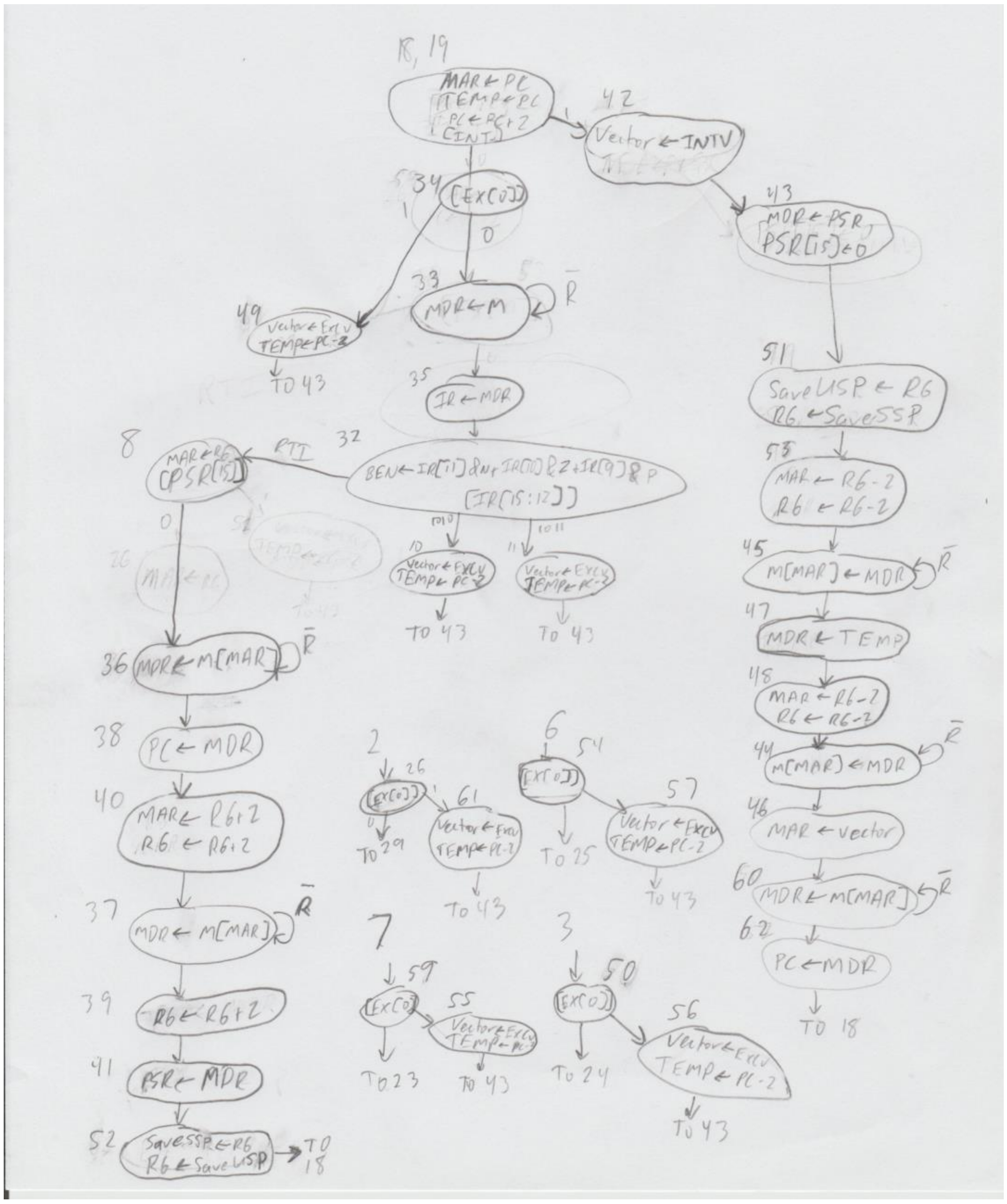
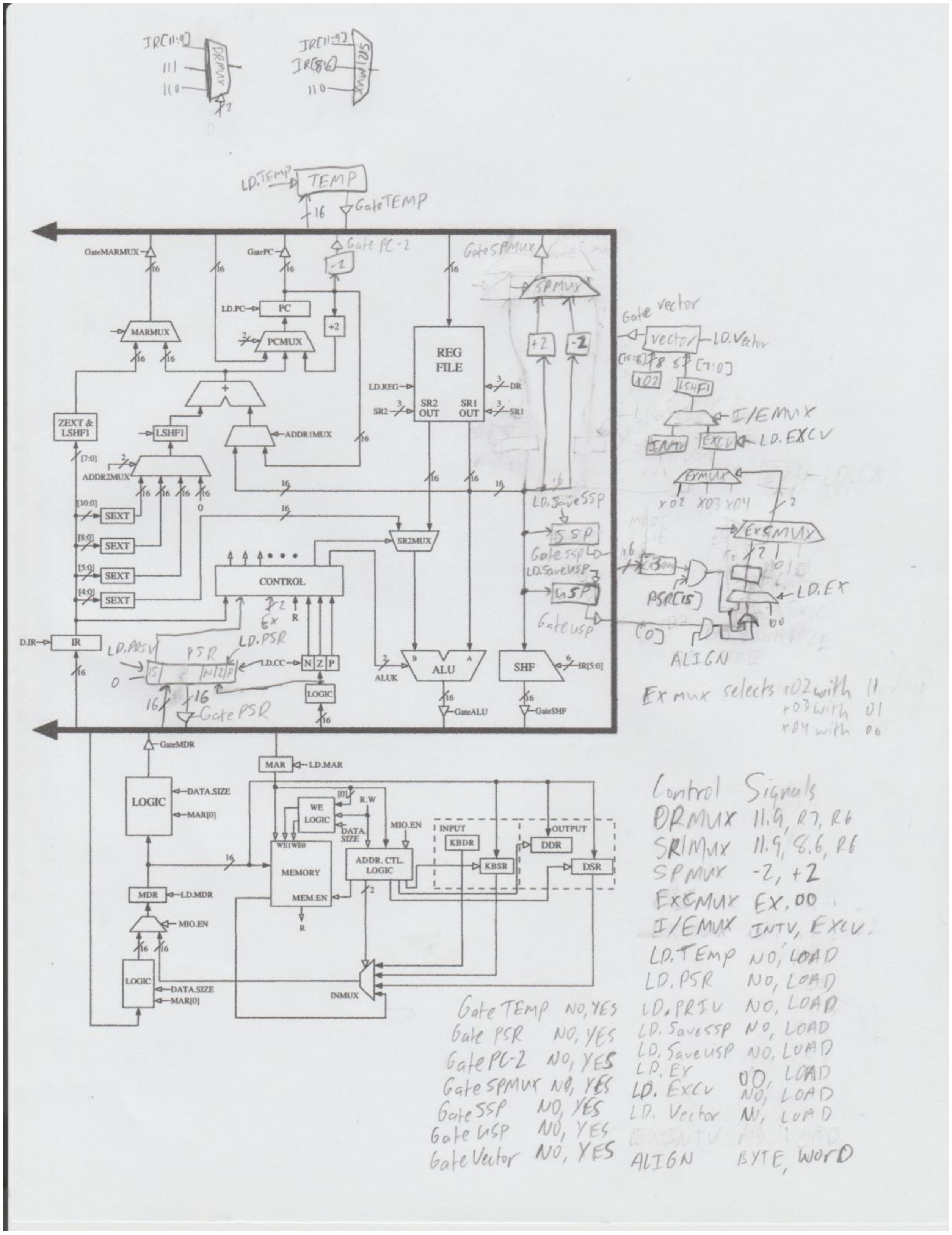
Noah Kessler



States 8-52 on the left implement the RTI instruction. They pop PC and PSR from the supervisor stack and switch R6 back to the user stack pointer. States 43-62 on the right are called when going to deal with an interrupt or exception. They save the user stack pointer, push PC and PSR onto the supervisor stack and change the privilege to supervisor mode. Then they load the appropriate vector into MAR based on the exception or interrupt and load PC with the starting address of the handler. States 18 and 19 now also store the old PC in a temp register and branch based on INT, which is set for an interrupt. States 34, 26, 50, 54 and 59 all check for exceptions and branch based on that. States 42, 49, 10, 11, 55, 56, 57 and 61 all load the vector for the interrupt/exception into a register, and all but 42 also store PC-2 in a register.



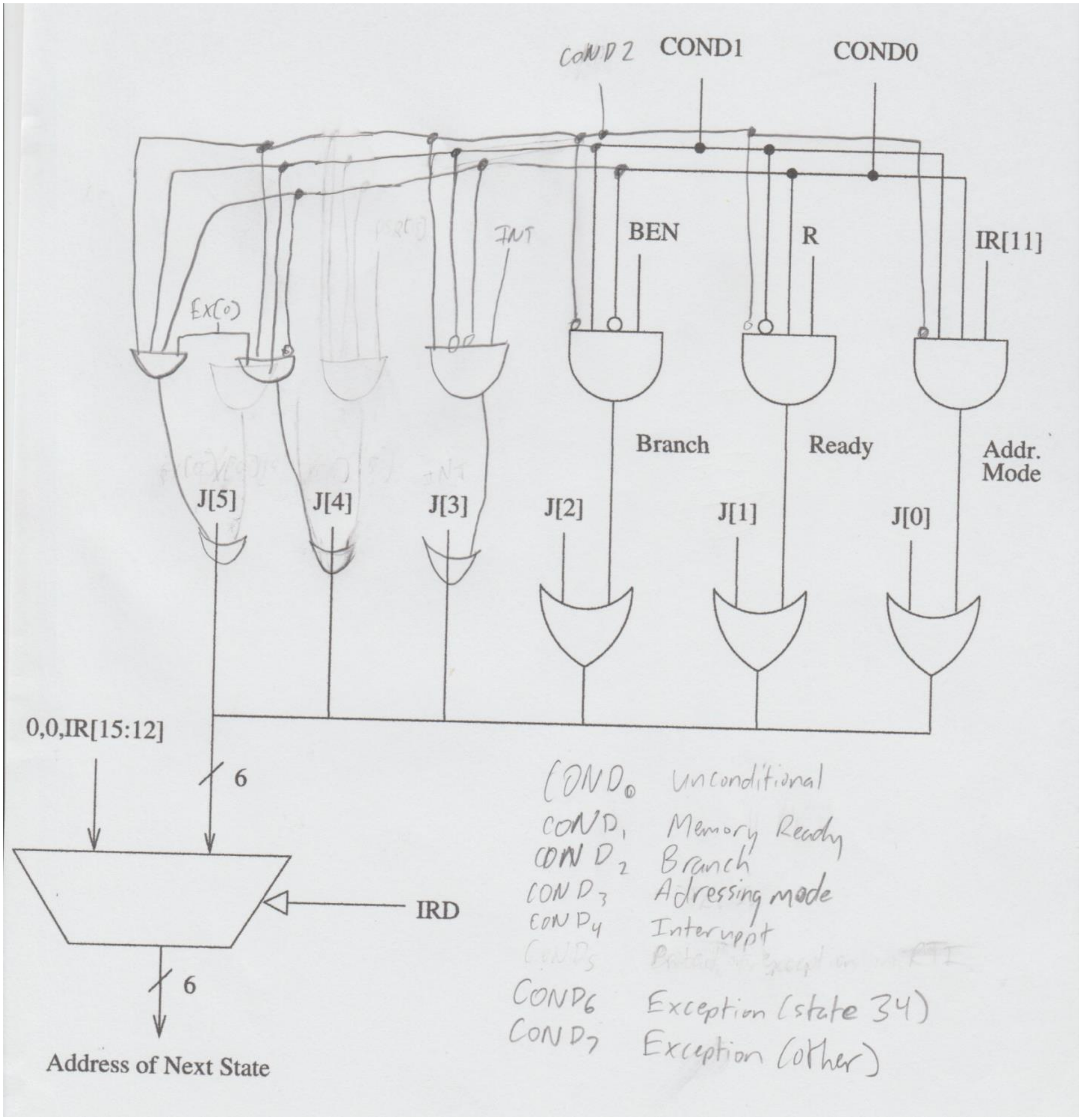
Ex mux selects x02 with 11
x03 with 01
x04 with 00

Control Signals

DRMUX	11.9, R7, R6
SR1MUX	11.9, 8.6, R6
SPMUX	-2, +2
ExcMUX	EX, 00
I/EMUX	INTV, EXCV
LD.TEMP	NO, LOAD
LD.PSR	NO, LOAD
LD.PRIV	NO, LOAD
LD.SaveSSP	NO, LOAD
LD.SaveUSP	NO, LOAD
LD.EX	00, LOAD
LD.EXCV	NO, LOAD
LD.Vector	NO, LOAD
ALIGN	BYTE, WORD

GateTEMP	NO, YES
Gate PSR	NO, YES
GatePC-2	NO, YES
GateSPMUX	NO, YES
GateSSP	NO, YES
GateUSP	NO, YES
GateVector	NO, YES

DRMUX and SRMUX both have R6 added as an input, and so both have one extra control bit. TEMP is a register which loads from the bus and can drive the bus, and it is used to store PC before it is pushed onto the stack when going to a handler. LD.TEMP loads it from the bus and GateTEMP drives the bus. SSP and USP are both registers which load from SR1 from the register file and can drive the bus. SSP stores the supervisor stack pointer and USP stores the user stack pointer. LD.SSP and LD.USP load the registers and GateSSP and GateUSP drive the bus. The PSR register stores the PSR. It can be loaded from the bus with LD.PSR and drive the bus with GatePSR. PSR[15] can be loaded with 0 directly with LD.PRIV. PSR[2:0] store the condition codes and are connected to the input of the condition code register. If LD.CC is set then the condition code register and PSR[2:0] are both loaded. If LD.PSR is set, then the condition codes are updated to match PSR[2:0]. SPMUX takes an input from SR1 of the register file and selects SR1+2 or SR1-2. GateSPMUX drives the bus with the output. The rest of the additional hardware is used to detect exceptions and load the correct vector for interrupts and exceptions. EX is a 2 bit register which is set by an exception. It is also used as a control to determine when to branch. EX is loaded with 0 when LD.EX is 0. When LD.EX is 1, EX[0] is set by taking bit 0 from the data on the bus AND ALIGN, which is set when MAR is being loaded with an address for a word of data, causing an unaligned access exception. EX[1:0] are both set if data from the bus, MAR, < 3000 and PSR[15] = 1, which is a protection exception. Protection sets both bits so that detecting an exception can be done with one bit, EX[0]. EXCMUX selects 0 when the control is 1 and EX when the control is 0. This output is the input to EXMUX, which selects the vector based on which interrupt is detected, and the result is stored in EXCV if LD.EXCV is set. INTV is loaded with 0x01. I/EMUX selects between the interrupt and exception vector to be loaded. The output is left shifted one and loaded into Vector[7:0] if LD.Vector is set. Vector[15:8] is loaded with 0x02. If GateVector is set then Vector drives the bus.



COND$_0$	Unconditional
COND$_1$	Memory Ready
COND$_2$	Branch
COND$_3$	Adressing mode
COND$_4$	Interuppt
COND$_5$	Protection exception or RTI
COND$_6$	Exception (state 34)
COND$_7$	Exception (other)

An additional condition bit is needed to select between 6 different conditions. Conditions 0, 1, 2 and 3 are the same as before. COND4 branches if the INT bit is set after an interrupt is detected. COND6 and COND7 both branch if EX[0] is set because an exception was detected. This needs 2 different conditions because of the state numbering. State 34 uses COND6 because it must come before state 35, which already has bit 5 of its address set. The rest of the states branching on EX[0] use COND7.