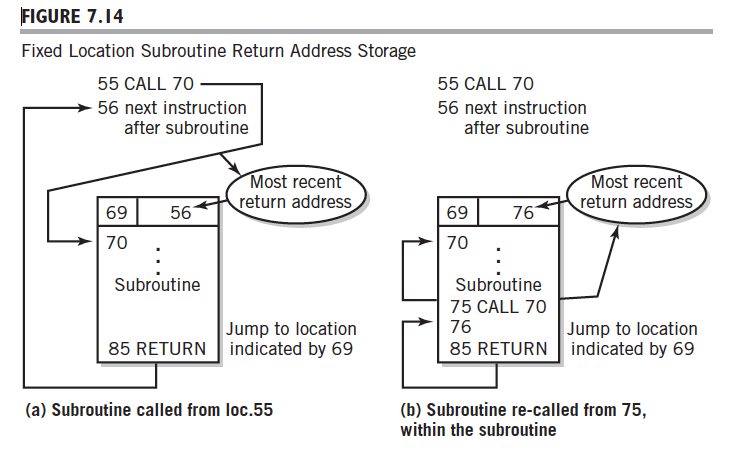
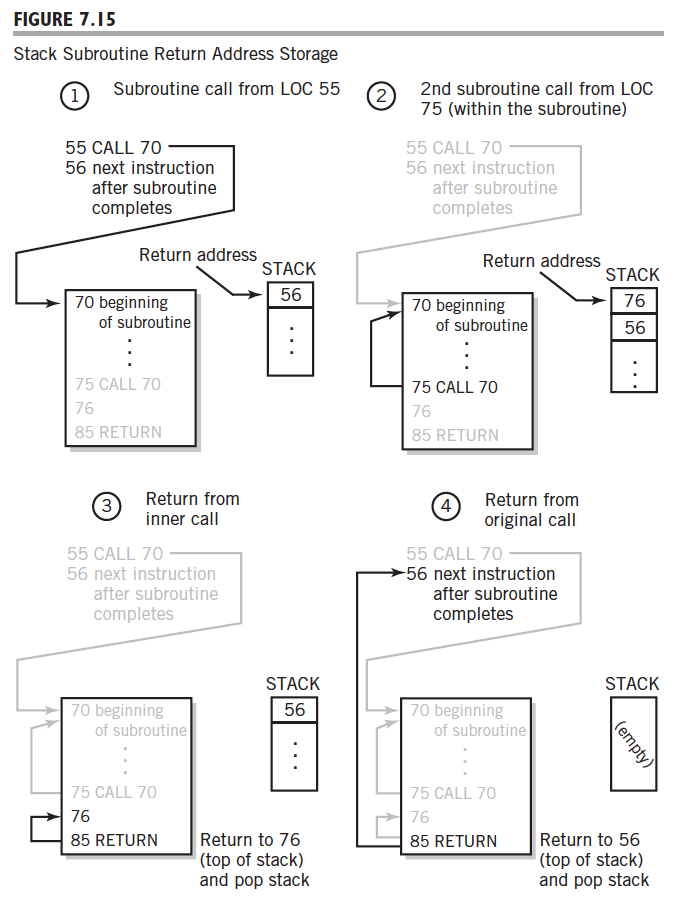
**2. Please explain how the stack is used for subroutine call and return (Chapter 7 “Stack Instructions” (Maximum 1 point)**

Program routines that are recursive must “call themselves”. Suppose the return address were stored in a fixed location. If the routine is called a second time, from within itself, the original returning address is lost and replaced by the new return address. The return address is stored on a stack. This time when the routine is again called, the original address is simply pushed down the stack, below the most recent address. We always return from the last called subroutine to the one just previous.





**3.** **Explain a computer's register-level architecture, including: (Maximum 1 point)**

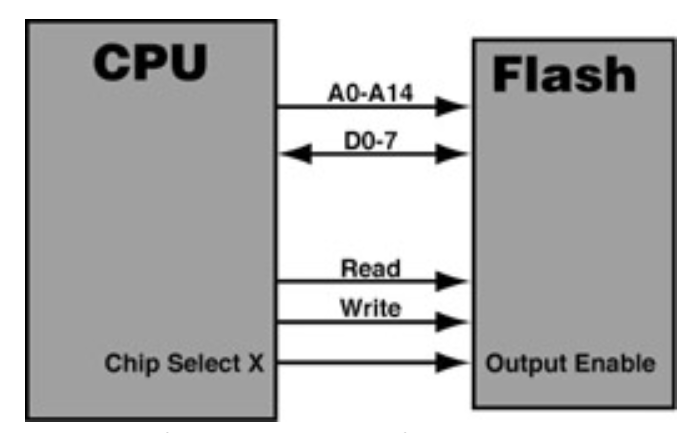
**a) CPU-memory interface**

CPU-memory interface includes the data bus, address bus and some control signals including Read, Write and Memory-Function-Complete (MFC). The CPU is interfaced to the data bus and address bus through the MDR and MAR registers.

The CPU uses the read and write signals to control the output drivers on the various memory and peripheral devices, and thus, controls the direction of the data bus.

The CPU-to-flash device interaction can be summarized with the following steps.

1. CPU places the desired address on the address pins.
2. CPU brings the read line active.
3. CPU brings the appropriate chip select line active.
4. The flash device places the data located at the specified address onto the data bus.
5. CPU reads in the data that has been placed on the bus by the flash device.
6. CPU releases the chip select line and processes the data.



**b) special-use registers** (**minimum 100 words**)

**Instruction Register (IR):** The instruction register holds the actual instruction being executed currently by the computer. In the Little Man Computer, this register was not used; the Little Man himself remembered the instruction he was executing. In a sense, his brain served the function of the instruction register.

**Memory Data Register (MDR):** The memory data register, sometimes known as the memory buffer register, will hold a data value that is being stored to or retrieved from the memory location currently addressed by the memory address register.

**Memory Address Register (MAR):** The memory address register holds the address of a memory location.

**Program Counter (PC):** The program counter holds the location of the next instruction to be fetched from memory. It is automatically incremented between supplying the address of the next instruction and the instruction being executed.

**Accumulator:** The accumulator is an internal CPU register used as the default location to store any calculations performed by the arithmetic and logic unit.

**c) addressing modes** (**minimum 100 words**)

The addressing modes refers to the way in which the operand of an instruction is specified. Information contained in the instruction code is the value of the operand or the address of the result/operand. Following are the main addressing modes that are used on various platforms and architectures.

1. Immediate Mode

The operand is an immediate value is stored explicitly in the instruction.

1. Index Mode

The address of the operand is obtained by adding to the contents of the general register (called index register) a constant value. The number of the index register, and the constant value are included in the instruction code. Index Mode is used to access an array whose elements are in successive memory locations. The content of the instruction code represents the starting address of the array and the value of the index register, and the index value of the current element. By incrementing or decrementing index register different element of the array can be accessed.

1. Indirect Mode

The effective address of the operand is the contents of a register or main memory location, location whose address appears in the instruction. Indirection is noted by placing the name of the register or the memory address given in the instruction in parentheses. The register or memory location that contains the address of the operand is a pointer. When an execution takes place in such mode, instruction may be told to go to a specific address. Once it's there, instead of finding an operand, it finds an address where the operand is located.

1. Absolute (Direct) Mode

The address of the operand is embedded in the instruction code.

1. Register Mode

The name (the number) of the CPU register is embedded in the instruction. The register contains the value of the operand. The number of bits used to specify the register depends on the total number of registers from the processor set.

1. Displacement Mode

Similar to index mode, except instead of a index register a base register will be used. Base register contains a pointer to a memory location. An integer (constant) is also referred to as a displacement. The address of the operand is obtained by adding the contents of the base register plus the constant. The difference between index mode and displacement mode is in the number of bits used to represent the constant. When the constant is represented a number of bits to access the memory, then we have index mode. Index mode is more appropriate for array accessing; displacement mode is more appropriate for structure (records) accessing.

1. Autoincrement /Autodecrement Mode

A special case of indirect register mode. The register whose number is included in the instruction code, contains the address of the operand. Autoincrement Mode = after operand addressing, the contents of the register are incremented. Decrement Mode = before operand addressing, the contents of the register are decrement.