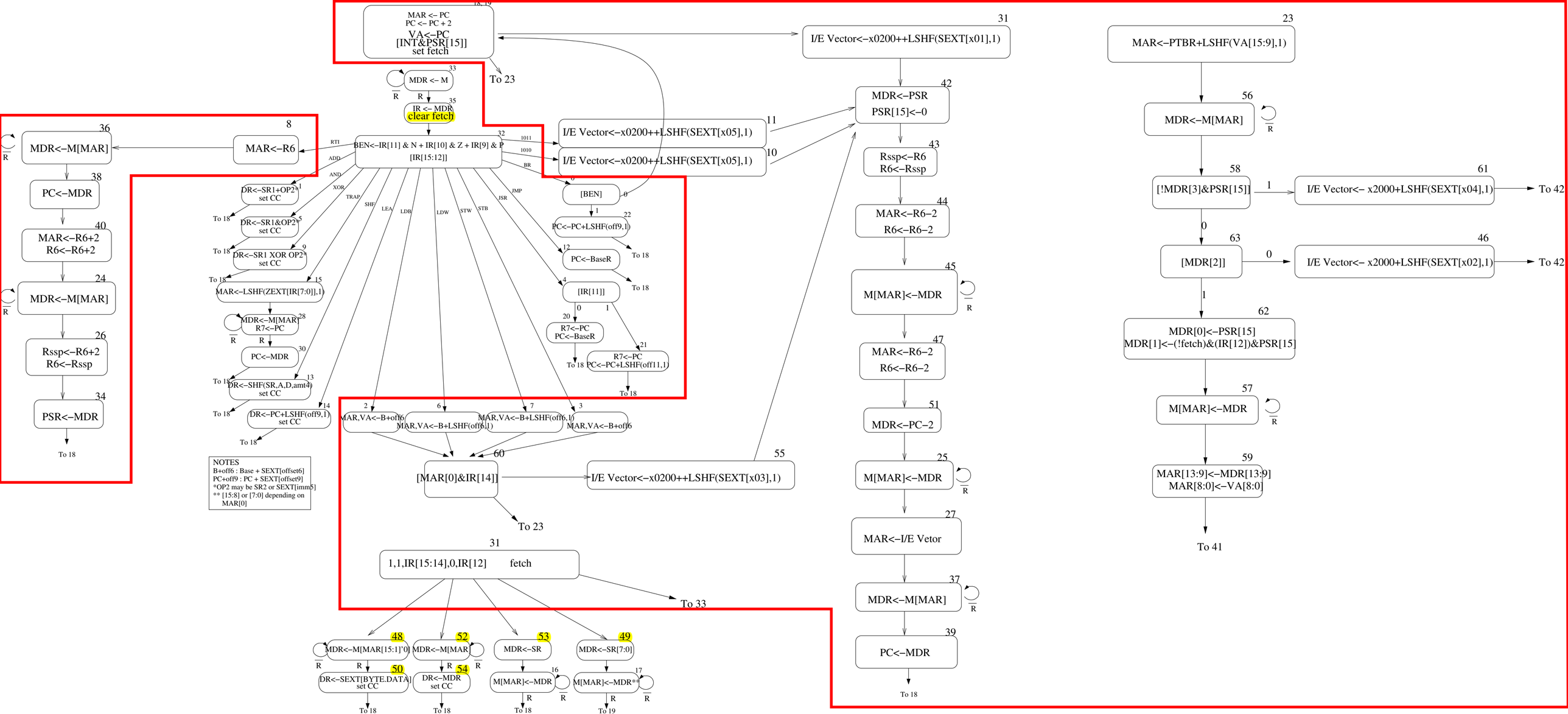
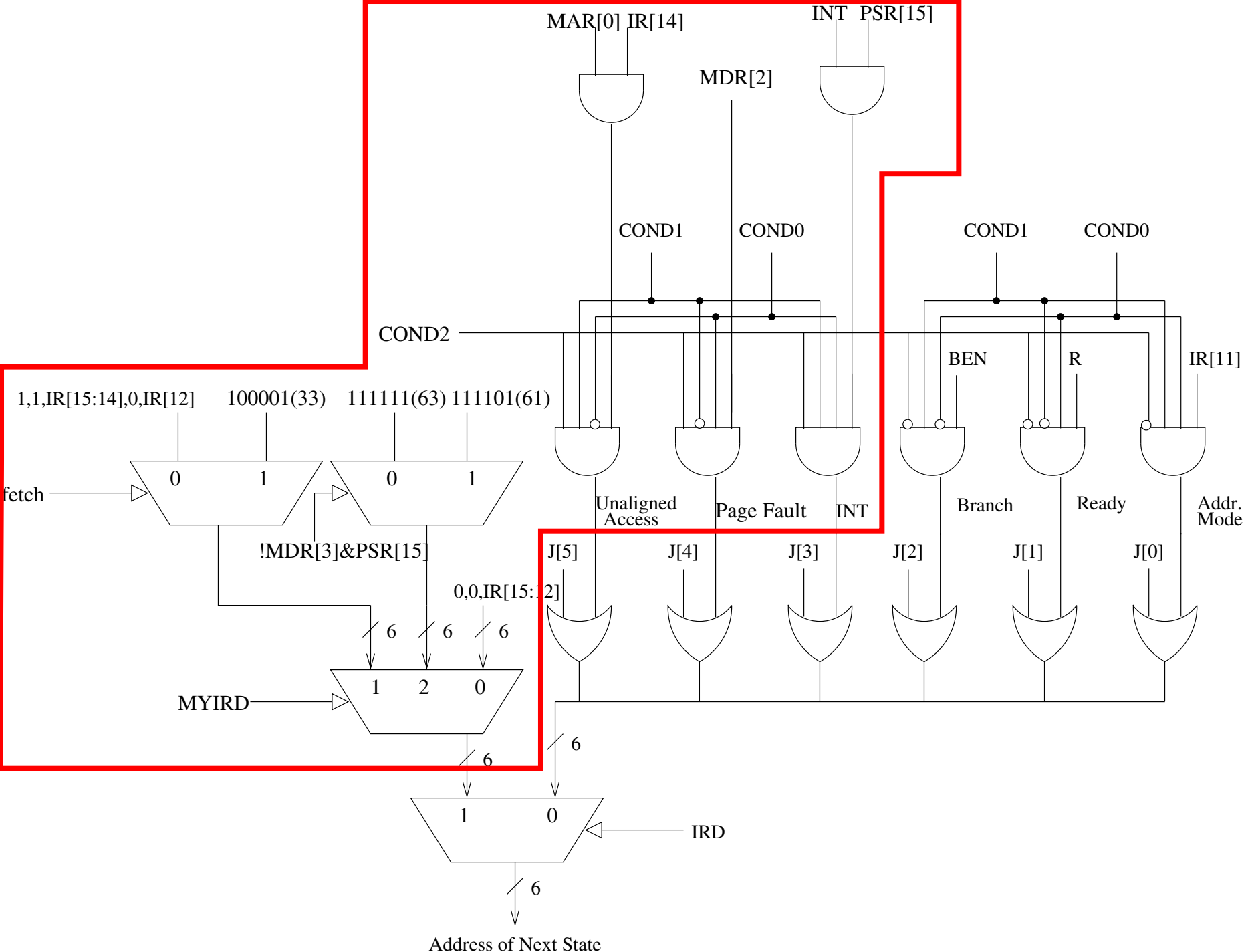
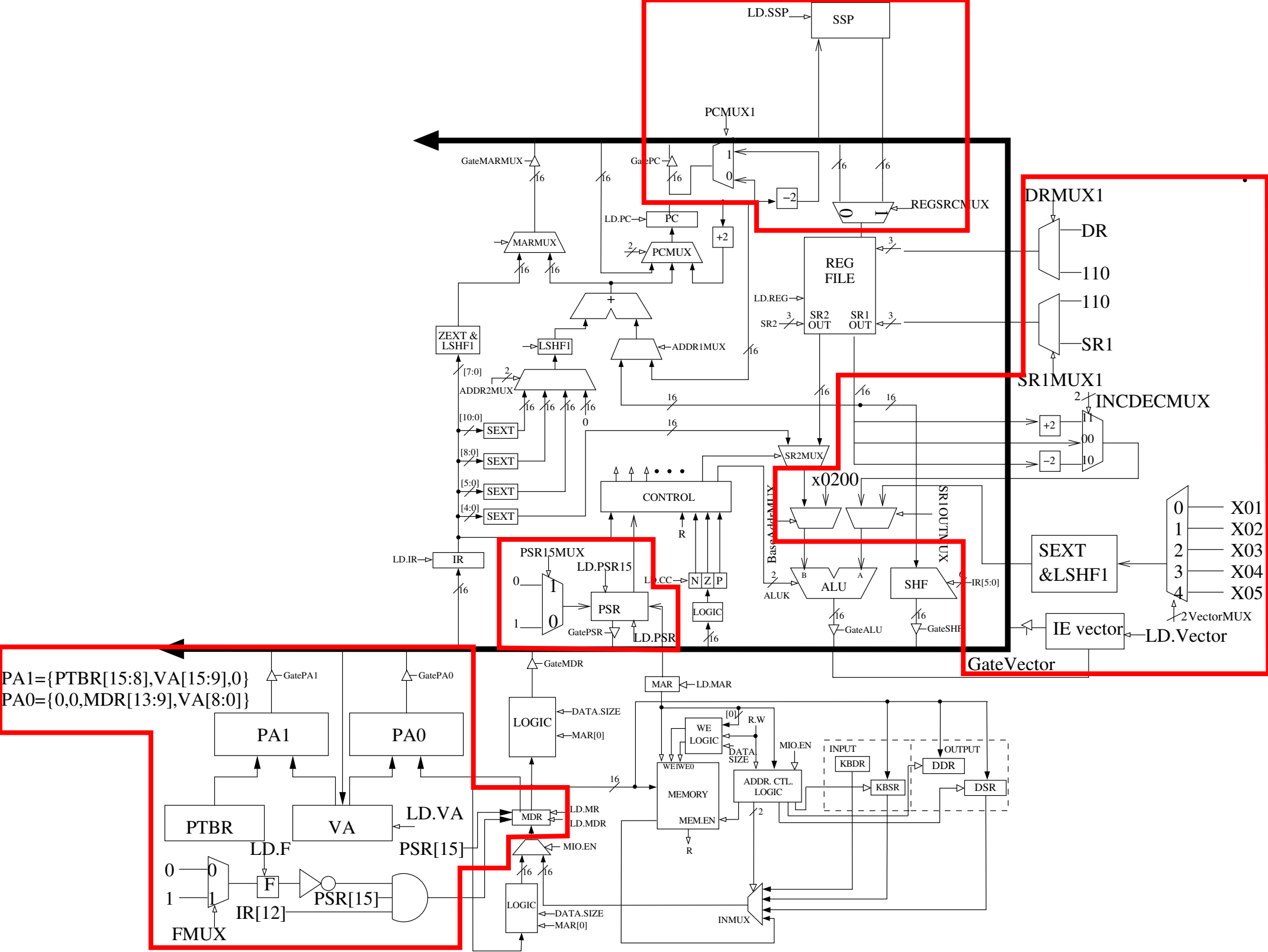

MAR <– PC
PC <– PC + 2
VA<–PC
[INT&PSR[15]]
set fetch
To 23
MDR <– M
IR <– MDR
clear fetch
BEN<–IR[11] & N + IR[10] & Z + IR[9] & P
[IR[15:12]]
RTI
ADD
AND
XOR
TRAP
SHF
LEA
LDB
LDW
STW
STB
JSR
JMP
1011
1010
BR
[BEN]
DR<–SR1+OP2*
set CC
DR<–SR1&OP2*
set CC
DR<–SR1 XOR OP2*
set CC
MAR<–LSHF(ZEXT[IR[7:0]],1)
MDR<–M[MAR]
R7<–PC
PC<–MDR
DR<–SHF(SR,A,D,amt4)
set CC
DR<–PC+LSHF(off9,1)
set CC
PC<–PC+LSHF(off9,1)
PC<–BaseR
[IR[11]]
R7<–PC
PC<–BaseR
R7<–PC
PC<–PC+LSHF(off11,1)
MAR,VA<–B+off6
MAR,VA<–B+LSHF(off6,1)
[MAR[0]&IR[14]]
I/E Vector<–x0200++LSHF(SEXT[x03],1)
I/E Vector<–x0200++LSHF(SEXT[x05],1)
I/E Vector<–x0200++LSHF(SEXT[x01],1)
To 18
NOTES
B+off6 : Basc + SEXT[offset6]
PC+off9 : PC + SEXT[offset9]
*OP2 may be SR2 or SEXT[imm5]
** [15:8] or [7:0] depending on MAR[0]
MDR<–M[MAR]
PC<–MDR
MAR<–R6+2
R6<–R6+2
Rssp<–R6+2
R6<–Rssp
PSR<–MDR
MAR<–R6
1,1,IR[15:14],0,IR[12] fetch
To 33
MDR<–M[MAR[15:1]'0']
DR<–SEXT[BYTE.DATA]
set CC
MDR<–M[MAR]
DR<–MDR
set CC
MDR<–SR
M[MAR]<–MDR
MDR<–SR[7:0]
M[MAR]<–MDR**
To 19
MDR<–PSR
PSR[15]<–0
Rssp<–R6
R6<–Rssp
MAR<–R6–2
R6<–R6–2
M[MAR]<–MDR
MAR<–R6–2
R6<–R6–2
MDR<–PC–2
M[MAR]<–MDR
MAR<–I/E Vetor
MDR<–M[MAR]
PC<–MDR
MAR<–PTBR+LSHF(VA[15:9],1)
MDR<–M[MAR]
[!MDR[3]&PSR[15]]
I/E Vector<– x2000+LSHF(SEXT[x04],1)
[MDR[2]]
I/E Vector<– x2000+LSHF(SEXT[x02],1)
MDR[0]<–PSR[15]
MDR[1]<–(!fetch)&(IR[12])&PSR[15]
M[MAR]<–MDR
MAR[13:9]<–MDR[13:9]
MAR[8:0]<–VA[8:0]
To 41
To 42


MAR[0] IR[14]
INT PSR[15]
MDR[2]
COND1
COND0
COND1
COND0
COND2
BEN
R
IR[11]
1,1,IR[15:14],0,IR[12]
100001(33)
111111(63)
111101(61)
0
1
0
1
fetch
Unaligned Access
Page Fault
INT
Branch
Ready
Addr. Mode
!MDR[3]&PSR[15]
J[5]
J[4]
J[3]
J[2]
J[1]
J[0]
0,0,IR[15:12]
6
6
6
1
2
0
MYIRD
6
6
1
0
IRD
6
Address of Next State


LD.SSP
SSP
PCMUX1
GateMARMUX
GatePC
1
0
−2
REGSRCMUX
DRMUX1
DR
110
LD.PC
PC
+2
MARMUX
2
PCMUX
REG
FILE
LD.REG
SR2
3
SR2
OUT
SR1
OUT
110
SR1
SR1MUX1
+
ZEXT &
LSHF1
LSHF1
ADDR1MUX
[7:0]
2
ADDR2MUX
INCDECMUX
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
SR2MUX
x0200
CONTROL
R
BaseAddrMUX
SR1OUTMUX
LD.IR
IR
PSR15MUX
LD.PSR15
LD.CC
N
Z
P
PSR
LOGIC
GatePSR
LD.PSR
ALUK
B
A
ALU
SHF
IR[5:0]
GateALU
GateSHF
SEXT
&LSHF1
X01
X02
X03
X04
X05
2VectorMUX
IE vector
LD.Vector
GateVector
PA1={PTBR[15:8],VA[15:9],0}
PA0={0,0,MDR[13:9],VA[8:0]}
GatePA1
GatePA0
PA1
PA0
PTBR
VA
LD.VA
LD.F
PSR[15]
F
IR[12]
FMUX
GateMDR
MAR
LD.MAR
LOGIC
DATA.SIZE
MAR[0]
WE
LOGIC
R.W
MIO.EN
INPUT
OUTPUT
KBDR
DDR
KBSR
DSR
WE1WE0
MEMORY
MEM.EN
ADDR. CTL.
LOGIC
LD.MR
LD.MDR
MDR
INMUX

Signal Name	Signal Values	
MYIRD/2	0	; 0,0,IR[15:12]
	1	; fetch(F) == 0: 1,1,IR[15:14],0,IR[12] ; fetch(F) == 1: 33 ; !MDR[3]&PSR[15] == 0: 63
	2	; !MDR[3]&PSR[15] == 1: 61
COND(COND2/1, COND1, COND0)	COND0~COND 3	; Original Cond Signal
	COND5	; Protection Exception
	COND6	; Unaligned Access
	COND7	; Interrupt
LD.SSP/1	NO, LOAD	
LD.VECTOR/1	NO, LOAD	
LD.PSR/1	NO, LOAD	
LD.PSR15/1	NO, LOAD	
GateVector/1	NO, YES	
GatePSR/1	NO, YES	
PCMUX1/1	PC	; Select PC
	PC-2	; Select PC-2
REGSRCMUX/1	BUS	; Select value from bus
	SSP	; Select value from SSP register
INCDECMUX/2	0, 1	; Select SR1OUT
	2	; Select SR1OUT - 2
	3	; Select SR1OUT + 2
VECTORMUX/3	0	; Select timer interrupt vector
	1	; Select page fault exception vector
	2	; Select unaligned-access exception vector
	3	; Select protection exception vector
	4	; Select unknown-opcode exception vector
SR1OUTMUX	0	; Select data from output of INCDECMUX
	1	; Select data from SEXT&LSHF1ed vector
BASEADDRMUX	0	; Select data from the output of SR2MUX
	1	; Select 0x0200
PSR15MUX	0	; Select value 1
	1	; Select value 0
DRMUX1	0	; Select value from output of DRMUX
	1	; Select 110
SR1MUX1	0	; Select value from output of SR1MUX
	1	; Select 110
LD.VA	NO, LOAD	
LD.MR	NO, LOAD	; Load (M)odify bit and R(eference) bit
LD.F	NO, LOAD	; Load F(etch) register
FMUX	0	; Indicate this memory access is not a fetch
	1	; Indicate this memory access is a fetch