9.16 a. How many trap service routines can be implemented in the LC-3? Why?

b. Why must a RTI instruction be used to return from a TRAP routine?

Why won’t a BR (Unconditional Branch) instruction work instead?

c. How many accesses to memory are made during the processing of a TRAP instruction? Assume the TRAP is already in the IR.

*Answer:*

1. The trap vector is 8 bits wide, so 256 trap service routines can be implemented.
2. The RTI instruction pops the top two values on the system stack into the PC and PSR, since the PC contains the address following the address of the TRAP instruction, control returns to the user program at the correct address.

The BR instruction cannot go back to the next instruction before the jump to continue the user program.

1. 3 accesses. When pushing PC&PSR, the current R6 must be stored in Save\_USP, and the contents in SSP loaded in R6, which is the 1st and 2nd access. The 8-bit trap vector expands to an address that corresponds to trap vector table. The memory location contains the value loaded to the PC, which is the 3rd access.

9.17 Refer to Figure 9.14, the HALT service routine.

a. What starts the clock after the machine is HALTed? Hint: How can the HALT service routine return after bit [15] of the Master Control Register is cleared?

b. Which instruction actually halts the machine?

c. What is the first instruction executed when the machine is started again?

d. Where will the RET of the HALT routine return to?

*Answer:*

1. After the 15th bit of the machine control register is cleared, the Halt service routine never returns because the clock has been stopped. So external mechanism is the only way to start the clock.
2. STI R0, MCR ; Store R0 into MC register
3. LD R1, SaveR1 ; Restore registers
4. The RET of the HALT routine return to the program that executed the HALT instruction. The PC will point to the address following the HALT instruction.