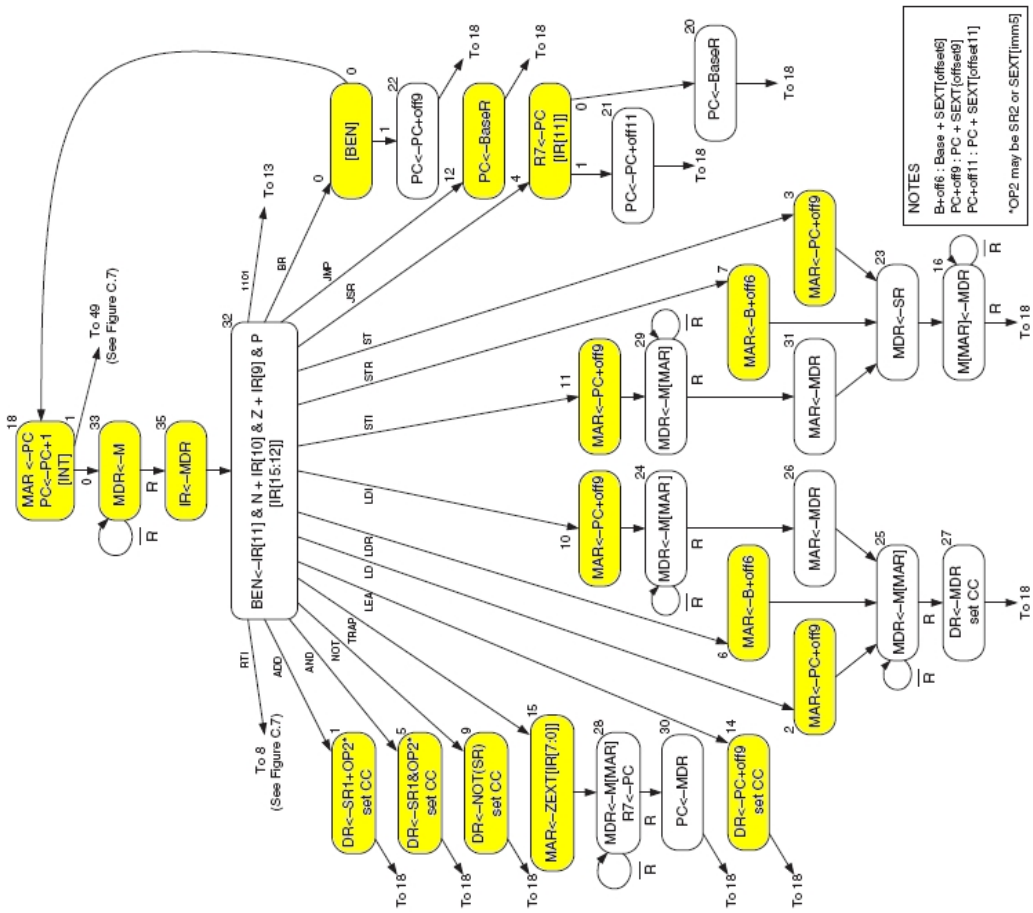
LC-3 FSM



NOTES: RTL corresponds to execution (after fetch!); JSRR not shown

LC-3 Instructions

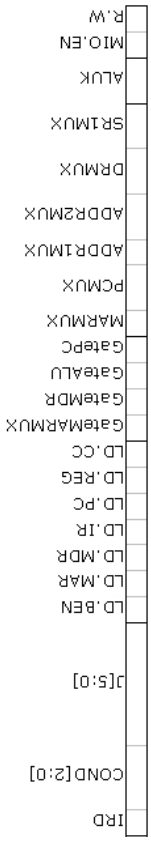
ADD	0001 DR SR1 0 00 SR2	ADD DR, SR1, SR2	DR ← SR1 + SR2, Setcc
ADD	0001 DR SR1 1 imm5	ADD DR, SR1, *imm5*	DR ← SR1 + SEXT(imm5), Setcc
AND	0101 DR SR1 0 00 SR2	AND DR, SR1, SR2	DR ← SR1 AND SR2, Setcc
AND	0101 DR SR1 1 imm5	AND DR, SR1, *imm5*	DR ← SR1 AND SEXT(imm5), Setcc
BR	0000 n z p PCoffset9	BR{nzp} *PCoffset9*	((n AND N) OR (z AND Z) OR (p AND P)): PC ← PC + SEXT(PCoffset9)
JMP	1100 000 BaseR 000000	JMP BaseR	PC ← BaseR
JSR	0100 1 PCoffset11	JSR *PCoffset11*	R7 ← PC, PC ← PC + SEXT(PCoffset11)
TRAP	1111 0000 trapvect8	TRAP *trapvect8*	R7 ← PC, PC ← M[ZEXT(trapvect8)]
LD	0010 DR PCoffset9	LD DR, *PCoffset9*	DR ← M[PC + SEXT(PCoffset9)], Setcc
LDI	1010 DR PCoffset9	LDI DR, *PCoffset9*	DR ← M[M[PC + SEXT(PCoffset9)]], Setcc
LDR	0110 DR BaseR offset6	LDR DR, BaseR, *offset6*	DR ← M[BaseR + SEXT(offset6)], Setcc
LEA	1110 DR PCoffset9	LEA DR, *PCoffset9*	DR ← PC + SEXT(PCoffset9), Setcc
NOT	1001 DR SR 111111	NOT DR, SR	DR ← NOT SR, Setcc
ST	0011 SR PCoffset9	ST SR, *PCoffset9*	M[PC + SEXT(PCoffset9)] ← SR
STI	1011 SR PCoffset9	STI SR, *PCoffset9*	M[M[PC + SEXT(PCoffset9)]] ← SR
STR	0111 SR BaseR offset6	STR SR, BaseR, *offset6*	M[BaseR + SEXT(offset6)] ← SR

LC-3 Datapath

LC-3 Datapath Control Signals

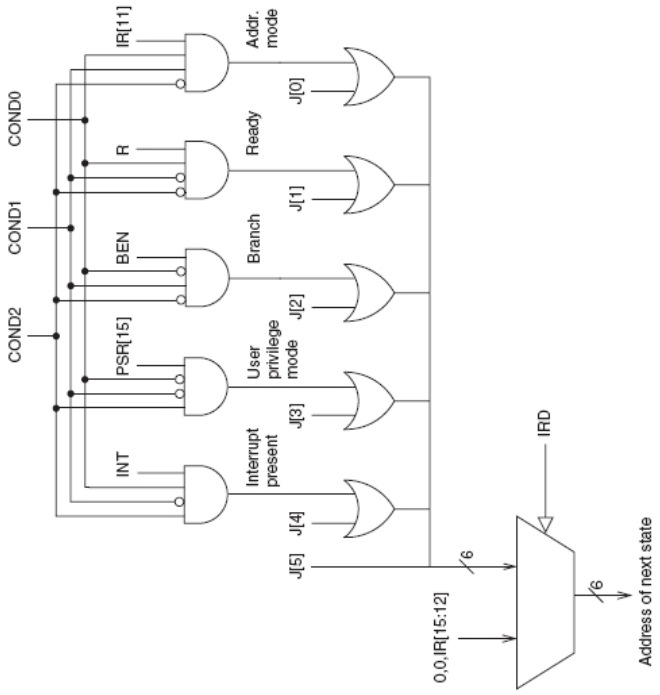
Signal	Description
LD.MAR	= 1, MAR is loaded
LD.MDR	= 1, MDR is loaded
LD.IR	= 1, IR is loaded
LD.PC	= 1, PC is loaded
LD.REG	= 1, register file is loaded
LD.BEN	= 1, updates Branch Enable (BEN) bit
LD.CC	= 1, updates status bits from system bus
GateMARMUX	= 1, MARMUX output is put onto system bus
GateMDR	= 1, MDR contents are put onto system bus
GateALU	= 1, ALU output is put onto system bus
GatePC	= 1, PC contents are put onto system bus
MARMUX	= 0, chooses ZEXT IR[7:0] = 1, chooses address adder output
MIO.EN	= 1, Enables memory, chooses memory output for MDR input = 0, Disables memory, chooses system bus for MDR input
R.W	= 1, M[MAR]<-MDR when MIO.EN = 1 = 0, MDR<-M[MAR] when MIO.EN = 1
ALUK	= 00, ADD = 01, AND = 10, NOT A = 11, PASS A
ADDR1MUX	= 0, chooses PC = 1, chooses reg file SR1 OUT
ADDR2MUX	= 00, chooses "0...00" = 01, chooses SEXT IR[5:0] = 10, chooses SEXT IR[8:0] = 11, chooses SEXT IR[10:0]
PCMUX	= 00, chooses PC + 1 = 01, chooses system bus = 10, chooses address adder output
DRMUX	= 00, chooses IR[11:9] = 01, chooses "111" = 10, chooses "110"
SR1MUX	= 00, chooses IR[11:9] = 01, chooses IR[8:6] = 10, chooses "110"

LC-3 Control Word Fields



LC-3 Microsequencer Control

Signal	Description
IRD	= 1, CAR ← 00||opcode (opcode = IR[15:12]), only during decode = 0, CAR ← J (plus 1,2,4,8,16 depending on COND bits)
COND	= 000, CAR ← J = 001, IF (R=1 and J[1]=0) THEN (CAR ← J plus 2) ELSE (CAR ← J) = 010, IF (BEN=1 and J[2]=0) THEN (CAR ← J plus 4) ELSE (CAR ← J) = 011, IF (IR[11]=1 and J[0]=0) THEN (CAR ← J plus 1) ELSE (CAR ← J)
J	6-bit next value for CAR (plus modifications depending on COND bits)



LC-3 TRAP Service Routines

Trap Vector	Assembler Name	Description
x20	GETC	Read a single character from the keyboard. The character is not echoed onto the console. Its ASCII code is copied into R0. The high eight bits of R0 are cleared.
x21	OUT	Write a character in R0[7:0] to the console display.
x22	PUTS	Write a string of ASCII characters to the console display. The characters are contained in consecutive memory locations, one character per memory location, starting with the address specified in R0. Writing terminates with the occurrence of x0000 in a memory location.
x23	IN	Print a prompt on the screen and read a single character from the keyboard. The character is echoed onto the console monitor, and its ASCII code is copied into R0. The high eight bits of R0 are cleared.
x24	PUTSP	Write a string of ASCII characters to the console. The characters are contained in consecutive memory locations, two characters per memory location, starting with the address specified in R0. The ASCII code contained in bits [7:0] of a memory location is written to the console first. Then the ASCII code contained in bits [15:8] of that memory location is written to the console. (A character string consisting of an odd number of characters to be written will have x00 in bits [15:8] of the memory location containing the last character to be written.) Writing terminates with the occurrence of x0000 in a memory location.
x25	HALT	Halt execution and print a message on the console.