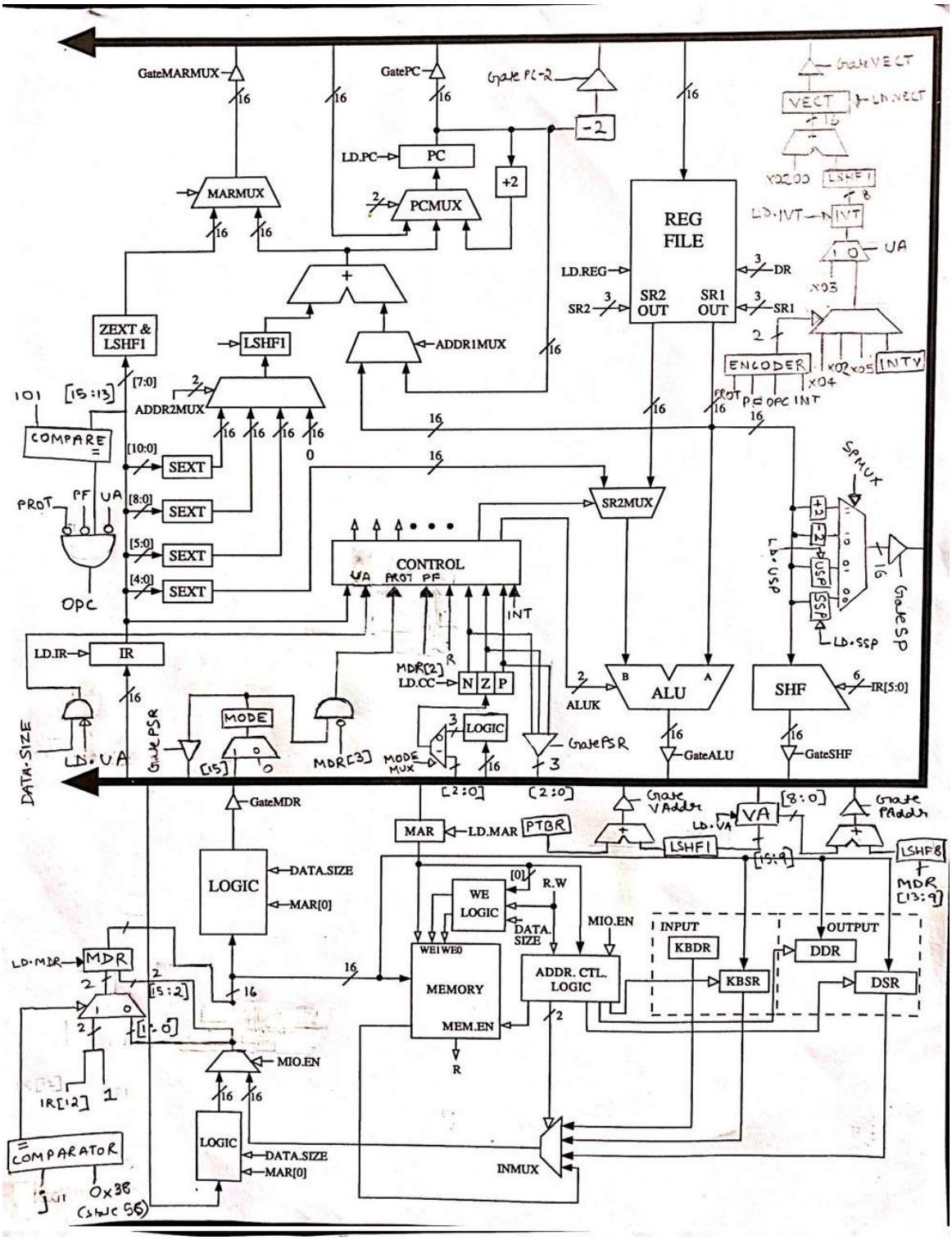
Adithya Ashok

Cycle-Level Simulation of LC3b Microarchitecture, with support for interrupts/exceptions and virtual memory/address translation, Spring 2021



LC3b Microarchitecture Document, Dr. Yale Patt:
http://users.ece.utexas.edu/~patt/05f.360N/handouts/360n.appC.pdf

Explanation of Datapath changes:

GATEPC-2 allows PC – 2 to be loaded onto the bus for interrupts.

MODE: PSR[15] dictates whether the program has user or supervisor privilege. Depending on **MODEMUX**, this register either loads PSR[15] from the BUS or a 0 to set PSR[15] to supervisor privilege for interrupts and exceptions.

The condition code logic was modified so it can load NZP from the bus when MODEMUX is 1, i.e. when the flags need to be set to a prior value that is in the BUS.

GATEPSR allows for the PSR to be loaded onto the bus for saving in the MDR later. The gate is responsible both for the current priority bit and condition codes.

UA, the unaligned access signal is generated by MAR[0] & DATA.SIZE, since it occurs for a word aligned access at an odd-numbered location. This is sent to the microsequencer.

INT is the external interrupt, sent to the microsequencer.

VECT is the register that contains the starting location of the service routine. It is determined as follows:

> First, we need to select the correct vector entry. **INTV** is the 8 bit interrupt vector supplied by the external interrupt. This, along with the determined exception vectors 0x02, 0x03, 0x03 are the inputs top a **4x1 MUX**.
>
> The select lines for the MUX are determined using a **PRIORITY ENCODER.** This takes the 4 input signals (same as microsequencer additions) for Unaligned access, Unknown Opcode and Protection Exception along with the Interrupt signal. Since it is a priority encoder, it will not cause errors with multiple signals being high and has the interrupt at the lowest priority and the others in any order.
>
> **OPC**, the unknown opcode signal is generated by comparing OPCODE >> 1 with 5. Since both 10 and 11 would generate a 1 with the equality comparator, it generates a signal for unknown opcode.
>
> The output for the **4x1 MUX** are the vector table entries, which are stored in the **IVT** register. These entries are then left shifted by 1 and added to 0x200 to generate the input for **VECT**.

Note: To minimize critical path, I included both the IVT register and the VECT in my design. In one cycle, IVT is loaded and in the later, VECT. This does not take any additional states.

Finally, **SPMUX** selects the values to be loaded onto the bus for saving into R6. Its select lines select between the various values that need to pushed on.

USP stores the user stack pointer, which can be loaded from R6.

SSP stores the system stack pointer, which can be loaded from R6.

The other two select lines increment and decrement the input i.e. R6 by 2, for when R6 needs to be incremented and decremented. (since the destination is also R6)

Therefore, the following is implemented through SPMUX along with the SSP and USP registers.

R6 = R6 + 2, R6 = R6 – 2, R6 = SSP, R6 = USP, USP = R6, SSP = R6

To allow SR1OUT and DR to access R6, changes need to be made to **SR1MUX** and **DRMUX**.

The select lines for the MUX are widened to 2 bits, where the high bit selects 110, i.e. R6.

In addition to the above, the following changes were made for lab 5.

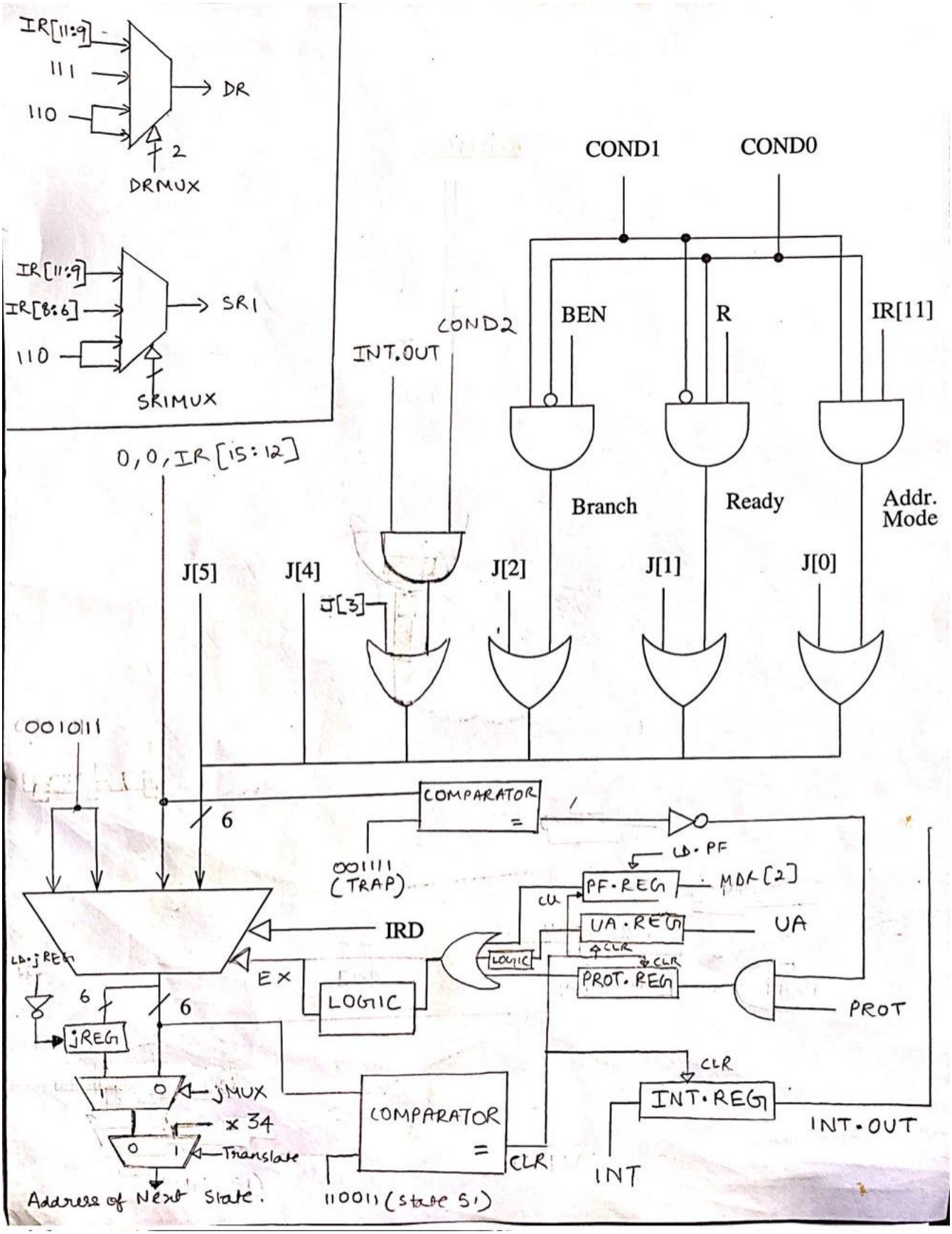
An additional mux was added to the interrupt handling hardware, the select line being UA and the output being its vector table entry on a high. On a low, it merely passes through the output of the 4x1 vector table, which has been changed to include the vector table entry of pagefaults.The priority encoder has been changed to prioritize protection exceptions over the newly added pagefault exception. The pagefault exception signal is generated based on the PTE valid bit ie. MDR[2].

PROT, the protection signal is generated by MODE == 1 & PTE[3] == 1 (i.e. MDR[3]). This signal is high when a protected page is accessed by a user program. This is sent to the microsequencer.

A virtual address register was added. For all memory accesses, the value is loaded into VA rather than MAR. The high 7 bits of the VA (page number) are left shifted by one and go into an adder with the newly added PTBR register, which is initialized to the page table base. The high 9 bits (offset) go into an adder with MDR[13:9] (left shifted by 8) to designate physical address. Gate VAddr gates the PTE address onto the bus, PAddr gates the physical address. Both are merely gating the adders.

In order to set reference and modify bits for PTEs, new logic was added to manipulate the MDR.

A comparator checks the j value to 56, to see if PTE has been loaded. The output of this comparator is the select line to a mux which changes the low 2 bits of the MDR input on a high. PTE[0] in the MDR becomes 1 and PTE[1] becomes IR12, which is only high for stores.


IR[11:9]
111
110
DR
2
DRMUX
IR[11:9]
IR[8:6]
110
SR1
SR1MUX
COND1
COND0
BEN
R
IR[11]
COND2
INT.OUT
Branch
Ready
Addr. Mode
0,0,IR[15:12]
J[5]
J[4]
J[3]
J[2]
J[1]
J[0]
001011
COMPARATOR
=
6
001111
(TRAP)
LD.PF
PF.REG
MDR[2]
CLR
UA.REG
UA
IRD
LOGIC
CLR
CLR
PROT.REG
PROT
EX
LOGIC
LD.JREG
6
6
JREG
1
0
JMUX
x 34
0
1
Translate
Address of Next State.
COMPARATOR
=
CLR
110011 (state 51)
CLR
INT.REG
INT.OUT
INT

The changes to the microsequencer are as follows:

As established before, the signals **INT, PROT, PF(page fault) and UA** are sent to the microsequencer.

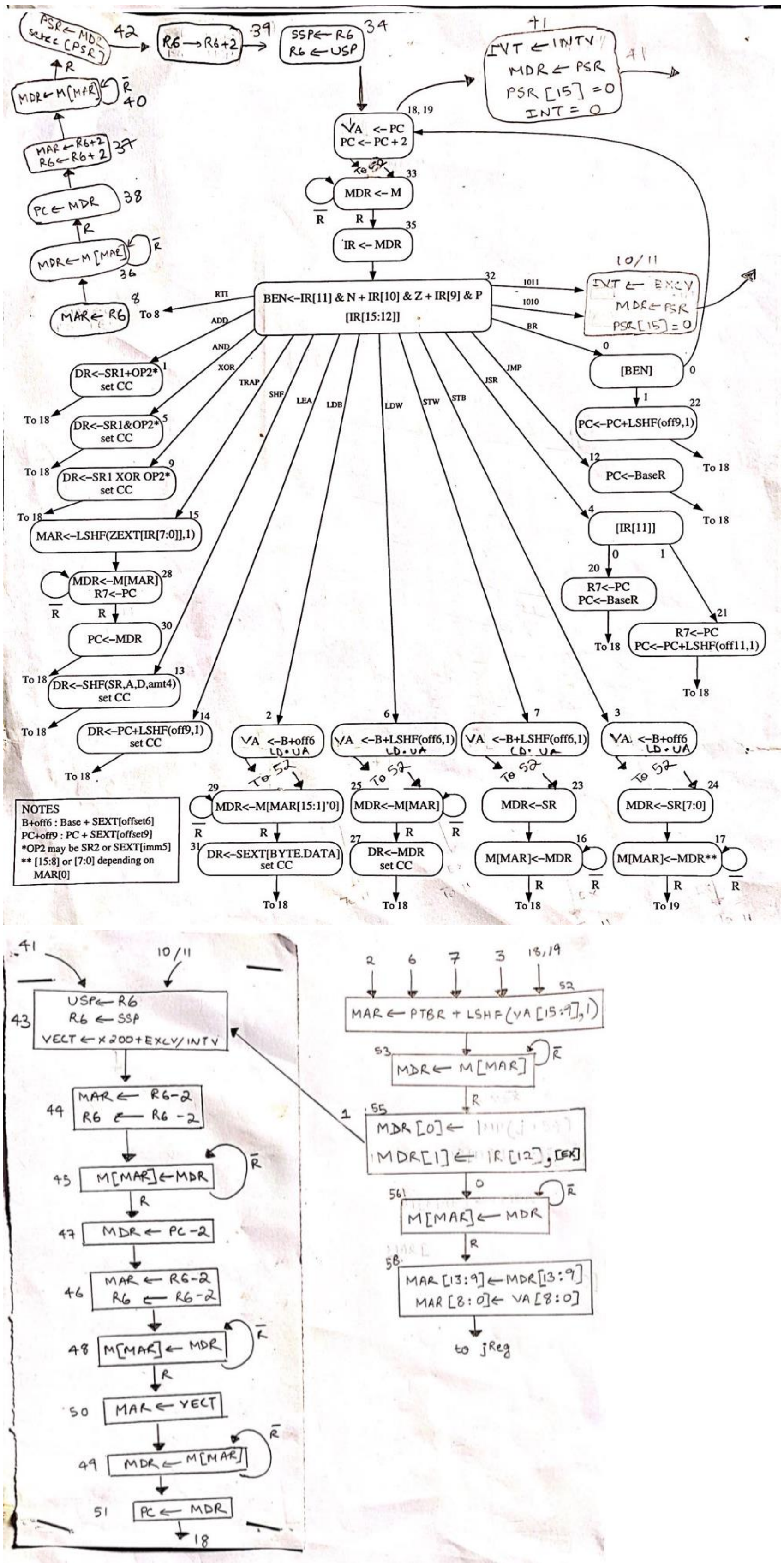
INT enters INT.REG, which stores the interrupt value for the duration of the interrupt handling. An additional condition bit is added, COND2, which goes into an AND gate with INT.OUT, the output of the interrupt signal register. During state 33, COND2 is high. If an interrupt was received, INT.OUT & COND2 = 1, which will make J = 41, which is the initial state for the interrupt handler.

If the instruction is not a TRAP and PROT is high, then PROT.REG is set. If UA is high, then UA.REG is set. If either of these are high, an exception is thrown. The LOGIC block ensures that the exception is checked for only in state 55 for loads and stores. JMP needs to be checked as well. If an exception has occurred, EX is set to 1. EX is an additional select line for the microsequencer MUX. When EX is 1, the next state is 10/11 which is the initial state for the exception handler. **For lab 5, control signal LD.UA is used to load the value of the DATA.SIZE in an earlier state i.e. state 2, 3, 6, 7. Then, the value in the register is ANDed in the logic block with VA[0] once it is calculated and the unaligned access exception is taken during state 55.**

At the end of the interrupt/exception handler, a comparator checks the final state and if it equals 51, it clears all registers, denoting that an interrupt no longer needs to be taken.

For **Lab 5**¸ the following changes were made to accommodate virtual memory. Whenever a virtual address translation needs to be made, the **Translate** signal is asserted so that the next state is State 52, which handles the address translation of what is in the VA register.

To return to the state after virtual memory translation, the next state is always saved in a register JREG unless LD.JREG is asserted. This signal is asserted all through the virtual address translation. Therefore, when the translation is complete, JREG continues to hold the accurate “next” state. To do this, the **JMUX** signal is asserted to load the value in JREG as next state.



PSR ← MDR
setcc [PSR]
42
R6 → R6+2
39
SSP ← R6
R6 ← USP
34
41
INT ← INTV
MDR ← PSR
PSR [15] = 0
INT = 0
41
18, 19
VA <- PC
PC <- PC + 2
To 52
33
MDR <- M
R
35
IR <- MDR
32
BEN<-IR[11] & N + IR[10] & Z + IR[9] & P
[IR[15:12]]
1011
1010
10/11
INT ← EXCV
MDR ← PSR
PSR [15] = 0
BR
0
[BEN]
0
1
22
PC<-PC+LSHF(off9,1)
To 18
12
PC<-BaseR
To 18
4
[IR[11]]
0
1
20
R7<-PC
PC<-BaseR
To 18
21
R7<-PC
PC<-PC+LSHF(off11,1)
To 18
MDR ← M[MAR]
R
40
MAR ← R6+2
R6 ← R6+2
37
PC ← MDR
38
MDR ← M[MAR]
36
MAR ← R6
8
RTI
To 8
ADD
AND
XOR
TRAP
SHF
LEA
LDB
LDW
STW
STB
JSR
JMP
1
DR<-SR1+OP2*
set CC
To 18
5
DR<-SR1&OP2*
set CC
To 18
9
DR<-SR1 XOR OP2*
set CC
To 18
15
MAR<-LSHF(ZEXT[IR[7:0]],1)
28
MDR<-M[MAR]
R7<-PC
R
30
PC<-MDR
To 18
13
DR<-SHF(SR,A,D,amt4)
set CC
To 18
14
DR<-PC+LSHF(off9,1)
set CC
To 18
2
VA <-B+off6
LD•UA
To 52
29
MDR<-M[MAR[15:1]'0]
R
31
DR<-SEXT[BYTE.DATA]
set CC
To 18
6
VA <-B+LSHF(off6,1)
LD•UA
To 52
25
MDR<-M[MAR]
R
27
DR<-MDR
set CC
To 18
7
VA <-B+LSHF(off6,1)
LD•UA
To 52
23
MDR<-SR
16
M[MAR]<-MDR
R
To 18
3
VA <-B+off6
LD•UA
To 52
24
MDR<-SR[7:0]
17
M[MAR]<-MDR**
R
To 19
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]
41
10/11
43
USP ← R6
R6 ← SSP
VECT ← x200+EXCV/INTV
44
MAR ← R6-2
R6 ← R6-2
45
M[MAR] ← MDR
R
47
MDR ← PC-2
46
MAR ← R6-2
R6 ← R6-2
48
M[MAR] ← MDR
R
50
MAR ← VECT
49
MDR ← M[MAR]
R
51
PC ← MDR
18
2
6
7
3
18,19
52
MAR ← PTBR + LSHF(VA[15:9],1)
53
MDR ← M[MAR]
R
1
55
MDR[0] ← 1
MDR[1] ← IR[12], [EX]
0
56
M[MAR] ← MDR
R
58
MAR[13:9] ← MDR[13:9]
MAR[8:0] ← VA[8:0]
to jReg

The state machine was modified in the following ways:

At state 33, if INT = 1, it branches to state 41 instead. Interrupt vector is loaded into the IVT register, PSR is loaded into MDR, MODE is set to 0 for supervisor.

If the OPCODE is an unknown opcode, it automatically goes to states 10/11 which are the exception handler initial state.

Both the interrupt and the exception handler go to state 43. From there, it follows the interrupt guidelines from the lab document.

The RTI instruction was implemented for OPCODE = 8

For lab 5, the following state changes were made. States 2, 3, 6, 7, 18, and 19 all lead to virtual memory translation rather than direct access. PTE is loaded into MDR, reference and modify bits are set, and the physical address is stored in the MAR. The state machine then goes to the state stored in the JREG, which would send it to states 33, 29, 25, 23, or 24 depending on the memory access.

Exceptions are taken after values are loaded into the PTE, in state 55. The unaligned exception is partially calculated before and stored in the UA register, where logic checks for the exception in state 55.