DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (I/2026)

Ayudantía 5

Ayudantes: Alberto Maturana (alberto.maturana@uc.cl), Mario Rojas (mario.denzel@estudiante.uc.cl, José Mendoza (jfmendoza@uc.cl))

Pregunta 1: Preguntas Conceptuales

a) Suponga que se hace uso del computador básico del curso para realizar las siguientes operaciones:

1. Multiplicación, a través de sumas iterativas, de los valores de los registros `A` y `B`, con `A` = 30 y `B` = 10, almacenando el resultado en A.
2. Aplicación de la operación *shift right* sobre el registro `A`, almacenando el resultado en `A`. ¿Cuál es el resultado, en representación decimal, del registro `A` luego de ejecutar las operaciones anteriores? Responda considerando que los registros almacenan valores de 8 bits sin signo.

b) ¿Cuál es el problema del *shift right* lógico con los números binarios en complemento de dos? ¿Es posible solucionarlo? ¿Que tipo de *shift right* ocupa nuestro computador básico?

Solución:

a) Considerando que los registros almacenan valores de 8 bits sin signo, el valor máximo que pueden almacenar es de 255 antes de que se genere *overflow*. Por lo tanto, dado que 30 x 10 = 300, se tendrá como resultado 300 - 256 = 44 en `A`. Luego, si se aplica un *shift right* sobre este valor, resultará en 44 // 2 = 22.

b) Como todos los bits del número se mueven una posición hacia la derecha, esto incluye al "bit de signo" que surge naturalmente del complemento de dos. Luego, como se inserta un bit igual a 0 en la posición más significativa, cambia el signo del número durante la operación. Es posible solucionar este problema mediante el uso del *shift right* aritmético: se mueven todos los bits una posición hacia la derecha, pero en vez de insertar un 0 en la posición más significativa, se inserta el bit más significativo del número original. A continuación, un ejemplo:

- $1010 \xrightarrow{\text{SHR}} 0101$
- $1010 \xrightarrow{\text{SHR-Ar}} 1101$

Esto permite que se mantenga el signo y, mejor aún, que la operación se siga comportando como una división entera por dos sobre el número original. El computador básico al trabajar

con números sin signo, no nos es necesario un SHR-Ar, por lo que nuestra instrucción SHR es lógica.

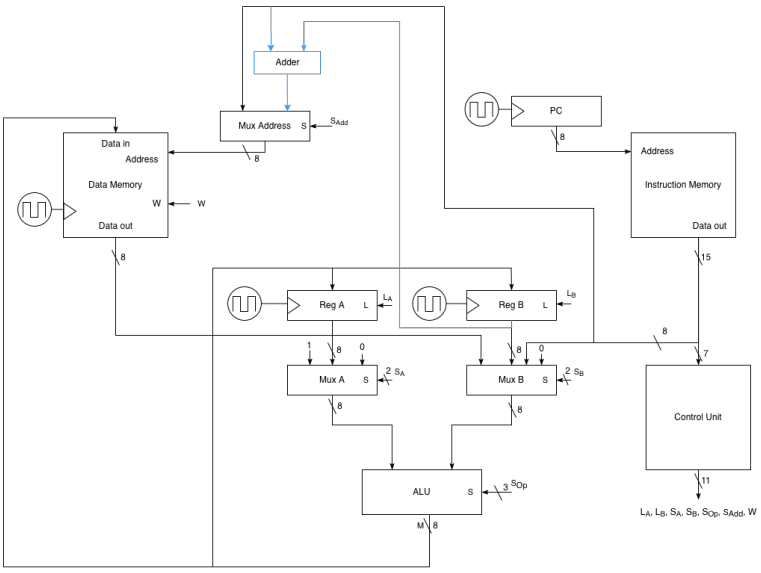
Pregunta 2: Modificación del computador básico I

Modifique la arquitectura del computador básico para implementar las instrucciones:

- `MOV A,(B+offset)`
- `MOV (B+offset),A`
- `MOV B,(B+offset)`
- `MOV (B+offset),B`

Es decir, instrucciones de direccionamiento indirecto con registro `B` y *offset*, siendo este último un literal. Para cada instrucción, deberá incluir la combinación completa de señales que la ejecutan. Por cada señal de carga y escritura, deberá indicar si se activan (1) o no (0); en las señales de selección, deberá indicar el nombre de la entrada escogida ("-" si no afecta).

Solución: Para la implementación de estas instrucciones, se puede modificar la conexión entre el registro `B` y el componente `Mux Address` para que ahora reciba el resultado de un sumador que sume el valor de `B` y el literal de la instrucción:

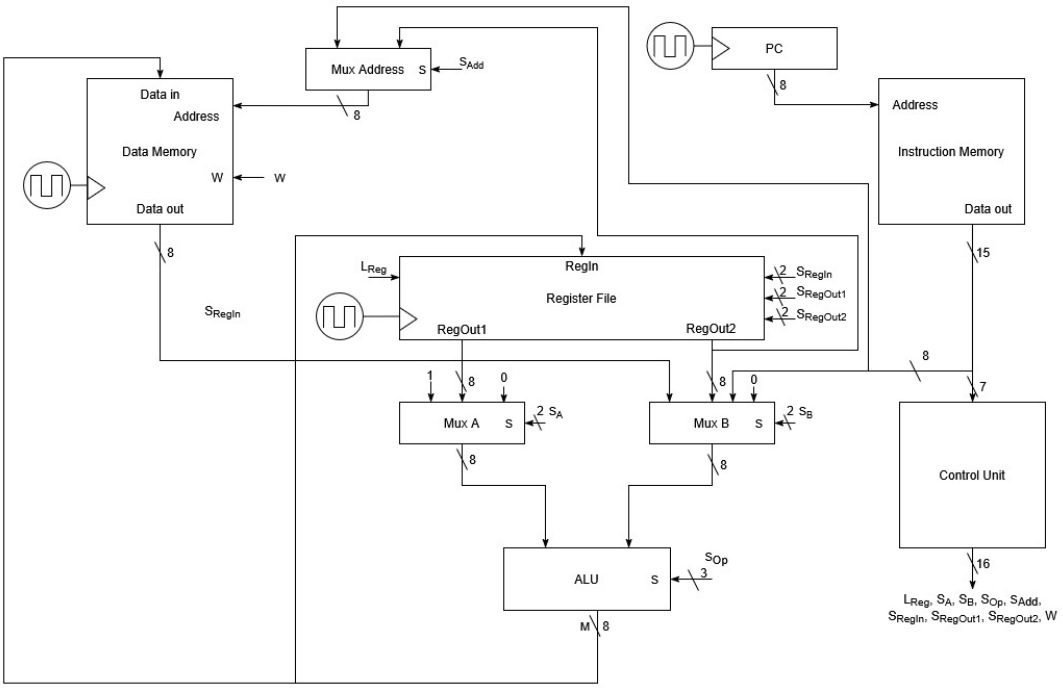


De esta forma, no es necesario agregar ni modificar señales, pero sí asegurar que el compilador haga uso del literal 0 para las instrucciones `MOV A,(B)`, `MOV B,(B)`, `MOV (B),A` y `MOV (B),B`. La tabla de señales es como sigue para las instrucciones implementadas:

Instrucción	L_A	L_B	W	S_A	S_B	S_{OP}	S_{Add}
`MOV A,(B+offset)`	1	0	0	ZERO	DOUT	ADD	B+offset
`MOV B,(B+offset)`	0	1	0	ZERO	DOUT	ADD	B+offset
`MOV (B+offset),A`	0	0	1	A	ZERO	ADD	B+offset
`MOV (B+offset),B`	0	0	1	ZERO	B	ADD	B+offset

Pregunta 3: Modificación del computador básico II

Una limitante del computador básico estudiado en clases es que solo hace uso de dos registros y, además, no se puede usar cualquiera de estos sobre todas las operaciones. En la práctica, las arquitecturas hacen uso de un *Register File*, componente con múltiples registros en su interior. A continuación, se muestra el diagrama del computador básico modificado para hacer uso de este componente, el que cuenta con 4 registros en su interior:



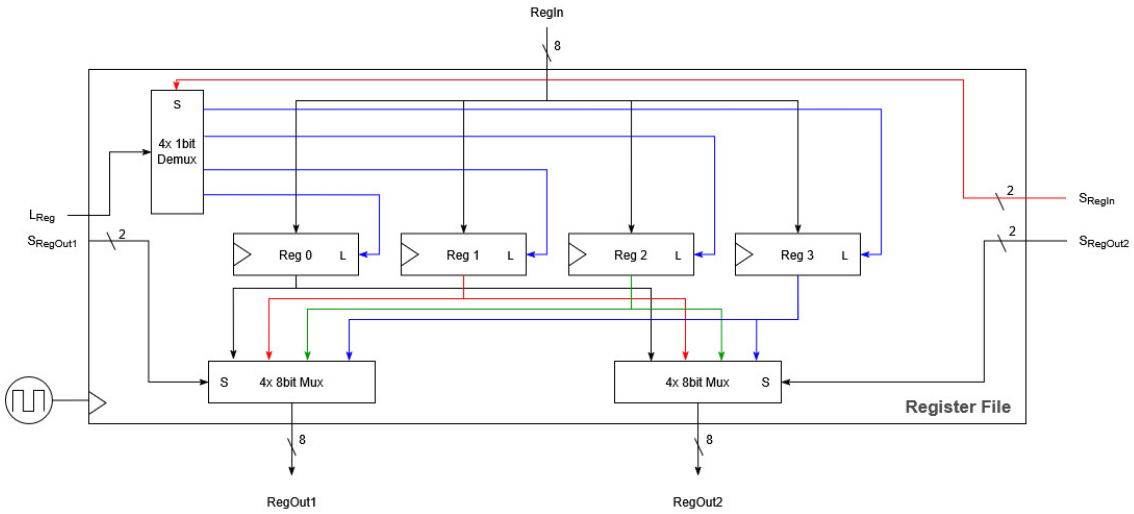
Se puede observar que, a través de las señales de control $S_{RegOut1}$ y $S_{RegOut2}$, se pueden seleccionar los registros a ser utilizados como operandos de la ALU; mientras que S_{RegIn} permite seleccionar el registro a ser sobreescrito si L_{Reg} es igual a 1. A partir de esta versión del computador básico, desarrolle las siguientes preguntas:

a) Elabore el diagrama interno del *Register File* para que cumpla con el comportamiento descrito. Puede agregar las conexiones y componentes que estime convenientes

b) Indique la combinación de señales completa para llevar a cabo la ejecución de las siguientes instrucciones:

- MOV Reg0, Reg3. Almacena en Reg0 el valor Reg3.
- MOV Reg2, Lit: Almacena en Reg2 el literal de la instrucción.
- MOV (Reg0), Reg1. Almacena en Mem[Reg0] el valor Reg1.
- XOR Reg0, Reg1, Reg2. Almacena en Reg0 el valor Reg1 XOR Reg2.
- NOT Reg1, Reg1. Almacena en Reg1 el valor NOT Reg1.

Solución: a) A continuación, se muestra una posible implementación del Register File:



En resumen:

- Se agrega el componente 4x 1bit Demux para transmitir la señal de carga LReg a solo uno de los cuatro registros internos. Se conecta a la señal S_{RegIn} para realizar la selección.
- Se conecta la señal de entrada RegIn a la entrada de todos los registros internos. Esto no es un problema, dado que su valor se actualizará solo si es que se les transmite la señal de carga habilitada según el valor de las señales L_{Reg} y S_{RegIn}.
- Se agregan dos componentes 4x 8bit Mux para seleccionar uno de los cuatro registros y transmitirlos a las salidas RegOut1 y RegOut2. Para realizar la selección, se conectan a las señales $S_{RegOut1}$ y $S_{RegOut2}$ respectivamente.
- Se asume que todos los registros se conectan a la misma señal de *clock*, no es necesario explicitar la conexión.

b) A continuación, la combinación de señales por instrucción:

Instrucción	L_{RegIn}	W	S_{RegIn}	$S_{RegOut1}$	$S_{RegOut2}$	S_A	S_B	S_{OP}	S_{Add}
MOV Reg0, Reg3	1	0	Reg0	-	Reg3	ZERO	RegOut2	ADD	-
MOV Reg2, Lit	1	0	Reg2	-	-	ZERO	LIT	ADD	-
MOV (Reg0), Reg1	0	1	-	Reg1	Reg0	Reg1	ZERO	ADD	RegOut2
XOR Reg0, Reg1, Reg2	1	0	Reg0	Reg1	Reg2	RegOut1	RegOut2	XOR	-
NOT Reg1, Reg1	1	0	Reg1	Reg1	-	RegOut1	-	NOT	-

Computador básico sin saltos

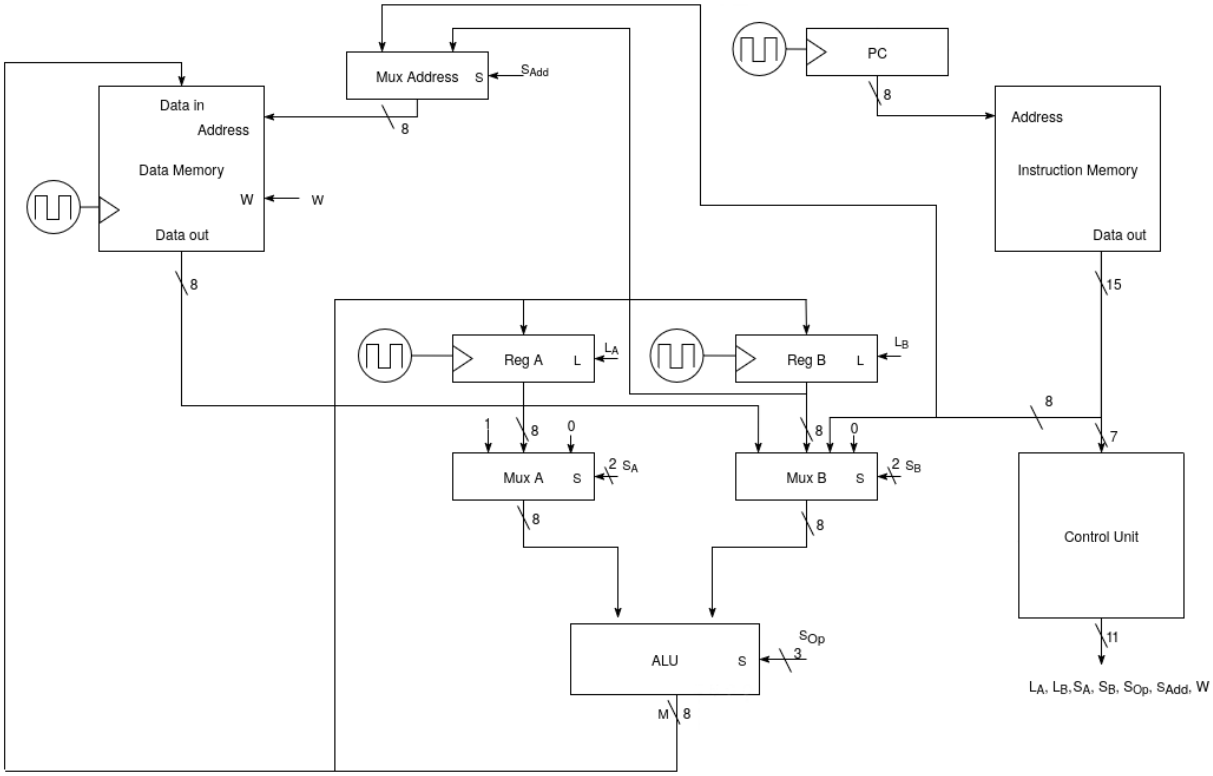


Tabla de instrucciones

Instrucción	Operandos	*Opcode*	Condición	L_A	L_B	S_A0,1	S_B0,1	S_{Op}0,1,2	S_{Add}0,1	W
MOV	A,B	0000000		1	0	ZERO	B	ADD	-	0
	B,A	0000001		0	1	A	ZERO	ADD	-	0
	A,Lit	0000010		1	0	ZERO	LIT	ADD	-	0
	B,Lit	0000011		0	1	ZERO	LIT	ADD	-	0
	A,(Dir)	0000100		1	0	ZERO	DOUT	ADD	LIT	0
	B,(Dir)	0000101		0	1	ZERO	DOUT	ADD	LIT	0
	(Dir),A	0000110		0	0	A	ZERO	ADD	LIT	1
	(Dir),B	0000111		0	0	ZERO	B	ADD	LIT	1
	A,(B)	0001000		1	0	ZERO	DOUT	ADD	B	0
	B,(B)	0001001		0	1	ZERO	DOUT	ADD	B	0
	(B),A	0001010		1	0	A	ZERO	ADD	B	1
ADD	A,B	0001011		1	0	A	B	ADD	-	0
	B,A	0001100		0	1	A	B	ADD	-	0
	A,Lit	0001101		1	0	A	LIT	ADD	-	0
	A,(Dir)	0001110		1	0	A	DOUT	ADD	LIT	0
	A,(B)	0001111		1	0	A	DOUT	ADD	B	0
	(Dir)	0010000		0	0	A	B	ADD	LIT	1
SUB	A,B	0010001		1	0	A	B	SUB	-	0
	B,A	0010010		0	1	A	B	SUB	-	0
	A,Lit	0010011		1	0	A	LIT	SUB	-	0
	A,(Dir)	0010100		1	0	A	DOUT	SUB	LIT	0
	A,(B)	0010101		1	0	A	DOUT	SUB	B	0
	(Dir)	0010110		0	0	A	B	SUB	LIT	1
AND	A,B	0010111		1	0	A	B	AND	-	0
	B,A	0011000		0	1	A	B	AND	-	0
	A,Lit	0011001		1	0	A	LIT	AND	-	0
	A,(Dir)	0011010		1	0	A	DOUT	AND	LIT	0
	A,(B)	0011011		1	0	A	DOUT	AND	B	0
	(Dir)	0011100		0	0	A	B	AND	LIT	1
OR	A,B	0011101		1	0	A	B	OR	-	0
	B,A	0011110		0	1	A	B	OR	-	0
	A,Lit	0011111		1	0	A	LIT	OR	-	0
	A,(Dir)	0100000		1	0	A	DOUT	OR	LIT	0
	A,(B)	0100001		1	0	A	DOUT	OR	B	0
	(Dir)	0100010		0	0	A	B	OR	LIT	1
NOT	A,A	0100011		1	0	A	-	NOT	-	0
	B,A	0100100		0	1	A	-	NOT	-	0
	(Dir)	0100101		0	0	A	B	NOT	LIT	1
XOR	A,B	0100110		1	0	A	B	XOR	-	0
	B,A	0100111		0	1	A	B	XOR	-	0
	A,Lit	0101000		1	0	A	LIT	XOR	-	0
	A,(Dir)	0101001		1	0	A	DOUT	XOR	LIT	0
	A,(B)	0101010		1	0	A	DOUT	XOR	B	0
	(Dir)	0101011		0	0	A	B	XOR	LIT	1
SHL	A,A	0101100		1	0	A	-	SHL	-	0
	B,A	0101101		0	1	A	-	SHL	-	0
	(Dir)	0101110		0	0	A	B	SHL	LIT	1
SHR	A,A	0101111		1	0	A	-	SHR	-	0
	B,A	0110000		0	1	A	-	SHR	-	0
	(Dir)	0110001		0	0	A	B	SHR	LIT	1
INC	B	0110010		0	1	ONE	B	ADD	-	0
	(B)	0110011		0	0	ONE	DOUT	ADD	B	1
	(Dir)	0110100		0	0	ONE	DOUT	ADD	LIT	1

Tabla 1: ISA del computador básico.

Feedback ayudantía

Escanee el QR para entregar feedback sobre la ayudantía.

