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.

a.256 service routines can be specified. Since TRAP vector has 8 bit ,2^8 = 256

b. 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.

And BR may not has the range to resume PC ,and even if BR can resume PC by Label ,it can't resume PSR, which contain the previous State, it must be resumed but BR can't do that.

c. if PSR[15]==1 ,then need to switch Stack Pointer, but it don't need Mem

then PUSH temp and PC into system stack,it's 2 Mem access

then it will fetch Mem[trapvec]'s address of sub service routine. it's 1 Mem access

So it has 3 Mem access, push temp and PC into system Stack， and get Mem[trapvec] -> PC

Operation

```
TEMP=PSR;
if (PSR[15] == 1)
   Saved_USP=R6 and R6=Saved_SSP;
   PSR[15]=0;
Push TEMP,PC† on the system stack
PC=mem[ZEXT(trapvect8)];
```

Description

If the the program is executing in User mode, the User Stack Pointer must be saved and the System Stack Pointer loaded. Then the PSR and PC are pushed on the system stack. (This enables a return to the instruction physically following the TRAP instruction in the original program after the last instruction in the service routine (RTI) has completed execution.) Then the PC is loaded with the starting address of the system call specified by trapvector8. The starting address is contained in the memory location whose address is obtained by zero-extending trapvector8 to 16 bits.

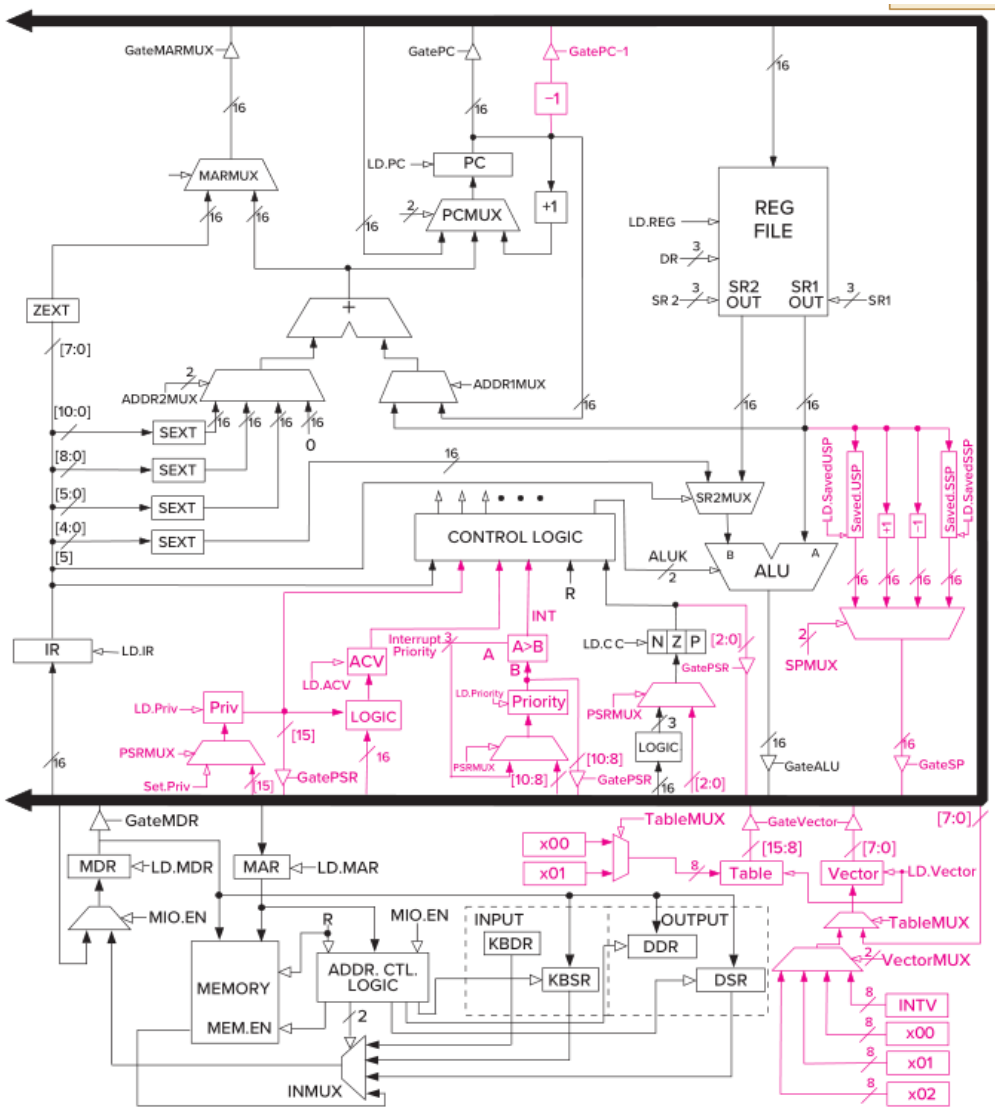

Figure C.8 LC-3 data path, including additional structures for interrupt control.

Section C.7.1 describes the flow of processing required to initiate an interrupt.
Section C.7.3 describes the flow of processing required to initiate an exception.

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 RTI of the HALT routine return to?

a. clock is stopped .none of instruction can be use. So only to start it by external ways, To set MCR[15]=1 and resume clock

b. STI R0,MCR ,it Stop the clock and halt the machine.

c. LD R1,SaveR1

d.to the PC where call HALT plus one. (PC+1,PC is where Call Halt)

```
01                  .ORIG   x0520     ; Where this routine resides
02                ST      R1, SaveR1 ; R1: a temp for MC register
03                ST      R0, SaveR0 ; R0 is used as working space
04
05  ; print message that machine is halting
06
07                LD      R0, ASCIINewLine
08                TRAP    x21
09                LEA     R0, Message
0A                TRAP    x22
0B                LD      R0, ASCIINewLine
0C                TRAP    x21
0D  ;
0E  ; clear bit 15 at xFFFE to stop the machine
0F  ;
10                LDI     R1, MCR    ; Load MC register into R1
11                LD      R0, MASK   ; R0 = x7FFF
12                AND     R0, R1, R0 ; Mask to clear the top bit
13                STI     R0, MCR    ; Store R0 into MC register
```

Figure 9.14 HALT service routine for the LC-3 (Fig. 9.14 continued on next page.)

```
14  ;
15  ; return from HALT routine.
16  ; (how can this routine return if the machine is halted above?)
17  ;
18                LD      R1, SaveR1 ; Restore registers
19                LD      R0, SaveR0
1A                RTI
1B  ;
1C  ; Some constants
1D  ;
1E  ASCIINewLine    .FILL   x000A
1F  SaveR0          .BLKW   1
20  SaveR1          .BLKW   1
21  Message         .STRINGZ   "Halting the machine."
22  MCR             .FILL   xFFFE      ; Address of MCR
23  MASK            .FILL   x7FFF      ; Mask to clear the top bit
24                  .END
```

Figure 9.14 HALT service routine for the LC-3 (continued Fig. 9.14 from previous page.)

First (lines 02 and 03), registers R1 and R0 are saved. R1 and R0 are saved because they are needed by the service routine. Then (lines 07 through 0C), the banner *Halting the machine* is displayed on the monitor. Finally (lines 10 through 13), the RUN latch (MCR[15]) is cleared by ANDing the MCR with 0111111111111111. That is, MCR[14:0] remains unchanged, but MCR[15] is cleared. *Question*: What instruction (or trap service routine) can be used to start the clock? *Hint:* This is a trick question! :-)