the zero flag is set and REPNE or REPNZ repeat the following instruction while the zero flag is not set.

- 1) : REP or REPZ
- 2) : REPE or REPZ
- 3) : REPE or RPZ
- 4) : RPE or REPZ

Correct Option : 2 From : Lecture 18

Question # 110

LES loads _____

- 1) : ES
- 2) : DS
- 3) : PS
- 4) : LS

Correct Option : 1 From : Lecture 20

Question # 111

LDS loads _____.

- 1) : ES
- 2) : DS
- 3) : PS
- 4) : LS

Correct Option : 2 From : Lecture 20

Question # 112

REP allows the instruction to be repeated _____ times allowing blocks of memory to be copied.

- 1) : DX
- 2) : CX
- 3) : BX
- 4) : AX

Correct Option : 2 From : Lecture 20

Question # 113

_____ pops IP, then CS, and then FLAGS.

- 1) : Ret n
- 2) : REZA
- 3) : REPE
- 4) : IRET

Correct Option : 4 From : Lecture 21

Question # 114

_____, Trap, Single step Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 2 From : Lecture 21

Question # 115

_____, NMI-Non Maskable Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 3 From : Lecture 21

Question # 116

To hook an interrupt we change the _____ corresponding to that interrupt.

- 1) : SX
- 2) : vector
- 3) : AX
- 4) : BX

Correct Option : 2 From : Lecture 22

Question # 117

_____ pops IP, then CS, and then FLAGS.

- 1) : Ret n
- 2) : REZA
- 3) : REPE
- 4) : IRET

Correct Option : 4 From : Lecture 21

Question #118

_____, Trap, Single step Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 2 From : Lecture 21

Question #119

_____, NMI-Non Maskable Interrupt

- 1) : INT 0
- 2) : INT 1
- 3) : INT 3
- 4) : INT 0

Correct Option : 3 From : Lecture 21

Question # 120

To hook an interrupt we change the _____ corresponding to that interrupt.

- 1) : SX
- 2) : vector
- 3) : AX
- 4) : BX

1. Assembly language is not a low level language.

- a. True
- b. False**

2. In case of COM File first command parameter is stored at _____ offset of program segment prefix.

- a. 0x80 (Not Confirm)**
- b. 0x82
- c. 0x84
- d. 0x86

3. Address always goes from

- a. Processor to memory
- b. Memory to processor**
- c. Memory to memory
- d. None of the above

4. The source register in OUT is

- a. AL or AX**
- b. BL or BX
- c. CL or CX
- d. DL or DX

5. By default CS is associated with

- a. SS
- b. BP
- c. CX
- d. IP**

6. Which of the following pins of parallel port are grounded

- a. 10-18
- b. 18-25**
- c. 25-32
- d. 32-39

7. In the instruction mov word [es:160], 0x1230, 30 represents the character

- a. A**
- b. B
- c. 0
- d. 1

8. On executing 0x21 0x3D, if file can't be opened then

- a. CF will contain 1**
- b. CF will contain 0
- c. ZF will contain 1
- d. ZF will contain 0

9. Which of the following IRQ is cascading interrupt

- a. IRQ 0
- b. IRQ 1
- c. IRQ 2**
- d. IRQ 3

10. The execution of instruction mov word [es:160], 0x1230, will print a character on the screen at

- a. First column of second row
- b. Second column of first row

- c. Second column of second row
d. First column of third row

=====

1)))SHR and SAL are same?

.True (correct)

.False

2)))mov ax,0 will set ZF flag

.True

.False

3)))In 9 pin DB connector ,which pic is assigned to TD.

. 1

. 2

. 3(correct)

. 4

4)))Lower 16 bits of EAX are labeled as

. AX(correct)

. BX

.EAX

.none of above

5))) which is the special prefix used for repeating a block

.rep(correct)

.repeat

.repb

.repe

6)) JA can not after cmp if unsigned destination is greater than source

.true

.false

Q=1

Conditional jump can only:

1. Far
2. short
3. near
4. all of the given

q=2:

Address is always go from:

1. Processor to memory

2. Memory to processor
3. Memory to memory
4. None of given

Q=3;

Programmable interrupt controllers have two ports 20 and 21.....port 20 is a control port while port 21 is

1. The interrupt make register
2. Interrupt port
3. Output port
4. Input port

Q=4:

In the instruction "move word[es:160],0x1230 represent the charechter.....

1. A
2. B
3. 0
4. 1

Q=5:

The 8088 processor divides interrupts into how many classes?

1. 2
2. 3
3. 4
4. 5

Q=6:

Which of the following is the pair of register used to access memory in string instruction?

1. DI and BP
2. SI and BP
3. DI and SI
4. DS and SI

Q=7:

In case of COM file,first command line parameter is stored at offset of program segment prefix'

1. 0x80
2. 0x82
3. 0x84
4. 0x86

Q=8:

The INT 0x13 service 0x03 is use to ...

1. Read disk sector
2. Write disk sector
3. Reset disk sector
4. Get drive parameters

Q=9:

After the execution of STOSWB,the CX wil be.....

1. Incremented by 1
2. Incremented by 2
3. Decrementd by 1
4. Decrementd by 2

Q=10

The execution of the instruction "mov word [ES:160],0x1230"will print a character on the screen at:

1. First column of second row
2. Second column of first row
3. Second column of second row
4. First column of third row

follow

• [Home](#)

•
•
•
•
•
•
•

RSS SUBSCRIBE

- **CS SOLVED QUIZZES**

- **CS101 SOLVED QUIZ**
- **CS201 SOLVED QUIZ**
- **CS301 SOLVED QUIZ**
- **CS302 SOLVED QUIZ**
- **CS304 SOLVED QUIZ**
- **CS401 SOLVED QUIZ**
- **CS403 SOLVED QUIZ**
- **CS504 SOLVED QUIZ**
- **CS610 SOLVED QUIZ**
- **CS601 SOLVED QUIZ**

- **MGT SOLVED QUIZZES**

- **MGT301 SOLVED QUIZ**
- **SUB CATEGORY TWO**
- **SUB CATEGORY THREE**

- **CATEGORY THREE**

- **PARENT CATEGORY ONE**

- **SUB CATEGORY ONE**
- **SUB CATEGORY TWO**
- **SUB CATEGORY THREE**

- **CHILD CATEGORY ONE**

- **UNCATEGORIZED**

- **WHATEVER**



SATURDAY, 19 NOVEMBER 2011

CS401 Mid Term Solved

ADDED JAN 6, 2010, UNDER: CS401 MID TERM SOLVED

Question # 1

There are three busses to communicate the processor and memory named as _____

- 1) : address bus.,data bus and data bus.
- 2) : addressing bus.,data bus and data bus.
- 3) : address bus.,datamove bus and data bus.
- 4) : **address bus.,data bus and control bus..**

Correct Option : 4 From : Lecture 1

Question # 2

The address bus is unidirectional and address always travels from processor to memory.

- 1) : **TRUE**
- 2) : FALSE
- 3) :
- 4) :

Correct Option : 1 From : Lecture 1

Question # 3

Data bus is bidirectional because_____

- 1) : To way
- 2) : Data moves from both, processor to memory and memory to processor,
- 3) : **Data moves from both, processor to memory and memory to data Bus,**
- 4) : None of the Given

Correct Option : 3 From : Lecture 1

Question # 4

Control bus_____

- 1) : is Not Important.
- 2) : is Important .
- 3) : **bidirectional.**
- 4) : unidirectional .

Correct Option : 3 From : Lecture 1

Question # 5

A memory cell is an n-bit location to store data, normally _____ also called a byte

- 1) : 4-bit
- 2) : **8-bit**
- 3) : 6-bit
- 4) : 80-bit

Correct Option : 2 From : Lecture 1

Question # 6

The number of bits in a cell is called the cell width._____ define the memory completely.

- 1) : **Cell width and number of cells,**

- 2) : cell number and width of the cells,
- 3) : width
- 4) : Height

Correct Option : 1 From : Lecture 1

Question # 7

for memory we define two dimensions. The first dimension defines how many _____bits are there in a single memory cell.

- 1) : **parallel**
- 2) : Vertical
- 3) : long
- 4) : short

Correct Option : 1 From : Lecture 1

Question # 8

_____ operation requires the same size of data bus and memory cell width.

- 1) : Normal
- 2) : **Best and simplest**
- 3) : first
- 4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 9

Control bus is only the mechanism. The responsibility of sending the appropriate signals on the control bus to the memory is of the_____.

- 1) : Data Bus
- 2) : **processor**
- 3) : Address Bus
- 4) : None of the Given

Correct Option : 2 From : Lecture 1

Question # 10

In "total: dw o" Opcode total is a _____

- 1) : Literal
- 2) : Variable
- 3) : **Label**
- 4) : Starting point

Correct Option : 3 From : Lecture 10

Question # 11

| 0 | --> | 1 | 1 | 0 | 1 | 0 | 0 | 0 | --> | C | is a example of _____

- 1) : Shl
- 2) : sar

3) : **Shr**

4) : Sal

Correct Option : 3 From : Lecture 10

Question # 12

| C | <-- | 1 | 1 | 0 | 1 | 0 | 0 | 0 | <-- | 0 | is a example of _____

1) : **Shl**

2) : sar

3) : Shr

4) : Sal

Correct Option : 1 From : Lecture 10

Question # 13

ADC has _____ operands.

1) : two

2) : **three**

3) : Five

4) : Zero

Correct Option : 2 From : Lecture 10

Question # 14

The basic purpose of a computer is to perform operations, and operations need _____.

1) : order

2) : nothing

3) : **operands**

4) : bit

Correct Option : 3 From : Lecture 2

Question # 15

Registers are like a scratch pad ram inside the processor and their operation is very much like normal _____.

1) : Number

2) : ooperations

3) : **memory cells**

4) : None of the Given

Correct Option : 3 From : Lecture 2

Question # 16

There is a central register in every processor called the _____ and The word size of a processor is defined by the width of its _____.

1) : **accumulator, accumulator**

2) : data bus, accumulator

3) : accumulator, Address Bus

4) : accumulator,memory

Correct Option : 1 From : Lecture 2

Question # 17

_____ does not hold data but holds the address of data

1) : Pointer, Segment, or Base Register

2) : **Pointer, Index, or Base Register**

3) : General Registers

4) : Instruction Pointer

Correct Option : 2 From : Lecture 2

Question # 18

“The program counter holds the address of the next instruction to be _____”

1) : **executed.**

2) : called

3) : deleted

4) : copy

Correct Option : 1 From : Lecture 2

Question # 19

There are _____ types of “instruction groups”

1) : **4**

2) : 5

3) : 3

4) : 2

Correct Option : 1 From : Lecture 2

Question # 20

These instructions are used to move data from one place to another.

1) : **TRUE**

2) : FALSE

3) :

4) :

Correct Option : 1 From : Lecture 2

Question # 21

“mov” instruction is related to the _____ *****.

1) : Arithmetic and Logic Instructions

2) : **Data Movement Instructions**

3) : Program Control Instructions

4) : Special Instructions

Correct Option : 2 From : Lecture 2

Question # 22

_____ allow changing specific processor behaviors and are used to play with it.

- 1) : **Special Instructions**
- 2) : Data Movement Instructions
- 3) : Program Control Instructions
- 4) : Arithmetic and Logic Instructions

Correct Option : 1 From : Lecture 2

Question # 23

8088 is a 16bit processor with its accumulator and all registers of _____.

- 1) : 32 bits
- 2) : 6 bits
- 3) : **16 bits**
- 4) : 64 bits

Correct Option : 3 From : Lecture 2

Question # 24

The _____ of a processor means the organization and functionalities of the registers it contains and the instructions that are valid on the processor.

- 1) : Manufactures
- 2) : **architecture**
- 3) : Deal
- 4) : None of the Given

Correct Option : 2 From : Lecture 2

Question # 25

Intel IAPX88 Architecture is _____

- 1) : **More then 25 old**
- 2) : New
- 3) : Not Good
- 4) : None of the Given

Correct Option : 1 From : Lecture 2

Question # 26

The iAPX88 architecture consists of _____ registers.

- 1) : 13
- 2) : 12
- 3) : 9
- 4) : **14**

Correct Option : 4 From : Lecture 3

Question # 27

General Registers are _____

1) : **AX, BX, CX, and DX**

2) : XA, BX, CX, and DX

3) : SS,SI and DI

4) : 3

Correct Option : 1 From : Lecture 3

Question # 28

AX means we are referring to the extended 16bit "A" register. Its upper and lower byte are separately accessible as _____.

1) : **AH and AL**

2) : A Lower and A Upper

3) : AL, AU

4) : AX

Correct Option : 1 From : Lecture 3

Question # 29

AX is General purpose Register where A stands for _____.

1) : Acadmic

2) : Ado

3) : Architecture

4) : **Accumulator**

Correct Option : 4 From : Lecture 3

Question # 30

The B of BX stands for _____ because of its role in memory addressing.

1) : Busy

2) : **Base**

3) : Better

4) : None of the Given

Correct Option : 2 From : Lecture 3

Question # 31

The D of DX stands for Destination as it acts as the destination in _____.

1) : **I/O operations**

2) : operations

3) : memory cells

4) : Memory I/O operations

Correct Option : 1 From : Lecture 3

Question # 32

The C of CX stands for Counter as there are certain instructions that work with an automatic count in the _____.

1) : DI register

- 2) : BX register
- 3) : **CX register**
- 4) : DX register

Correct Option : 3 From : Lecture 3

Question # 33

_____ are the index registers of the Intel architecture which hold address of data and used in memory access.

- 1) : SI and SS
- 2) : PI and DI
- 3) : SI and IP
- 4) : **SI and DI**

Correct Option : 4 From : Lecture 3

Question # 34

In Intel IAPX88 architecture _____ is the special register containing the address of the next instruction to be executed.

- 1) : AX
- 2) : PI
- 3) : **IP**
- 4) : SI

Correct Option : 3 From : Lecture 3

Question # 35

SP is a memory pointer and is used indirectly by a set of _____.

- 1) : **instructions**
- 2) : Pointers
- 3) : Indexes
- 4) : Variables

Correct Option : 1 From : Lecture 3

Question # 36

_____ is also a memory pointer containing the address in a special area of memory called the stack.

- 1) : SP
- 2) : **BP**
- 3) : PB
- 4) : AC

Correct Option : 2 From : Lecture 3

Question # 37

_____ is bit wise significant and accordingly each bit is named separately.

- 1) : AX
- 2) : FS
- 3) : IP

4) : **Flags Register**

Correct Option : 4 From : Lecture 3

Question # 38

When two 16bit numbers are added the answer can be 17 bits long, this extra bit that won't fit in the target register is placed in the _____ where it can be used and tested

1) : **carry flag**

2) : Parity Flag

3) : Auxiliary Carry

4) : Zero Flag

Correct Option : 1 From : Lecture 3

Question # 39

Program is an ordered set of instructions for the processor.

1) : **TRUE**

2) : FALSE

3) :

4) :

Correct Option : 1 From : Lecture 3

Question # 40

For Intel Architecture "operation destination, source" is way of writing things.

1) : **TRUE**

2) : FALSE

3) :

4) :

Correct Option : 1 From : Lecture 3

Question # 41

Operation code " add ax, bx " _____.

1) : Add the bx to ax and change the bx

2) : Add the ax to bx and change the ax

3) : **Add the bx to ax and change the ax**

4) : Add the bx to ax and change nothing

Correct Option : 3 From : Lecture 3

Question # 42

The maximum memory iAPX88 can access is _____.

1) : **1MB**

2) : 2MB

3) : 3MB

4) : 128MB

Correct Option : 1 From : Lecture 4

Question # 43

The maximum memory iAPX88 can access is 1MB which can be accessed with _____.

1) : 18 bits

2) : **20 bits**

3) : 16 bits

4) : 2 bits

Correct Option : 2 From : Lecture 4

Question # 44

_____ address of 1DEDo where the opcode B80500 is placed.

1) : **physical memory**

2) : memory

3) : efective

4) : None of the Given

Correct Option : 1 From : Lecture 4

Question # 45

16 bit of Segment and Offset Addresses can be converted to 20bit Address i.e

Segment Address with lower four bits zero + Offset Address with _____ four bits zero = 20bit Physical Address

1) : Middle

2) : lower

3) : Top

4) : **upper**

Correct Option : 4 From : Lecture 4

Question # 46

When adding two 20bit Addresses a carry if generated is dropped without being stored anywhere and the phenomenon is called address_____.

1) : **wraparound**

2) : mode

3) : ping

4) : error

Correct Option : 1 From : Lecture 4

Question # 47

segments can only be defined a 16byte boundaries called _____ boundaries.

1) : **segment**

2) : paragraph

3) : Cell

4) : RAM

Correct Option : 1 From : Lecture 4

Question # 48

in a Program CS, DS, SS, and ES all had the same value in them. This is called _____.

- 1) : equal memory
- 2) : **overlapping segments**
- 3) : segments hiding
- 4) : overlapping SI

Correct Option : 2 From : Lecture 4

Question # 49

"db num1" size of the memory is _____

- 1) : **1byte**
- 2) : 4bit
- 3) : 16bit
- 4) : 2byte

Correct Option : 1 From : Lecture 5

Question # 50

1-----[org 0x0100]

2-----mov ax, [num1] ; load first number in ax

3-----mov bx, [num2] ; load second number in bx

4-----add ax, bx _____

5-----int 0x21

6-----

7-----num1: dw 5

8-----num2: dw 10

Comments for the 4 are :

- 1) : No comments Will be
- 2) : ; accumulate sum in add
- 3) : ; **accumulate sum in ax**
- 4) : ; accumulate sum in Bx

Correct Option : 3 From : Lecture 5

Question # 51

In "mov ax, bx" is _____ Addressing Modes.

- 1) : Immediate
- 2) : Indirect
- 3) : Direct
- 4) : **Register**

Correct Option : 4 From : Lecture 5

Question # 52

In "mov ax, [bx]" is _____ Addressing Modes

1) : **Based Register Indirect**

2) : Indirect

3) : Base Indirect

4) : Immediate

Correct Option : 1 From : Lecture 5

Question # 53

In "mov ax, 5" is _____ Addressing Modes

1) : **Immediate**

2) : Indirect

3) : Indirect

4) : Register

Correct Option : 1 From : Lecture 6

Question # 54

In "mov ax, [num1+bx]" is _____ ADDRESSING

1) : OFFSET+ Indirect

2) : Register + Direct

3) : Indirect + Reference

4) : **BASEd REGISTER + OFFSET**

Correct Option : 4 From : Lecture 7

Question # 55

"base + offset addressing" gives This number which came as the result of addition is called the _____.

1) : Address

2) : mode

3) : **effective address**

4) : Physical Address

Correct Option : 3 From : Lecture 7

Question # 56

"mov ax, [cs:bx]" associates _____ for this one instruction

1) : CS with BX

2) : **BX with CS**

3) : BX with AX

4) : None of the Given

Correct Option : 2 From : Lecture 7

Question # 57

For example

BX=0100

DS=FFFo

And Opcode are;

move [bx+0x0100], Ax

now what is the effective memory address;

1) : 0020

2) : **0200**

3) : 0300

4) : 0x02

Correct Option : 2 From : Lecture 7

Question # 59

In “ mov [1234], al ” is _____ Addressing Modes.

1) : Immediate

2) : Indirect

3) : **Direct**

4) : Register

Correct Option : 3 From : Lecture 8

Question # 60

In “ mov [SI], AX ” is _____ Addressing Modes.

1) : Basef Register Indirect

2) : Indirect

3) : **Indexed Register Indirect**

4) : Immediate

Correct Option : 3 From : Lecture 8

Question # 61

In “ mov ax, [bx - Si] ” is _____ ADDRESSING

1) : Basef Register Indirect

2) : Indirect

3) : Direct

4) : **illegal**

Correct Option : 4 From : Lecture 8

Question # 62

In “ mov ax, [BL] ” there is error i.e. _____

1) : Address must be 16bit

2) : Address must be 8bit

3) : Address must be 4bit

4) : **8 bit to 16 bit move illegal**

Correct Option : 4 From : Lecture 8

Question # 63

In "mov ax, [SI+DI]" there is error i.e. _____

- 1) : **Two indexes can't use as Memory Address**
- 2) : index can't use as Memory Address
- 3) : I don't Know
- 4) : None of the Given

Correct Option : 1 From : Lecture 8

Question # 64

In JNE and JNZ there is difference for only _____;

- 1) : **Programmer or Logic**
- 2) : Assembler
- 3) : Debugger
- 4) : IAPX88

Correct Option : 1 From : Lecture 9

Question # 65

JMP is Instruction that on executing take jump regardless of the state of all flags is called _____

- 1) : Jump
- 2) : Conditional jump
- 3) : **Unconditional jump**
- 4) : Stay

Correct Option : 3 From : Lecture 9

Question # 66

When result of the source subtraction from the destination is zero, zero flag is set i.e. ZF=1 its mean that;

- 1) : **DEST = SRC**
- 2) : DEST != SRC
- 3) : DEST < SRC
- 4) : DEST > SRC

Correct Option : 1 From : Lecture 9

Question # 67

When an unsigned source is subtracted from an unsigned destination and the destination is smaller, borrow is needed which sets the _____.

- 1) : carry flag i.e CF = 0
- 2) : **carry flag i.e CF = 1**
- 3) : Carry Flag + ZF=1
- 4) : None of the Given

Correct Option : 2 From : Lecture 9

Question # 68

In the case of unassigned source and destination when subtracting and in the result ZF =1 OR CR=1 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : **UDEST ? USRC**
- 4) : DEST > SRC

Correct Option : 3 From : Lecture 9

Question # 69

In the case of unassigned source and destination when subtracting and in the result ZF =0 AND CR=0 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST < USRC
- 4) : UDEST > USRC

Correct Option : 4 From : Lecture 9

Question # 70

In the case of unassigned source and destination when subtracting and in the result CR=0 then _____

- 1) : DEST = SRC
- 2) : DEST != SRC
- 3) : UDEST < USRC
- 4) : **UDEST ? USRC**

Correct Option : 4 From : Lecture 9

Question # 71

_____ This jump is taken if the last arithmetic operation produced a zero in its destination. After a CMP it is taken if both operands were equal.

- 1) : **Jump if zero(JZ)/Jump if equal(JE)**
- 2) : Jump if equal(JE)
- 3) : Jump if zero(JZ)
- 4) : No Jump fot This

Correct Option : 1 From : Lecture 9

Question # 72

_____ This jump is taken after a CMP if the unsigned source is smaller than or equal to the unsigned destination.

- 1) : JBE(Jump if not below or equal)
- 2) : **JNA(Jump if not above)/JBE(Jump if not below or equal)**
- 3) : JNA(Jump if not above)
- 4) : No Jump fot This

Correct Option : 2 From : Lecture 9

-
-
-

Looking Something? Search Here

SearchSign UpSign In

image: https://storage.ning.com/topology/rest/1.0/file/get/1894214041?profile=RESIZE_710x

We are here with you hands in hands to facilitate your learning & don't appreciate the idea of copying or replicating solutions. Read More>>

IMAGE:

HTTPS://API.NING.COM/FILES/P0E6PRAPIVQXF22DIUAALSAQN*NOXBHSOJTH70EXFGJGKAMGMJMRXA4EEE*UAXWYTLNIHNJMP2-IHDCQ2LS505LABRZQWQ82/FINALLASTBANNERLASTUPDATED.JPG

Virtual University of Pakistan

WWW.VUSTUDENTS.NING.COM



HOME

CHAT

MYPAGE

MEMBERS

FORUM

STUDY ZONE

BLOGS

VIDEOS

PHOTOS

EVENTS

TOP MOST

CONTACT

Looking For Something at vustudents.ning.com? Click Here to Search

image:

https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=64&height=64&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1



CS401 Computer Architecture and Assembly Language Programming Quiz No 1 Solution and Discussion Spring 2014 Due Date: May 08, 2014

Posted by + M.Tariq Malik on April 26, 2014 at 9:57am in CS401 Computer Architecture and Assembly Language ProgrammingBack to CS401 Computer Architecture and Assembly Language Programming Discussions

CS401 Computer Architecture and Assembly Language Programming Quiz No 1 Solution and Discussion Spring 2014 Due Date: May 08, 2014

Quiz No.01 will open on May 07, 2014 and the last date of taking quiz is May 08, 2014

Like 2 members like this Share

Share This With Friends.....

Share0 0

Tweet0 Share0 Share2

+ Click Here To Join also Our facebook study Group.

This Content Originally Published by a member of VU Students.

+ Prohibited Content On Site + Report a violation + Report an Issue

image: <https://www.animatedimages.org/data/media/111/animated-arrow-image-0103.gif>

..How to Join Subject Study Groups & Get Helping Material?..

image: <https://www.animatedimages.org/data/media/111/animated-arrow-image-0104.gif>



Views: 3073

See Your Saved Posts Timeline

Replies to This Discussion

image:

https://api.ning.com/files/BvvGf9m5PCwpwb1EG9XwhQihCFPSNPkwLNBTbVZHUAnYc5iRYaWz9emJBbBLT1Kqob5UDEML61gCyjnAcfMXgkdP3wGcg45a/1078630746.jpeg?xgip=48%3A13%3A337%3A337%3B%3B&profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

Permalink Reply by ĴĩÃ ĆĤ on April 26, 2014 at 11:39am

thanx 4 info

image: <http://www.bkserv.net/images/Smile.gif>

image: <http://www.bkserv.net/images/Smile.gif>

image: https://api.ning.com/files/vYV2N1KqCyWx-uRXHAn34fAn*zA8xwXJDGrhjQEfkvETBMTTibOuz0CT2Zk*PWhcgH9GT2*p4Grb9gT1LVk*uHXd5EgPjW2O/559187146.jpeg?xgip=9%3A0%3A380%3A380%3B%3B&profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

Permalink Reply by abira ali on May 7, 2014 at 2:26pm

Question # 2

Instruction Pointer holds the address of the



Previous instruction to be executed

Current instruction

Next instruction to be executed

None of the given

Question # 3 of 10

Register whose each bit specify a different meaning is

Accumulator Register

vuzs

Pointer Register

Index register

Flag register



Question # 4 of 10

By default CS is associated with

SS

BP

CX

IP

Question # 5 of 10

Memory to Memory operation is allowed

True

False

Question # 6 of 10



Size Mismatch Error is a syntax error

False (Size mismatch is logical error)

True

Question # 7 of 10

unconditional jump can be

near

short

far

all of the given

Question # 8 of 10

Register are storage cell



Outside the processor

Both inside and outside the processor

Inside the processor

None of the given

www.vuzs.net

Question # 9 of 10

Register to Register Operation is not allowed

True

False

www.vuzs.net

The operation of CMP is to Subtract source from destination

True



False

Size Mismatch Error is a syntax error

False

True

Unconditional jump

Execute in every condition whether true or false

If the condition is true

If the condition is false

None of the given

Which type of Rotation it is "Every bit moves one position to the right and the bit dropped from the right is inserted at the left. This bit is also copied into the carry flag."

ROL

RCR

RCL

None of the given

Assembly language is not a low level language.

true

False

In JA jump is not taken after a CMP if the unsigned destination is larger than the unsigned source.

True

False



Group of bits processor uses to inform memory which element to read/write is collectively known as

Control bus

Data bus

Address bus

RAM

Memory to Memory operation is allowed

True

False

90 is the op-code of

 Do nothing
Add

Subtract

Multiplication

we can not add two base register i.e. (bx+bp) or cant use in an instruction

True

False

Intel follow

Littel endian

Big endian



Both littel endian and big endian

None of the given

SHL and SAL are same

True

False

The first 16-bit processor produced by "Intel" was 8085

True

False



The first 16-bit processor produced by "Intel" was 8085

True

False

www.vuzs.net

The extension of assembly language file is

.doc

.com

.lst

.asm



When a large number is subtracted from a smaller number, a borrow is needed; in this case which flag will be set

ZF

CF

SF

OF

All the addressing mechanisms in iAPX88 return a number called _____ address.

Effective address

Physical address

Direct address

None of the given



Which type of shifting is "Inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag."

SHL

SAL

SAR

None of the given

`mov [1234],ax` is an example of

Direct addressing

Base register indirect

Base+index

None of the given



Registers are also called scratch pad ram

True

False

The basic function of register is to?

Hold the operand

Hold the operator

Hold both the operator and operand

None of the given



www.vuzs.net

The jump is taken if the last arithmetic operation changed the sign unexpectedly.

JO

JNO

JNZ

JZ

In JA jump is not taken after a CMP if the unsigned destination is larger than the unsigned source.

True

False

which type of rotation it is "The carry flag is inserted from the left, every bit moves one position to the right, and the right most bit is dropped in the carry flag. "

RCR

ROL

RCL

ROR

which type of rotation it is "The carry flag is inserted from the left, every bit moves one position to the right, and the right most bit is dropped in the carry flag. "

RCR

ROL

RCL

ROR



This jump is taken if the last arithmetic operation produced a number in its destination that has even parity , Which jump is taken

JP

JPE

JNP

both JP and JPE

In direct addressing the memory address given in the instruction is

When a large number is subtracted from a smaller number, a borrow is needed; in this case which flag will be set

ZF

 CF

SF

OF

SHL and SAL are same

True

False

we can not Subrtace index register from the base register(bx-si)in assemblby language vuzs

True

False

Group of bits processor uses to inform memory which element to read/write is collectively known as



Control bus

Data bus

Address bus

RAM

image:

https://api.ning.com/files/mL7vEimd6q1FFRitFm5rLQihCFPSNPkwLNBTbVZHUAnYc5iRYaWz9emUHutIAuNHbO5Nmxp3F1FxMBmg*RTwgYfRvHI0ahB7/1063873508.jpeg?xgip=0%3A0%3A255%3A255%3B%3B&profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

Permalink Reply by + Mărie Rajput + on May 7, 2014 at 5:12pm

ye sab question ap ko aj k quizz ma aey hain ????

kitny marks ka quizz tha ???

image:

https://api.ning.com/files/iIZAd8bUR*ap85F47P1pmdlJpHtPpcGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcFrE9d-EeIjoS8TnL7it4zsvJ9UqmwuhZZ/vu.jpg?width=48&height=48&crop=1%3A1

Permalink Reply by cancerian on May 7, 2014 at 8:30pm

8 chapters cover 22 lectures. I observed that first four lectures cover first chapter so please give the breakup of other eight chapters like

Chapter # 1 (lectures # 1,2,3,4)

Chapter #. 2 (?)

Chapter # 3

Chapter #. 4

Chapter #5

Chapter # 6

Chapter # 7

Chapter # 8

image: <https://mail.google.com/mail/u/2/images/cleardot.gif>

image:

https://api.ning.com/files/mL7vEimd6q1FFRitFm5rLQihCFPSNPkwLNBTbVZHUAnYc5iRYaWz9emUHutIAuNHbO5NmXP3F1FxMBmg*RTwgYfRvHI0ahB7/1063873508.jpeg?xgip=0%3A0%3A255%3A255%3B%3B&profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

Permalink Reply by + Mărie Rajput + on May 8, 2014 at 6:09am

check out lms regarding this, go to Course website > Lessons

you will find all lessons.

image:

https://api.ning.com/files/rLQihCFPSNPkwLNBTbVZHUAnYc5iRYaWz9em8ZYcIsMOyXzqGK7VbRI0ybWlcF4eIMGmst9m1vEHiWLVQHANwn*wMYxAv3T-/1217870021.jpeg?xgip=0%3A0%3A477%3A477%3B%3B&profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

Permalink Reply by + ❁ 'cm(nisa)` ❁ on May 8, 2014 at 1:23pm

10 marks ka quiz hai +Mărie Rajput+

image: <http://www.bkserv.net/images/Smile.gif>

image:

https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

Permalink Reply by + M.Tariq Malik on May 7, 2014 at 10:40pm

Question # 1 of 10 (Start time: 12:57:07 PM) Total Marks: 1

Mechanism used to drop carry for making the calculated address valid is known as:

Select correct option:

Carry Overload

Overflow

Address Wraparound

None of the above

Question # 2 of 10 (Start time: 12:58:36 PM) Total Marks: 1

we can not Subrtace index register from the base register(bx-si)in assemblby language

Select correct option:

True

False

Question # 3 of 10 (Start time: 12:59:56 PM) Total Marks: 1

Physical address calculation depends on

Select correct option:

Base address

Effective address

Offset Address

None of the above

Question # 4 of 10 (Start time: 01:01:14 PM) Total Marks: 1

Simple CMP instruction uses _____ operation

Select correct option:

Addition

Division

Subtraction

Multiplication

Question # 5 of 10 (Start time: 01:02:45 PM) Total Marks: 1

SS is by default associated with

Select correct option:

BP

IP

SP

BP

Question # 6 of 10 (Start time: 01:04:14 PM) Total Marks: 1

When a 32 bit number is divided by a 16 bit number, the remainder is of

Select correct option:

4 bits

8 bits

16 bits

32 bits

Question # 7 of 10 (Start time: 03:53:38 PM) Total Marks: 1

Which of the following is not a valid instruction in assembly language?

Select correct option:

MOV AX, 55

MOV AX, BX

MOV CS, 0xb800

MOV BX, AX

Question # 8 of 10 (Start time: 03:55:09 PM) Total Marks: 1

Memory to Memory operation is allowed

Select correct option:

True

False

The other directive is "define word" or "dw" with the same syntax as "db" but reserving a whole word of ___ bits instead of a byte.

Select correct option:

32

8

16

64

Question # 10 of 10 (Start time: 03:58:10 PM) Total Marks: 1

we can not add two base register i.e. (bx+bp) or cant use in an instruction

Select correct option:

True

False

BR,

Answers:



1) Address Wraparound

2) True

3) Effective address

4) Subtaction

5) BP

6) 16 bits

7) MOV CS, 0xb800

8) False

9) 8 bits

10) True>

image:

https://api.ning.com/files/iIZAd8bUR*ap85F47P1pmdIJpHtPpcGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcFrE9d-EeIjoS8TnL7it4zsvJ9UqmwuhZZ/vu.jpg?width=48&height=48&crop=1%3A1



2aqlargmrfxmzPermalink Reply by 2aqlargmrfxmz on May 7, 2014 at 11:11pm

Tariq Bhai question No 9 ka ans 8 bits nai 16 bits ha

image: <http://www.bkserv.net/images/Smile.gif>

image: <http://www.bkserv.net/images/Smile.gif>

image: <http://www.bkserv.net/images/Smile.gif>

image:

https://api.ning.com/files/rLQihCFPSNPkwLNBTbVZHUAnc5iRYaWz9em8ZYcIsMOyXzqGK7VbRI0ybwlcf4eIMGMst9m1vEHiWLVQHANwn*wMYxAv3T-/1217870021.jpeg?xgip=0%3A0%3A477%3A477%3B%3B&profile=RESIZE_180x180&width=48&height=48&crop=1%3A1

Permalink Reply by + ☼ 'cm(NISQ)`` ☼ on May 8, 2014 at 1:25pm

Hmm 16 hi hai

image: <http://www.bkserv.net/images/Smile.gif>

image:

https://api.ning.com/files/iIZAd8bUR*ap85F47P1pmdIJpHtPpcGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcFrE9d-EeIjoS8TnL7it4zsvJ9UqmwuhZZ/vu.jpg?width=48&height=48&crop=1%3A1



1quypvq75nb4qPermalink Reply by 1quypvq75nb4q on May 8, 2014 at 4:28pm

Today main chat room ma ya quiz solved kiya tha sub nay ??? ager koi ans theek nai ha to correct ker den..thax

1. When a large number is subtracted from a smaller number, a borrow is needed; in this case which flag will be se

cf

2. jump is not position relative but is absolute

far

3. Group of bits processor uses to inform memory which element to read/write is collectively known as

address bus

4. A complete _____ is called a pass over the array

ITERATION

5. There are _____ types of address wraparound

2



6. All the addressing mechanisms in iAPX88 return a number called _____ address.

Effective

7. Whenever we need access to a memory location whose address is not known until run-time we use _____.

INDEX REGISTOR

8. DX plays an important role in arithmetic _____.

DIVISION

9. If BL contains 00000101 then after a Single Right Shift, BL will contain

00000011

10. To multiply a number in a register by 2 the number is _____.

Shifted right one bit

image:

https://api.ning.com/files/iIZAd8bUR*ap85F47P1pmdIJpHtPpcGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcFrE9d-EeIjoS8TnL7it4zsvJ9UqmwuhZZ/vu.jpg?width=48&height=48&crop=1%3A1

1quypvq75nb4qPermalink Reply by 1quypvq75nb4q on May 8, 2014 at 6:13pm

SP is associated (by default) with _____.

CS

Intel follows _____.

Little endian

The maximum amount of memory accessible using 8085 processor is _____.

64 kb

Which of the following addressing scheme has been used in the instruction MOV [BX], AX?.....

base register direct access

Instruction Pointer holds the address of the.....

next instruction to be executed

Sending the appropriate signals on the control bus to the memory is the responsibility of _____.

processor control

Register whose each bit specifies a different meaning is-----

flag Register

The iAPX88 processor supports _____ modes of memory access.

7

CX register is mostly used as a

counter control

Constant can never be used as _____.

source

destination

both source and destination

immediate source



Which one of the following is an illegal instruction?

MOV ax,[bx+bp]

In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag ?

RCR

RCL

ROR

ROL

In _____ operation, a carry flag is inserted from the left moving every bit one position to the right, with the right most bit is dropped in the carry flag".

RCR

ROL

RCL

ROR

Which of the following is not a valid instruction in assembly language?

MOV CS, 0xb800

 Data bus is _____.

bidirectional confirm

Simple CMP instruction uses _____ operation.

Subtraction

The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

Shift Logical Right (SHR)

Shift Arithmetic Right (SAR)

Shift Arithmetic Left (SAL)

Shift Logical Left (SHL)

After the execution of SAR instruction, _____.

The msb is replaced by a 0



The msb retains its original value

The msb is replaced by 1

The msb is replaced by the value of CF

Which of the following instruction is effectively same as to multiply the value of AX by 8?

MUL AX, 3

The shift logical left operation is the exact _____ of shift logical right.

opposite

The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

Shift Logical Left (SHL)

SHR

SHL



The basic function of register is to

Hold the operand

When a 32 bit number is divided by a 16 bit number, the remainder is of

8

n _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag ?

ROL

RCR

RCL

ROR

mov [bp], al" moves the one byte contents of the AL register to the address contained in BP register in the current _____.

Stack Segment



Data Segment

Code Segment

Extra SegmentExtra Segment

Which of the following shift operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag ?

SHL

In case of short jump, the offset is stored in _____ .

1 2 4 16 bytes?

image:

https://api.ning.com/files/iIZAd8bUR*ap85F47P1pmdlJpHtPpcGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcFrE9d-EeIjoS8TnL7it4zsvJ9UqmwuhZZ/vu.jpg?width=48&height=48&crop=1%3A1

1quypvq75nb4qPermalink Reply by 1quypvq75nb4q on May 8, 2014 at 7:58pm

The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place. The bit dropped from the right is caught in the carry basket.

Shift Logical Right (SHR)

Shift Arithmetic Right (SAR)



Shift Arithmetic Left (SAL)

Shift Logical Left (SHL)

< Previous

1

2

Next >

Page

2

RSS

Welcome Guest

image: https://api.ning.com/files/WltTFfACAjceq0Z9scF-2P8-PEOCiskrhevMG69xLcPEO2dN6EUNXc1D02aw3mZyUCwZS78JwFbU4CraNbOH5mwVvAbouPgH/registernow_blue.png?width=175

Sign Up

or Sign In

LATEST ACTIVITY

30+ new items

image:  https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS507 Information Systems

10 hours ago

image:

https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFvGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS507 Information Systems

10 hours ago

image:

https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFvGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS507 Information Systems

10 hours ago

image:

https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS507 Information Systems

10 hours ago

image: https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik's 8 discussions were featured

CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material

CS508 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material

 MCM101 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material

5 more...

10 hours ago

image: https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFvGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik liked + M.Tariq Malik's discussion CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material

10 hours ago

image: https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFvGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik added a discussion to the group CS507 Information Systems

image: https://api.ning.com/files/F2gXzo9gtlVpHeOGhm0X5j*BfvL9u5mRqmw-CkZ3NaBkbzOJrkZDprWBIJ-B4hNk6XEs518vC-lp7*UZ-H38TIWLTN2Mv475/imagesCAFNEF1J.jpg?profile=RESIZE_180x180&crop=1%3A1&width=64

Thumbnail

CS507 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material

10 hours ago

4 1

image: https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFvGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS508 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS508 Modern Programming Languages

10 hours ago

image:
https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFvGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS508 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS508 Modern Programming Languages

10 hours ago

image:
https://storage.ning.com/topology/rest/1.0/file/get/292124640?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1



Muhammad Bilal liked zohaib iftikhar's discussion ...* HALAAT K QADMON PE KABHI QALANDER NAHI GIRTA ...*

10 hours ago

image:
https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS508 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS508 Modern Programming Languages

10 hours ago

image: https://storage.ning.com/topology/rest/1.0/file/get/2742202890?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

image: https://api.ning.com/files/bR8nA-I0SJG6CIZy-G9bFxGXvn7hyT4N02K91fZsBDwSGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcF-opkYm*m2jWzSgMBAS/ddd.png?size=64&crop=1%3A1

+ M.Tariq Malik replied to + M.Tariq Malik's discussion CS508 Mid Term Papers Spring 2019 (15~28-June 2019) & All Solved Past Papers, Solved MCQs & Helping Material in the group CS508 Modern Programming Languages

10 hours ago

[More Latest Activity? Click Here](#)

[RSS](#)



STUDY CORNER FOR DIGISKILLS STUDENTS

image: https://storage.ning.com/topology/rest/1.0/file/get/2200595117?profile=RESIZE_710x

image: https://storage.ning.com/topology/rest/1.0/file/get/2200942483?profile=RESIZE_710x

TODAY TOP MEMBERS

1

image:

https://storage.ning.com/topology/rest/1.0/file/get/2665646886?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

+ ツ

2

image:

https://storage.ning.com/topology/rest/1.0/file/get/1568148567?profile=RESIZE_180x180&width=32&height=32&crop=1%3A1

+ ≧ J K ≧ B.E D 1.5

3

image:

https://api.ning.com/files/iIZAd8bUR*ap85F47P1pmdIJpHtPpcGf9m5PCwpwb1EG9XyJgwVosLWYX7URUBcFrE9d-EeIjoS8TnL7it4zsvJ9UqmwuhZZ/vu.jpg?width=32&height=32&crop=1%3A1

A



RSS

View All

image: https://storage.ning.com/topology/rest/1.0/file/get/1894642504?profile=RESIZE_710x

In Left - Shift - Operation , the left most bit _____

Select correct option

will be dropped

will go into CF

will be moved to the right - most position

will always be 1

Question # 2 of 10 (Start time:

09 :16 :53 PM) Total Marks: 1

" mov [bp] , al " moves the one byte contents of the AL register to the address contained in BP register in the current _____ .

Select correct option :

Stack Segment

Data Segment

Code Segment

Extra Segment

Question # 3 of 10 (Start time:

09 :17 :33 PM) Total Marks: 1

CX register is mostly used as a

Select correct option :

counter register

flag register

base register

destination register

Question # 4 of 10 (Start time:

09 :18 :55 PM) Total Marks: 1

By default CS is associated with

Select correct option :

SS

BP

CX

IP

Question # 5 of 10 (Start time:

09 :20 :30 PM) Total Marks: 1

Which of the following shift operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag ?

Select correct option :

SHL

SAL

SAR

SHR

Question # 6 of 10 (Start time:

09 :21 :58 PM) Total Marks: 1

The _____ operation is about shifting every bit one place to the right with a copy of the most significant bit left at the most significant place . The bit dropped from the right is caught in the carry basket .

Select correct option :

Shift Logical Left (SHL)

Shift Logical Right (SHR)

Shift Arithmetic (SAR)

Shift Arithmetic Left (SAL)

Question # 7 of 10 (Start time:

09 :23 :17 PM) Total Marks: 1

In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag ?

Select correct option :

ROL

RCR

RCL

ROR

Question # 8 of 10 (Start time:

09 :24 :15 PM) Total Marks: 1

Which one of the following is an illegal instruction?

Select correct option

MOV AX ,BX

MOV AX ,65

MOV ax ,[bx + bp]

Mov BX , 10

Question # 9 of 10 (Start time:

09 :24 :54 PM) Total Marks: 1

The shift logical left operation is the exact _____ of shift logical right .

Select correct option :

Similar

Opposite

implementation

comparison

Question # 10 of 10 (Start time :

09 :25 :41 PM) Total Marks: 1

Physical address calculation depends on

Select correct option :

Base address

Effective address

Offset Address

Segment Address

1

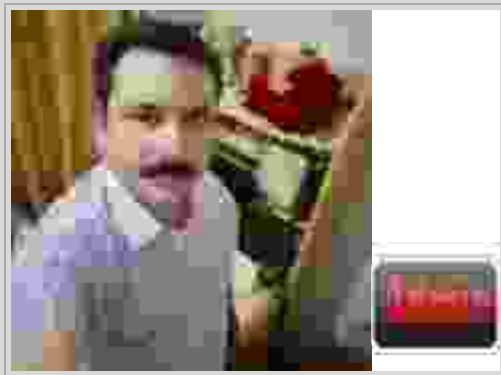


CS401 Computer Architecture and Assembly Language Programming

2020 members

[Description](#)

[Comments](#)[Discussions](#)[Members](#)



CS401 Online Quiz No 02 Solution & Discussion Last Date:17-12-2014

by [+ M.Tariq Malik](#)

Dec 8, 2014

CS401 Online Quiz No 02 Solution & Discussion Last Date:17-12-2014

CS401 - Computer Architecture and Assembly Language Programming Online Quiz 2 Solution Fall 2014 of Virtual University (VU)

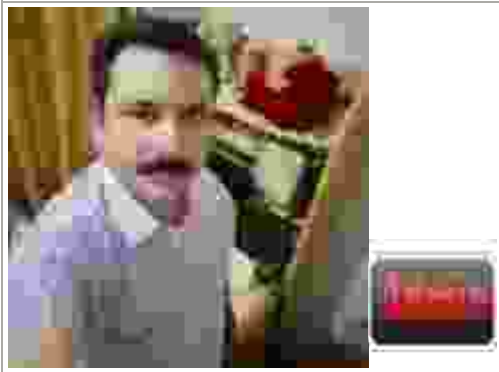
Dear Students

This is to inform that quiz 02 will be opened on 16th December, 2014 and last date to attempt quiz will be 17th December, 2014.

Instructions:

- You can start attempting the quiz at any time but within given date(s) by clicking the quick link for Quiz on VU-LMS as it will become enabled within the mentioned dates. As soon as the time will be over, it will automatically be disabled and will not be available to attempt it.

[Load Previous Replies](#)



• [+ M. Tariq Malik](#)

CS401 Quiz 2

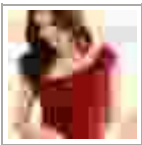
Dec 17, 2014



• [nina](#)

XOR can also be used as a _____ to invert selective bits.
Making operations

Dec 17, 2014



• [Syeda Shahzadi Bukhari \(BS 8th \)](#)

Question # 1 of 10 (Start time: 09:38:11 PM) Total Marks: 1

In case of near jump, the relative address is stored in _____ bits.

Select correct option:

4

8

16 right

32

Question # 2 of 10 (Start time: 09:38:39 PM) Total Marks: 1
The Jump command that does not depend on FLAG register is
Select correct option:

JCXZ
JO
JNE.....right
JP

Question # 3 of 10 (Start time: 09:40:04 PM) Total Marks: 1
In SCAS Example, We use SCASB with _____ and a zero in AL register to
find a zero byte in a string.
Select correct option:

REPNE ...right
SCAS
MOV
CALL

Question # 4 of 10 (Start time: 09:40:24 PM) Total Marks: 1
How much byte/bytes cannot be pushed and popped from the stack at once.
Select correct option:

Single
Double
Three right not sure
Four

Question # 5 of 10 (Start time: 09:41:55 PM) Total Marks: 1
Which of the following are the two variants of STOS instruction?
Select correct option:

STOSB and STOSW right
STOS and STOSES
STOS1 and STOS2
STOSA and STOSB

Question # 6 of 10 (Start time: 09:43:00 PM) Total Marks: 1
_____ jump is not position relative but is absolute
Select correct option:

Near
Short
Far right
Extra

Question # 7 of 10 (Start time: 09:43:22 PM) Total Marks: 1
DW can store _____ bit value in it.
Select correct option:

8
16
24
32 ..right

Question # 8 of 10 (Start time: 09:44:13 PM) Total Marks: 1
LDS instruction loads _____ register.
Select correct option:

ES
DS right
CX

Question # 9 of 10 (Start time: 09:44:46 PM) Total Marks: 1
Our computers screen is like a 2-D array having _____ rows and
_____ columns.
Select correct option:

25, 40
25, 80
80, 25 right

Question # 10 of 10 (Start time: 09:46:08 PM) Total Marks: 1
To convert the case of a character, we add or subtract _____
from its ASCII code.
Select correct option:

0x10
0x20.....right
0x30
0x41

Dec 17, 2014

[1 member likes this](#)

4



ALL CS401 solved Quiz no 1 and 2 (2013 and 2014) in one discussion

by [+ WASI\(S.Admin\) +](#)
May 27, 2014

Cs 401 Quiz no 1 (2014)@ wasi

Question # 1 of 10 (Start time: 12:57:07 PM) Total Marks: 1
Mechanism used to drop carry for making the calculated address valid is known as:

Select correct option:

Carry Overload

Overflow

Address Wraparound

None of the above

Question # 2 of 10 (Start time: 12:58:36 PM) Total Marks: 1
we can not Subrtace index register from the base register(bx-si)in assemblby language

Select correct option:

True

False

Question # 3 of 10 (Start time: 12:59:56 PM) Total Marks: 1
Physical address calculation depends on

Select correct option:

Base address

Effective address

Offset Address

None of the above

Question # 4 of 10 (Start time: 01:01:14 PM) Total Marks: 1
Simple CMP instruction uses ____ operation

Select correct option:

Addition

Division

Subtraction

Multiplicaion

Question # 5 of 10 (Start time: 01:02:45 PM) Total Marks: 1

SS is by default associated with

Select correct option:

BP

IP

SP

BP

Question # 6 of 10 (Start time: 01:04:14 PM) Total Marks: 1

When a 32 bit number is divided by a 16 bit number, the remainder is of

Select correct option:

4 bits

8 bits

16 bits

32 bits

Question # 7 of 10 (Start time: 03:53:38 PM) Total Marks: 1

Which of the following is not a valid instruction in assembly language?

Select correct option:

MOV AX, 55

MOV AX, BX

MOV CS, 0xb800

MOV BX, AX

Question # 8 of 10 (Start time: 03:55:09 PM) Total Marks: 1

Memory to Memory operation is allowed

Select correct option:

True

False

Question # 9 of 10 The other directive is “define word” or “dw” with the same syntax as “db” but reserving a whole word of __ bits instead of a byte.

Select correct option:

32

8

16

64

Question # 10 of 10 (Start time: 03:58:10 PM) Total Marks: 1

we can not add two base register i.e. (bx+bp) or cant use in an instruction

Select correct option:

True

False



• [+ WASI\(S.Admin\) +](#)

Cs 401 Quiz no 1 (2014)@ wasi

Question # 1 of 10 (Start time: 12:57:07 PM) Total Marks: 1

Mechanism used to drop carry for making the calculated address valid is known as:

Select correct option:

Carry Overload

Overflow

Address Wraparound

None of the above

Question # 2 of 10 (Start time: 12:58:36 PM) Total Marks: 1

we can not Subrtace index register from the base register(bx-si)in assembly language

Select correct option:

True

False

Question # 3 of 10 (Start time: 12:59:56 PM) Total Marks: 1

Physical address calculation depends on

Select correct option:

Base address

Effective address

Offset Address

None of the above

Question # 4 of 10 (Start time: 01:01:14 PM) Total Marks: 1

Simple CMP instruction uses _____ operation

Select correct option:

Addition

Division

Subtraction

Multiplicaion

Question # 5 of 10 (Start time: 01:02:45 PM) Total Marks: 1

SS is by default associated with

Select correct option:

BP

IP

SP

BP

Question # 6 of 10 (Start time: 01:04:14 PM) Total Marks: 1

When a 32 bit number is divided by a 16 bit number, the remainder is of

Select correct option:

4 bits

8 bits

16 bits

32 bits

Question # 7 of 10 (Start time: 03:53:38 PM) Total Marks: 1

Which of the following is not a valid instruction in assembly language?

Select correct option:

MOV AX, 55

MOV AX, BX

MOV CS, 0xb800

MOV BX, AX

Question # 8 of 10 (Start time: 03:55:09 PM) Total Marks: 1

Memory to Memory operation is allowed

Select correct option:

True

False

Question # 9 of 10 The other directive is “define word” or “dw” with the same syntax as “db” but reserving a whole word of ___ bits instead of a byte.

Select correct option:

32
8
16
64

Question # 10 of 10 (Start time: 03:58:10 PM) Total Marks: 1
we can not add two base register i.e. (bx+bp) or cant use in an instruction
Select correct option:

True
False

Cs 401 Quiz no 1 (2014)@ wasi

1. When a large number is subtracted from a smaller number, a borrow is needed; in this case which flag will be

Ans cf

2. jump is not position relative but is absolute.

Ans far

3. Group of bits processor uses to inform memory which element to read/write is collectively known as.

Ans address bus

4. A complete _____ is called a pass over the array.

Ans ITERATION

5. There are _____ types of address wraparound.

Ans 2

6. All the addressing mechanisms in iAPX88 return a number called _____ address.

Ans Effective

7. Whenever we need access to a memory location whose address is not known until run-time we use _____.

Ans INDEX REGISTOR

8. DX plays an important role in arithmetic _____.

Ans DIVISION

9. If BL contains 00000101 then after a Single Right Shift, BL will contain

Ans 00000011

10. To multiply a number in a register by 2 the number is _____.

Ans Shifted right one bit

11. SP is associated (by default) with _____.

Ans cs

12. Intel follows _____.

Ans Little endian

13 The maximum amount of memory accessible using 8085 processor is _____.

Ans 64 kb

14. Which of the following addressing scheme has been used in the instruction MOV [BX], AX?.....

Ans base register direct access

15. Instruction Pointer holds the address of the.....

Ans next instruction to be executed

16. Sending the appropriate signals on the control bus to the memory is the responsibility of _____.

Ans processor control

17. Register whose each bit specifies a different meaning is-----

Ans flag Register

18. The iAPX88 processor supports _____modes of memory access.

Ans 7

19. CX register is mostly used as a

Ans counter confirm

20.Which one of the following is an illegal instruction?

Ans MOV ax,[bx+bp]

21. Which of the following is not a valid instruction in assembly language?

Ans MOV CS, 0xb800

22. Data bus is _____.

Ans bidirectional confirm

23.Simple CMP instruction uses _____ operation.

Ans Subtraction

24. Which of the following instruction is effectively same as to multiply the value of AX by 8?

Ans MUL AX, 3

25. The shift logical left operation is the exact _____ of shift logical right.

Ans oposite

26. When a 32 bit number is divided by a 16 bit number, the remainder is of

Ans 8

27. Which of the following shift operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag ?

Ans SHL

28. In _____ every bit moves one position to the right and the bit dropped from the right is inserted at the left and also copied into the carry flag ?

ROR

RCL

Ans ROR

ROL

29. In _____ operation, a carry flag is inserted from the left moving every bit one position to the right, with the right most bit is dropped in the carry flag".

Ans RCR

ROL

RCL

ROR

[4](#)

Both DS and ES can be used to access the video memory. However we commonly keep DS for accessing our data, and load ES with the segment of video memory. Select correct option:

True

False

PAGE 81

Question # 2 of 10 (Start time: 03:17:23 PM) Total Marks: 1
How many characters standard ASCII has?
Select correct option:

512

256

128

64

During the CALL operation, the current value of the instruction pointer is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer.
 Select correct option:

- True**
- False

VGA stands for
 Select correct option:

- Video Graphic Accumulator
- Video Graphics Adapter**
- Visual Graphics Adapter
- Video Graphics Application

Video Graphics Adapter

_____ transfers the word at the current top of stack (pointed to by SP) to the destination operand and then increments SP by two to point to the new top of stack.
 Select correct option:

- PUSH
- POP**
- CALL
- None of the given

The execution of the instruction "mov word [ES : 160], 0x1230" will print a character on the screen at:
 Select correct option:

- First column of second row**
- Second column of first row
- Second column of second row
- First column of third row

The Operation of pop ax is AX <-- [SP] SP <-- SP-2
 Select correct option:

True

False

_____ function decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.
Select correct option:

POP

PUSH

RET

ADD

Far calls are called intra segment calls.
Select correct option:

True

False

STOS is often used to clear a block of memory or fill it with a constant.
Select correct option:

True

False

How many characters were defined by standard ASCII?

Select correct option:

132

124

122

128

Stack clearing by the caller needs an extra instruction on behalf of the caller after every call made to the subroutine, unnecessarily increasing instructions in the program.
Select correct option:

True

False

The direction of movement is controlled with the _____ in the flags register. If this flag is cleared the direction is from lower addresses towards higher addresses and if this flag is set the direction

is from higher addresses to lower addresses.
Select correct option:

Direction	Flag	(DF)
Control	Flag	(CF)
Carry	Flag	(CF)
Non of above		

ASCII stands for _____.
Select correct option:

American	Standard	Code	for	Information	Interaction
American	Standard	Code	for	Information	Interchange
American	Standard	Communication	for	Integer	Interchange
American Scientific Communication for Integer Interaction					

During the CALL operation, the current value of the instruction pointer is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer.
Select correct option:

Extended ASCII has 256 characters with assigned numbers from
Select correct option:

1	to	255
0	to	256
0	to	255
1 to 256		

Hexadecimal is the prevalent and standard format for representation of characters in computers.
Select correct option:

True
False

The execution of the instruction "mov word [ES : 160], 0x1230" will print a character on the screen at:
Select correct option:

First	column	of	second	row
Second	column	of	first	row
Second	column	of	second	row
First column of third row				

In the instruction "mov word [es:160], 0x1230", 12 means:
Select correct option:

green	color	on	black	background
green	color	on	blue	background
black	color	on	green	background
blue color on green background				

This top of stack is contained in the ____ register.
Select correct option:

SP
BP
AX
BX

PUSH increments SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.
Select correct option:

True
False

Page 71

The reduction in code size and the improvement in speed are the two reasons why block processing instructions were introduced in the _____ processor.
Select correct option:

8088
8085
8080
iAPX386

MOVS is used to _____ a block of memory.
Select correct option:

Save
Move

Delete
Push

The operation of PUSH is not similar to CALL however with a register other than the instruction pointer.
Select correct option:

True
False

Reference: operation of PUSH is similar to CALL however with a register other than the instruction pointer. page 69

Both DS and ES can be used to access the video memory. However we commonly keep DS for accessing our data, and load ES with the segment of video memory.
Select correct option:

True
False

Which bit of the attributes byte represents the blue component of foreground color ?
Select correct option:

0
1
2
3

SCAS compares a source byte or word in register AL or AX with the _____ string element addressed by ES: DI and updates the flags.

Source

Destination

Flag

Register

page 92

An element is pushed on the stack SP is decremented by ____ as the 8088 stack works on word sized elements.

Three

two

four

five

page 68

To access the arguments from the stack, the immediate idea that strikes is to ____ them off the stack.

push

pop

add

insert

We use _____ to access the parameters that are stay on the stack with out popping them.

pop

DS not sure

PUSH

BP

How many characters were defined by standard ASCII?

132

124

128

ASCII table is the contiguous arrangement of the uppercase alphabets (41-5A), the lowercase alphabets (61-7A), and the numbers_____

31-40

29-39

30-39

page no 80

The purpose of MOVS instruction is to move a memory location to register

True

False

_____ decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.

push

pop

call

None

This top of stack is contained in the ____ register.

SP

BP

AX

To access the arguments from the stack, the immediate idea that strikes is to ____ them off the stack.

Select _____ correct _____ option:

Push

Pop

Add

Insert

Parameters _____ clears _____ from _____ the _____ stack _____ by _____ the

Select _____ correct _____ option:

caller

callee

caller _____ and _____ callee

None of the given

There _____ are _____ just _____ block _____ processing _____ instructions _____ in _____ 8088.

Select _____ correct option:
6
5
4
3

_____ decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP.
Select _____ correct option:
PUSH
POP
CALL
None of the Above

The operation of PUSH is not similar to CALL however with a register other than the instruction pointer.
Select _____ correct option:
True
False

The direction of movement is controlled with the _____ in the flags register. If this flag is cleared the direction is from lower addresses towards higher addresses and if this flag is set the direction is from higher addresses to lower addresses.
Select _____ correct option:

Direction	Flag	(DF)
Control	Flag	(CF)
Carry	Flag	(CF)
Non of above		

Hexadecimal is the prevalent and standard format for representation of characters in computers.
Select _____ correct option:
True
False

MUL (multiply) Instruction performs an unsigned multiplication of the source operand and the _____.
Select _____ correct option:
Accumulator
Carry

Word
Base

We can convert any digit to __ by adding 0x30 in the digit.
Select correct option:

Character

ASCII

EBCDIC

Standard Code

A typical stack is an area of computer memory with a fixed origin and a variable

True

False

Local variables should be created when the subroutine is called and discarded afterwards.

True

False

Both DS and ES can be used to access the video memory. However we commonly keep DS for accessing our data, and load ES with the segment of video memory.

True

False

Elements are removed from the stack in the reverse order to the order of their addition.

True

False

The Operation of Push is if "we push ax" then $SP \leftarrow SP + 2$ $[SP] \leftarrow AX$

True

False

The operations of placing items on the stack and removing them from there are called push and ret.

True

False

A _____ is an area of memory that holds all local variables and parameters used by any function and remembers the order in which functions are called so that function returns occur correctly.

Instruction Pointer

Stack

Data Segment

Base Register

Stack is a data structure that behaves in a first in last _____ manner.

In

Out

Push

Add

RET do not pops the word at the top of the stack (pointed to by register SP) into the instruction pointer but increments SP by two.

True

False

--

Question No: 1 (Marks: 1) - Please choose one

After

the execution of SAR instruction

▶ The msb is replaced by a 0

- ▶ The msb is replaced by 1
- ▶ The msb retains its original value
- ▶ The msb is replaced by the value of CF

Question No: 2 (Marks: 1) - Please choose one

RETF

will pop the offset in the

- ▶ BP
- ▶ IP
- ▶ SP
- ▶ SI

Question No: 3 (Marks: 1) - Please choose one

The

routine that executes in response to an INT instruction is called

▶ ISR

- ▶ IRS
- ▶ ISP
- ▶ IRT

Question No: 4 (Marks: 1) - Please choose one

The

first instruction of "COM" file must be at offset:

- ▶ 0x0010
- ▶ 0x0100
- ▶ 0x1000
- ▶ 0x0000

Question No: 5 (Marks: 1) - Please choose one

"Far"

jump is not position relative but is _____

▶ memory dependent

▶ Absolute

▶ temporary

▶ indirect

Question No: 6 (Marks: 1) - Please choose one

Only

_____ instructions allow moving data from memory to memory.

▶ string

▶ word

▶ indirect

▶ stack

Question No: 7 (Marks: 1) - Please choose one

After

the execution of instruction "RET 2"

▶ SP is incremented by 2

▶ SP is decremented by 2

▶ SP is incremented by 4

▶ SP is decremented by 4

Question No: 8 (Marks: 1) - Please choose one

DIV

instruction has

▶ Two forms

▶ Three forms

▶ Four forms

▶ Five forms

Question No: 9 (Marks: 1) - Please choose one

When

the operand of DIV instruction is of 16 bits then implied dividend will be of

- ▶ 8 bits
- ▶ 16 bits
- ▶ 32 bits
- ▶ 64 bits

Question No: 10 (Marks: 1) - Please choose one

After

the execution of MOVS instruction which of the following registers are updated

- ▶ SI only
- ▶ DI only
- ▶ SI and DI only
- ▶ SI, DI and BP only

Question No: 11 (Marks: 1) - Please choose one

In

8088 architecture, whenever an element is pushed on the stack

- ▶ SP is decremented by 1
- ▶ SP is decremented by 2
- ▶ SP is decremented by 3
- ▶ SP is decremented by 4

Question No: 12 (Marks: 1) - Please choose one

When

a very large number is divided by very small number so that the quotient is larger than the space provided, this is called

- ▶ Divide logical error
- ▶ Divide overflow error
- ▶ Divide syntax error
- ▶ An illegal instruction

Question No: 13 (Marks: 1) - Please choose one

In the

word designated for one screen location, the higher address contains

► The character code

- The attribute byte
- The parameters
- The dimensions

Question No: 14 (Marks: 1) - Please choose one

Whic

h of the following options contain the set of instructions to open a window to the video memory?

- mov AX, 0xb008

mov ES, AX

► mov AX, 0xb800

mov ES, AX

- mov AX, 0x8b00

mov ES, AX

- mov AX, 0x800b

mov ES, AX

Question No: 15 (Marks: 1) - Please choose one

In a

video memory, each screen location corresponds to

- One byte

► Two bytes

- Four bytes
- Eight bytes

Question No: 16 (Marks: 1) - Please choose one

The

execution of the instruction "mov word [ES : 0], 0x0741" will print character "A" on screen , background color of the screen will be

► Black

- White
- Red
- Blue

Question No: 17 (Marks: 2)

Why

is it necessary to provide the segment and offset address in case of FAR jump ?

Segment and offset must be given to a far jump. Because, sometimes we may need to go from one code segment to another, and near and short jumps cannot take us there. Far jump must be used and a two byte segment and a two byte offset are given to it. It loads CS with the segment part and IP with the offset part.

Question No: 18 (Marks: 2)

What'

s your understanding about Incrementing and Decrementing Stack?

Whenever an element is pushed on the stack SP is decremented by two and whenever an element is popped on the stack SP is incremented by two.

A decrementing stack moves from higher addresses to lower addresses as elements are added in it while an incrementing stack moves from lower addresses to higher addresses as elements are added.

As the 8088 stack works on word sized elements. Single bytes cannot be pushed or popped from the stack.

Question No: 19 (Marks: 2)

Num

ber2:

IF DF=0 what its represent and IF DF=1 what its represent ?

The direction of movement is controlled with the Direction Flag (DF) in the flags register. If this flag is cleared DF=0, the direction is from lower addresses towards higher addresses and if this flag is set DF=1, the direction is from higher addresses to lower addresses. If DF is cleared, DF = 0 this is called the autoincrement mode of string instruction, and if DF is set, DF=1, this is called the autodecrement mode. There are two instructions to set and clear the direction flag.

Question No: 20 (Marks: 3)

What

is the Difference between CALL and RET

The CALL instruction allows temporary diversion and therefore reusability of code.

The word return holds in its meaning that we are to return from where we came and need no explicit destination.

Therefore RET takes no arguments and transfers control back to the instruction following the CALL that took us in this subroutine.

Question No: 21 (Marks: 3)

Tell

the Formula to scroll up the screen

rep movsw

scroll up

```
scrollup: push bp
mov bp,sp
push ax
push cx
push si
push di
push es
push ds
mov ax, 80 ; load chars per row in ax
mul byte [bp+4] ; calculate source position
mov si, ax ; load source position in si
push si ; save position for later use
shl si, 1 ; convert to byte offset
mov cx, 2000 ; number of screen locations
sub cx, ax ; count of words to move
mov ax, 0xb800
mov es, ax ; point es to video base
mov ds, ax ; point ds to video base
xor di, di ; point di to top left column
cld ; set auto increment mode
rep movsw ; scroll up
mov ax, 0x0720 ; space in normal attribute
pop cx ; count of positions to clear
rep stosw ; clear the scrolled space
pop ds
pop es
pop di
pop si
pop cx
pop ax
pop bp
ret 2
```

Question No: 22 (Marks: 5)

Expla

in how extended shifting is performed

Using our basic shifting and rotation instructions we can effectively shift a 32bit number in memory word by word. We cannot shift the whole number at once since our architecture is limited to word operations. The algorithm we use consists of just two instructions and we name it extended shifting.

```
num1: dd 40000
shl word [num1], 1
rcl word [num1+2], 1
```

The DD directive reserves a 32bit space in memory; however the value we placed there will fit in 16bits. So we can safely shift the number left 16 times.

The least significant word is accessible at num1 and the most significant word is accessible at num1+2.

The two instructions are carefully crafted such that the first one shifts the lower word towards the left and the most significant bit of that word is dropped in carry. With the next instruction we push that dropped bit into the least significant bit of the next word effectively joining the two 16bit words.

The final carry after the second instruction will be the most significant bit of the higher word, which for this number will always be zero.

Question No: 23 (Marks: 5)

Write

a subroutine to calculate the string length.?

subroutine to calculate the length of a string

; takes the segment and offset of a string as parameters

strlen: push bp

mov bp,sp

push es

push cx

push di

les di, [bp+4] ; point es:di to string

mov cx, 0xffff ; load maximum number in cx

xor al, al ; load a zero in al

repne scasd ; find zero in the string

mov ax, 0xffff ; load maximum number in ax

sub ax, cx -- --

The first instruction of COM file must be at offset:

a. 0x0010

b. 0x0100

c. 0x1000

d. 0x0000

2. The iAPX88 architecture consists of ____ registers.

a. 12

b. 14

c. 16

d. 18

3. When two 16-bit numbers are added the answer can be 17 bits long, this extra bit that won't fit in the target register is placed in the where it can be used and tested.

a. Carry flag

b. parity flag

c. auxiliary carry

d. zero flag

4. Only instruction allow moving data from memory to memory

a. string

b. word

c. indirect

d. stack

5. Allow changing specific processor behaviors and are used to play with it.

a. Special instructions

b. data movement instructions

c. program control instructions

d. arithmetic and logic instructions

6. 8088 is a 16-bit processor with its accumulator and all registers of ____.
- a. 32 bits
 - b. 6 bits
 - c. 16 bits
 - d. 8 bits
7. In the instruction `cmp ax,bx` the contents of ____ are changed.
- a. Ax
 - b. bx
 - c. cx
 - d. flag register
8. All the addressing mechanisms in iAPX88 return a number called ____ address.
- a. Effective
 - b. Faulty
 - c. indirect
 - d. direct
9. `Mov byte[num1],5` is ____ instruction.
- a. Legal
 - b. illegal
 - c. stack based
 - d. memory indirect
10. The memory address always moves from
- a. processor to memory
 - b. memory to processor

c. memory to peripheral

d. peripheral to processor

11. An offset alone is not complete without

a. segment

b. code label

c. index register

d. data label

12. Code segment is associated to ____ register by default.

a. IP

b. SS

c. BP

d. CX

13. The iAPX88 processor supports ____ modes of memory access.

a. 5

b. 6

c. 7

d. 8

14. A 32-bit processor has accumulator of ____

a. 8 bits

b. 16 bits

c. 32 bits

d. 64 bits

15. After execution of JCXZ instruction CX will changed with flag affect.

a. CF

b. OF

c. DF

d. None

16. Far jump is not position relative but is ____ memory

a. dependent

b. absolute

c. temporary

d. indirect

17. If the address of memory location num1 is 0117 and its content is 0005 then after execution of the instruction `mov bx, num1` bx will contain.

a. 0005

b. 0117

c. num1

d. 1701

18. Assembly the cx register is used normally as a ____ register.

a. Source

b. counter

c. index

d. pointer

19. Which is the unidirectional bus?

a. Control bus

b. data bus

c. address bus

d. none

20. ____ register holds the address of next instruction is to be executed

a. base pointer

b. code segment

c. source index

d. program counter

21. JC and JNC test the ____

a. flag

b. carry

c. parity

d. zero sign

22. Which bit sets the character “blinking” on the screen?

a. 5

b. 6

c. 7

d. 8

23. Mov ax, 5 has:

a. 1 operand

b. 2 operands

c. 3 operands

d. 4 operands

24. Index registers are used to store ____.

- a. data
- b. intermediate result
- c. address
- d. both data and address

25. The bits of the ____ work independently and individually.

- a. Index register
- b. base register
- c. flags register
- d. accumulator

26. The operation of cmp is to:

- a. subtract source from destination
- b. subtract destination from source
- c. add 1 to the destination
- d. add source and destination

27. The registers IP, SP, BP, SI, DI and BX all can contain a ____ offset.

- a. 8 bits
- b. 16 bits
- c. 32 bits
- d. 64 bits

28. Regarding assembler, which statement is true:

- a. assembler converts mnemonics to the corresponding OPCODE
- b. assembler converts OPCODE to the corresponding mnemonics
- c. assembler executes the assembly code all at once

d. assembler executes the assembly code step by step

29. If BB is the OPCODE of the instruction which states to “move a constant value to ax register”, the hexadecimal representation (using little Endian notation) of the instruction mov ax, 336 (150 in hexadecimal number system) will be:

a. 0XBB0150

b. 0X5001BB

c. 0X01BB50

d. 0XBB5001

30. Assembly language is:

a. low level programming language

b. high level programming language

c. also known as machine language

d. not considered closer to the computer

31. There are ____ registers in lapx88 architecture that can hold address of data.

a. 1

b. 2

c. 3

d. 4

32. Which part of this B80500 encoded instruction is an opcode?

a. Opcode is 0500

b. opcode is B80500

c. opcode is B8

d. opcode is 05

33. In ____ operation the carry flag is inserted from the right causing every bit to move one location to its left and the most significant bit occupying the carry flag.

a. Rotate through carry right(RCR)

b. Rotate through carry left(RCL)

c. Rotate left (ROL)

d. Rotate right (ROR)

34. In ____ operation, a carry flag is inserted from the left moving every bit one position to the right, with the right most bit is dropped in the carry flag.

a. RCR

b. ROL

c. RCL

d. ROR

35. CS and IP are both ____ bit registers.

a. 8

b. 4

c. 16

d. 32

36. Motorola follows ____.

a. big endian

b. little endian

c. both

d. None

37. Intel follows ____.

a. Little endian

b. big endian

c. both

d. None

38. The shift logical right operation inserts.

a. A zero from right

b. a zero from left

c. a one from right

d. a one from left

39. Shifting the -15 two-bit SAR:

a. -7

b. 7

c. -8

d. 8

40. In left shift operation the most left bit____

a. will drop

b. will go to CF

c. will come to the right most

d. will be always 1

41. To reserve 8-bits in memory ____ directive is used.

a. db

b. dw

c. dn

d. dd

42. In the mov ax, 5 5 is the ____ operand.

- a. source
- b. destination
- c. memory
- d. register

43. Which flags are not used for mathematical operations?

- a. Carry, interrupt and trap flag
- b. direction interrupt and trap flag
- c. direction overflow and trap flag
- d. direction interrupt and sign flag

44. The number of bits required to access 1MB of memory are

- a. 16 bits
- b. 20 bits
- c. 32 bits
- d. depends on the processor architecture

45. cx register is:

- a. count register
- b. data register
- c. index register
- d. base register

46. Which of the following is not true about registers?

- a. their operation is very much like memory
- b. intermediate results may also be stored in registers
- c. they are also called scratch pad ram

d. none

47. Types of jump are:

a. short, near

b. short, near, far

c. near, far short, far

48. 8088 is a ___ bit processor.

a. 8

b. 16

c. 32

d. 64

49. $|0| \rightarrow |1|1|0|1|0|0|0| \rightarrow |C|$ is an example of:

a. SHL

b. SHR

c. SAR

d. SAL

50. Memory is determined by _____ pair and not alone.

a. Segment-offset

b. segment-code

c. offset-code

d. offset addressing

51. In rotate right operation every bit moves one position to the right and the bit dropped from the right is inserted at the left and:

a. dropped in CF

b. moves to AL

c. don't go anywhere

d. none

52. There are three buses to communicate the processor and memory named as:

a. address, line, data bus

b. address, control, line bus

c. address, control, data bus

d. none

53. The address bus is unidirectional and address always travel from processor to memory.

a. True

b. False

54. Data bus is bidirectional because:

a. to way

b. data moves from both: processor to memory and memory to processor

c. data moves from both: processor to memory and memory to data bus

d. none

55. Control bus:

a. is one way

b. unidirectional

c. bidirectional

d. none

56. A memory cell is an n-bit location to store data, normally ____ also called a byte.

a. 4-bit

b. 8-bit

c. 16-bit

d. 32-bit

57. The number of bits in a cell is called the cell width. ____ define the memory completely.

a. Cell width and number of cells

b. cell number

c. width

d. height

58. For memory we define two dimensions. The first dimension defines how many ____ bits are there in a single memory cell.

a. Parallel

b. vertical

c. long

d. short

59. if ax contains decimal -2 and bx contains decimal 2 then after the execution of the instruction: cmp ax, bx JA label

a. jump will be taken

b. zero flag will set

c. ZF will contain value 4

d. Jump will not be taken

60. If D is 35 is shift to left 2 bits the new value:

a. 35

b. 70

c. 140

d. 17

61. In general, the memory cell cannot be wider than the width of the data bus.

a. True

b. False

62. ____ bus carries the intent of the processor that it wants to read or to write.

a. Control

b. Address

c. Data

d. Both control and data

63. The responsibility of sending the appropriate signals on the control bus to the memory is of the ____.

a. Control Bus

b. Peripherals

c. Processor

d. Memory

64. There are temporary storage places inside the processor called ____.

a. Memories

b. registers

c. peripherals

d. none

65. We can have precisely ____ address on the address bus and consequently precisely ____ element on the data bus.

a. one, one

b. one, two

c. two, one

d. two, two

66. Traditionally all mathematical and logical operations are performed on the ____.

a. Processor

b. register

c. Accumulator

d. None

67. Whenever we need access to a memory location whose address is not known until run-time we need an ____ register.

a. Index

b. Flag

c. accumulator

d. none

68. The instruction cli clears the ____ flag.

a. Interrupt

b. overflow

c. direction

d. carry

69. The instruction sti sets the ____ flag.

a. Carry

b. interrupt

c. parity

d. overflow

70. iAPX88 stands for: “Intel Advanced Processor Extensions 88”

71. iAPX386 is a ____ bit processor.

a. 8

b. 16

c. 32

d. 64

72. First processor 8080 was ____ bit processor.

a. 8

b. 16

c. 32

d. 64

73. The A of AX register stands for: Accumulator

74. The B of BX register stands for: Base

75. The C of CX register stands for: Counter

76. The D of DX register stands for: Destination

77. SI and DI are 16-bit and cannot be used as 8-bit register pairs like ax, bx, cx and dx.

a. True

b. False

78. Which of the following is true about Parity?

a. Parity is the number of “one” bits in a binary number

b. Parity is either odd or even

c. Both a and b

d. none

79. The collection of 4-bits is called.

a. Word

b. nibble

c. byte

d. none

80. During addition or subtraction if a carry goes from one nibble to the next which flag is set?

a. Auxiliary

b. carry

c. trap

d. parity

81. Which flag is set if the last mathematical or logical instruction has produced a zero in its destination.

a. Carry

b. parity

c. direction

d. zero

82. To start a comment ____ is used in assembly.

a. Colon (:)

b. hyphen (-)

c. semicolon (;)

d. asterisk (*)

83. The process through which the segment register can be explicitly specified is known as:

- a. segment addressing
- b. segment override prefix**
- c. segment indexing
- d. offset indexing

84. If BL contains 00000101 then after a single right shift. BL will contain:

- a. 00000011
- b. 00000010**
- c. 10000011
- d. 10000010

85. In assembly language JNZ is used to:

- a. jump if the zero flag is not set**
- b. jump if the zero flag is set
- c. jump if the sign flag is set
- d. jump if the sign flag is not set

86. SP is associated (by default) with:

- a. DS
- b. SS**
- c. ES
- d. CS

87. The stack pointer contains the address of the word that is currently on ____.

- a. Top of the stack**
- b. down of the stack

c. top and down both

d. any position in the stack

88. Which one of the following is an illegal instruction?

a. `Mov ax, bx`

b. `Mov ax, 65`

c. `Mov ax, [bx+bp]`

d. `Mov bx, 10`

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

Question # 1 of 10

`mov [si+300], ax` is an example of Indexed Register Indirect + Offset

True

False

Question # 2 of 10 (Start time: 06:51:43 PM)

we can not Subrtace index register from the base register(bx-si)in assemblby language

true

false

In direct addressing the memory address given in the instruction is



Fixed

correct

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

Variable

Register

Empty

The first 16-bit processor produced by “Intel” was 8085

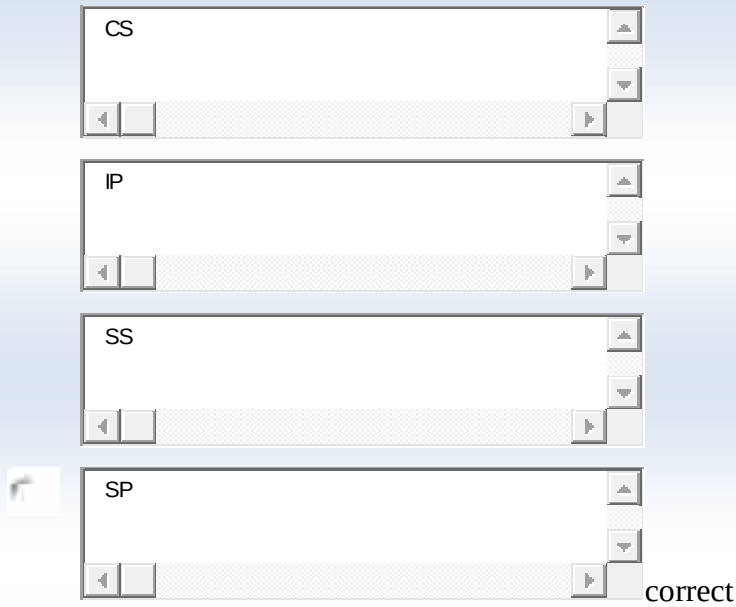
True

False

correct

BP by default associated with

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language



CS

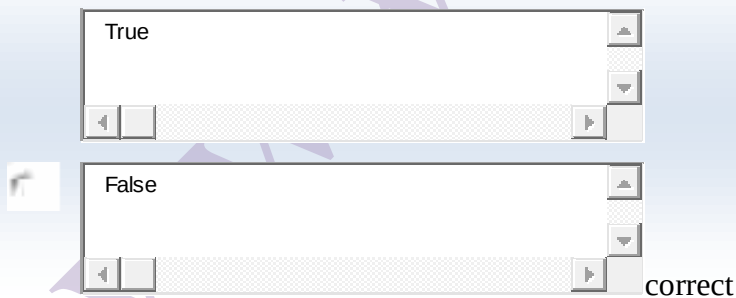
IP

SS

SP

correct

Register to memory operation is not allowed



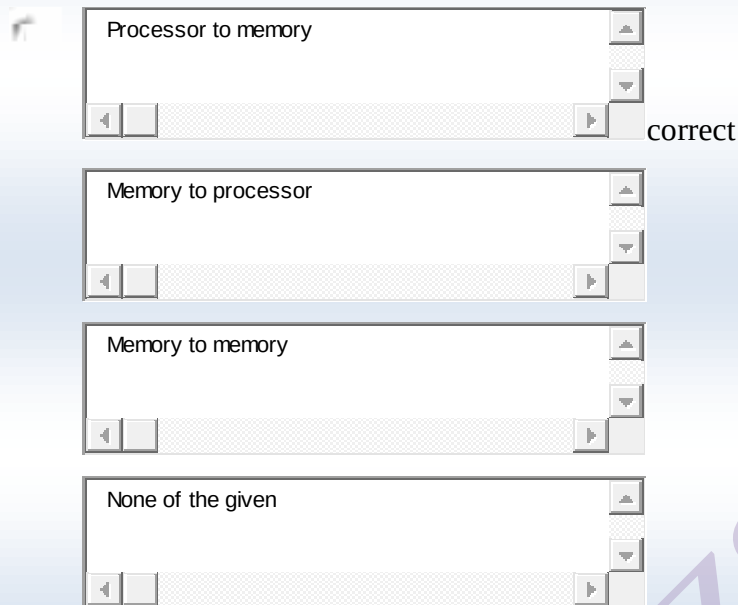
True

False

correct

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

Address is always go from



Processor to memory

Memory to processor

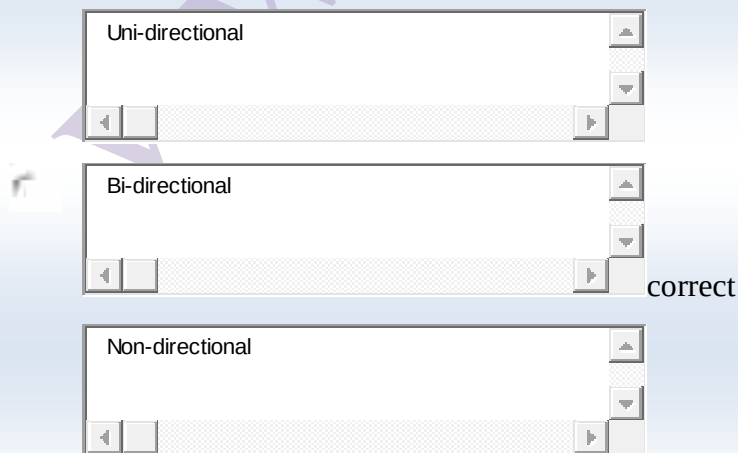
Memory to memory

None of the given

correct

The image shows a screenshot of a quiz interface. It contains four radio button options: 'Processor to memory', 'Memory to processor', 'Memory to memory', and 'None of the given'. The 'Memory to processor' option is selected, and the word 'correct' is displayed to the right of the options.

Data bus is



Uni-directional

Bi-directional

Non-directional

correct

The image shows a screenshot of a quiz interface. It contains three radio button options: 'Uni-directional', 'Bi-directional', and 'Non-directional'. The 'Bi-directional' option is selected, and the word 'correct' is displayed to the right of the options.

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

None of the given

www.vuzs.net

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

Quiz - Mozilla Firefox 4.0 Beta B

http://quiz.vu.edu.pk/QuizQuestion.aspx?ver=c5fde013-a38a-41f3-82f8-e01a60e90622

MC10040461 | Sajid Mahmood

Time Left 40

Quiz Start Time: 02:04 PM

Question # 2 of 10 [Start time: 02:08:01 PM]

Total Marks:

Architecture consists of

Select correct option:

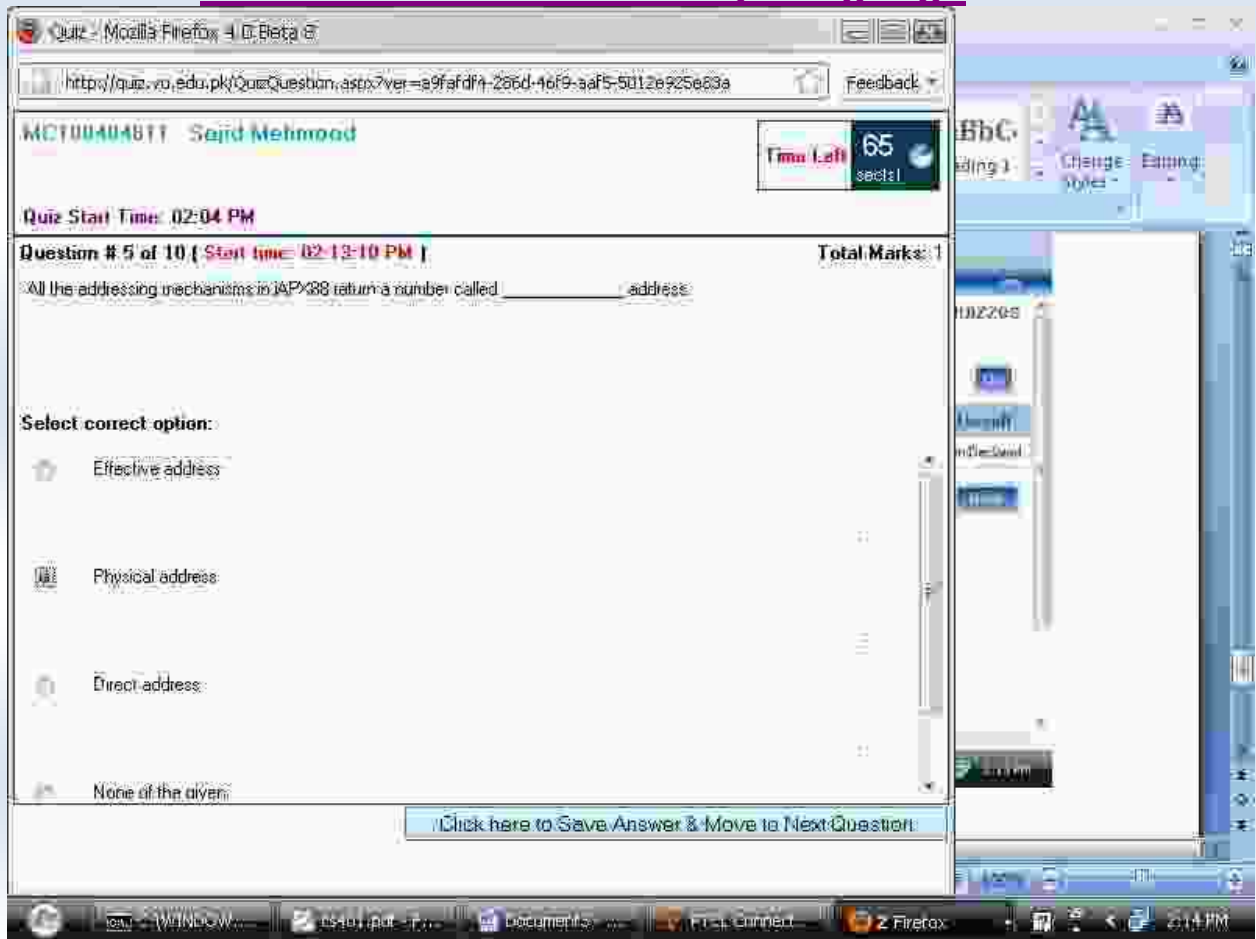
☐ 14 Registers

☐ 16 Registers

☐ 32 Registers

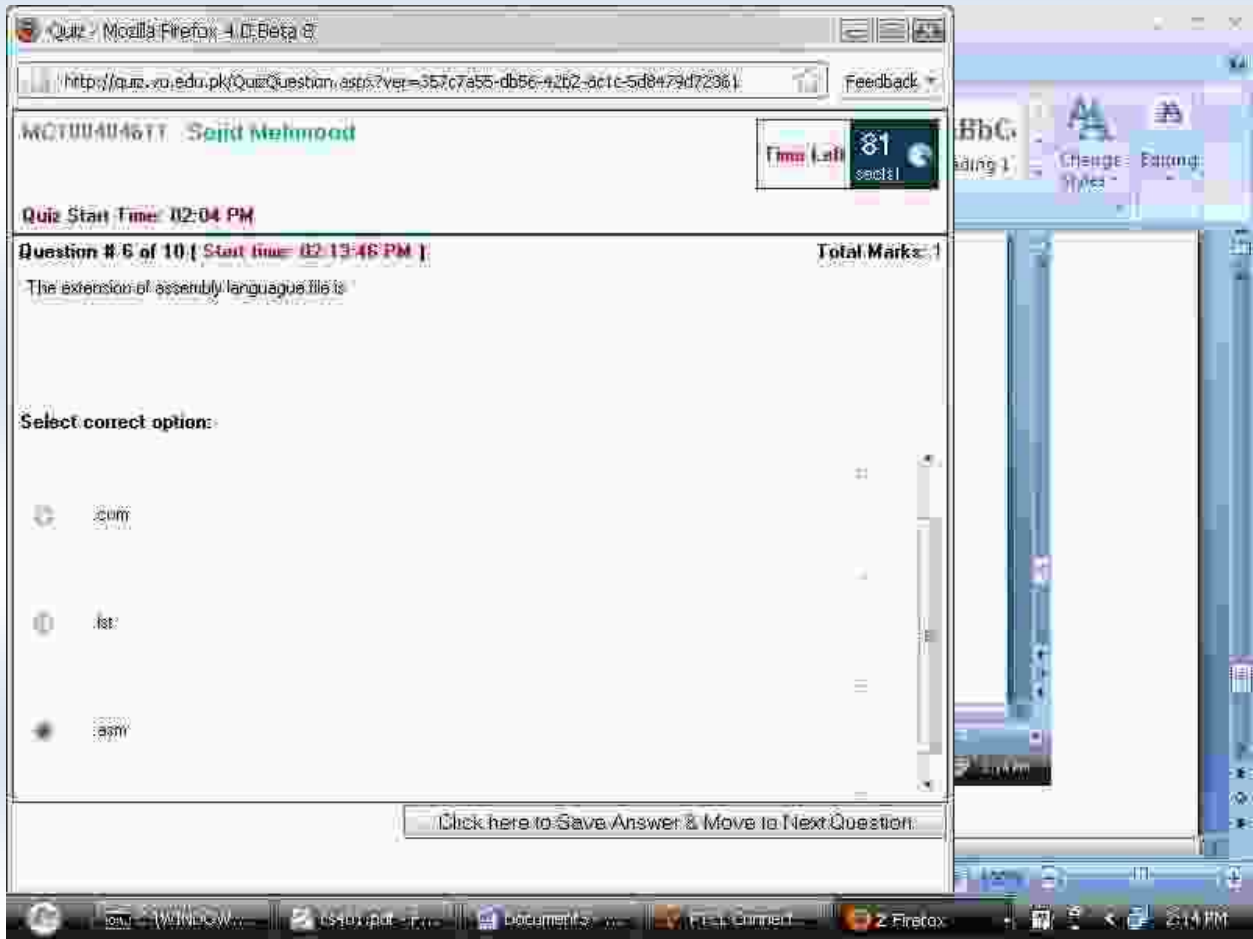
Click here to Save Answer & Move to Next Question

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

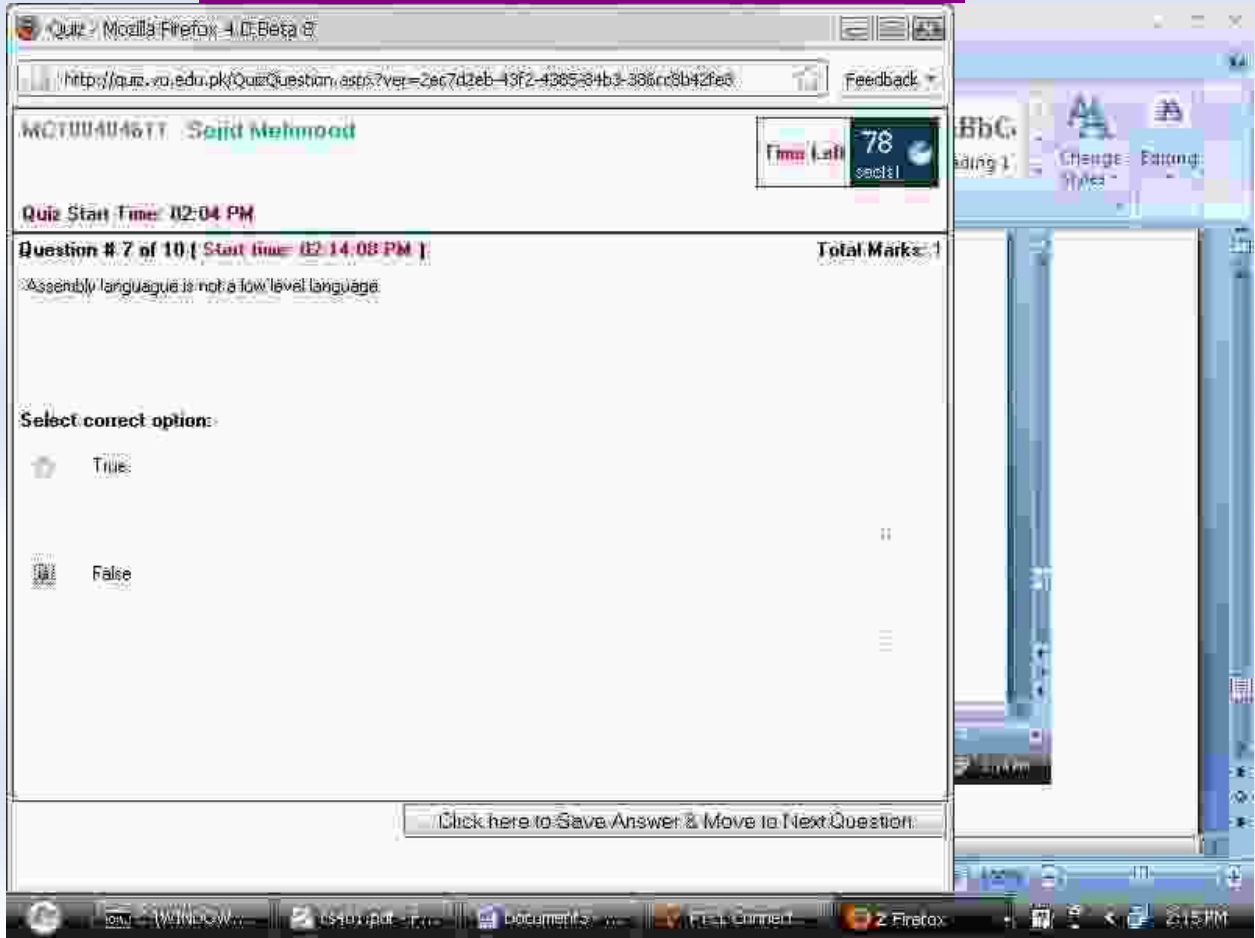


CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

1



CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language



CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

Quiz - Mozilla Firefox 4.0 Beta B

http://quiz.vu.edu.pk/QuizQuestion.aspx?ver=F711d6bd-ff87-478e-8638-3f585c9e3a83

MC100401461 | Sajid Mahmood

Time Left: 46

Quiz Start Time: 02:04 PM

Question # 9 of 10 | Start time: 02:15:37 PM | Total Marks:

Select correct option:

☐ Big endian

☐ Little endian

☐ Both of them

☐ None of the given

Click here to Save Answer & Move to Next Question

taskbar: WINDOW... CS401.pdf - P... documents... FTL Connect Z Firefox 2:17 PM

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

Quiz - Mozilla Firefox 4.0 Beta B

http://quiz.vu.edu.pk/QuizQuestion.aspx?ver=c81b6035-9182-48c9-94d6-94a1410b8c9a

MC100401461 | Sajid Mahmood

Time Left: 66

Quiz Start Time: 02:04 PM

Question # 10 of 10 | Start time: 02:16:23 PM | Total Marks: 1

Registers are also called:

Select correct option:

☒ True

☐ False

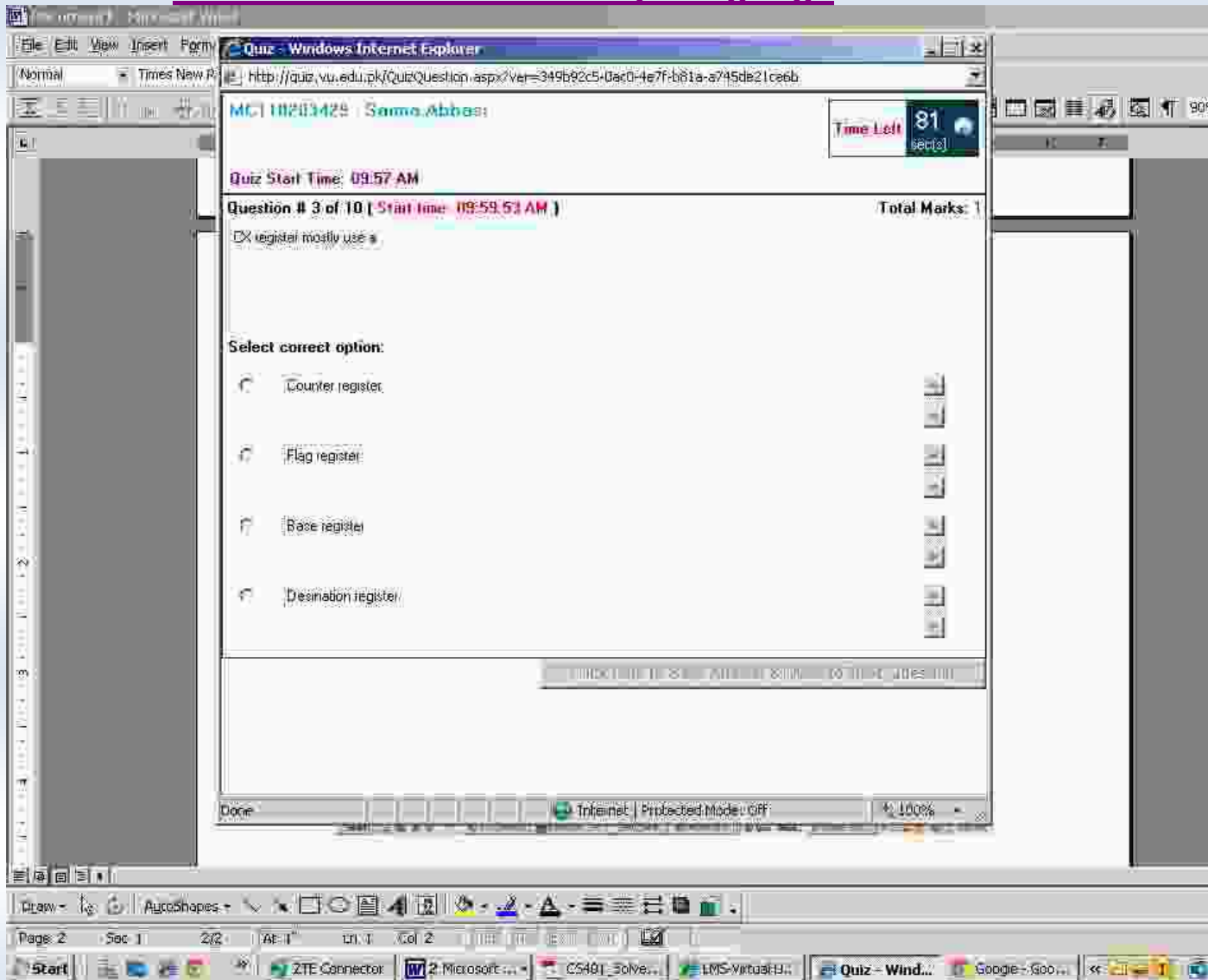
Click here to Save Answer & Move to Next Question

www.vuzs.net

The screenshot shows a Windows desktop environment. On the left, there is a sidebar with icons for 'Computer', 'Home', 'My Recent Places', 'IT Teachers', 'Recycle Bin', 'Desktop Icons', 'Network Magic Folders', 'Adobe Reader', 'Internet Explorer', 'Sheet Messages', 'PowerPoint', 'Downloads', 'Google Chrome', and 'Before Windows 7'. The main area displays a quiz window titled 'Quiz - Windows Internet Explorer'. The address bar shows 'http://quiz.vu.edu.pk/QuizQuestion.aspx?Ver=543656ee-57de-4672-b638-1c2c98b871d1'. The quiz content includes a question number 'MC118203425', a 'Same Address' label, a 'Time Left' of '89 sec(s)', and a 'Quiz Start Time' of '09:57 AM'. The question text is 'Question # 1 of 10 (Start time: 09:57:25 AM)' and 'Group of bus processor uses to inform memory which element to read/write is collectively known as'. Below the question, there are four radio button options: 'Control bus', 'Data bus', 'Address bus', and 'RAM'. The 'Data bus' option is selected. The bottom of the window shows a taskbar with icons for 'Done', 'Internet', 'Protected Mode', and 'CFF', along with a '100%' zoom level.

4

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language



1

CS401 Solved MCQs from Online Quizzes- Computer Architecture and Assembly Language

memory cell can not be wider

Quiz - Windows Internet Explorer

http://quiz.vu.edu.pk/QuizQuestion.aspx?Ver=d6c52674-1808-4639-bc1c-0ad60e9444c

MCQ 11203429 Samee Abbas

Time Left 64 (secs)

Quiz Start Time: 09:57 AM

Question # 8 of 10 (Start time: 10:05:46 AM) Total Marks: 1

Memory cell can not be wider than the data bus.

Select correct option:

☐ True

☐ False

Done

Internet Protected Mode: OFF

100%

Basic Registers in Computer processor

www.writeurl.com x Programming Languages (kb)

All data must be represented in a register before it can be processed. For example ... In general the **memory cell cannot be wider than** the width of the **data bus** ...

Start

ZTE Connector

Microsoft Word

CS401_Solve...

LMS-VirtualH...

Quiz - Wind...

memory cell c...

9:07 PM

1

\

DOWNLOAD MIDTERM

PAST PAPERS BY WAQAR SIDDHU

More in PDF From

VU Answer

Get All Solutions.

Why BP (Base Pointer) default segment is SS (Stack Segment)?

Answer ([Please click here to Add Answer](#))



“Interrupts are asynchronous” What does it mean?

Answer ([Please click here to Add Answer](#))



What are the results after performing the following instructions, where ax = 00110011 and bx = 00010001

1. and ax,bx
2. or ax,bx
3. xor ax,bx

Answer ([Please click here to Add Answer](#))





What is the working of Screen Scrolling program and write the name of the most important instruction used in that program?

Answer ([Please click here to Add Answer](#))



Write an assembly language program that display Zero (0) on the whole DOS screen.

Answer ([Please click here to Add Answer](#))



Write an assembly language program that clears the computer screen.

Answer ([Please click here to Add Answer](#))



How 16-bit DIV operation is different from 32-bit DIV operation?

Answer ([Please click here to Add Answer](#))



Which of the following are bitwise operators?

AND
MOV
ADD
MUL

Answer ([Please click here to Add Answer](#))





How the value of Instruction Pointer register is saved and updated when a CALL instruction is executed?

Answer ([Please click here to Add Answer](#))



For what purpose "INT 1" is reserved ?

Answer ([Please click here to Add Answer](#))



Suppose AL contains 11001011b and CF= 1. Give the new contents of AL after each of the following instructions is executed. Assume the preceding initial conditions for each part of this question.

- SHL AL,1
- a. SHR AL,1
- b. ROL AL,CL ;if CL contains 2
- c. SAR AL,CL ;if CL contains 2
- d. RCR AL,CL ;if CL contains 3

Answer ([Please click here to Add Answer](#))



Following instructions are taken from the program of printing a "hello word" at desired screen location. What functionality is achieved by the following code lines?

[Initially SI is pointing to the string and DI is pointing to the top left column of the screen]

```
mov al, [si]
mov [es:di], ax
add di, 2
```

Answer ([Please click here to Add Answer](#))



[Initially SI is pointing to the string and DI is pointing to the top left column of the screen]

```
mov al, [si]
mov [es:di], ax
add di, 2
add si, 1
```

Answer ([Please click here to Add Answer](#))



OR
NOT
SUB
XOR
DIV
ADC

Answer ([Please click here to Add Answer](#))



What does the instruction "lds si, [bp+4]" do?

Answer (Please [click here](#) to Add Answer)



point ds:si to the first string

Write down the procedure to clear the selective bit.

Answer (Please [click here](#) to Add Answer)



which bit we want convert into zero we make and opration on it whic
value we want to covert into zro we apply and opration with value 0 t
other valuse fil with one.

Made By: Waqar Siddhu

How string instruction work on a block of data?

Answer ([Please click here to Add Answer](#))



What is wrong with this program?

```
; bubble sort subroutine taking parameters from stack  
[org 0x0100]
```

Answer ([Please click here to Add Answer](#))





In case of branching techniques used in Intel's architecture, which instruction is used to provide temporary diversion and how?

Answer ([Please click here to Add Answer](#))





What will be the affect of executing REPE and REPNE instructions on zero flag? With which instructions you can use both of these REPE and REPNE prefixes?

Answer ([Please click here to Add Answer](#))



Why BP (Base Pointer) default segment is SS (Stack Segment)?

Answer ([Please click here to Add Answer](#))



Write a single instruction that can be used as an alternative of the following two instructions?

```
DEC CX  
JNZ NextChar
```

Answer ([Please click here to Add Answer](#))



What are the instructions use by the Assembly language for Permanent diversion and temporarily diversion?

Answer ([Please click here to Add Answer](#))



Which interrupts are generated as a result of

1. Dividing a number by zero
2. If the trap flag is set

Answer ([Please click here to Add Answer](#))



What is Divide Overflow Error? Complete the following code such that it will cause Divide Overflow Error.

```
mov BL, ____  
mov AX, ____  
DIV BL
```

Answer ([Please click here to Add Answer](#))



Write an assembly language program that clears the computer screen.

Answer ([Please click here to Add Answer](#))

