EE 213
3/2/2014

JNC, JC, and JNZ Instructions for the WIMP51

For the beginning of the project I looked up the Hex code for the JNC, JC, JNZ, as well as JZ so that I could compare with how it was created with the other jump instructions needed. The Hex codes, as well as the binary codes, for each new instruction are on the 'Code for Testing' page.

The next step was to go over each write enable: ACC_WE, AUX_WE, IR_WE, REG_IN, REG_EN, C_WE, and PC_WE. With each one I checked to see if the code for the instructions needed to be off or on for Fetch , Decode, or Execute. After checking my code with all the write enables I found that my code was active or inactive as it should be for each write enable except for PC_WE and ACC_WE.

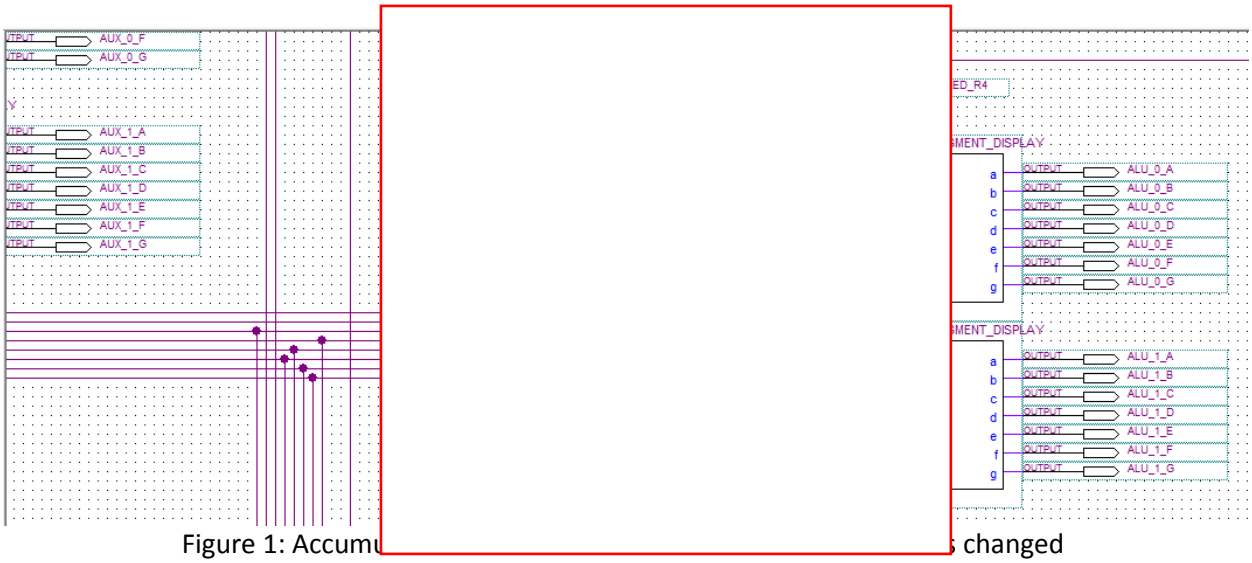


Figure 1: Accumu s changed

For the accumulator write ena Fetch, Decode, and Execute. However, the accumulator u eded rather than using code to show what is needed. E he accumulator so I had to add each one. In Figure 2 it sh with two AND gates combined through an OR gate. Figure e already created JZ code.

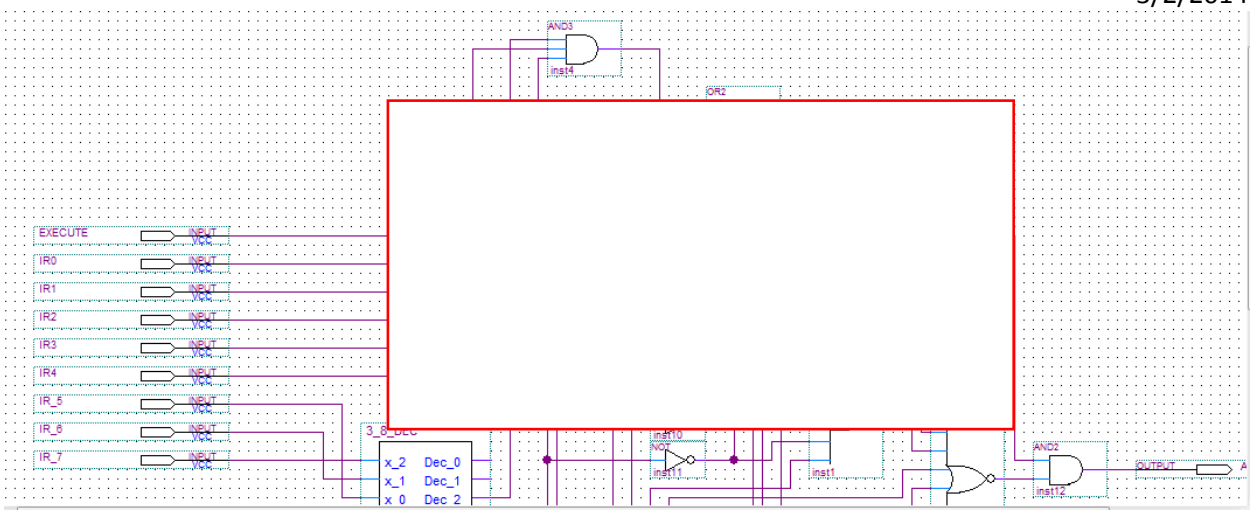


Figure 2: Instructions added to the ACC_WE JNC and JC

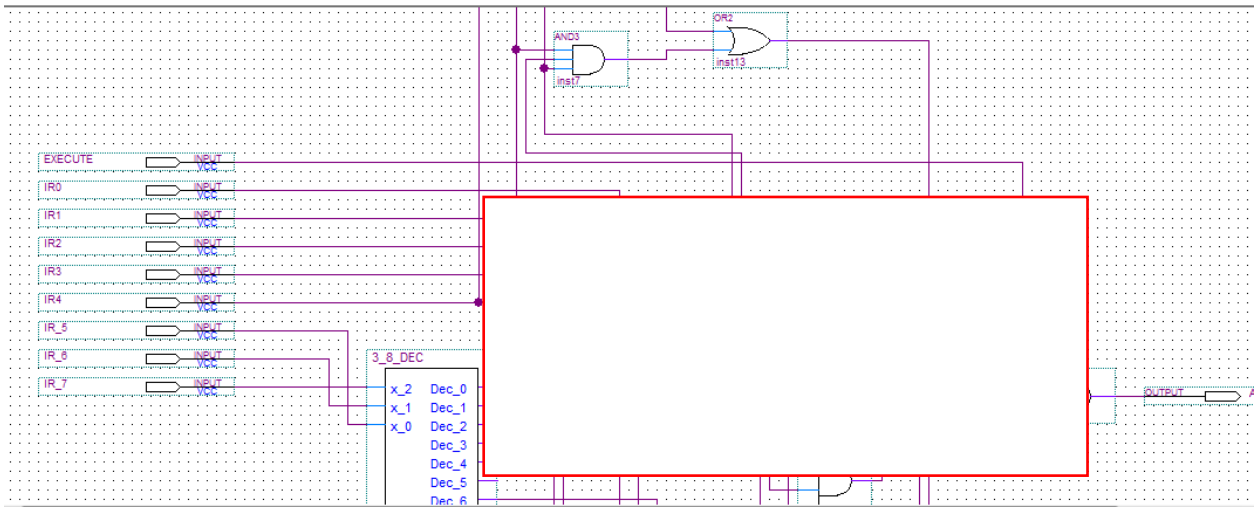


Figure 3: JNZ added to eh ACC_WE

The next change was to the PC_WE. After creating the new code for the PC_WE I had to create a new input for IR2. With that I had to make a new PC_WE and bring it into the program and wire IR2 to it from the ALU.

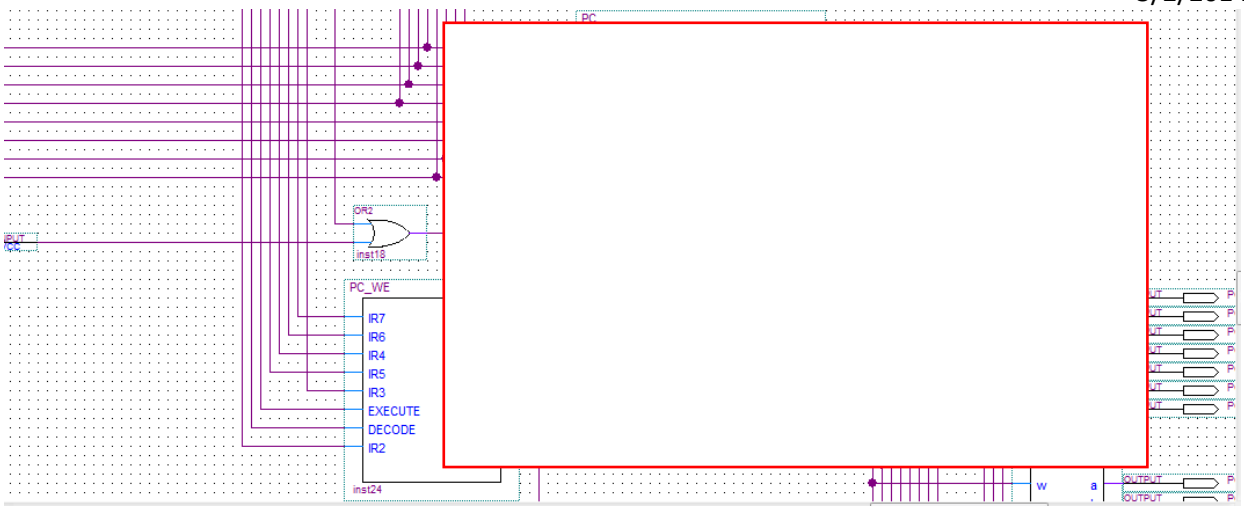


Figure 4: Program Counter Write Enable

For the PC_WE all three instructions had to be added. For the instruction JNZ to differentiate it from other instructions I had to bring in IR2 which is shown in Figure 6. JNC and JC are shown at the top of Figure 5 combine into two AND gates moved into an OR. JNZ is shown in the middle branching off of the already created JZ instruction.

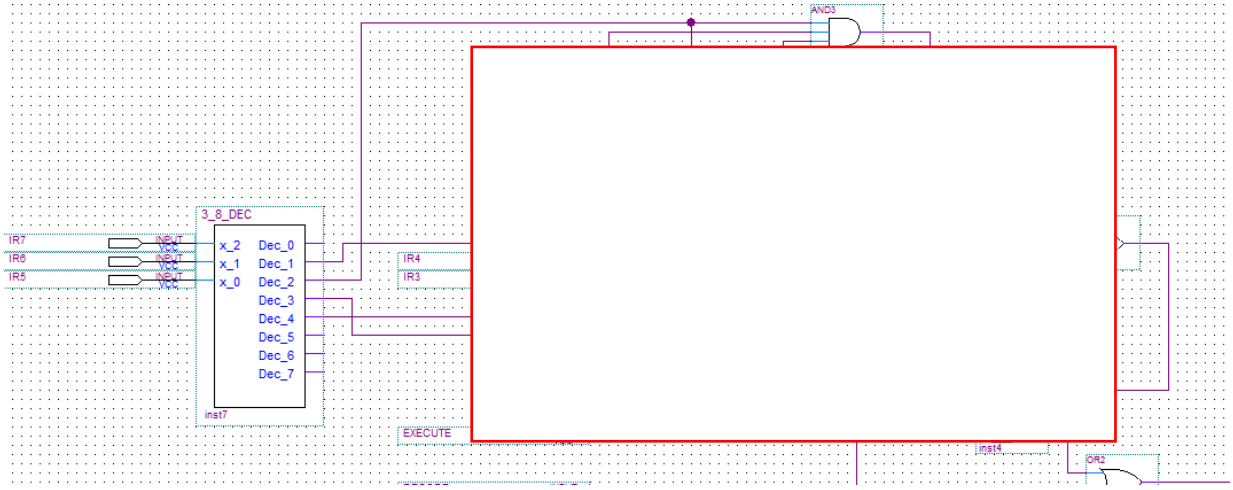


Figure 5: JNC, JC, and JNZ added at the top and middle

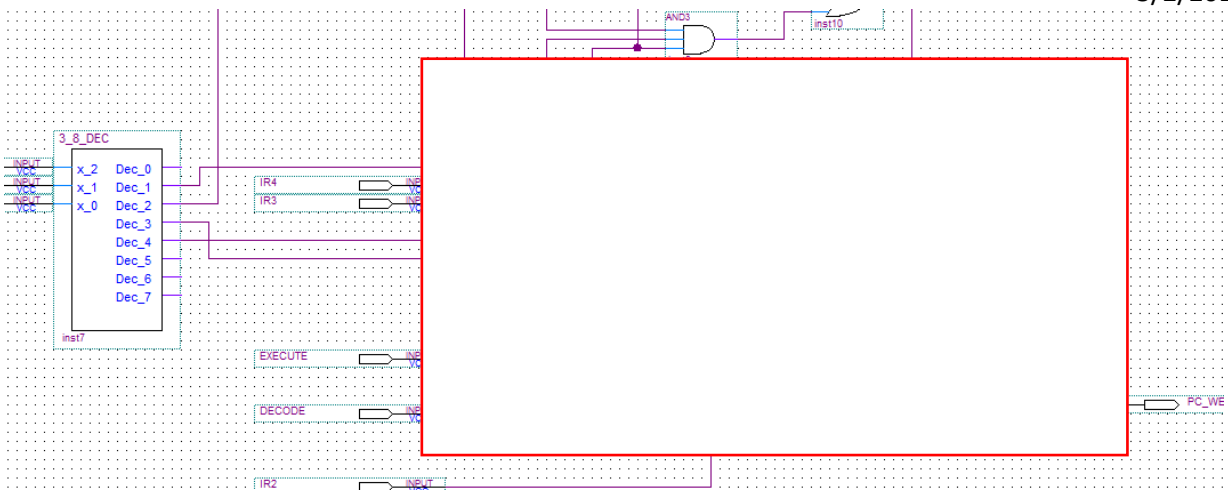


Figure 6: Shows the IR2 input added to the program

The jump instruction that already existed was located in the PC_ALU and that was where the rest of the new jump instructions would be added. Figure 7 below shows the new PC_ALU block that had to be input into the program after I made the changes for the carry bit to be brought in. The input for the carry bit was called COUT and rewired for the new block.

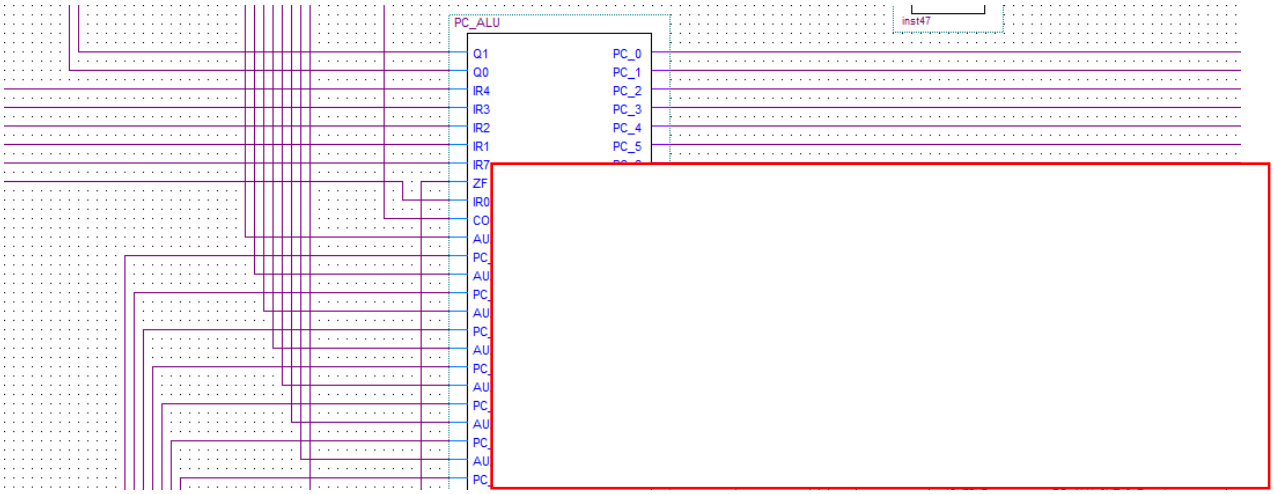


Figure 7: new PC_ALU block added showing COUT

For the PC_ALU the only jump instruction was originally JZ. For JZ it looked into a zero coming for a jump and so for JNZ I added a NOT to the zero. Figure 11 shows that for the JC and JNC I added a carry bit coming into the same OR as the zero commands were. Figure 9 shows that what I added to the PC_ALU was for the JZ and JNZ instructions to use IR4 and IR3 put through an AND gate as 00 respectively, and for JC and JNC they were each put through an AND gate as well as 10 respectively. After it is decided whether 00 or 10 is going through then it decides whether zero or carry is active or not.

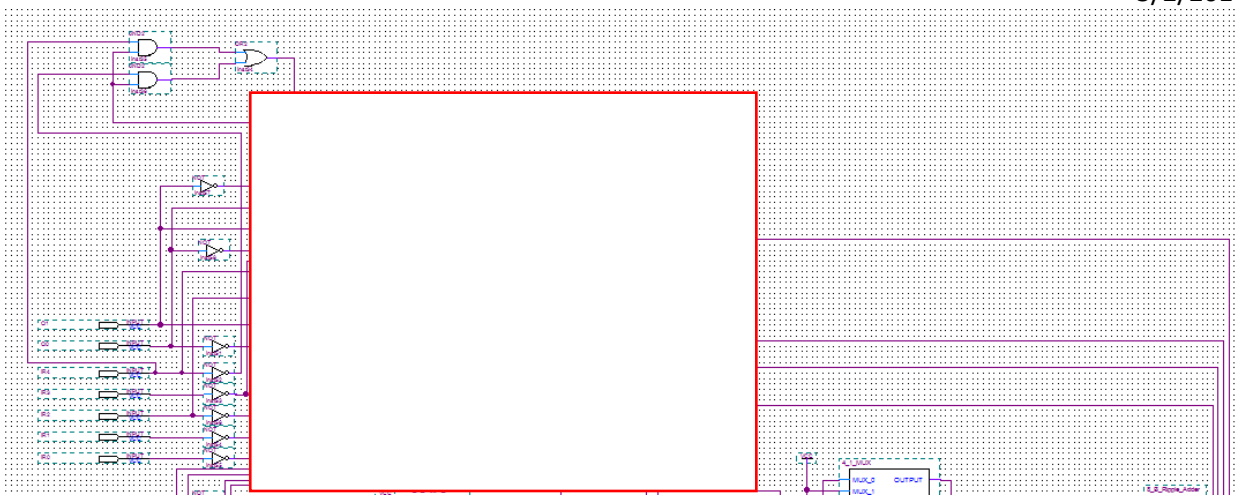
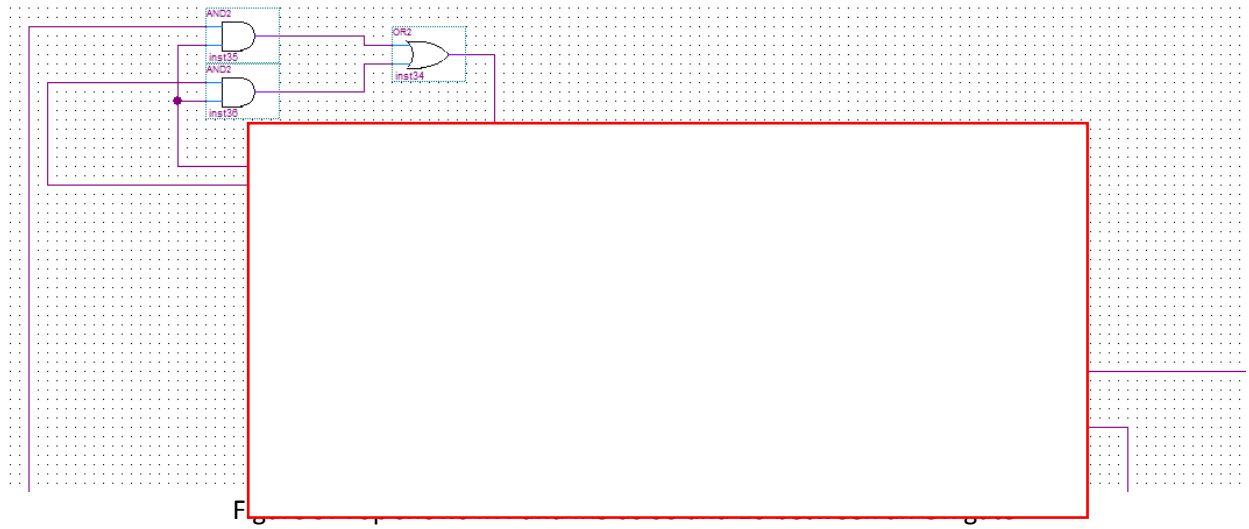


Figure 8: Zoomed out look to show most of the changes in the PC_ALU



F

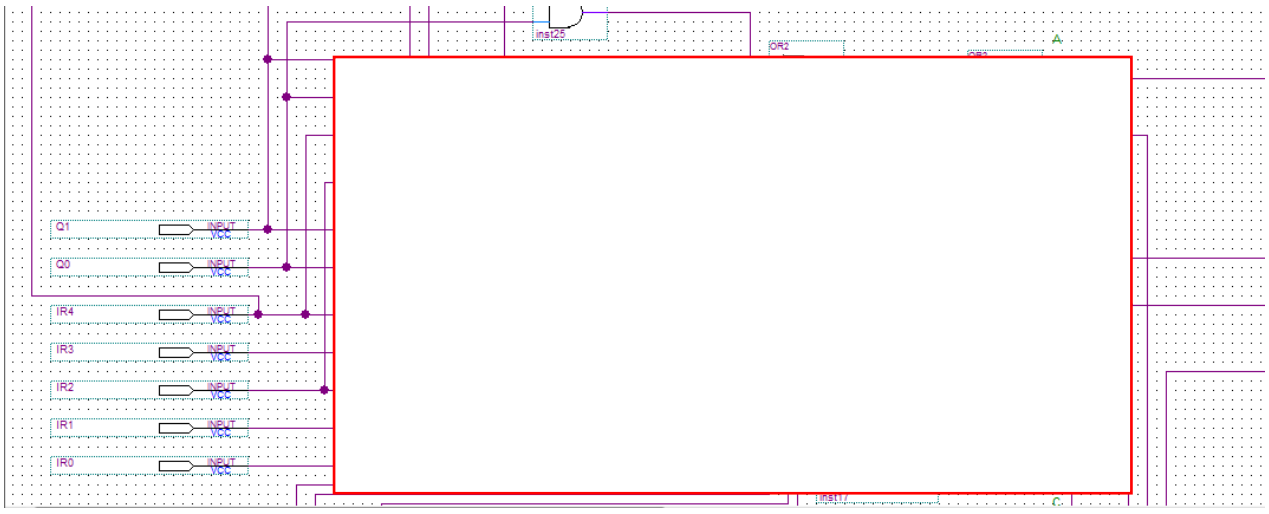


Figure 10: Shows the connection for IR4 and IR3

Figure 11: Shows the connection for the carry and zero bit

All the new instructions and previous instructions were tested with the code in the next section. Each test was ran until ended by a SJMP which would cycle through the last command. Every jump operated as needed and would proceed or not proceed depending on what the circumstance called for.

Code for Testing

JNC: Hex - 50 Binary - 01010000

JNZ: Hex - 70 Binary - 01110000

JC: Hex - 40 Binary - 01000000

	Instruction Set:	Address:	Machine Code:	
	MOV A, #05H	00	74	
		01	05	A = #05H
	JNZ N_1	02	70	
		03	05	jump to 09
N_2	SETB C	04	D3	CY = 1
	JC N_3	05	40	
		06	0A	jump to 11
	MOV A,#00H	07	74	
		08	00	A = #00H
N_1	JNZ N_2	09	70	
		0A	F9	jump to 04
N_5	CLR C	0B	C3	CY = 0
	JNC N_6	0C	50	
		0D	0A	jump to 18
N_4	SETB C	0E	D3	CY = 1
	JC N_5	0F	40	
		10	FA	jump to 0B
N_3	CLR C	11	C3	CY = 0
	MOV A,#00H	12	74	
		13	00	A = #00H
	ADD A,#79H	14	34	
		15	79	A = #79H
	JNC N_4	16	50	
		17	F6	jump to 0E
N_6/				
BACK	SJMP BACK	18	80	
		19	FE	