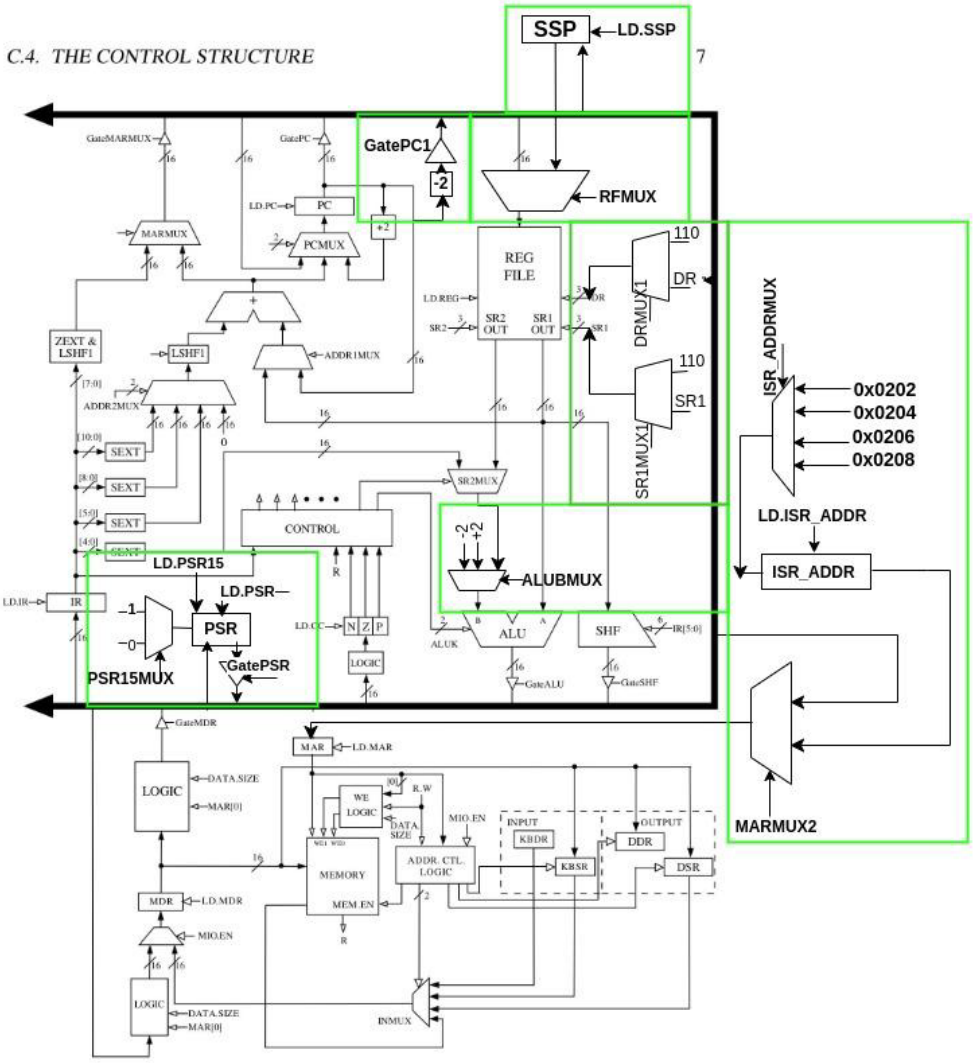
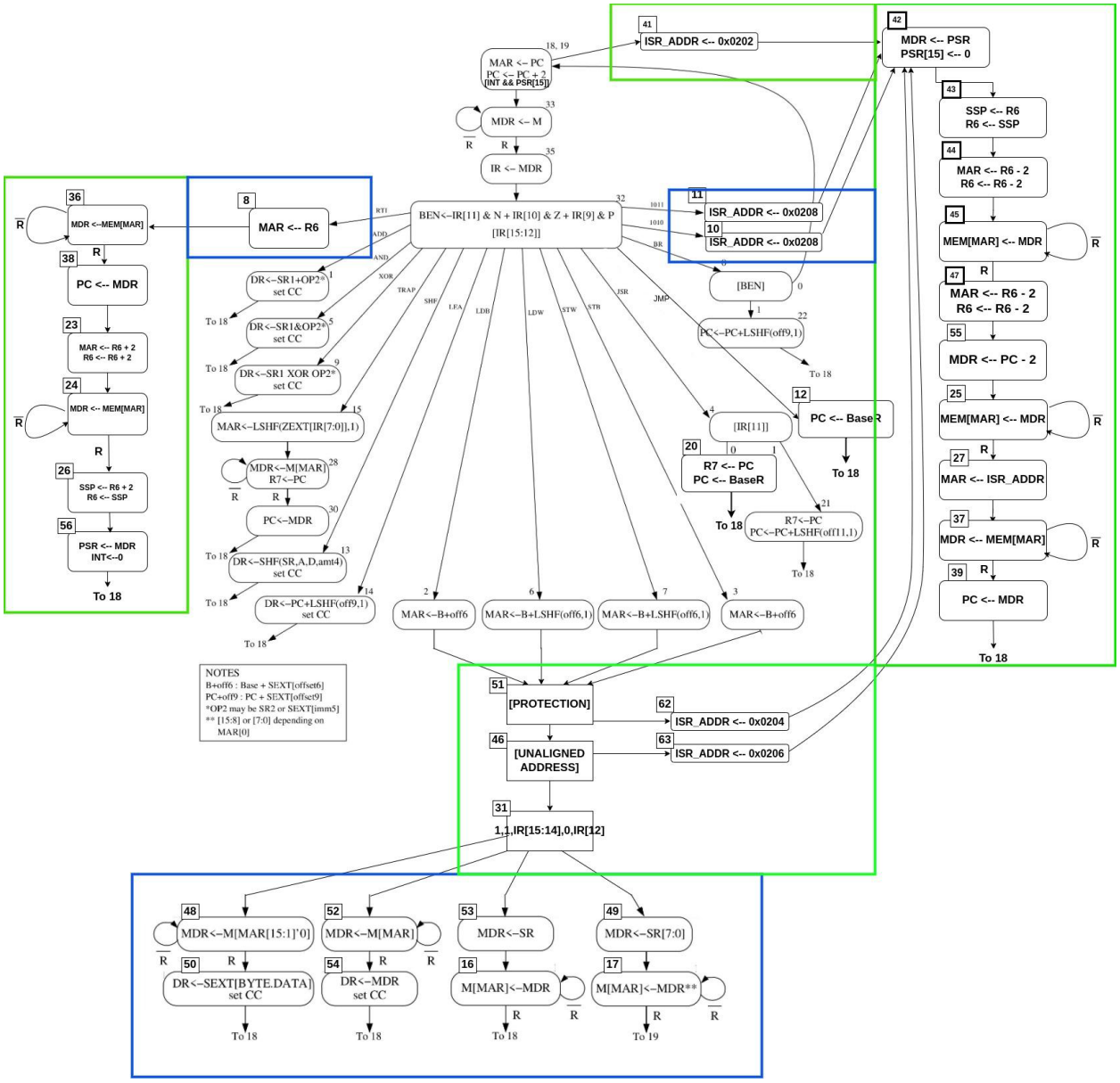
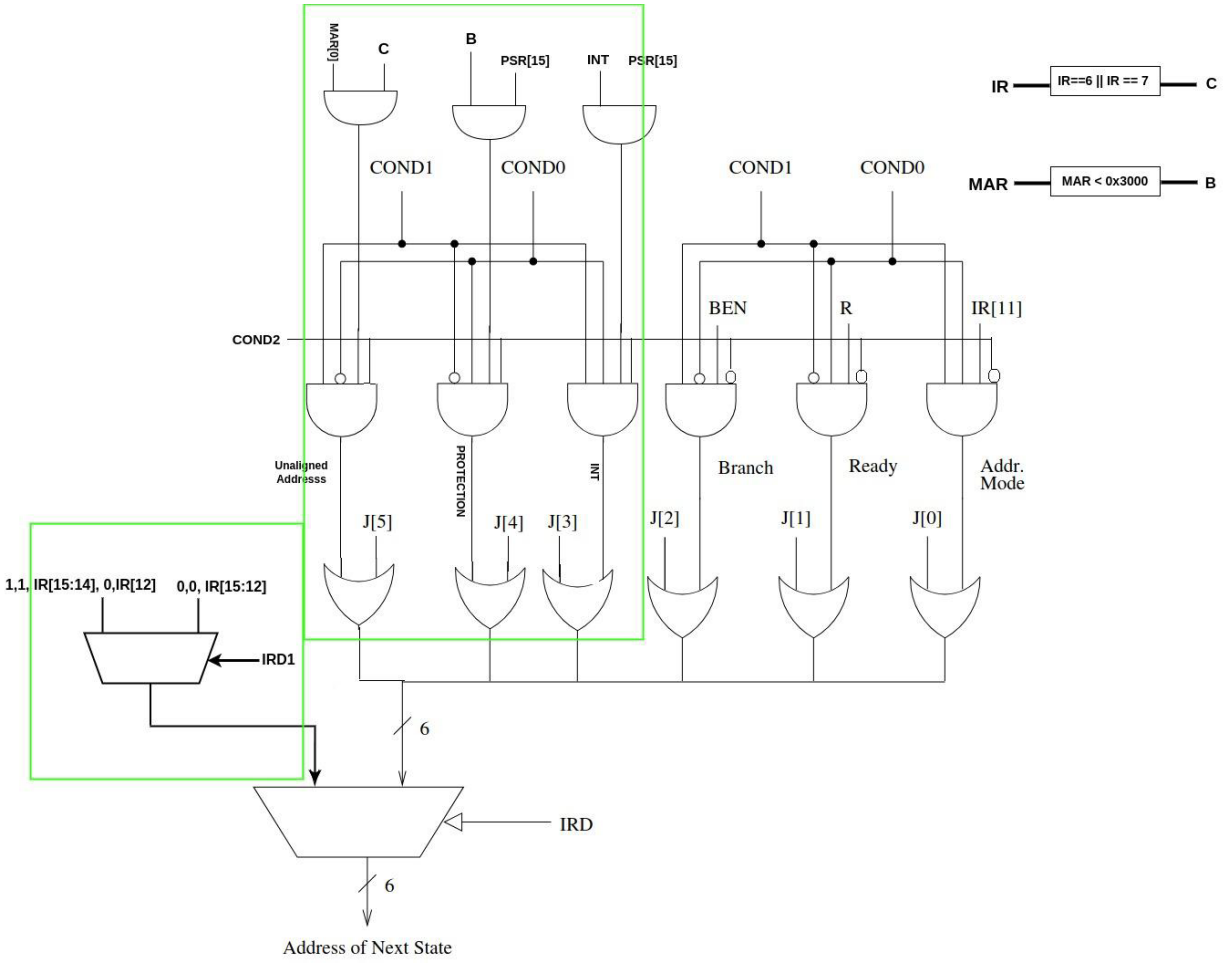


Figure C.3: The LC-3b data path

An SSP Register, controlled by the LD.SSP signal is added. This register captures data from the bus and serves as an input to the RFMUX, which toggles between bus data and the SSP Register output to load SSP data into R6.
GatePC1 transmits the Program Counter (PC), decremented by 2, onto the bus.
The ALUBMUX selects between src2, +2, and -2, facilitating the increment or decrement of R6 by 2.
SR1MUX1 and DRMUX1 set the source and destination register to R6, respectively, enabling R6 to function as a stack pointer.
ISR_ADDRMUX switches between 0x0202, 0x0204, 0x0206, and 0x0208 (Address in the interrupt vector table), depending on the exception or interrupt, loading the selected address into ISR_ADDR when LD.ISR_ADDR is high.

MARMUX2 selects between bus data and the ISR_ADDR output to feed into the MAR register, aiding in ISR address loading.
PSR15MUX manages the PSR[15] bit, indicating user vs. supervisor privilege. LD.PSR15 loads PSR15MUX's output into PSR[15]. Additionally, LD.PSR loads the PSR from the bus, while GatePSR outputs the PSR onto the bus.


41
ISR_ADDR <-- 0x0202
42
MDR <-- PSR
PSR[15] <-- 0
18, 19
MAR <-- PC
PC <-- PC + 2
[INT && PSR[15]]
33
MDR <-- M
35
IR <-- MDR
36
MDR <--MEM[MAR]
8
MAR <-- R6
32
BEN<--IR[11] & N + IR[10] & Z + IR[9] & P
[IR[15:12]]
11
ISR_ADDR <-- 0x0208
10
ISR_ADDR <-- 0x0208
38
PC <-- MDR
23
MAR <-- R6 + 2
R6 <-- R6 + 2
24
MDR <-- MEM[MAR]
26
SSP <-- R6 + 2
R6 <-- SSP
56
PSR <-- MDR
INT<--0
To 18
DR<--SR1+OP2*
set CC
DR<--SR1&OP2*
set CC
DR<--SR1 XOR OP2*
set CC
MAR<--LSHF(ZEXT[IR[7:0]],1)
MDR<--M[MAR]
R7<--PC
PC<--MDR
DR<--SHF(SR,A,D,amt4)
set CC
DR<--PC+LSHF(off9,1)
set CC
MAR<--B+off6
MAR<--B+LSHF(off6,1)
MAR<--B+LSHF(off6,1)
MAR<--B+off6
[BEN]
PC<--PC+LSHF(off9,1)
[IR[11]]
PC <-- BaseR
R7 <-- PC
PC <-- BaseR
R7<--PC
PC<--PC+LSHF(off11,1)
43
SSP <-- R6
R6 <-- SSP
44
MAR <-- R6 - 2
R6 <-- R6 - 2
45
MEM[MAR] <-- MDR
47
MAR <-- R6 - 2
R6 <-- R6 - 2
55
MDR <-- PC - 2
25
MEM[MAR] <-- MDR
27
MAR <-- ISR_ADDR
37
MDR <-- MEM[MAR]
39
PC <-- MDR
NOTES
B+off6 : Base + SEXT[offset6]
PC+off9 : PC + SEXT[offset9]
*OP2 may be SR2 or SEXT[imm5]
** [15:8] or [7:0] depending on
MAR[0]
51
[PROTECTION]
62
ISR_ADDR <-- 0x0204
46
[UNALIGNED
ADDRESS]
63
ISR_ADDR <-- 0x0206
31
1,1,IR[15:14],0,IR[12]
48
MDR<--M[MAR[15:1]'0]
50
DR<--SEXT[BYTE.DATA]
set CC
52
MDR<--M[MAR]
54
DR<--MDR
set CC
53
MDR<--SR
16
M[MAR]<--MDR
49
MDR<--SR[7:0]
17
M[MAR]<--MDR**
To 19



Added IRD1 Mux to select states i.e. 48, 52, 53, and 49 from the state 31. The IR[13] (2nd bit) is tied to 0 so that it can be enabled for state transitions 48 → 50 and 52 → 54.

Requirement → Protection, Unaligned Address, and Interrupt Conditions
The above requirements led to the addition of the COND2 signal.

COND = 110 (Unaligned Address Exception)
COND = 101 (Protection Exception)
COND = 111 (Timer Interrupt)

The Unaligned Address Exception is triggered when MAR[0] = 1 and the instruction is either STW or LDW.
The Protection Exception is triggered if the address is less than 0x3000 and the system is in user mode (i.e., PSR[15] = 1).
The Timer Interrupt is triggered when the interrupt vector is 0x01 and the system is in user mode (i.e., PSR[15] = 1).

Signal Name	Signal Values	
LD.MAR/1:	NO, LOAD	
LD.MDR/1:	NO, LOAD	
LD.IR/1:	NO, LOAD	
LD.BEN/1:	NO, LOAD	
LD.REG/1:	NO, LOAD	
LD.CC/1:	NO, LOAD	
LD.PC/1:	NO, LOAD	
GatePC/1:	NO, YES	
GateMDR/1:	NO, YES	
GateALU/1:	NO, YES	
GateMARMUX/1:	NO, YES	
GateSHF/1:	NO, YES	
PCMUX/2:	PC+2	;select pc+2
	BUS	;select value from bus
	ADDER	;select output of address adder
DRMUX/1:	11.9	;destination IR[11:9]
	R7	;destination R7
SR1MUX/1:	11.9	;source IR[11:9]
	8.6	;source IR[8.6]
ADDR1MUX/1:	PC, BaseR	
ADDR2MUX/2:	ZERO	;select the value zero
	offset6	;select SEXT[IR[5:0]]
	PCoffset9	;select SEXT[IR[8:0]]
	PCoffset11	;select SEXT[IR[10:0]]
MARMUX/1:	7.0	;select LSHF(ZEXT[IR[7:0]],1)
	ADDER	;select output of address adder
ALUK/2:	ADD, AND, XOR, PASSA	
MIO.EN/1:	NO, YES	
R.W/1:	RD, WR	
DATA.SIZE/1:	BYTE, WORD	
LSHF1/1:	NO, YES	
LD.PSR/1:	NO, LOAD	
LD.PSR15/1:	NO, LOAD	
LD.SSP/1:	NO, LOAD	
LD.ISR_ADDR/1:	NO, LOAD	
GatePSR/1:	NO,YES	
GatePC1/1:	NO,YES	
RFMUX/1:	BUS, SSP	
ALUBMUX/2:	B, +2, -2	
ISR_ADDRMUX/2:	0x0202, 0x0204, 0x0206, 0x0208	
PSR15MUX/1:	1, 0	
SR1MUX1/1:	SR, R6	
DR1MUX1/1:	DR, R6	
MARMUX2/1	BUS, ISR_ADDR	

Signal Name	Signal Values	
J/6:		
IRD/1:	NO, YES	
IRD1/1: NO, YES		

Table C.2: Microsequencer control signals

J/6
COND/3

COND_0 : Unconditionally
COND_1 : Memory Ready
COND_2 : Branch
COND_3 : Addressing Mode
COND_5 : Protection Exception
COND_6 : Unaligned Exception
COND_7 : Timer Interrupt

IRD/1 NO/YES
IRD1/1 NO/YES

LD.PSR → Loads the bus data on the PSR
LD.PSR15 → Loads either 0 or 1 on the PSR[15] bit
LD.SSP → Loads the bus data on the SSP register.
LD.ISR_ADDR → Loads the ISR_ADDRMUX output on the ISR_ADDR register
GatePSR → Loads the PSR register data on the bus.
GatePC → Loads the PC register data on the bus
RFMUX → Select between SSP and bus as Register File Input
ALUBMUX → Select between Reg File 2nd output, +2, and -2 as an ALU input
ISR_ADDRMUX →Select between 0x0202, 0x204, 0x0206, and 0x0208 as ISR address
PSR15MUX →Select between 0 and 1 for the PSR[15] bit
SR1MUX1 → Select between SRMUX output and R6 as a source register
DR1MUX1 → Selects between DRMUX output and R6 as the destination register

MARMUX2 → Selects between data on the bus and ISR_ADDR register