Indian Institute of Technology, Guwahati

CS 348 Implementation of Programming Languages Lab

Quiz 1

Duration: 55 minutes, Date: 01-02-2023

Answer all questions

1. Let us assume that an assembly language program (p1.asm) has a global _start defined in the .text section. Similarly, another assembly language program (p2.asm) has a global _start defined in its own .text section. Let the object files (say p1.o) created using assembler for p1.asm. Assume p2.asm is calling some module in P1.asm. Can executable file be generated by assembler for p2.asm? Justify your answer.

2. Is it possible for an Assembly Language program meant for an x86 processor to run on an arm architecture processor of the same bit size? Explain your thought process enumerating the steps involved.

3. What are the differences between EXTERN and GLOBAL directives. How are they inter-related to each other? Give an example where you use these directives to effectively show their relationship.

4. Explain the differences between internal and external interrupts with examples. How would you write an Interrupt Sub Routine such that the control returns to the program that initiates the interrupt versus when the interrupt results in an exit?

5. You have a 32-bit processor. Will this processor be able to run a 16-bit assembly code? Will this be able to run a 64-bit assembly code. Give your reasons with explanation.

6. Consider the assembly language code snippet:

   Section .data

   Msg db "Test %I", 0x0a, 0x00

   Explain each part of the code snippet, including the hex.

7. Consider the following 32-bit assembly language code, where ESP = 36. The following instructions are then carried out.

   Push 14

   Push 42

   pop

   Push EIP

   Push 51

   Pop eax

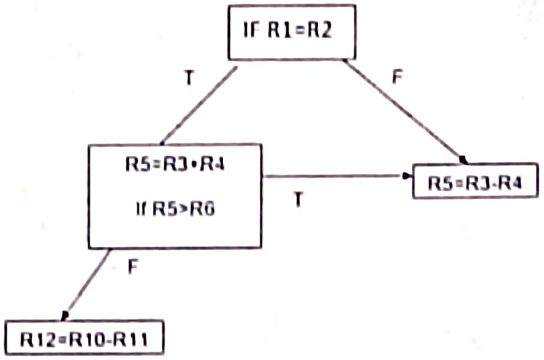
   Pop ebx

   Mul 14

   Mov EIP, ebx

   What are the values in eax, ebx, esp? Also, which instruction does EIP point to after the mov operation?

8. Convert the following Flow Chart to 32-bit assembly language code snippet:



9. What instructions can be used to modify the value of Stack Pointer (ESP/RSP) excluding push, pop, call and ret. Show with examples.

10. Write the x86 assembly code to the following C code. Assume that the base address of the array "arr" is 0x100 (hexadecimal).

```
int arr[4] = {1,2,3,4};
for(int i=0;i<4;i++)
{
        arr[i]--;
}
```

11. Write down the major differences between near jump and short jump in assembly language.

12. What is the effect of the following instruction?

mov ecx, [esi + eax]

Best wishes

QUIZ 1 Solution

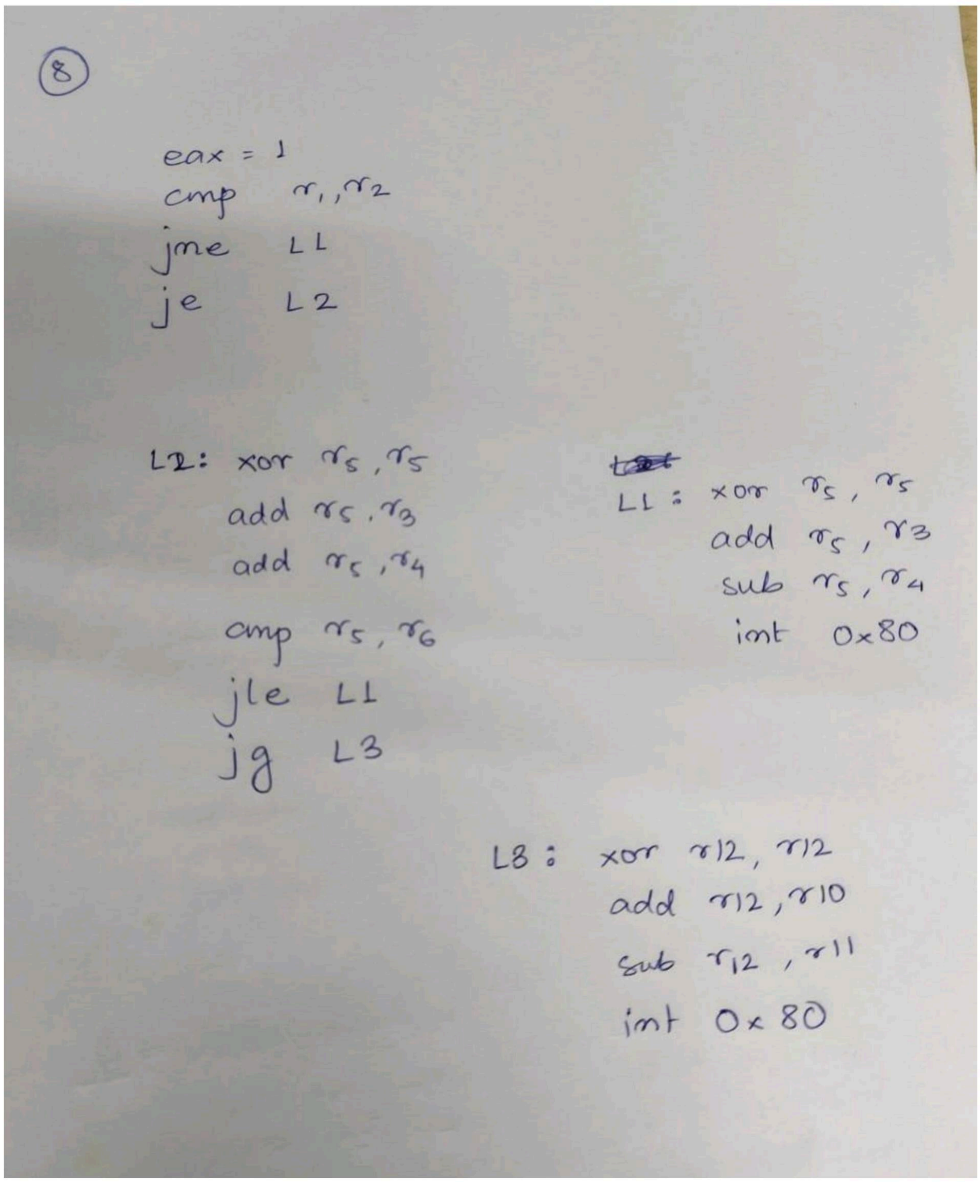
1. Executable files cannot be generated as two global _start is illegal. The processor won't know which start to begin with.
2. Yes, it is possible. To ensure that, an intermediate code must be generated which is machine independent. This intermediate code can then be converted to machine dependent object code.
3. Extern is used to declare a symbol which is not defined anywhere in the module being assembled but is assumed to be defined in some other module and needs to be referred to by this one. GLOBAL is the other end of EXTERN: if one module declares a symbol as EXTERN and refers to it, then in order to prevent linker errors, some other module must define the symbol and declare it as GLOBAL. Some assemblers use the name PUBLIC for this purpose.

Example: A global variable can be imported as extern into a separate program, by importing the original library that contains the global variable.

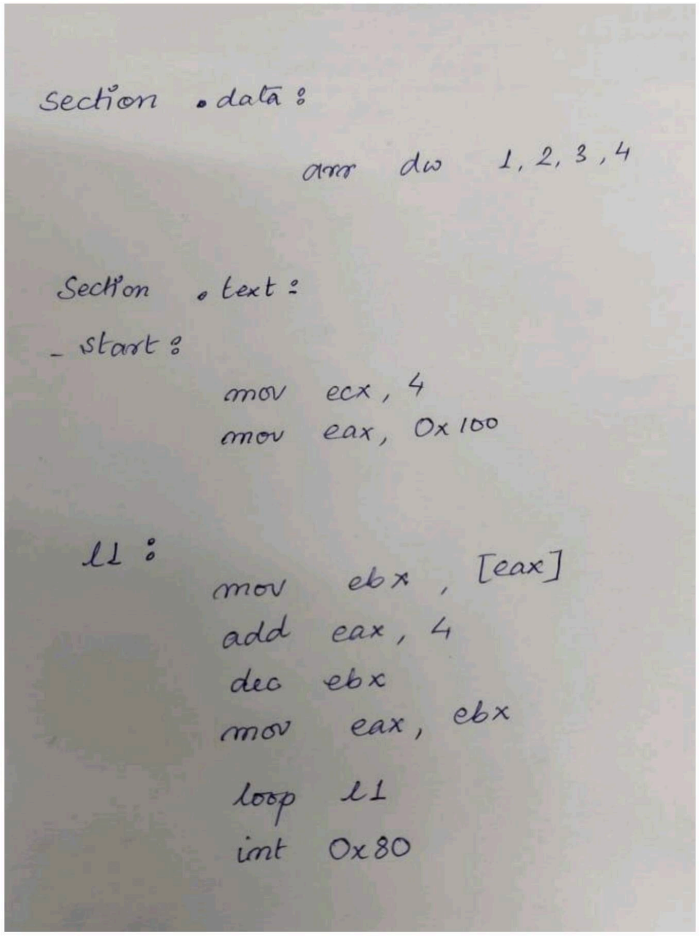
4. An interrupt caused due to internal instructions that are embedded into the execution instruction of a program is called an internal interrupt. An external interrupt is caused when the program running in the CPU is interrupted as a result of external interference from the user, peripherals, hardware or network.

Int 0x80 with eax=1 will exit, while eax=4 with corresponding values for ebx, ecx and edx will result in a return to the program that initiated it.

5. Yes, the 32-bit processor can successfully run a 16-bit assembly code. Modifications will have to be made. However, it will not be able to run a 64-bit Assembly code directly. The assembly code that runs on an assembler specifically designed to be processor independent can however be run on any processor, irrespective of its bit size.
6. Msg = variable name, db = data Byte, “Test %I" = actual string with an integer that can be printed, 0x0a = 10 (used as new line \n), 0x00 = indicates end of line.
7. Eax = 714, ebx = eip, esp = 32, eip points to instruction push eip.
8.



9. You can use ‘mov’, ‘add’, ‘sub’, ‘or’, ‘and’ to modify stack pointer.
10.



11. Near jump—A jump to an instruction within the current code segment (the segment currently pointed to by the CS register), sometimes referred to as an intra-segment jump.

    Short jump—A near jump where the jump range is limited to –128 to +127 from the current EIP value.

    Far jump—A jump to an instruction located in a different segment than the current code segment but at the same privilege level, sometimes referred to as an intersegment jump.

    Task switch—A jump to an instruction located in a different task.

12. Move the (2/4/8 bytes) of data from address (esi+eax) to ecx.