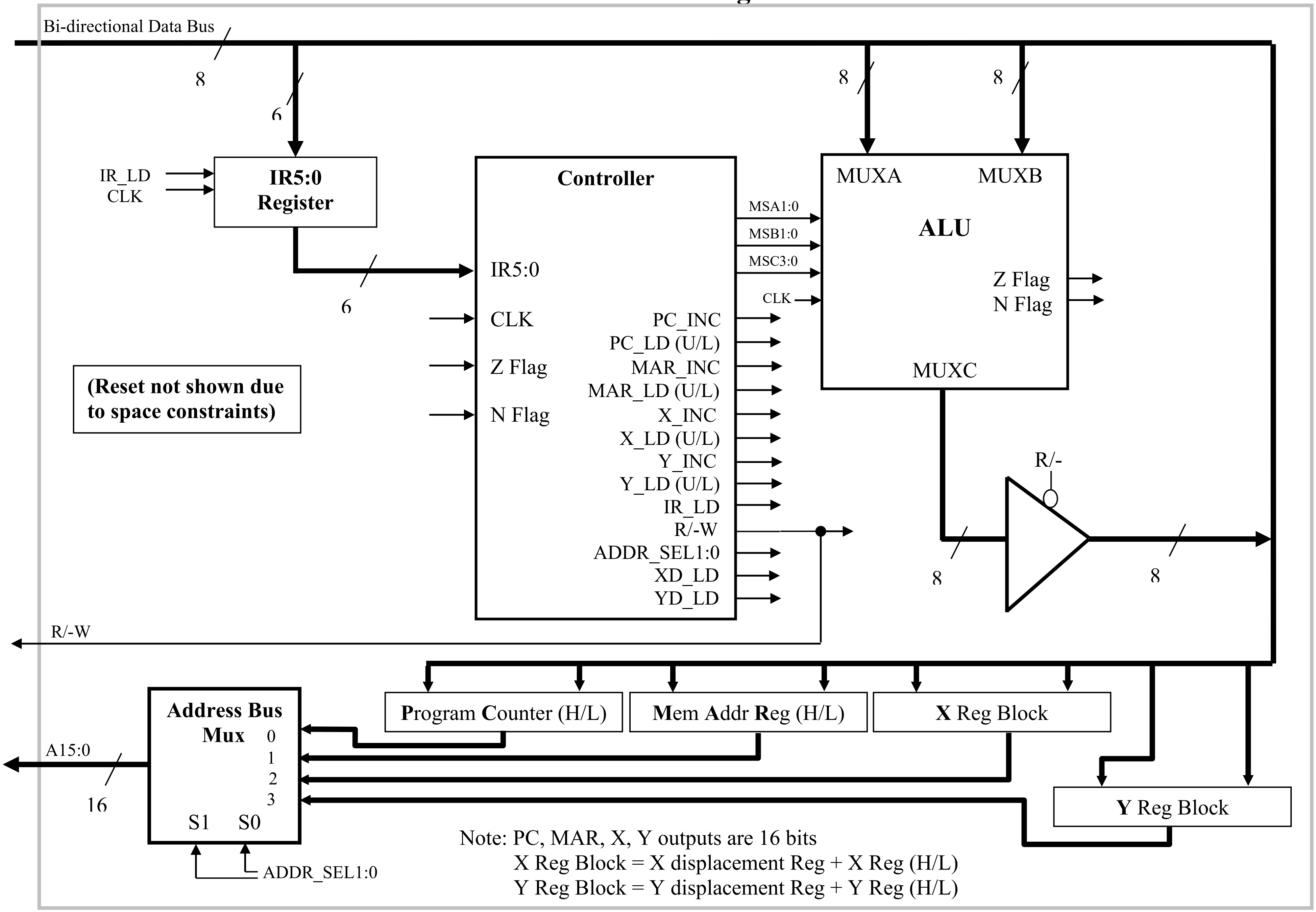
G-CPU Block Diagram



(Reset not shown due to space constraints)

Note: PC, MAR, X, Y outputs are 16 bits
X Reg Block = X displacement Reg + X Reg (H/L)
Y Reg Block = Y displacement Reg + Y Reg (H/L)

G-CPU Instruction Set

Data Movement Instructions:

Machine Codes (hex)	Instruction	Operand	Description	# of States
00	TAB	none	Transfer A to B (inherent addressing)	2
01	TBA	none	Transfer B to A (inherent addressing)	2
02 mm	LDAA #data	8-bit data	Load A with immediate data (immediate addr.)	3
03 mm	LDAB #data	8-bit data	Load B with immediate data (immediate addr.)	3
04 ll hh	LDAA addr	16-bit address	Load A with data from memory location addr (extended addressing)	5
05 ll hh	LDAB addr	16-bit address	Load B with data from memory location addr (extended addressing)	5
06 ll hh	STAA addr	16-bit address	Store data in A to memory location addr (extended addressing)	5
07 ll hh	STAB addr	16-bit address	Store data in B to memory location addr (extended addressing)	5
08 ii jj	LDX #data	16-bit data	Load X with immediate data (immediate addr.)	4
09 ii jj	LDY #data	16-bit data	Load Y with immediate data (immediate addr.)	4
0A ll hh	LDX addr	16-bit addr	Load X with data from memory location addr. (extended addressing)	6
0B ll hh	LDY addr	16-bit addr	Load Y with data from memory location addr. (extended addressing)	6
0C dd	LDAA dd,X	8-bit displacement	Load A with data from memory location pointed to by X + dd (indexed addressing)	4
0D dd	LDAA dd,Y	8-bit displacement	Load A with data from memory location pointed to by Y + dd (indexed addressing)	4
0E dd	LDAB dd,X	8-bit displacement	Load B with data from memory location pointed to by X + dd (indexed addressing)	4
0F dd	LDAB dd,Y	8-bit displacement	Load B with data from memory location pointed to by Y + dd (indexed addressing)	4
10 dd	STAA dd,X	8-bit displacement	Store data in A to memory location pointed to by X + dd (indexed addressing)	4
11 dd	STAA dd,Y	8-bit displacement	Store data in A to memory location pointed to by Y + dd (indexed addressing)	4
12 dd	STAB dd,X	8-bit displacement	Store data in B to memory location pointed to by X + dd (indexed addressing)	4
13 dd	STAB dd,Y	8-bit displacement	Store data in B to memory location pointed to by Y + dd (indexed addressing)	4

G-CPU Instruction Set

ALU Related Instructions:

Machine Codes (hex)	Instruction	Operand	Description	# of States
14	SUM_BA	none	Sum A, B and place in A (inherent addressing)	2
15	SUM_AB	none	Sum A, B and place in B (inherent addressing)	2
16	AND_BA	none	AND A, B and place in A (inherent addressing)	2
17	AND_AB	none	AND A, B and place in B (inherent addressing)	2
18	OR_BA	none	OR A, B and place in A (inherent addressing)	2
19	OR_AB	none	OR A, B and place in B (inherent addressing)	2
1A	COMA	none	Complement contents in A (inherent addressing)	2
1B	COMB	none	Complement contents in B (inherent addressing)	2
1C	SHFA_L	none	Shift A left by one-bit (inherent addressing)	2
1D	SHFA_R	none	Shift A right by one-bit (inherent addressing)	2
1E	SHFB_L	none	Shift B left by one-bit (inherent addressing)	2
1F	SHFB_R	none	Shift B right by one-bit (inherent addressing)	2
30	INX	none	Increment X (inherent addressing)	2
31	INY	none	Increment Y (inherent addressing)	2

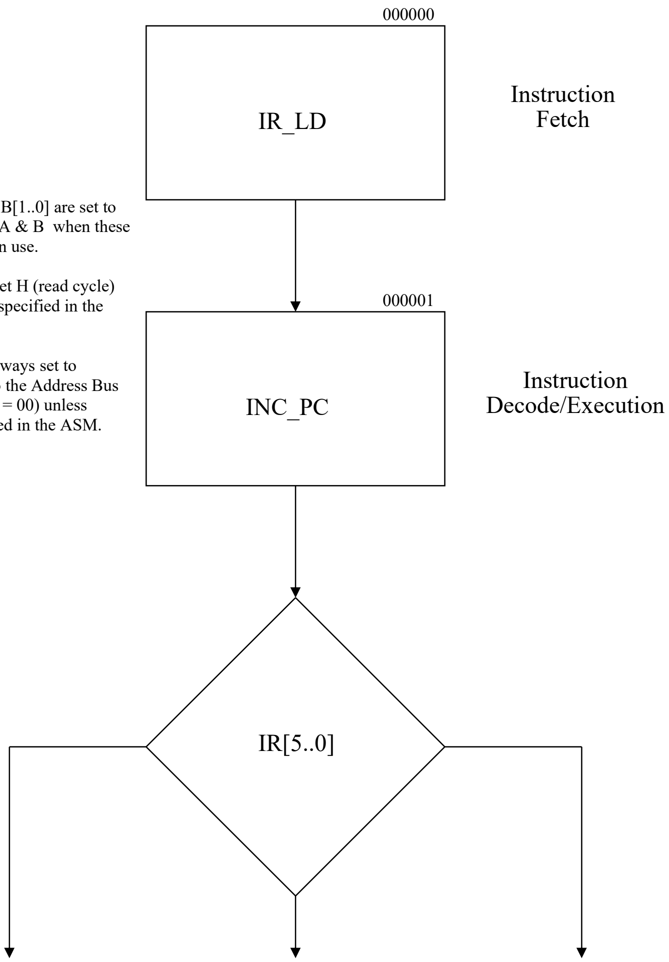
Branch Instructions:

Machine Codes (hex)	Instruction	Operand	Description	# of States
20 bb	BEQ	addrL	Branch if A = 0, i.e., Z Flag = 1 (absolute addressing)	3
21 bb	BNE	addrL	Branch if A ≠ 0, i.e., Z Flag = 0 (absolute addressing)	3
22 bb	BN	addrL	Branch if A is negative, i.e., N Flag = 1 (absolute addressing)	3
23 bb	BP	addrL	Branch if A is positive (or zero), i.e., N Flag = 0 (absolute addressing)	3

Special Notes

1. Z flag and N flag are only set and cleared by the contents in register A.
2. A branch is accomplished by moving the operand address "addr" to the lower byte of the PC. The upper byte of the PC remains unchanged after a branch.
3. The Branch Instructions use absolute addressing where only the low byte of the address is used as an operand. If the branch condition is met, the high byte of the PC is unchanged and the low byte takes the value of the operand (addrL).
4. Explanations of the operands shown in the Machine Codes:
    - mm — 8-bit immediate data value
    - ii — Low-order byte of a 16-bit data
    - jj — High-order byte of a 16-bit data
    - ll — Low-order byte of a 16-bit address
    - hh — High-order byte of a 16-bit address
    - dd — 8-bit displacement value
    - bb — Low-order byte of a 16-bit address for a branch instruction

G-CPU Controller Flow Charts



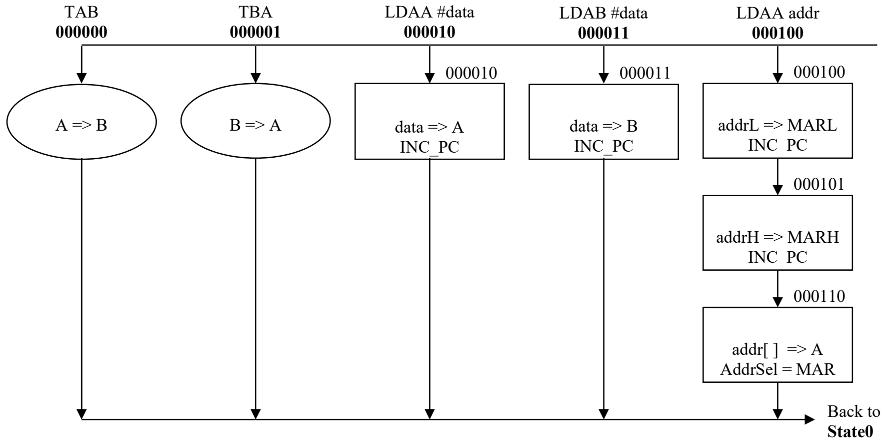
Special Notes:

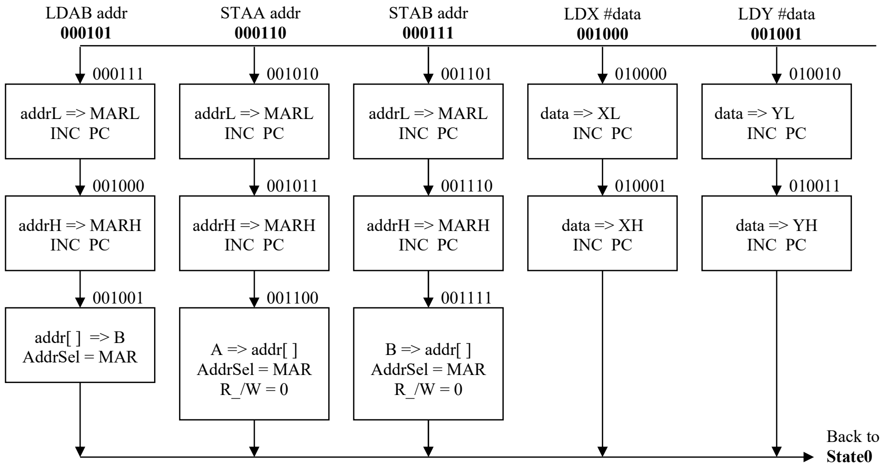
1. MSA[1..0] & MSB[1..0] are set to protect registers A & B when these registers are not in use.
2. R_/W is always set H (read cycle) unless otherwise specified in the ASM chart.
3. ADDR_SEL is always set to connect the PC to the Address Bus (i.e. ADDR_SEL = 00) unless otherwise specified in the ASM.

Instructions Listed on Next Pages…

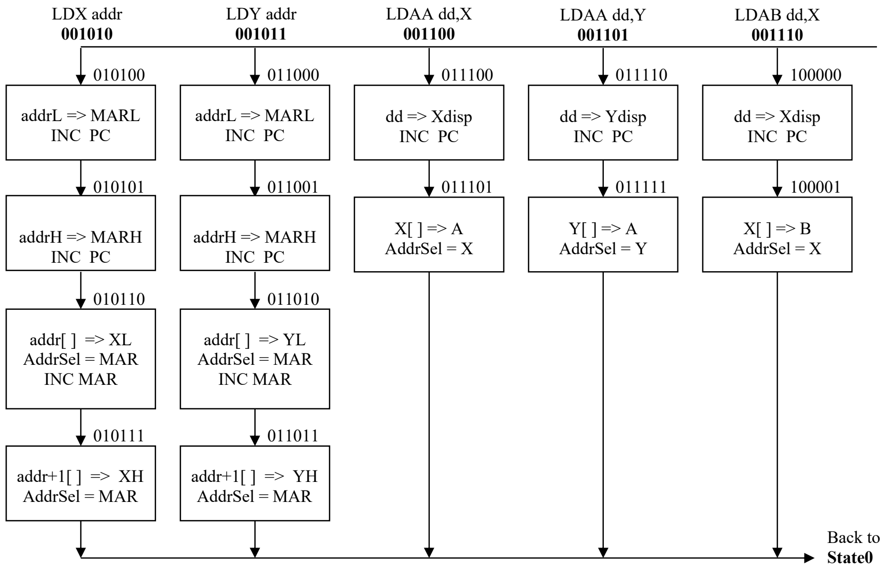
G-CPU Controller Flow Charts

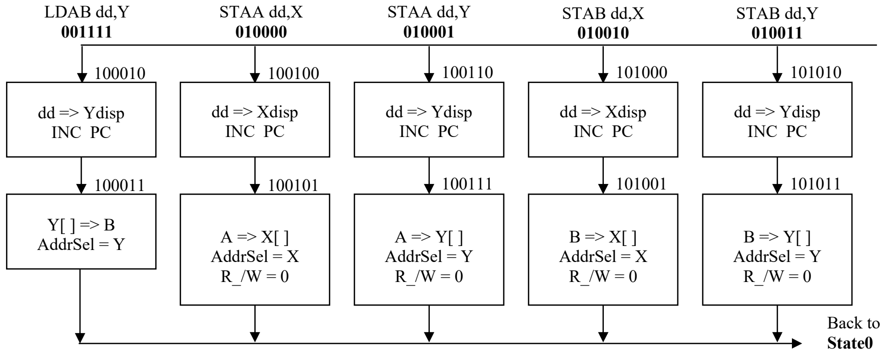
Data Movement Instructions:



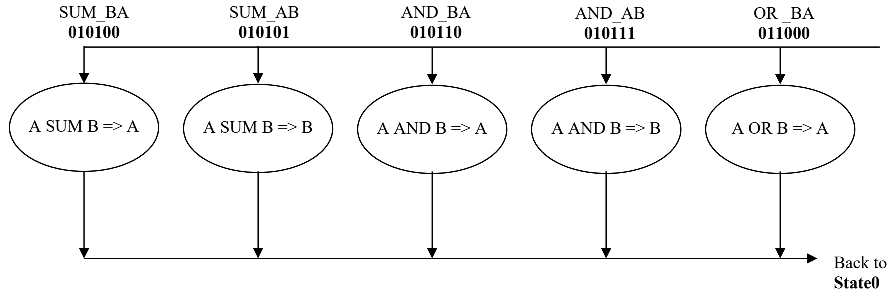


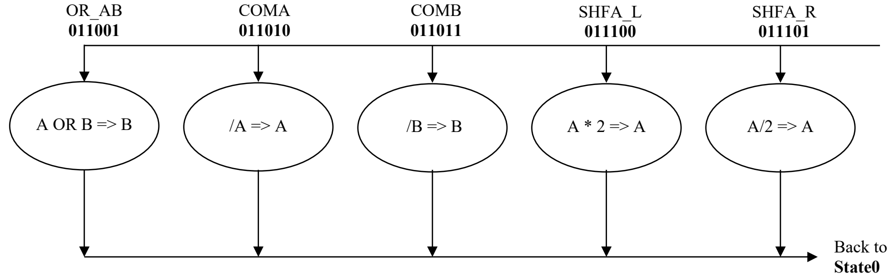
Data Movement Instructions (continued):

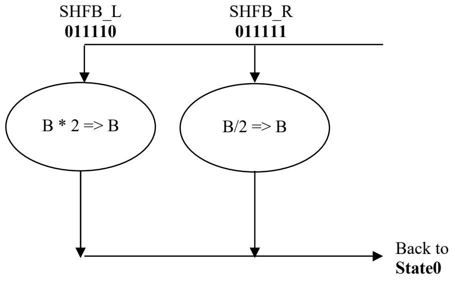




ALU Related Instructions:

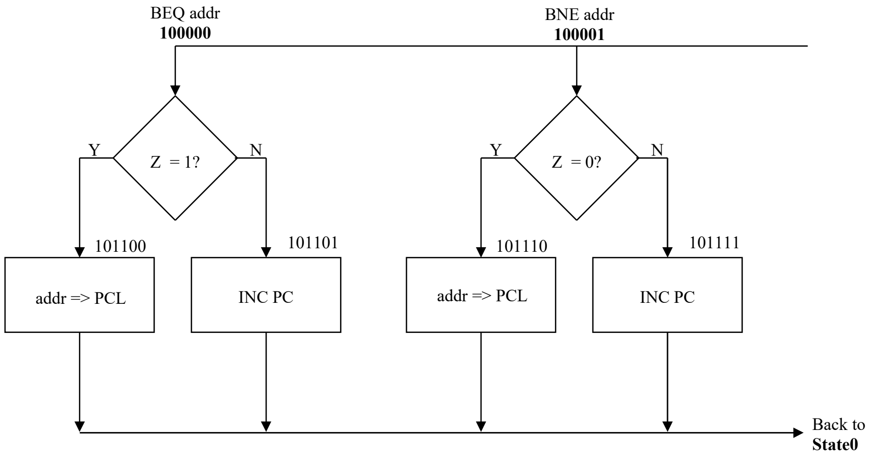


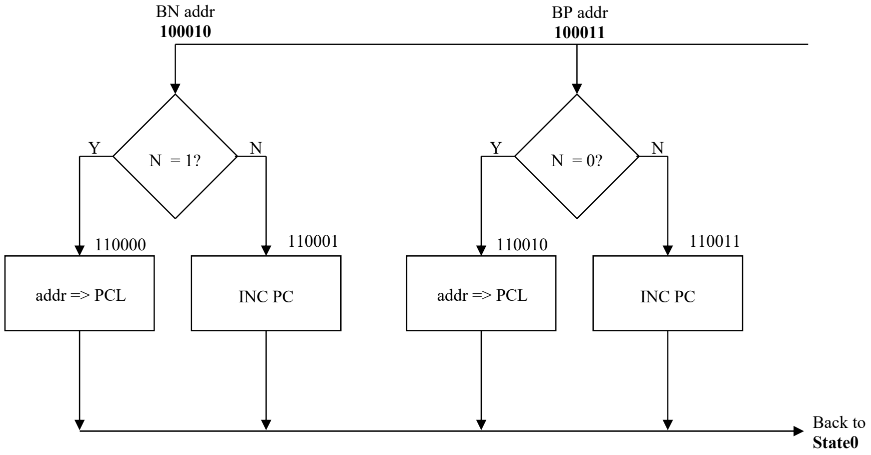




G-CPU Controller Flow Charts

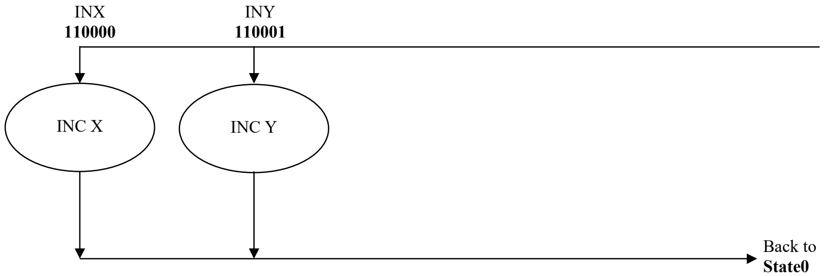
Branch Instructions:





G-CPU Controller Flow Charts

Additional Instructions:



G-CPU Controller Next State Table

Pres State	Opcode	Flags	Next State	Mux Select			Control		REG INC	ADDR SEL	PC	MAR	X,Y Loading		Disp Regs	
Q[5..0]	IR[5..0]	Z N	D[5..0]	MSA [1..0]	MSB [1..0]	MSC [3..0]	IR LD	R /W	PC MAR X Y	ADDR SEL [1..0]	PC LD L/U	MAR LD L/U	X LD L/U	Y LD L/U	XD_LD YD_LD	Present State Function
000000	******	**	000001	01	10	0000	1	1	0000	00	00	00	00	00	00	generic instruction fetch
000001	000000	**	000000	01	01	0000	0	1	1000	00	00	00	00	00	00	Transfer A to B (TAB)
000001	000001	**	000000	10	10	0000	0	1	1000	00	00	00	00	00	00	Transfer B to A (TBA)
000001	000010	**	000010	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAA #data, state 1
000010	******	**	000000	00	10	0000	0	1	1000	00	00	00	00	00	00	LDAA #data, state 2
000001	000011	**	000011	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAB #data, state 1
000011	******	**	000000	01	00	0001	0	1	1000	00	00	00	00	00	00	LDAB #data, state 3
000001	000100	**	000100	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAA addr, state 1
000100	******	**	000101	01	10	0000	0	1	1000	00	00	10	00	00	00	LDAA addr, state 4
000101	******	**	000110	01	10	0000	0	1	1000	00	00	01	00	00	00	LDAA addr, state 5
000110	******	**	000000	00	10	0000	0	1	0000	01	00	00	00	00	00	LDAA addr, state 6
000001	000101	**	000111	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAB addr, state 1
000111	******	**	001000	01	10	0000	0	1	1000	00	00	10	00	00	00	LDAB addr, state 7
001000	******	**	001001	01	10	0000	0	1	1000	00	00	01	00	00	00	LDAB addr, state 8
001001	******	**	000000	01	00	0000	0	1	0000	01	00	00	00	00	00	LDAB addr, state 9
000001	000110	**	001010	01	10	0000	0	1	1000	00	00	00	00	00	00	STAA addr, state 1
001010	******	**	001011	01	10	0000	0	1	1000	00	00	10	00	00	00	STAA addr, state A
001011	******	**	001100	01	10	0000	0	1	1000	00	00	01	00	00	00	STAA addr, state B
001100	******	**	000000	01	10	0000	0	0	0000	01	00	00	00	00	00	STAA addr, state C
000001	000111	**	001101	01	10	0000	0	1	1000	00	00	00	00	00	00	STAB addr, state 1
001101	******	**	001110	01	10	0000	0	1	1000	00	00	10	00	00	00	STAB addr, state D
001110	******	**	001111	01	10	0000	0	1	1000	00	00	01	00	00	00	STAB addr, state E
001111	******	**	000000	01	10	0001	0	0	0000	01	00	00	00	00	00	STAB addr, state F
000001	001000	**	010000	01	10	0000	0	1	1000	00	00	00	00	00	00	LDX #data, state 1
010000	******	**	010001	01	10	0000	0	1	1000	00	00	00	10	00	00	LDX #data, state 10
010001	******	**	000000	01	10	0000	0	1	1000	00	00	00	01	00	00	LDX #data, state 11
000001	001001	**	010010	01	10	0000	0	1	1000	00	00	00	00	00	00	LDY #data, state 1
010010	******	**	010011	01	10	0000	0	1	1000	00	00	00	00	10	00	LDY #data, state 12
010011	******	**	000000	01	10	0000	0	1	1000	00	00	00	00	01	00	LDY #data, state 13
000001	001010	**	010100	01	10	0000	0	1	1000	00	00	00	00	00	00	LDX addr, state 1
010100	******	**	010101	01	10	0000	0	1	1000	00	00	10	00	00	00	LDX addr, state 14
010101	******	**	010110	01	10	0000	0	1	1000	00	00	01	00	00	00	LDX addr, state 15
010110	******	**	010111	01	10	0000	0	1	0100	01	00	00	10	00	00	LDX addr, state 16
010111	******	**	000000	01	10	0000	0	1	0000	01	00	00	01	00	00	LDX addr, state 17
000001	001011	**	011000	01	10	0000	0	1	1000	00	00	00	00	00	00	LDY addr, state 1
011000	******	**	011001	01	10	0000	0	1	1000	00	00	10	00	00	00	LDY addr, state 18

G-CPU Controller Next State Table

Pres State	Opcode	Flags	Next State	Mux Select			Control		REG INC	ADDR SEL	PC	MAR	X,Y Loading		Disp Regs	
Q[5..0]	IR[5..0]	Z N	D[5..0]	MSA [1..0]	MSB [1..0]	MSC [3..0]	IR LD	R /W	PC MAR X Y	ADDR SEL [1..0]	PC LD L/U	MAR LD L/U	X LD L/U	Y LD L/U	XD_LD YD_LD	Present State Function
011001	******	**	011010	01	10	0000	0	1	1000	00	00	01	00	00	00	LDY addr, state 19
011010	******	**	011011	01	10	0000	0	1	0100	01	00	00	00	10	00	LDY addr, state 1A
011011	******	**	000000	01	10	0000	0	1	0000	01	00	00	00	01	00	LDY addr, state 1B
000001	001100	**	011100	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAA dd,X state 1
011100	******	**	011101	01	10	0000	0	1	1000	00	00	00	00	00	10	LDAA dd,X state 1C
011101	******	**	000000	00	10	0000	0	1	0000	10	00	00	00	00	00	LDAA dd,X state 1D
000001	001101	**	011110	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAA dd,Y state 1
011110	******	**	011111	01	10	0000	0	1	1000	00	00	00	00	00	01	LDAA dd,Y state 1E
011111	******	**	000000	00	10	0000	0	1	0000	11	00	00	00	00	00	LDAA dd,Y state 1F
000001	001110	**	100000	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAB dd,X state 1
100000	******	**	100001	01	10	0000	0	1	1000	00	00	00	00	00	10	LDAB dd,X state 20
100001	******	**	000000	01	00	0000	0	1	0000	10	00	00	00	00	00	LDAB dd,X state 21
000001	001111	**	100010	01	10	0000	0	1	1000	00	00	00	00	00	00	LDAB dd,Y state 1
100010	******	**	100011	01	10	0000	0	1	1000	00	00	00	00	00	01	LDAB dd,Y state 22
100011	******	**	000000	01	00	0000	0	1	0000	11	00	00	00	00	00	LDAB dd,Y state 23
000001	010000	**	100100	01	10	0000	0	1	1000	00	00	00	00	00	00	STAA dd,X state 1
100100	******	**	100101	01	10	0000	0	1	1000	00	00	00	00	00	10	STAA dd,X state 24
100101	******	**	000000	01	10	0000	0	0	0000	10	00	00	00	00	00	STAA dd,X state 25
000001	010001	**	100110	01	10	0000	0	1	1000	00	00	00	00	00	00	STAA dd,Y state 1
100110	******	**	100111	01	10	0000	0	1	1000	00	00	00	00	00	01	STAA dd,Y state 26
100111	******	**	000000	01	10	0000	0	0	0000	11	00	00	00	00	00	STAA dd,Y state 27
000001	010010	**	101000	01	10	0000	0	1	1000	00	00	00	00	00	00	STAB dd,X state 1
101000	******	**	101001	01	10	0001	0	1	1000	00	00	00	00	00	10	STAB dd,X state 28
101001	******	**	000000	01	10	0001	0	0	0000	10	00	00	00	00	00	STAB dd,X state 29
000001	010011	**	101010	01	10	0000	0	1	1000	00	00	00	00	00	00	STAB dd,Y state 1
101010	******	**	101011	01	10	0001	0	1	1000	00	00	00	00	00	01	STAB dd,Y state 2A
101011	******	**	000000	01	10	0001	0	0	0000	11	00	00	00	00	00	STAB dd,Y state 2B
000001	010100	**	000000	11	10	0010	0	1	1000	00	00	00	00	00	00	SUM_BA state 1
000001	010101	**	000000	01	11	0010	0	1	1000	00	00	00	00	00	00	SUM_AB state 1
000001	010110	**	000000	11	10	0011	0	1	1000	00	00	00	00	00	00	AND_BA state 1
000001	010111	**	000000	01	11	0011	0	1	1000	00	00	00	00	00	00	AND_AB state 1
000001	011000	**	000000	11	10	0100	0	1	1000	00	00	00	00	00	00	OR_BA state 1
000001	011001	**	000000	01	11	0100	0	1	1000	00	00	00	00	00	00	OR_AB state 1
000001	011010	**	000000	11	10	0101	0	1	1000	00	00	00	00	00	00	COMA state 1
000001	011011	**	000000	01	11	0110	0	1	1000	00	00	00	00	00	00	COMB state 1
000001	011100	**	000000	11	10	0111	0	1	1000	00	00	00	00	00	00	SHFA_L state 1

G-CPU Controller Next State Table

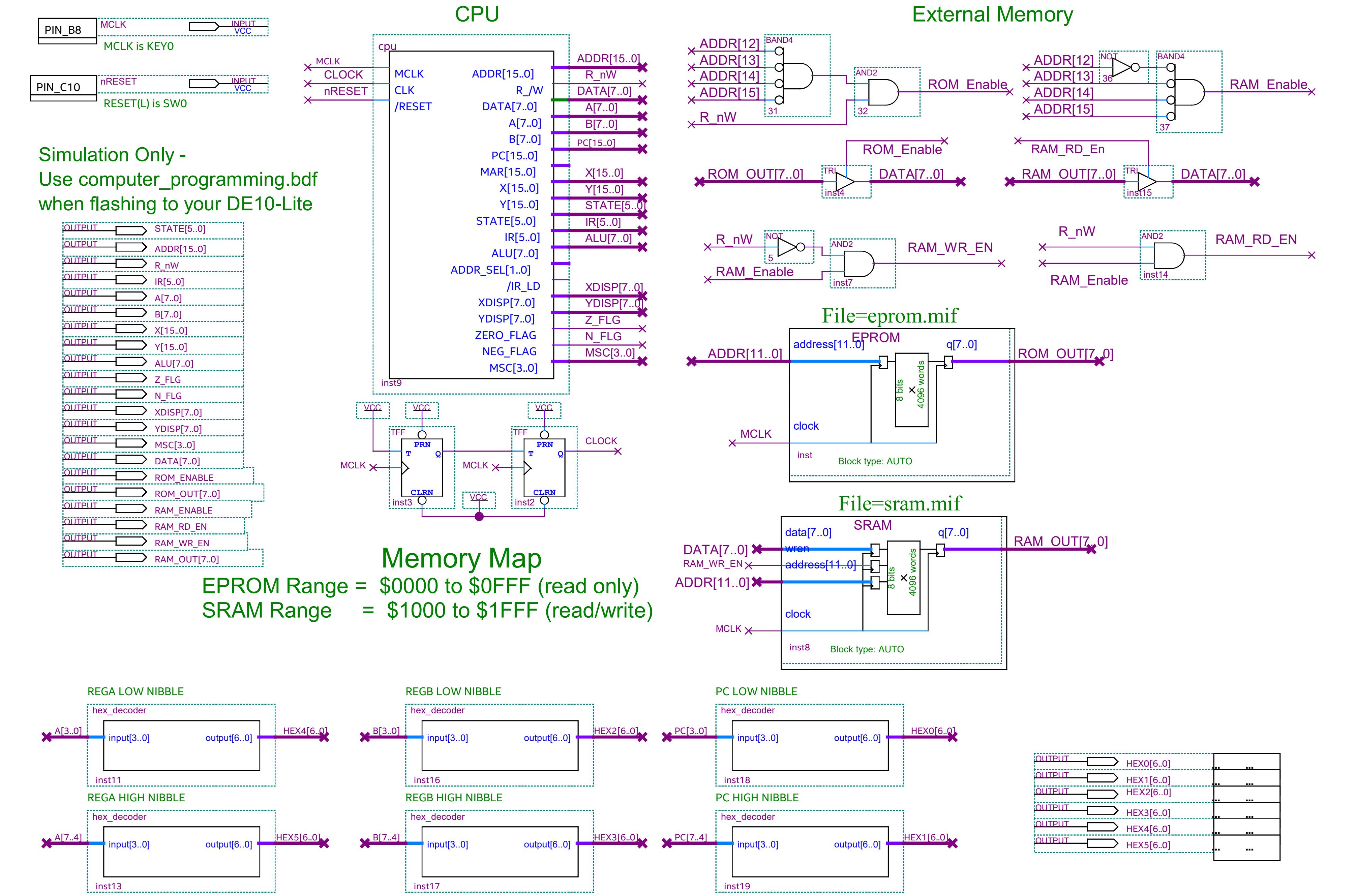
Pres State	Opcode	Flags	Next State	Mux Select			Control		REG INC	ADDR SEL	PC	MAR	X,Y Loading		Disp Regs	
Q[5..0]	IR[5..0]	Z N	D[5..0]	MSA [1..0]	MSB [1..0]	MSC [3..0]	IR LD	R /W	PC MAR X Y	ADDR SEL [1..0]	PC LD L/U	MAR LD L/U	X LD L/U	Y LD L/U	XD_LD YD_LD	Present State Function
000001	011101	**	000000	11	10	1000	0	1	1000	00	00	00	00	00	00	SHFA_R state 1
000001	011110	**	000000	01	11	1001	0	1	1000	00	00	00	00	00	00	SHFB_L state 1
000001	011111	**	000000	01	11	1010	0	1	1000	00	00	00	00	00	00	SHFB_R state 1
000001	100000	1*	101100	01	10	0000	0	1	1000	00	00	00	00	00	00	BEQ addr state 1
000001	100000	0*	101101	01	10	0000	0	1	1000	00	00	00	00	00	00	BEQ addr state 1
101100	******	**	000000	01	10	0000	0	1	0000	00	10	00	00	00	00	BEQ addr state 2C
101101	******	**	000000	01	10	0000	0	1	1000	00	00	00	00	00	00	BEQ addr state 2D
000001	100001	0*	101110	01	10	0000	0	1	1000	00	00	00	00	00	00	BNE addr state 1
000001	100001	1*	101111	01	10	0000	0	1	1000	00	00	00	00	00	00	BNE addr state 1
101110	******	**	000000	01	10	0000	0	1	0000	00	10	00	00	00	00	BNE addr state 2E
101111	******	**	000000	01	10	0000	0	1	1000	00	00	00	00	00	00	BNE addr state 2F
000001	100010	*1	110000	01	10	0000	0	1	1000	00	00	00	00	00	00	BN addr state 1
000001	100010	*0	110001	01	10	0000	0	1	1000	00	00	00	00	00	00	BN addr state 1
110000	******	**	000000	01	10	0000	0	1	0000	00	10	00	00	00	00	BN addr state 30
110001	******	**	000000	01	10	0000	0	1	1000	00	00	00	00	00	00	BN addr state 31
000001	100011	*0	110010	01	10	0000	0	1	1000	00	00	00	00	00	00	BP addr state 1
000001	100011	*1	110011	01	10	0000	0	1	1000	00	00	00	00	00	00	BP addr state 1
110010	******	**	000000	01	10	0000	0	1	0000	00	10	00	00	00	00	BP addr state 32
110011	******	**	000000	01	10	0000	0	1	1000	00	00	00	00	00	00	BP addr state 33
000001	110000	**	000000	01	10	0000	0	1	1010	00	00	00	00	00	00	Increment X (INX)
000001	110001	**	000000	01	10	0000	0	1	1001	00	00	00	00	00	00	Increment Y (INY)

computer_simulation.bdf

Date: April 05, 2024

computer_simulation.bdf

Project: computer



Simulation Only -
Use computer_programming.bdf
when flashing to your DE10-Lite

Memory Map

EPROM Range = $0000 to $0FFF (read only)

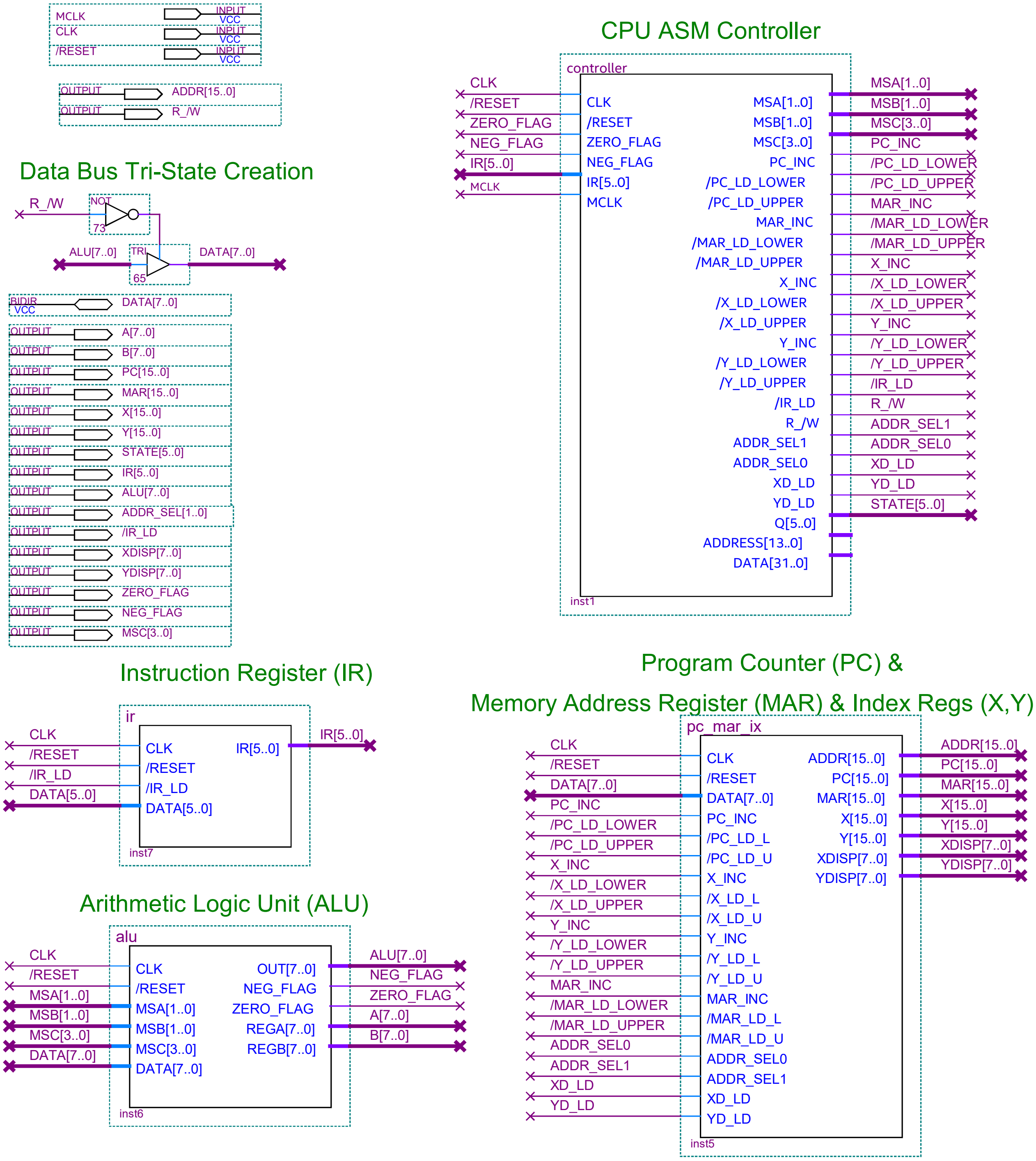
SRAM Range = $1000 to $1FFF (read/write)

Date: April 05, 2024

CPU

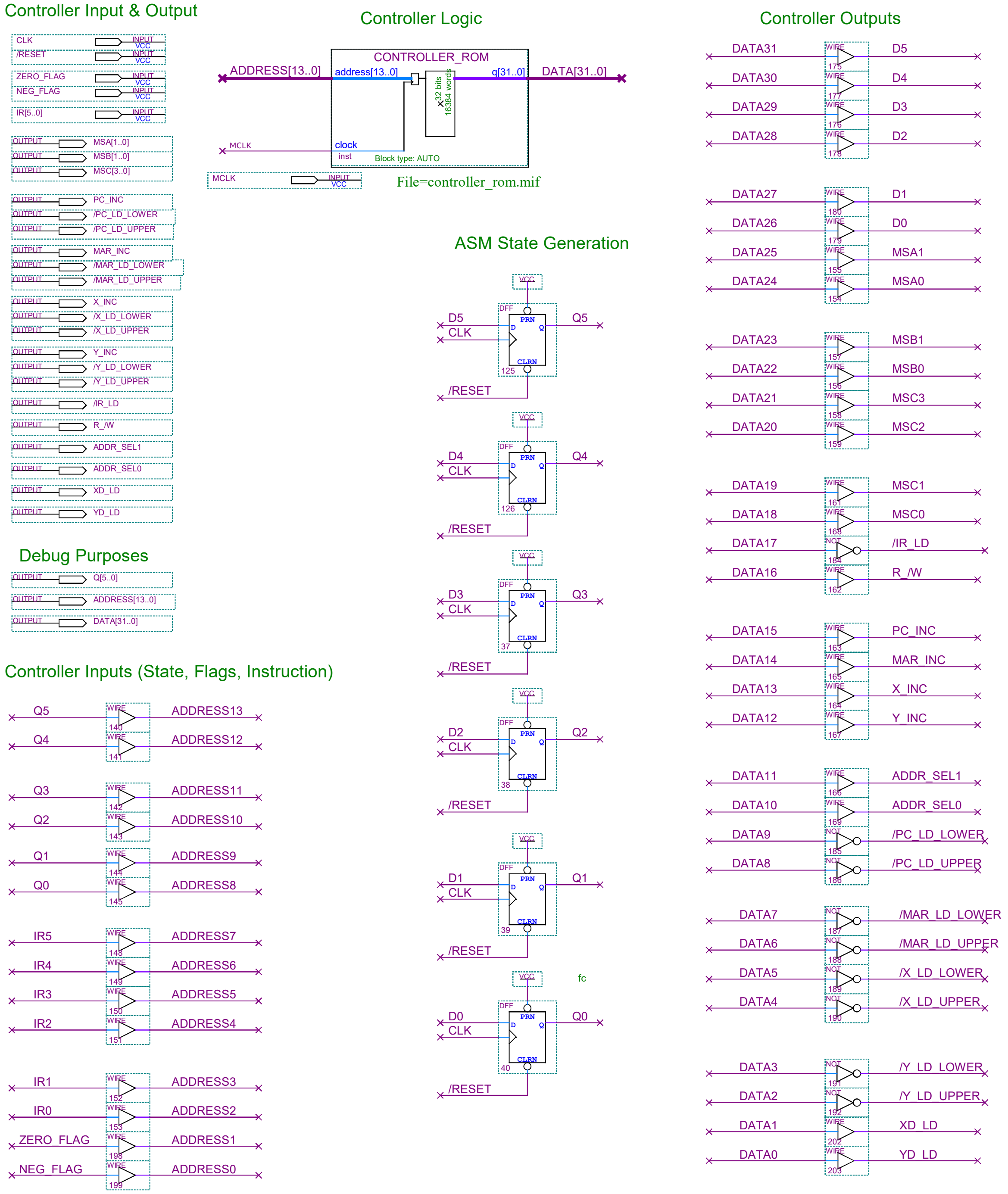
cpu.bdf

Project: computer

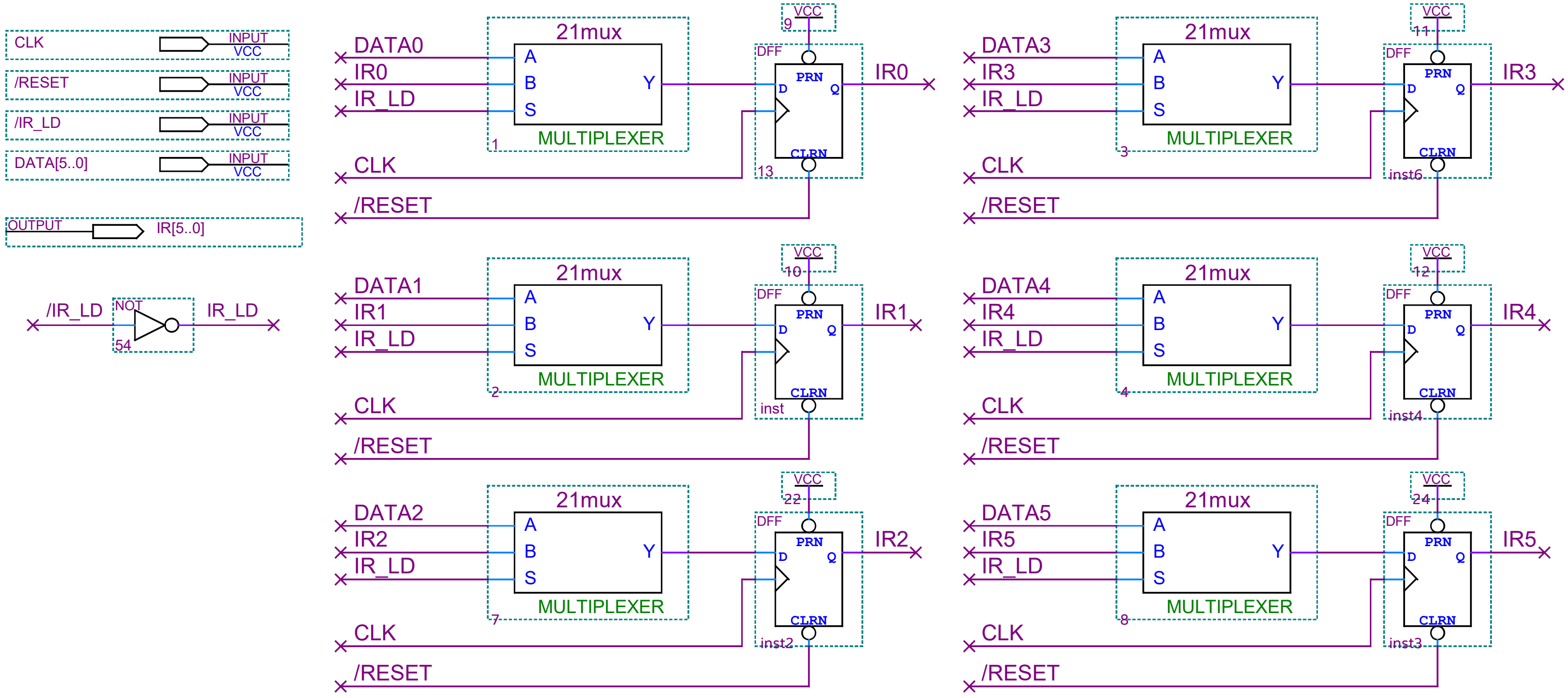


Controller

Date: April 05, 2024 controller.bdf Project: computer



Instruction Register (IR)



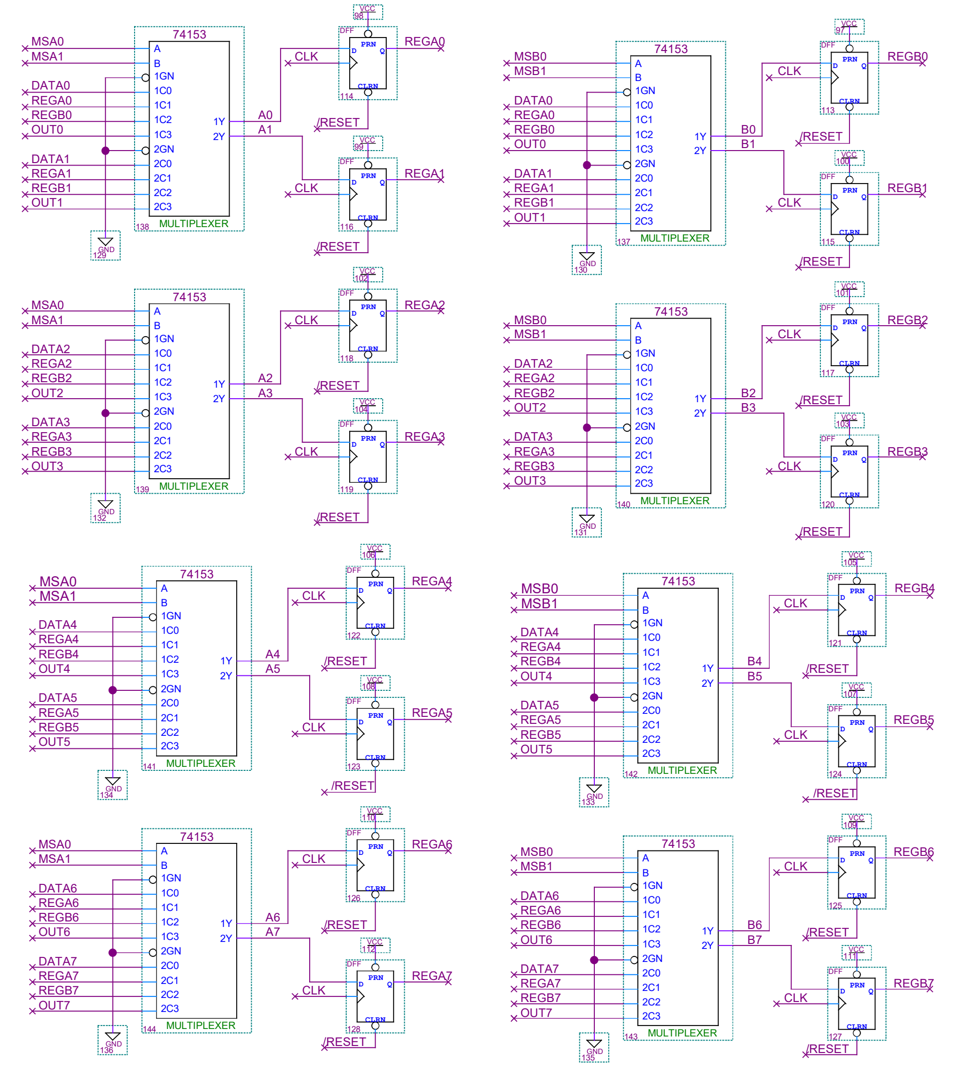
Date: April 05, 2024

ALU

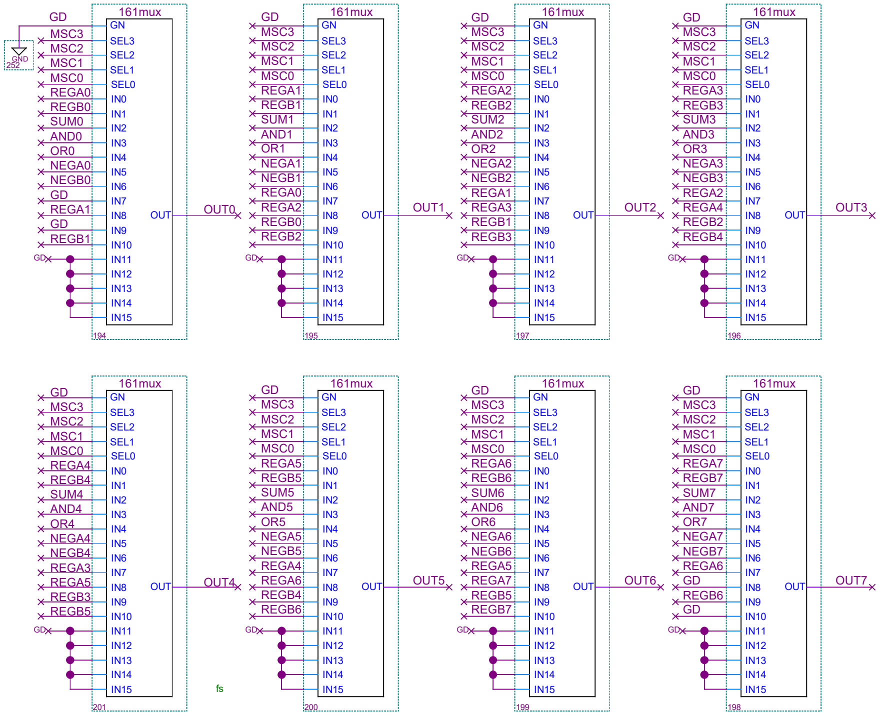
alu.bdf

Project: computer

Mux A, Mux B, Reg A, Reg B Block



Mux C Block



MSC3:0	Selection
0 0 0 0	REGA
0 0 0 1	REGB
0 0 1 0	Sum(A,B)
0 0 1 1	AND(A,B)
0 1 0 0	OR(A,B)
0 1 0 1	COMA
0 1 1 0	COMB
0 1 1 1	SHFA_Left
1 0 0 0	SHFA_Right
1 0 0 1	SHFB_Left
1 0 1 0	SHFB_Right

Input & Output Signals

Debug Signals

Sum & Carry Flag Generation

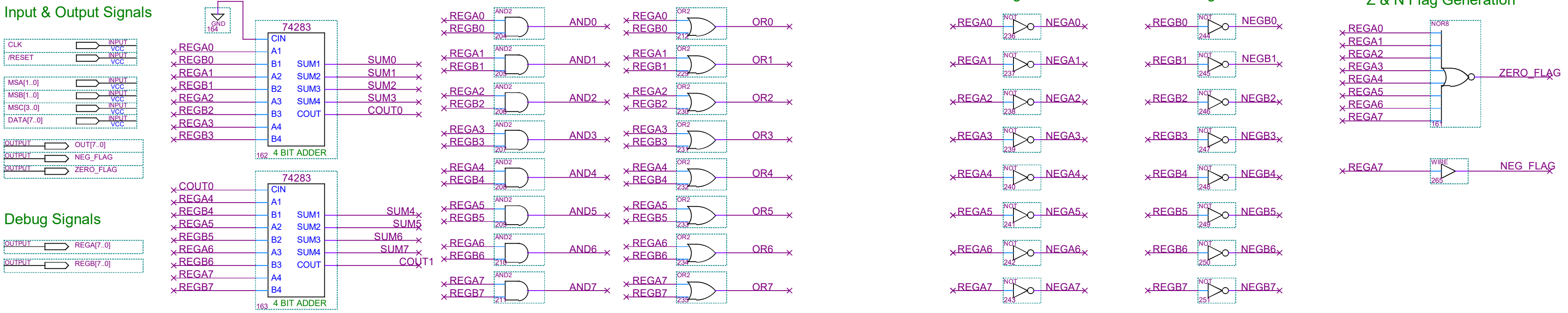
AND Generation

OR Generation

Negate A

Negate B

Z & N Flag Generation

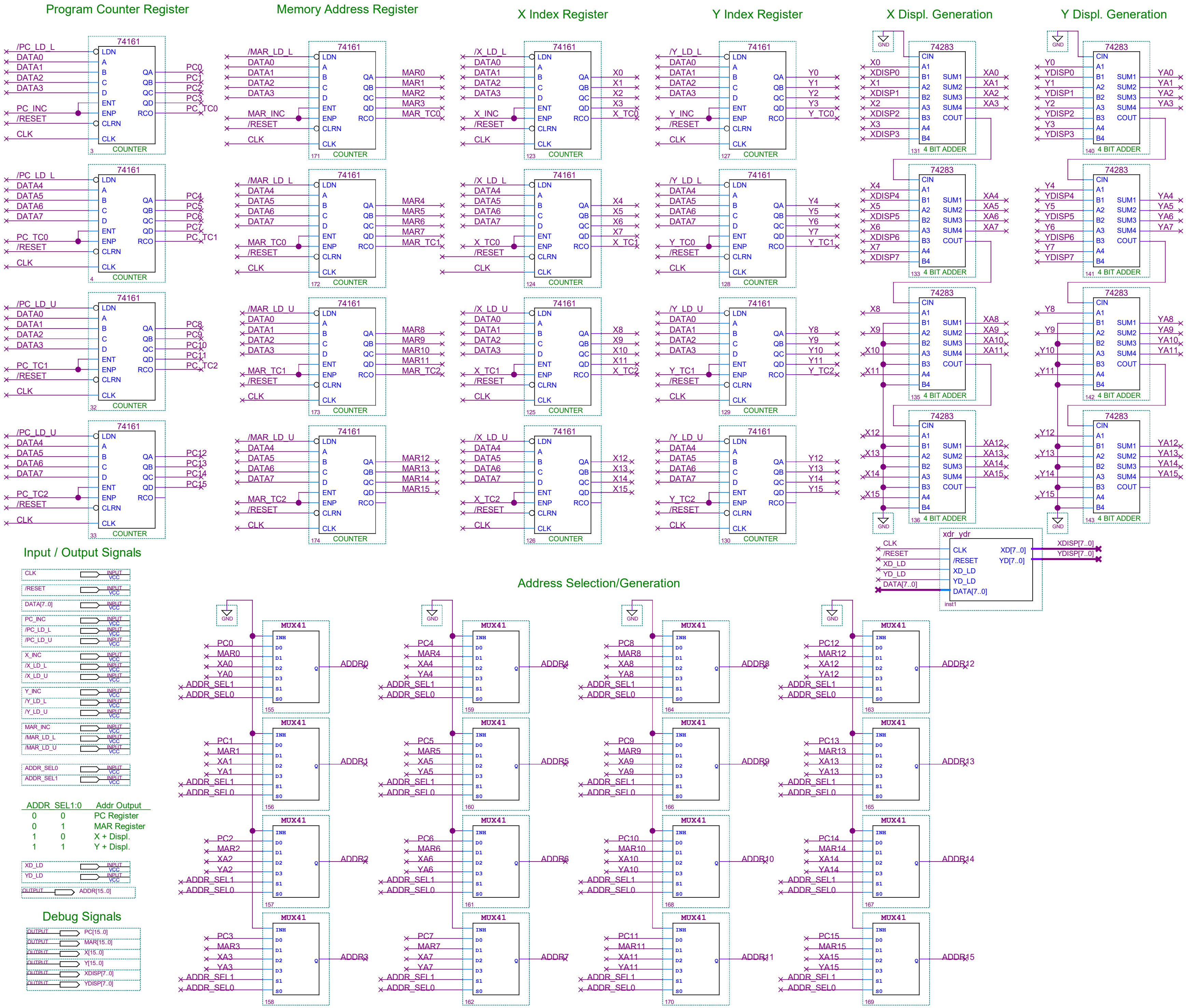


Program Counter (PC) & Memory Address Register (MAR) & Index Regs (X,Y)

Date: April 05, 2024

pc_mar_ix.bdf

Project: computer



ADDR_SEL1:0		Addr Output
0	0	PC Register
0	1	MAR Register
1	0	X + Displ.
1	1	Y + Displ.

Date: April 05, 2024

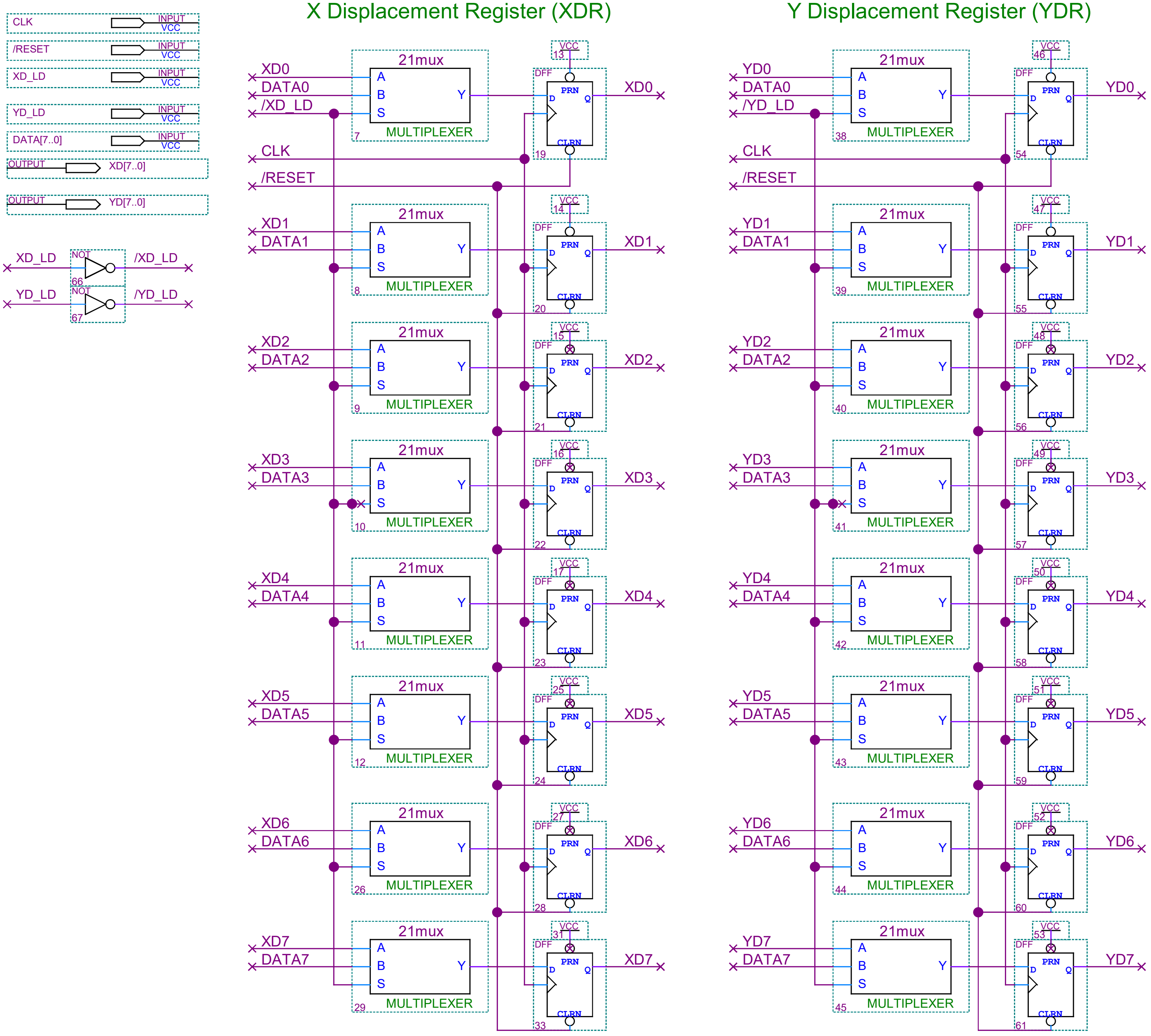
xdr_ydr

xdr_ydr.bdf

Project: computer

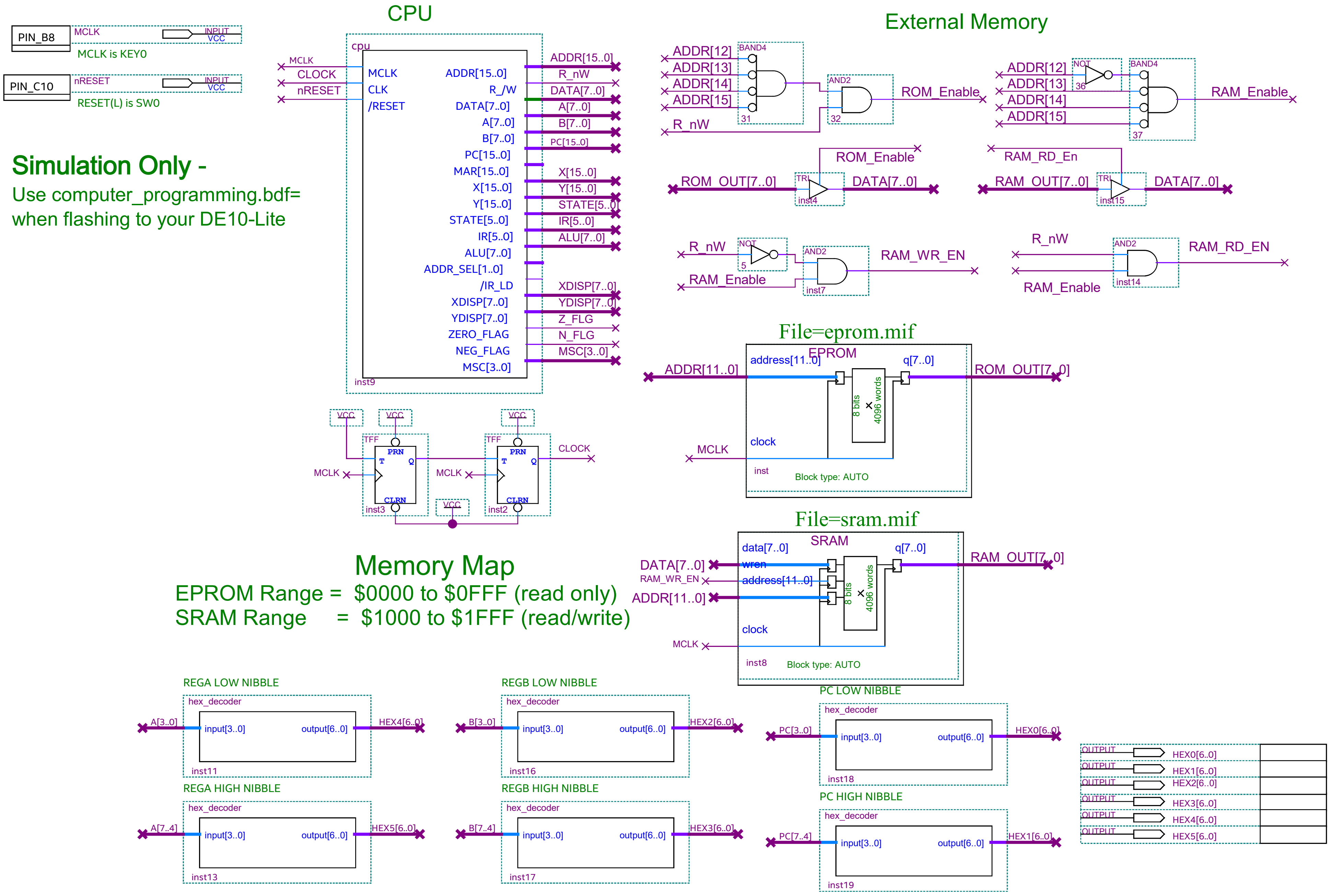
X Displacement Register (XDR)

Y Displacement Register (YDR)



computer_programming.bdf

Date: April 05, 2024 computer_programming.bdf Project: computer



Simulation Only -

Use computer_programming.bdf=
when flashing to your DE10-Lite

Memory Map

EPROM Range = $0000 to $0FFF (read only)
SRAM Range = $1000 to $1FFF (read/write)