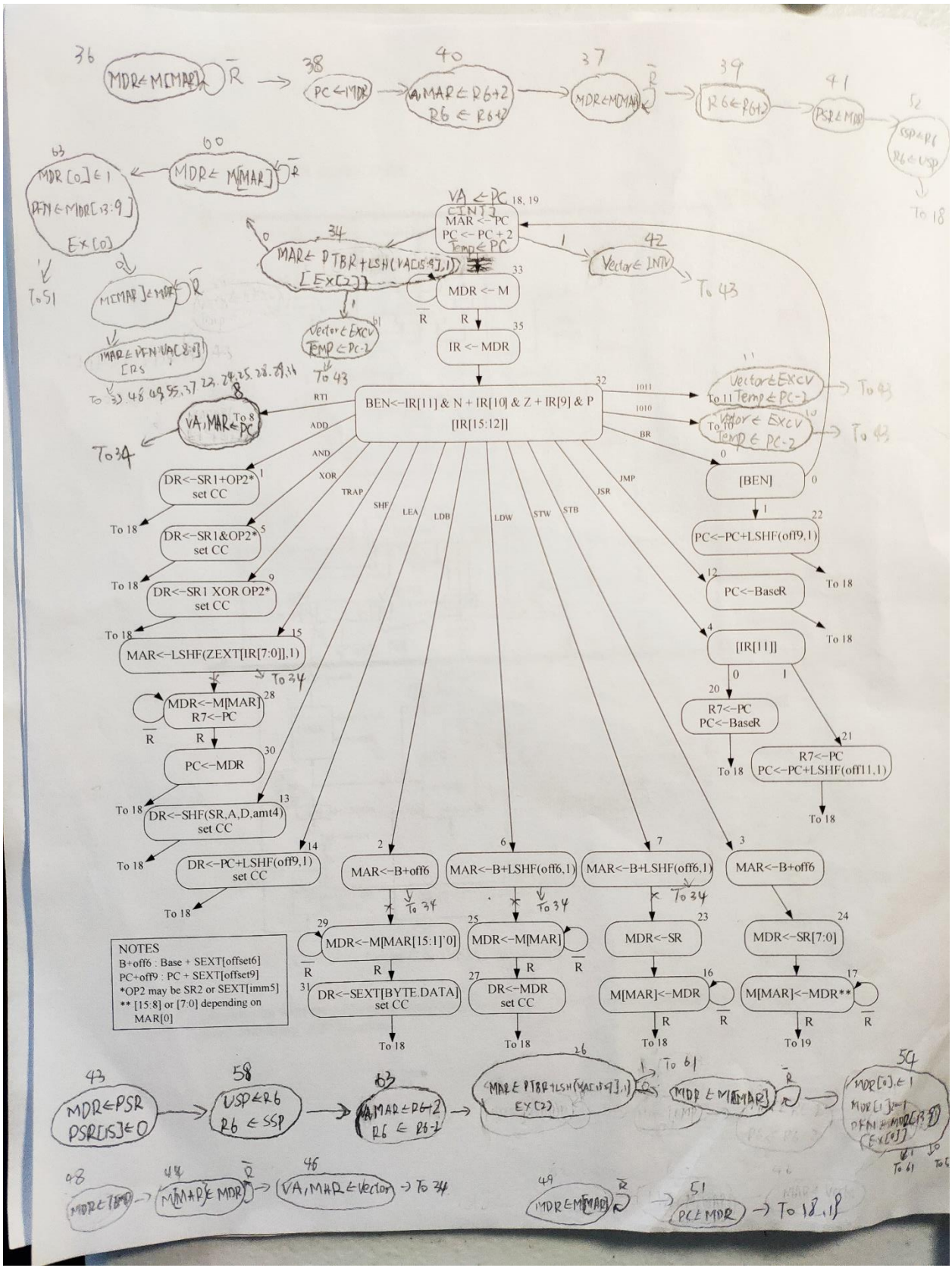
BEN<-IR[11] & N + IR[10] & Z + IR[9] & P
[IR[15:12]]
RTI
ADD
AND
XOR
TRAP
SHF
LEA
LDB
LDW
STW
STB
JSR
JMP
BR
1011
1010
DR<-SR1+OP2*
set CC
DR<-SR1&OP2*
set CC
DR<-SR1 XOR OP2*
set CC
MAR<-LSHF(ZEXT[IR[7:0]],1)
MDR<-M[MAR]
R7<-PC
PC<-MDR
DR<-SHF(SR,A,D,amt4)
set CC
DR<-PC+LSHF(off9,1)
set CC
MAR<-B+off6
MDR<-M[MAR[15:1]'0]
DR<-SEXT[BYTE.DATA]
set CC
MAR<-B+LSHF(off6,1)
MDR<-M[MAR]
DR<-MDR
set CC
MDR<-SR
M[MAR]<-MDR
MAR<-B+off6
MDR<-SR[7:0]
M[MAR]<-MDR**
[BEN]
PC<-PC+LSHF(off9,1)
PC<-BaseR
[IR[11]]
R7<-PC
PC<-BaseR
R7<-PC
PC<-PC+LSHF(off11,1)
Vector<-EXCV
Temp<-PC-2
VA<-PC
[INT]
MAR<-PC
PC<-PC+2
Temp<-PC
Vector<-INTV
MDR<-M
IR<-MDR
VA,MAR<-PC
MAR<-PTBR+LSHF(VAC[15:9],1)
[EX[2]]
MDR[0]<-1
PSR[15]<-0
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]

The memory translation is achieved by states starting from 26 and 34. Loads start at state 34 and store begin at state 26. They are separated because the modify bit is only set when there is a store. The unaligned access exception is checked along with the load of address of PTE. The page frame number is saved to obtain the physical address of data. Then it goes back to the normal states to complete the memory access with actual address.

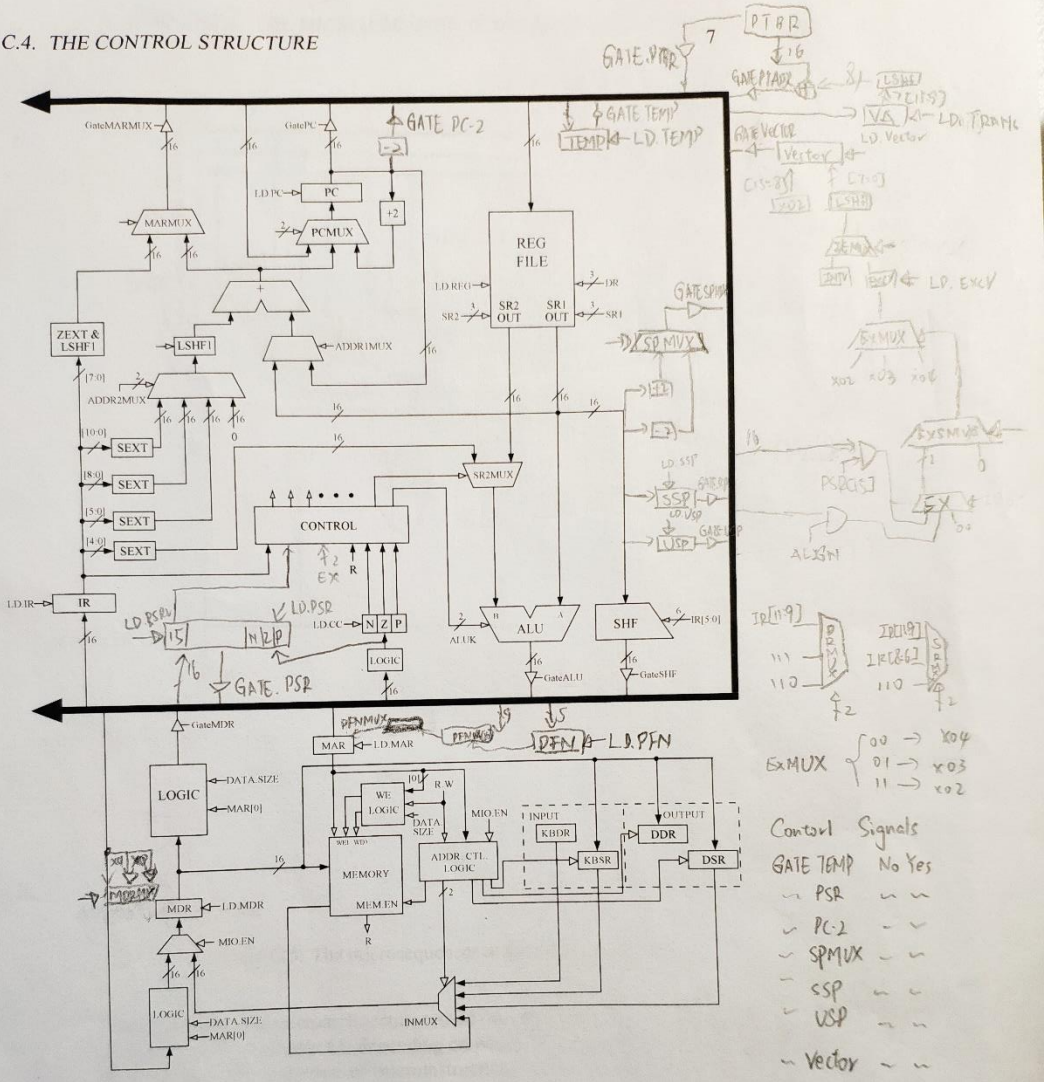

Figure C.3: The LC-3b data path

provide you with the additional flexibility of more states, so we have selected a control store consisting of 2^6 locations.

Several new control signals and registers have been added to achieve address translation. PTBR is used as page table base register and it is controlled by GatePTBR. VA stores the virtual address and GATEVA drives the bus with the address stored in VA register. PFN is used to store the page frame number once it is gotten from memory and it is loaded with LD.PFN. When PFNMUX is set, MAR[8:0] is loaded from the bus with VA[8:0] and the higher bits MAR[13:9] is loaded with the PFN.

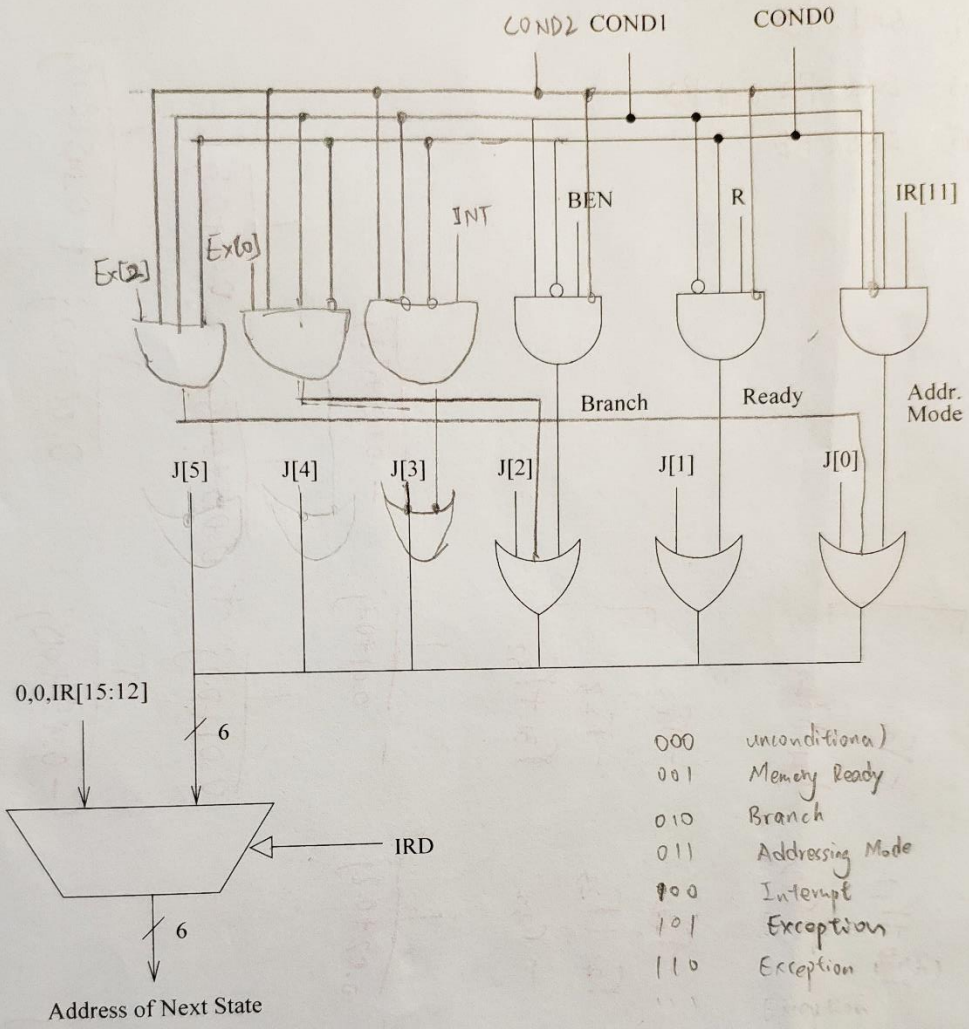

Figure C.5: The microsequencer of the LC-3b base machine

unused opcodes, the microarchitecture would execute a sequence of microinstructions, starting at state 10 or state 11, depending on which illegal opcode was being decoded. In both cases, the sequence of microinstructions would respond to the fact that an instruction with an illegal opcode had been fetched.

Several signals necessary to control the data path and the microsequencer are not among those listed in Tables C.1 and C.2. They are DR, SR1, BEN, and R. Figure C.6 shows the additional logic needed to generate DR, SR1, and BEN.

The remaining signal, R, is a signal generated by the memory in order to allow the

In this design, we have more conditions than before. Only 2 condition bits are not enough so we have to add one more condition bit. The six conditions are shown in the picture.