207 Past Question 1 Solved

1. Explain 3 categories of statement in Assembly Language
    1. ASSEMBLY DIRECTIVE: tell the assembler about the various aspects of the assembly process
    2. EXECUTABLE STATEMENT: tell the processor what to do. Each instruction consists of an operation code (opcode). Each executable instruction generates one machine language instruction
    3. MACROS: ACT AS TEXT SUBSTITUTION MECHANISM, THE CONSIST OF BOTH ASSEMBLY DIRECTIVES AND EXECUTABLE STATEMENTS****
2. Write down Assembly language instruction format and explain each part with appropriate examples
   1. [LABEL] [MNEMONIC] [OPERAND] [COMMENT]
   1. **LABEL: IT IDENTIFIES A LINE OF CODE, SO IT CAN BE REFERED TO IN A MACRO**
   2. **MNEMONIC: ALSO CALLED OPECODE, IT IS THE ONLY COMPULSORY PART OF AN ASSEMBLY LANGUAGE INSTRUCTION, IT TELLS THE MACHINE WHAT OPERATION TO PERFORM WHETHER ADDITION, SUBTRACTION AND SO ON.**
   3. **OPERAND: IT INFORMS THE COMPUTER WHERE TO GET THE DATA IT WANTS TO MANIPULATE AND WHERE TO STORE THE FINAL DATA**
   4. **COMMENT: IT IS USED TO INFORM THE PROGRAMMER OR OTHER USERS WHAT THE PARTICULAR LINE OF CODE IS DOING, IT IS DENOTED BY A SEMICOLON ;**
3. Difference between machine language and assembly language
   Assembly language and machine code are both low-level programming languages Assembly language is a symbolic representation of machine code instructions. It uses mnemonics to represent the underlying machine code instructions, making it easier for humans to read and write. Machine code, on the other hand, is the actual binary code that is executed by the computer's processor. It is not easily readable by humans and is usually generated by an assembler program from assembly language source code. In summary, assembly language is a human-readable representation of machine code, while machine code is the actual code executed by the processor.
4. Explain how processor perform its functions or execute instructions
   The process through which the processor controls the execution of instructions is referred as the fetch-decode-execute cycle, or the execution cycle. It consists of three continuous steps:
    - Fetching the instruction from memory
    - Decoding or identifying the instruction
    - Executing the instruction
5. Explain briefly logic components of a single chip microprocessor
   1. **Arithmetic Logic Unit (ALU):** performs mathematical operations such as addition, subtraction, and logical operations such as AND, OR, NOT.
   2. **Control Unit:** responsible for fetching instructions from memory and decoding them to execute the correct operation. It controls the flow of data within the microprocessor.
   3. **Register**: small, fast memory units that store data used by the ALU and control unit

4. **Bus**: a set of connections that allow the microprocessor to communicate with memory and other peripheral devices.
5. **Memory**: holds the instructions and data used by the microprocessor.
6. **Input/Output (I/O) ports:** allow the microprocessor to communicate with external devices such as keyboard, mouse, and monitor.
7. **Clock**: a timing signal that synchronizes the operation of the microprocessor.

6. Give 3 examples of 16bits, 32bits and 64bits registers
    1. **16 BIT: AX, BX, CX
    2. **32 BIT: EAX, EBX, ECX**
    3. **64 BIT; RAX, RBX, RCX****
7. Explain 3 modes of execution in an high level programming language
   In high-level programming languages, there are typically two modes of execution:
    1. **Interpreter mode**: In this mode, the program is executed line by line, with each line of code being interpreted and executed as it is read. This mode is typically slower than the compiler mode, but it allows for more flexibility and easier debugging, as errors can be caught and corrected as the program is executed.
    2. **Compiler mode:** In this mode, the program is first translated into machine code (or object code) by a compiler. This machine code can then be executed directly by the computer's processor. This mode is typically faster than interpreter mode, as the program is executed directly by the processor, but it can be less flexible and debugging can be more difficult, as errors may not be caught until the program is executed.
8. What is the difference between processor and microprocessor CHIP
   A microprocessor chip and a processor are related but different concepts. A microprocessor chip is a physical piece of silicon that contains the transistors and other components that make up a microprocessor. It is a chip that can be inserted into a motherboard or other device to provide processing capabilities.

   A processor, on the other hand, refers to the entire package of components that perform the processing functions. It includes the microprocessor chip, as well as other components such as cache memory, buses, and support circuits. A processor is the central processing unit (CPU) of a computer, it is the "brain" that performs the instructions of a computer program.

   In summary, a microprocessor chip is a physical piece of silicon that contains the transistors and other components that make up a microprocessor, while the processor is the entire package of components that perform the processing functions, including the microprocessor chip, the memory, buses, and support circuits.
9. What are constants. Give 2 examples of integer, and character constants
   CHARACTERS IN OUR PROGRAM WHICH RETAINS ITS SAME VALUE THROUGHOUT THE PROGRAM. INTEGER:2, 3, CHARACTER A, B**
10. Write an assembly language program to find the sum of all numbers between 1 & 10 starting from the rear and divide by 5. Leave the answer in Ax

```
.model small
.stack 100h
.code
```

```
 mov ax 10
 mov bx, 0

        myloop:
                add bx, ax
                dec ax
                cmp ax, 0
                jg myloop

mov ax, bx
div ax,  5

end
```

```

```

11. What are the errors in the following and find the appropriate correction
    1. **mov ax, 3d:** 3d is not recognized as a valid digit by most assemblers. it it more standard to say mov ax, 3
    2. **mov cx, ch:** cx is a register and ch is the high-order byte of the cx register. TO access the value of ch as a seperate entity, you need to use a different register such as dx.
    3. **mov eax, 1h:** It is not necessarily wrong, but the standard way of representing a hexidecimal constant in x86 assembly is to prefix it with 0x. A more standard instruction would be mov eax, 0x1
12. Explain each line of code:

```
.686 !The directive to specify the processor type.
.model flat, stdcall ! Specifies the memory model and callilng convention to be used.
This memory model is flat.
.stack 4096 ! Sets the size of the stack to 4096 bytes
ExitProcess PROTO, dwExitCode: DWORD !This line declares a function protype for the
'ExitProcess' function. It takes a single argument, 'dwExitCode', which is a 32-bit
DWORD
.code
main PROC !Starts the definition of the main function, labeled 'main'
        mov eax, 10000h !moves the hexadecimal value, 0x10000 into the EAX
        add eax, 40000h !adds the hexadecimal value 0x400000 to the value stored in
the EAX
        sub eax, 20000h ! subtracts the hexadecimal value 020000 from the value stored
in the EAX
        push 0 ! Pushed the value 0 onto the  stack
        call Exit !calls the 'ExitProcess' function, passing the value '0' as the
argument
main ENDP !ends the definition of the main function
END main !Indicates the end of the program and
```

13. Difference between variables and identifiers
    An identifier is a name given to a variable, function, or any other item in a computer program.

A variable, is a container that holds a value, it can be assigned different values during the execution.

14. Instruction to evaluate 5+(6-2) leaving result in ax

```
.code
mov bx, 6
sub bx, 2
mov ax, 5
add ax, bx
```

15. Explain the following
    1. Address Bus
        1. It is used to identify where information is being sent or fetched. It identifies the source and destination of data on the data bus. If the cpu wants to read a byte of data from the memory, it puts the address of the desired data on the address lines.
    2. Data Bus
        1. Carries the information being transmitted 1 bit at a time. The number of lines in a data bus is referred to as the width of the data bus.
    3. Control Bus
       Describes the manner in which information is sent. Control access to data on the data and address buses. Control signals transmits both command and timing information. Command is the type of operation that should be performed on the data. While timing indicate the validity of data and address information.
16. Four Basic microprocessor registers
    Registers are grouped into three broad categories
    - General Registers
        - Data Registers: EAX, EBX, ECX, AX, BX, CX, AL, AH
        - Pointer Registers: EIP, ESP, EBP, IP, SP, Bp
        - Index Register: ESI, EDI
    - Control Registers
    - Segment Registers
17. Explain how processor perform its functions or execute instructions with appropriate registers mentioned
    The processor performs its functions through a series of steps:
    1. **Fetching:** The processor fetches an instruction from memory and stores it in the instruction register (IR).
    2. **Decoding:** The processor decodes the instruction stored in the IR to determine what operation it represents and what data it operates on.
    3. **Execution:** The processor performs the operation specified by the instruction, using data from the data register (DR) or memory. The result of the operation is stored in the DR.
    4. Repeat: The processor repeats the fetch-decode-execute-store cycle until it reaches a HALT instruction, which signals the end of a program.
18. When a computer is said to be a 32bit or 64bit system what does it mean
    The terms "32-bit" and "64-bit" refer to the way a computer's processor (also called CPU),

handles information. The number refers to the size of the data words that the processor can handle, with a 32-bit processor being able to handle data words of up to 32 bits, and a 64-bit processor being able to handle data words of up to 64 bits.
In general, a 64-bit system can handle more memory, larger data sets, and is generally faster than a 32-bit system. This is because a 64-bit processor can process more information at once, compared to 32-bit systems.