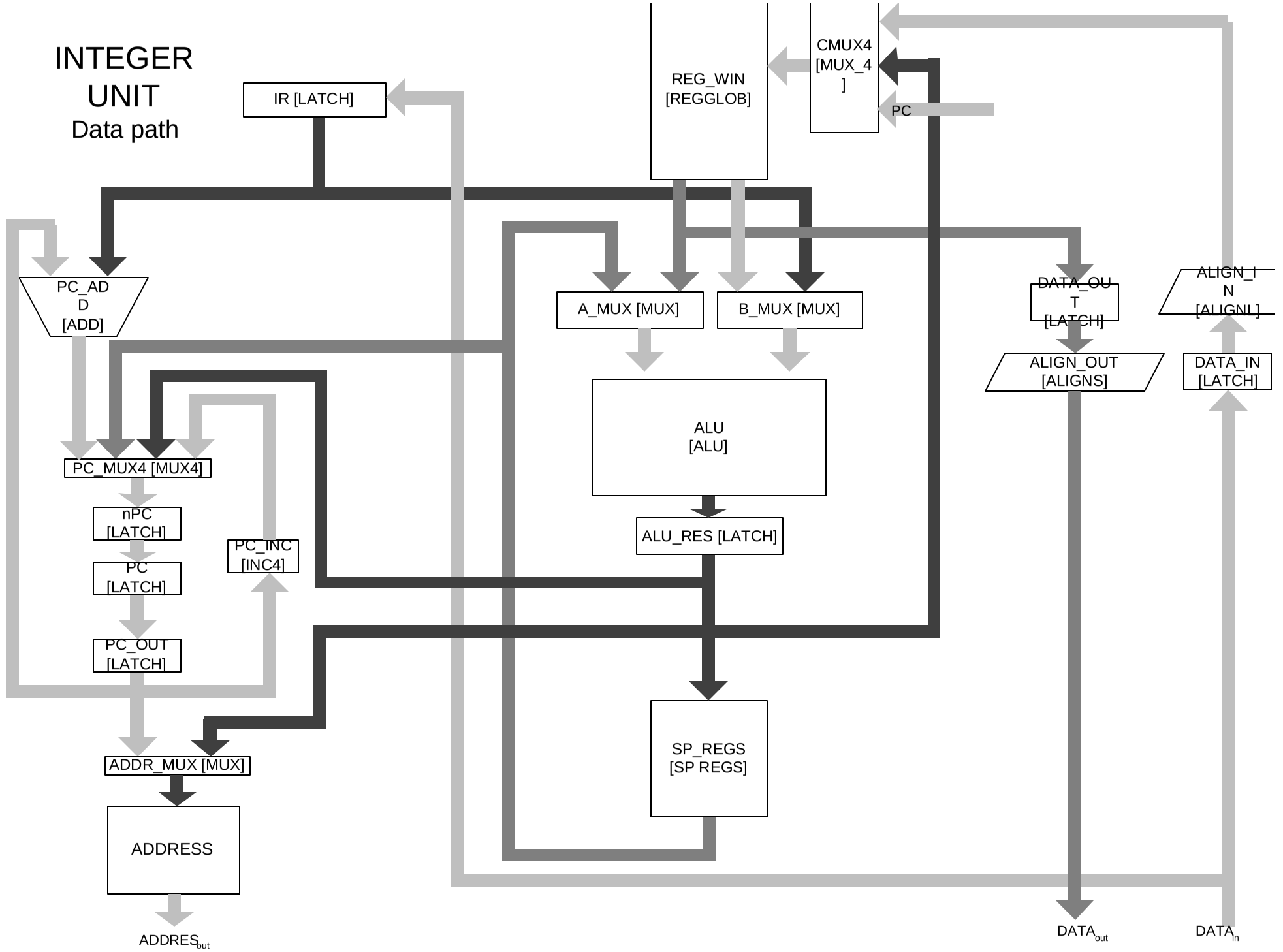
INTEGER
UNIT
Data path
IR [LATCH]
REG_WIN
[REGGLOB]
CMUX4
[MUX_4
]
PC
PC_AD
D
[ADD]
A_MUX [MUX]
B_MUX [MUX]
DATA_OU
T
[LATCH]
ALIGN_I
N
[ALIGNL]
ALIGN_OUT
[ALIGNS]
DATA_IN
[LATCH]
ALU
[ALU]
PC_MUX4 [MUX4]
nPC
[LATCH]
PC_INC
[INC4]
PC
[LATCH]
ALU_RES [LATCH]
PC_OUT
[LATCH]
SP_REGS
[SP REGS]
ADDR_MUX [MUX]
ADDRESS
ADDRES$_{out}$
DATA$_{out}$
DATA$_{in}$

UNIDAD DE CONTROL

La siguiente tabla enumera los puertos de entrada y salida de la unidad de control e indica el componente y conjunto de puertos a los que se deben conectar.

Señales de entrada:

De	A	Nombre	Se conecta a
0	31	IR	IR.OUT0..31
32	63	PSR	SP_REGS.PSR0..31
32	32	BUS_BUSY_IN	BUS.BUSY
33	33	BUS_DACK_IN	BUS.DTACK
34	34	BUS_ERR	BUS.ERR

Señales de salida:

De	A	Nombre	Se conecta a
0	4	A_SEL0..4	REG_WIN.ASEL0..ASEL4
5	9	B_SEL0..4	REG_WIN.BSEL0..BSEL4
10	14	C_SEL0..4	REG_WIN.CSEL0..CSEL4
15	15	C_EN	REG_WIN.CEN
16	16	C_MUX0	CMUX4.SELA
17	17	C_MUX1	CMUX4.SELB
18	18	C_MUX2	CMUX4.SELC
19	19	C_MUX3	CMUX4.SELD
20	20	A_MUX	A_MUX.SELA/SELB
21	21	B_MUX	B_MUX. SELA/SELB
22	22	ENABLE_MUL	ALU.ENABLE_MUL
23	23	ENABLE_ALU	ALU.ENABLE_ALU
24	24	ENABLE_SHFT	ALU.ENABLE_SFHT
25	25	ENABLE_SETH	ALU.ENABLE_SETH
26	29	ALU_FCOD0..4	ALU.FCOD0..4
30	30	ALU_RES_EN	ALU_RES.EIN
31	31	PC_MUX0	PC_MUX4.SELA
32	32	PC_MUX1	PC_MUX4.SELB
33	33	PC_MUX2	PC_MUX4.SELC
34	34	PC_MUX3	PC_MUX4.SELD
35	35	NPC_EN	NPC.EIN
36	36	PC_EN	PC.EIN
37	37	PC_OUT_EN	PC_OUT.EIN
38	38	ADDR_MUX	ADDR_MUX. SELA/SELB
39	39	IR_EN	IR.EIN
40	40	DATA_IN_EN	DATA_IN.EIN
41	41	DATA_OUT_EN	DATA_OUT.EIN

42	43	ALIGN_OUT_SIZE0..1	ALIGN_OUT.SIZE0..1
44	44	ALIGN_OUT_SIGN	ALIGN_OUT.SIGN
45	46	ALIGN_IN_SIZE0..1	ALIGN_IN.SIZE0..1
47	49	SP_REGSIN0..2	SP_REGS.INREG0..2
50	52	SP_REGSOUT0..2	SP_REGS.OUTREG0..2
53	53	TBR_TRAP_EN	SP_REGS.TB_TRAB_EN
54	54	INCDEC_FCOD	SP_REGS.INCDEC_FCOD
55	55	S=PS	SP_REGS.S=PS
56	56	PS=S	SP_REGS.PS=S
57	57	S=1	SP_REGS.S=1
58	58	SET_ET	SP_REGS.SET_ET
59	59	CLEAR_ET	SP_REGS.CLEAR_ET
60	60	SP_REGS_EN	SP_REGS.SP_REGS_EN
61	61	BUS_AS	BUS.AS
62	62	BUS_RD/WR	BUS.RD/WR
63	63	BUS_RESET	BUS.RESET
64	64	BUS_BUSY_OUT	BUS.BUSY
65	65	BUS_ERR	BUS.ERR
66	69	BUS_SEL0..3	BUS.SEL0..3

Especificación de la Unidad de Control

Luego de un fetch, PC apunta a la próxima instrucción.

Señales de entrada	Comportamiento
INSTRUCCIÓN: NOP 10987654321098765432109876543210 IR: 00000001000000000000000000000000 PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Nada. Traer la próxima instrucción. [Esperar a BUS_BUSY_IN = 0] T_0: PC_Out.Enable = 1, ADDR_MUX.SEL = PC_OUT, RD, BUS_BUSY_OUT = 1, RDM.Enable = 1 T_1: PC.Enable = 1 T_2: PC_MUX4.SEL = PC_INC, nPC.Enable = 1 [Esperar a BUS_DACK_IN = 1] T_3: IR.Enable = 1
INSTRUCCIÓN: STB 10987654321098765432109876543210 IR: 11AAAAA000101BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Escribir el contenido del primer byte del registro A en la dir Reg[B]+Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1, ALIGN_OUT.SIZE = Byte, [Esperar a BUS_BUSY_IN = 0] BUS_SEL = DECOD(Bits más bajos de address A_0 A_1), BUS_BUSY_OUT = 1, WR Continuar como en NOP
INSTRUCCIÓN: STH 10987654321098765432109876543210 IR: 11AAAAA000110BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Escribir el contenido de los dos primeros bytes del registro A en la dir Reg[B]+Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1

	T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1, ALIGN_OUT.SIZE = HalfWord, [Esperar BUS_BUSY_IN = 0] BUS_SEL = DECOD2(Bit más bajo de address) BUS_BUSY_OUT = 1, WR Continuar como en NOP
INSTRUCCIÓN: ST 10987654321098765432109876543210 IR: 11AAAAA000100BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Escribir el contenido del registro A en la dir Reg[B]+Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1, ALIGN_OUT.SIZE = Word, [Esperar BUS_BUSY_IN = 0] BUS_SEL = 15 (todos 1´s) BUS_BUSY_OUT = 1, WR Continuar como en NOP
INSTRUCCIÓN: STB 10987654321098765432109876543210 IR: 11AAAAA000101BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Escribir el contenido del primer byte del registro A en la dir Reg[B] +C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1, ALIGN_OUT.SIZE = Byte, [Esperar a BUS_BUSY_IN = 0] BUS_SEL = DECOD(Bits más bajos de address A_0 A_1), BUS_BUSY_OUT = 1, WR Continuar como en NOP
INSTRUCCIÓN: STH 10987654321098765432109876543210 IR: 11AAAAA000110BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Escribir el contenido los primeros 2 bytes del registro A en la dir Reg[B] +C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1, ALIGN_OUT.SIZE = HalfWord, [Esperar BUS_BUSY_IN = 0] BUS_SEL = DECOD2(Bit más bajo de address) BUS_BUSY_OUT = 1, WR Continuar como en NOP
INSTRUCCIÓN: ST 10987654321098765432109876543210 IR: 11AAAAA000100BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Escribir el contenido del registro A en la dir Reg[B] + C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1, ALIGN_OUT.SIZE = Word, [Esperar BUS_BUSY_IN = 0] BUS_SEL = 15 (todos 1´s) BUS_BUSY_OUT = 1, WR Continuar como en NOP
INSTRUCCIÓN: LDSB 10987654321098765432109876543210 IR: 11AAAAA001001BBBBB000000000CCCCC	Acciones: Cargar en el registro A el byte de la dir Reg[B] +Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C,

PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Byte, ALIGN_IN.SIGN =1, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDSH 3130292827262524232221201918171615141312111009080706050403020100 IR: 11AAAAA001010BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el half word de la dir Reg[B] +Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Half, ALIGN_IN.SIGN =1, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDUB 3130292827262524232221201918171615141312111009080706050403020100 IR: 11AAAAA000001BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el byte de la dir Reg[B] +Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Byte, ALIGN_IN.SIGN =0, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDUH 3130292827262524232221201918171615141312111009080706050403020100 IR: 11AAAAA000010BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el half word de la dir Reg[B] +Reg[C]. T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Half, ALIGN_IN.SIGN =0, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LD 3130292827262524232221201918171615141312111009080706050403020100	Acciones: Cargar en el registro A el word de la dir Reg[B] +Reg[C].

IR: 11AAAAA000000BBBBB000000000CCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	T_0: REG_WIN.A_SEL = B,REG_WIN. B_SEL.SEL = C, A_MUX.SEL = REG_WIN, B_MUX.SEL = REG_WIN ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Word, ALIGN_IN.SIGN =0, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDSB 10987654321098765432109876543210 IR: 11AAAAA001001BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el byte de la dir Reg[B] +C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Byte, ALIGN_IN.SIGN =1, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDSH 10987654321098765432109876543210 IR: 11AAAAA001010BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el half word de la dir Reg[B] +C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Half, ALIGN_IN.SIGN =1, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDUB 10987654321098765432109876543210 IR: 11AAAAA000001BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el byte de la dir Reg[B] +C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Byte, ALIGN_IN.SIGN =0, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LDUH	Acciones: Cargar en el registro A el half word de la dir Reg[B] + C.

10987654321098765432109876543210 IR: 11AAAAA000010BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Half, ALIGN_IN.SIGN =0, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP
INSTRUCCIÓN: LD 10987654321098765432109876543210 IR: 11AAAAA000000BBBBB1CCCCCCCCCCCCC PSR: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Acciones: Cargar en el registro A el word de la dir Reg[B] + C. T_0: REG_WIN.A_SEL = B, A_MUX.SEL = REG_WIN, B_MUX.SEL = IR_LATCH ALU.FCOD = ADD, ALU_RES.Enable = 1 T_1: ADDR_MUX.SEL = ALU_RES, DATA_OUT.Enable = 1, RDM.Enable = 1 [Esperar BUS_BUSY_IN = 0] BUS_BUSY_OUT = 1, RD T_2: [Esperar BUS_DACK_IN = 1] DATA_IN.Enable = 1, ALIGN_IN.Size = Word, ALIGN_IN.SIGN =0, CMUX_4.SEL = ALIGN_IN, REGWIN.C_SEL = A, REGWIN.C_EN = 1 Continuar como en NOP

Cosas pendientes:

1. Conectar ALU_RES a DATA_OUT,
2. Ver como manegar los códigos de condición de la ALU (agregar una señal de la UC que sea ALU_CARRYIN)
3. Agregar un modelo que permita seleccionar los 13 primeros bits del registro IR
4. Agregar un registro RDM