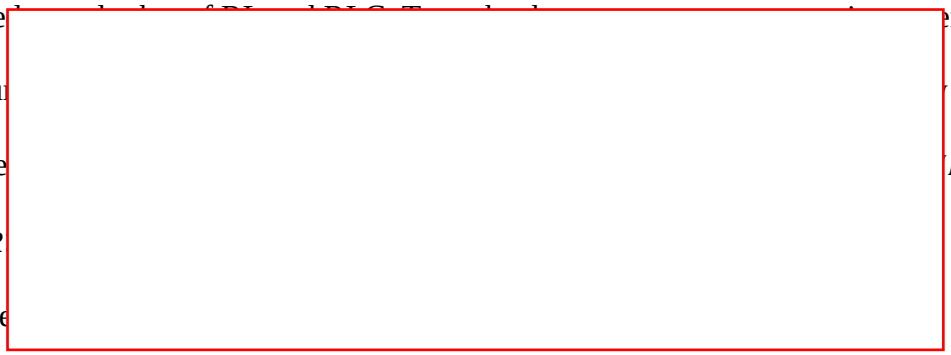
Project 1 Report: WIMP51 "The Extended" Version

The goal of this project was to increase the functionality of a program named WIMP51. The original WIMP51 is a program that simulates thirteen different functions of an 805l microprocessor. Table 1 in appendix A lists those functions. The extended version of WIMP51 has 5 extra functions that manipulate the accumulator: rotate right (RR), rotate right through carry (RRC), rotate left (RL), rotate left thought carry (RLC), and Compliment (CPL). In order to achieve this goal changes had to be made to the AUX write enable, the PC write enable, and the ALU.

First, the AUX write enable, which previously excluded only SWAP A, CLR C, and SETB C during the execute or decode cycles, was changed to also exclude RR, RRC, RL, RLC, and CPL

gister file, and he AUX, w $\cdot \overline{\overline{IR3}}) \cdot (\overline{\overline{\overline{IR7} \cdot \overline{IR}}}$.

Next the PC write enable needed to be altered. This is because the simplified code of the ADDC matche ent twice in one cyle, du codes. The PC write e $IR4) \cdot (\overline{\overline{IR3}} \cdot IR2 + IR$. This can be se



Next, The major changes came in the ALU. The LA select of the ALU needed to have a larger pr r was increase used to control v lled CPL. Th the updated LA select.

The next part in the ALU to change was the 4 to1 MUXs. More funtionality means bigger M s. Figure 4 shows t uted through a not gate EL block needed t oded as 1 at line I_6. nplimented accumu he accumu ng bits are roatated e 8:1 MUXs is the line h the L_A_SE n shares this opcode, these bits are all that is required to set I_4 of the priority encoder high, and select MUX 4. For the instructions RL, and RLC MUX 5 is the line that connects the bits rotating left. Inside the L_A_SEL the Opcode 001 is decoded and set to I_3 since this is also the Opcode for the instructions RL, and RLC this can be anded with not IR2, and $\overline{IR3}$. So that I_5 will not go high when the ADDC instructions are performed. The rest of the L_A_SEL block can remain the same for I_3, I_2, and I_1. Since the MUX lines that they select have not been changed.

Original[redacted]ing the Carry bit was that the data rotated into the [redacted]ver the Carry was rotated into. When the RLC instrcti[redacted]ccumulator MSB 0, and LSB 0, during the execute cycl[redacted]tead of getting a 1 at LSB, a 0 would pass into carry a[redacted]positive edge triggered D flip flop was placed after the [redacted]rotate occurs, and does not update when the Carry [redacted]

A 4:1 M[redacted]riority encoder was used for the select lines. The input to the mux 0,1, 2, and 3 line is Cout from the adder, and either clear or set C for I_1, th[redacted]active then th[redacted]opcode is the s[redacted]en RL is perf[redacted] instruc[redacted] and the car[redacted]ction 0011 0[redacted]uts to the Ca[redacted]opcode 000 wh[redacted]Flag.

Table 1 – Insctuction set for WIMP51 Extended

MOV	A,#D	01110100	dddddddd
ADDC	A,#D	00110100	dddddddd
MOV	Rn,A	11111nnn	
MOV	A,Rn	11101nnn	
ADDC	A,Rn	00111nnn	
ORL	A,Rn	01001nnn	
ANL	A,Rn	01011nnn	
XRL	A,Rn	01101nnn	
SWAP	A	11000100	
CLR	C	11000011	
SETB	C	11010011	
SJMP	Rel	10000000	dddddddd
JZ	Rel	01100000	dddddddd
RR	A	00000011	
RRC	A	00010011	
RL	A	00100011	
RLC	A	00110011	
CPL	A	11110100	

Table 2 – Test Program 1 (all but JZ)

PC	Machine code	Instruction	Accumulator
00	74	MOV A,#99H	
01	99		99
02	03	RR A	CC
03	C3	CLRC	
04	34	ADDC #40H	
05	04		D0
06	23	RL A	A1
07	F8	MOV R0,A	
08	38	ADDC A, R0	42
09	F9	MOV R1,A	
0A	13	RRC A	A1
0B	D3	SETC	
0C	33	RLC A	43
0D	E8	MOV A,R0	A1
0E	F4	CPL A	5E
0F	C4	SWAP A	E5
10	49	ORL A,R1	E7
11	58	ANL A,R0	A1
12	68	XRL A,R0	00
13	80	SJMP STOP	
14	FE		

Table 3 – Test Program 2 JZ

PC	Machine Code	Instruction	Accumulator
00	74	MOV A,#00H	
01	00		00
02	60	JZ	
03	03	03	
04	XX	XX	
05	XX	XX	
06	XX	XX	
07	80	SJMP STOP	
08	FE		

Figure 1 – AUX Write Enable

Figure 2 – PC Write Enable

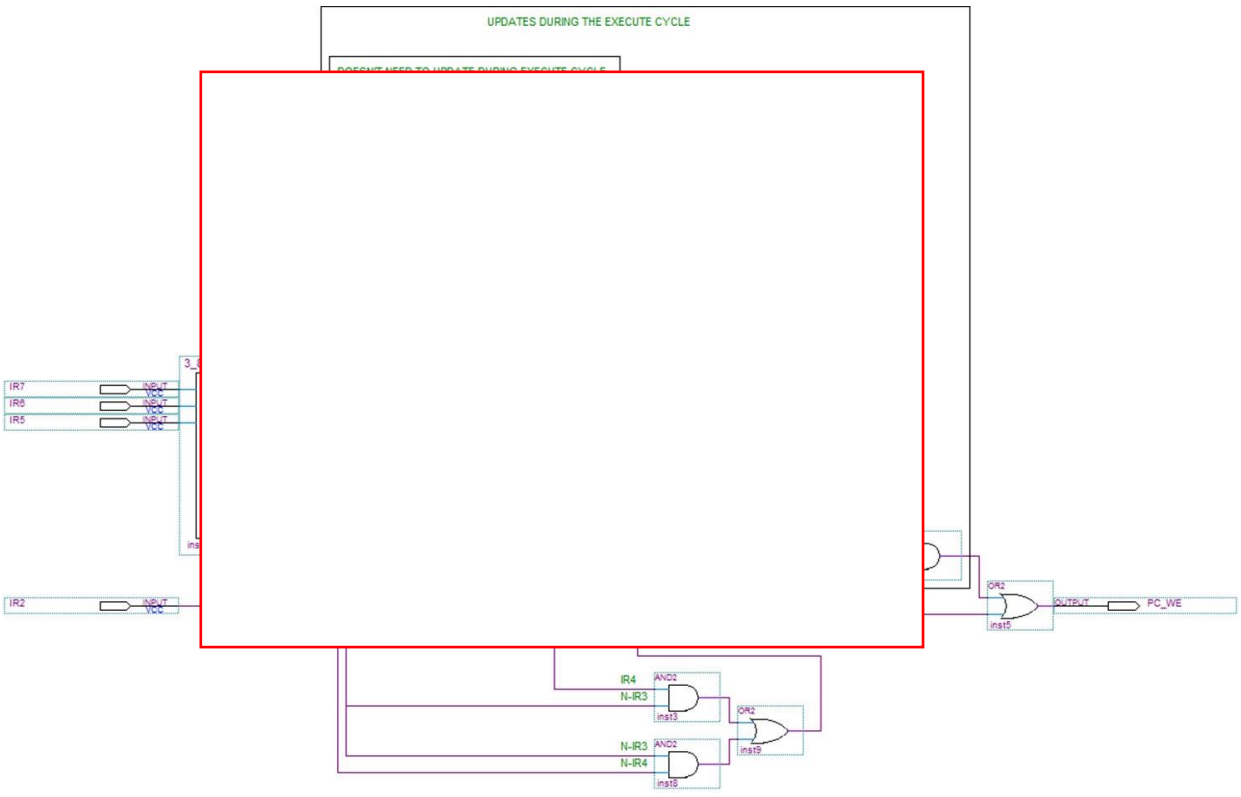


Figure 3 – LA Select

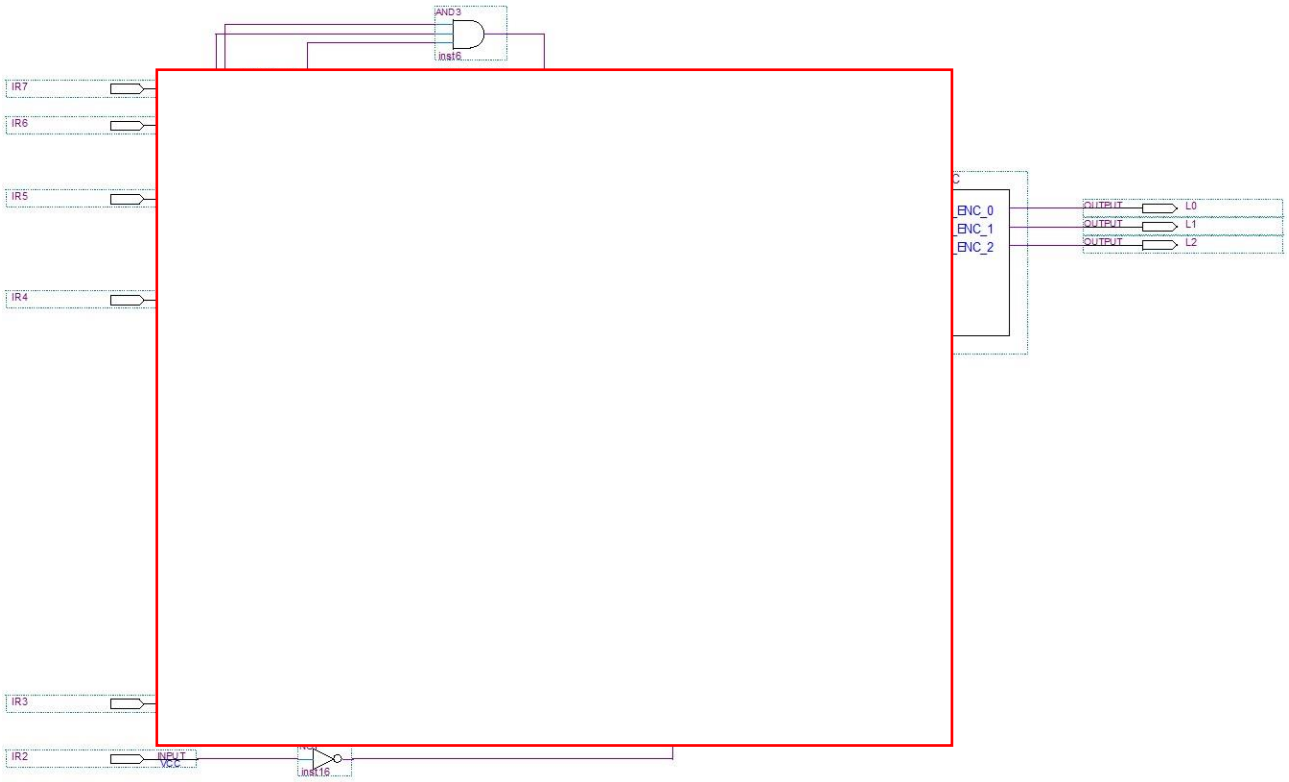


Figure 4 - ALU