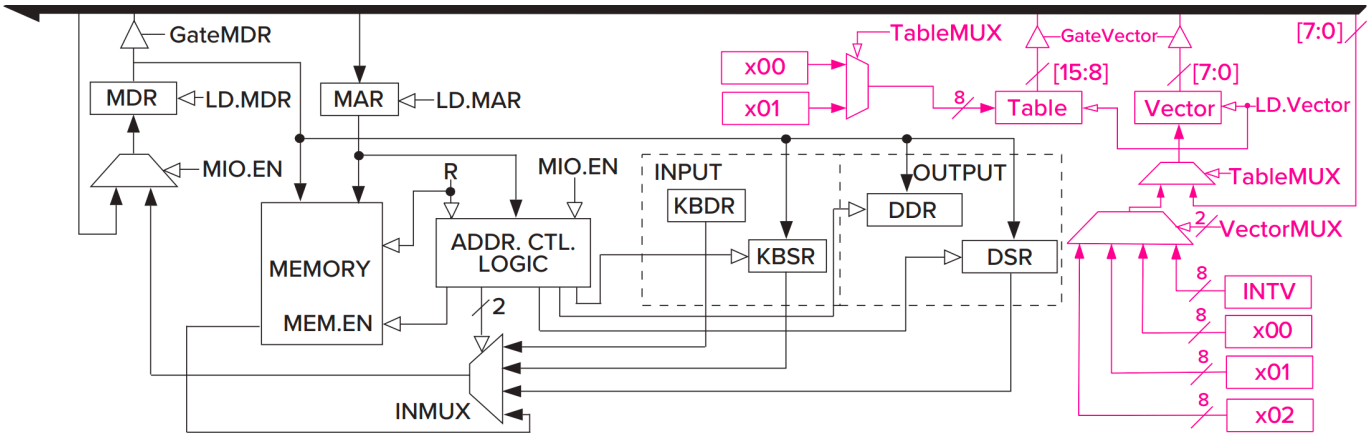
Problem

访问O/S和I/O时，PSR[15]需要设置为0。

Apendix C: The Microarchitecture of the LC-3



R: read/write

Table C.3 Truth Table for Address Control Logic

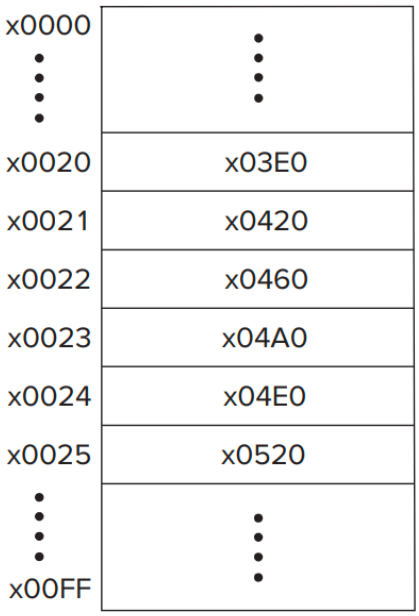
MAR	MIO.EN	R.W	MEM.EN	IN.MUX	LD.KBSR	LD.DSR	LD.DDR
xFE00	0	R	0	x	0	0	0
xFE00	0	W	0	x	0	0	0
xFE00	1	R	0	KBSR	0	0	0
xFE00	1	W	0	x	1	0	0
xFE02	0	R	0	x	0	0	0
xFE02	0	W	0	x	0	0	0
xFE02	1	R	0	KBDR	0	0	0
xFE02	1	W	0	x	0	0	0
xFE04	0	R	0	x	0	0	0
xFE04	0	W	0	x	0	0	0
xFE04	1	R	0	DSR	0	0	0
xFE04	1	W	0	x	0	1	0
xFE06	0	R	0	x	0	0	0
xFE06	0	W	0	x	0	0	0
xFE06	1	R	0	x	0	0	0
xFE06	1	W	0	x	0	0	1
other	0	R	0	x	0	0	0
other	0	W	0	x	0	0	0
other	1	R	1	mem	0	0	0
other	1	W	1	x	0	0	0

MIO: Memory/Input/Output

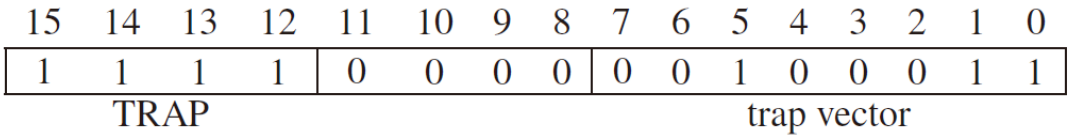
CH9: I/O

Trap Mechanism

- A set of **service routines** executed on behalf of user programs by the operating system.
- A **table of the starting addresses** of these 256 service routines.

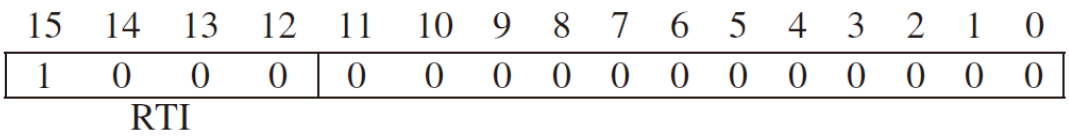


- The **TRAP instruction**



  - push PSR and PC into system stack (why push stack?) (trap in trap)
  - PSR[15] set 0, PSR[10:8] unchanged (so the PL of trap is the same as caller program) if from user mode to supervisor mode change USP to SSP
  - 8-bit vector zero-extension, get start address in trap vector table (x0000-x00FF) (why x0000-x00FF ?) (8 bit zero extension)

- A **linkage** back to the user program



  - pop PSR and PC from system stack
  - set PSR[15] 0 or 1, change SP if needed

State Machine

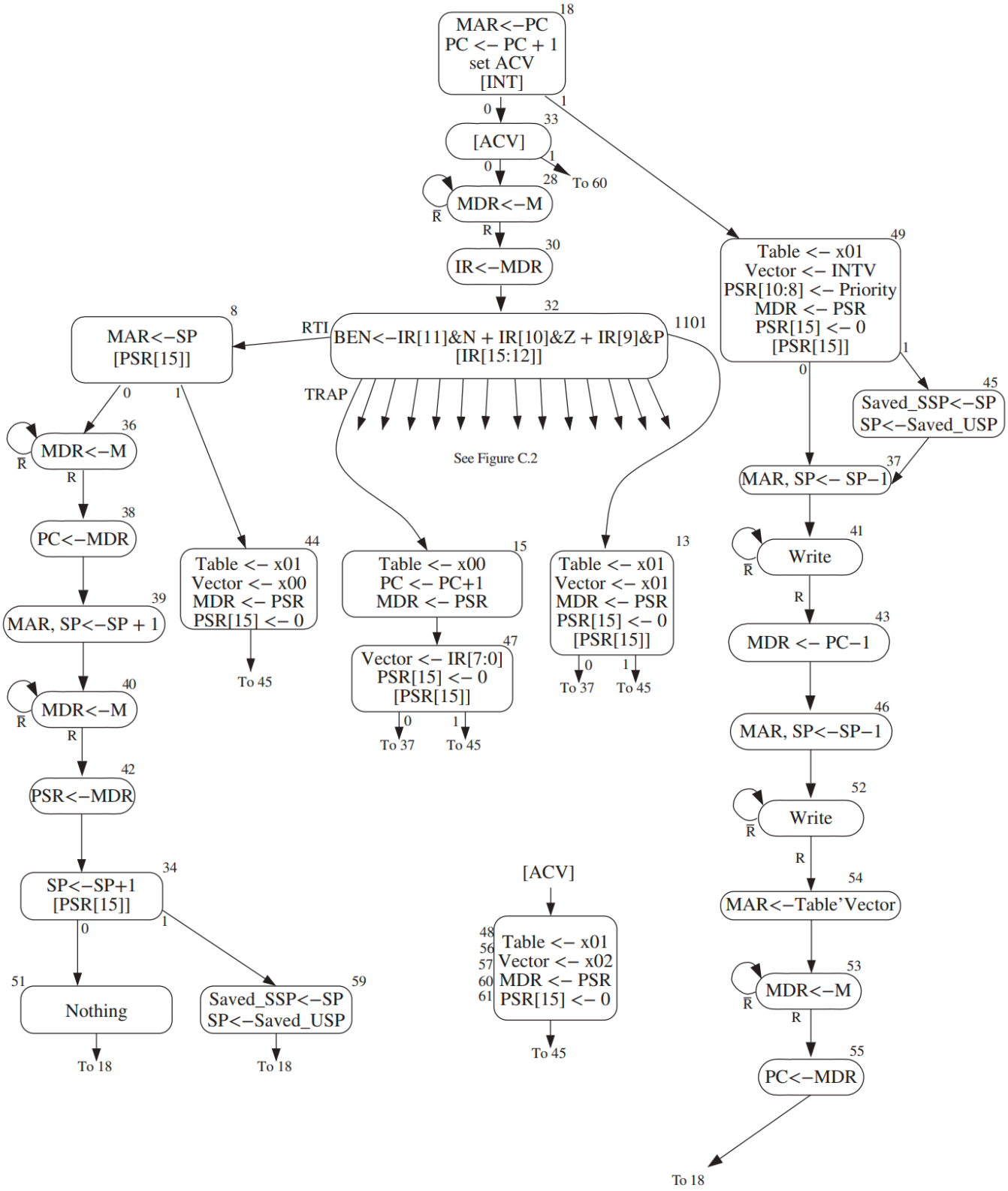


Figure C.7 LC-3 state machine showing interrupt control.

> 45有误
>
> ACV (access violence) INT (interrupt signal) BEN (branch enable)
> SP: R6
>
> in state 43, why should we push PC-1 into stack, but not PC？(中断的指令还没做，所以返回地址要把PC++的效果抵消)

The Trap Routine for Character Input

```
01                .ORIG   x04A0
02   START        JSR     SaveReg
03                LD      R2,Newline
04                JSR     WriteChar
05                LEA     R1,PROMPT
06   ;
07   ;
08   Loop         LDR     R2,R1,#0    ; Get next prompt char
09                BRz     Input
0A                JSR     WriteChar
0B                ADD     R1,R1,#1
0C                BRnzp   Loop
0D   ;
0E   Input        JSR     ReadChar
0F                ADD     R2,R0,#0    ; Move char to R2 for writing
10                JSR     WriteChar   ; Echo to monitor
11   ;
12                LD      R2, Newline
13                JSR     WriteChar
14                JSR     RestoreReg
15                RTI                 ; RTI terminates the trap routine
16   ;
17   Newline      .FILL   x000A
18   PROMPT       .STRINGZ  "Input a character>"
19   ;
1A   WriteChar    LDI     R3,DSR
1B                BRzp    WriteChar
1C                STI     R2,DDR
1D                RET                 ; JMP R7 terminates subroutine
1E   DSR          .FILL   xFE04
1F   DDR          .FILL   xFE06
20    ;
21   ReadChar     LDI     R3,KBSR
22                BRzp    ReadChar
23                LDI     R0,KBDR
24                RET
25   KBSR         .FILL   xFE00
26   KBDR         .FILL   xFE02
27    ;
28    SaveReg       ST      R1,SaveR1
29                  ST      R2,SaveR2
2A                  ST      R3,SaveR3
2B                  ST      R4,SaveR4
2C                  ST      R5,SaveR5
2D                  ST      R6,SaveR6
2E                  RET
2F    ;
30    RestoreReg    LD      R1,SaveR1
31                  LD      R2,SaveR2
32                  LD      R3,SaveR3
33                  LD      R4,SaveR4
34                  LD      R5,SaveR5
35                  LD      R6,SaveR6
36                  RET
37    SaveR1        .FILL   x0000
```

```
38    SaveR2          .FILL   x0000
39    SaveR3          .FILL   x0000
3A    SaveR4          .FILL   x0000
3B    SaveR5          .FILL   x0000
3C    SaveR6          .FILL   x0000
3D                    .END
```

Figure 9.15 The LC-3 trap service routine for character input (our final answer!).

Tip

- Opcode and operands are mandatory（必须的）
- The RTI instruction (opcode = 1000, with no operands)