**End of Ch # 7 Exercises: Solutions**

**7.2, page# 231;**

Suppose that the following instructions are found at the given locations in memory:

1. Show the contents of the IR, the PC, the MAR, the MDR, and A at the conclusion of instruction 20.
2. Show the contents of each register as each step of the fetch-execute cycle is performed for instruction 21.

**Solution:**

**a.** Working from the F/E cycle for instruction 20,

PC  MAR

MDR  IR  final value of IR = 150

IR [address]  MAR  final value of MAR = 50

MDR  A  final value of MDR and A = 724

PC + 1 PC  final value of PC = 21

**b.**  PC MAR MDR IR A

PC  MAR 21 21 724 150 724

MDR  IR 21 21 351 351 724

IR [address]  MAR 21 51 351 351 724

A + MDR  A 21 51 006 351 730

PC + 1  PC 21 51 006 351 730

**7.4, page# 232;**

What are there two different registers (MAR and MDR) associated with memory? How much memory can this computer address?

**Solution:**

There are two different registers associated with memory because each memory location has an address that identifies it and the data that is stored there, just as each mailbox in the LMC has both an address and the slip of paper containing the data stuffed into the slot.

**7.11, page# 232;**

Suppose that the instruction format for a modified Little Man Computer requires two consecutive locations for each instruction. The high-order digits of the instruction are located in the first mail slot, followed by the low –order digits. The IR is large enough to hold the entire instruction and can be addressed as IR [high] and [IR] low to load it. You may assume that the op code part of the instruction uses IR [high] and that the address is found in IR [low]. Write the fetch-execute cycle for an ADD instruction on this machine.

**Solution:**

Since it is only possible to access one memory location at a time, it will be necessary to access memory twice to fetch the entire instruction word. The instruction is in locations pointed to by PC and PC + 1. Therefore, the F-E cycle for an ADD instruction in this machine can be represented by

PC  MAR

MDR  IR [high]

PC + 1  PC fetch

PC MAR

MDR  IR [low]

IR [low]  MAR

A + MDR A

PC + 1  PC

Note that the PC is, indeed, incremented twice by this instruction.

**7.12, page# 233;**

The Little Prince Computer (LPC) is a mutant variation on the LMC. The LPC is so named because the differences are a royal plan). The LPC has one additional instruction. The extra instruction requires two consecutive words:

0XX

0YY

This instruction, known as move, moves data directly from location XX to location YY without affecting the value in the accumulator. To execute this instruction, the Little Prince would need to store the XX data temporarily. He can do this by writing the value on a piece of paper and holding it until he retrieves the second address. The equivalent in a real CPU might be called the intermediate address register, or IAR. Write the fetch-execute cycle for the LPC *MOVE* instruction.

**Solution:**

There are five basic steps to be performed:

a. fetch the instruction

b. retrieve the data from memory location XX

c. save the retrieved data in IAR

d. fetch the next location to get address YY

e. store the data from IAR to address YY

The following F-E cycle will do the job. The steps above are identified at the right:

PC  MAR

MDR  IR (step a)

IR [add]  MAR (step b)

MDR  IAR (step c)

PC + 1  PC

PC  MAR

MDR  IR (step d)

IR [add]  MAR

IAR MDR (step e)

PC + 1  PC

**7.13, page# 233.**

Generally, the distance that a programmer wants to move from the current instruction location on a BRANCH ON CONDITION is fairly small. This suggests that it might be appropriate to design the BRANCH instruction is such a way that the new location is calculated relative to the current instruction location. For example, we could design a different LMC instruction 8CX. The C digit would specify the condition on which to branch, and X would be a single-digit relative address. Using 10’s complement, this would allow a branch of -5 to +4 locations from the current address. If we were currently executing this instruction at location 24,803 would cause a branch on negative to location 27. Write a fetch-execute cycle for this exercise, and you may also assume that the complementary addition is handled correctly. The single-digit address X, is still found in IR [address]

PC  MAR

MDR  IR

if A < 0 PC + IR [add]  PC

else PC +1  PC