Laboratorio - Organización de la CPU

Objetivos

- Ejercitar conceptos vistos en el curso sobre diseño de circuitos digitales.
- Familiarizarse con el funcionamiento interno de la CPU a través de la implementación de varios componentes internos de su microarquitectura.

Descripción de la tarea

Se desea implementar una CPU simple de una arquitectura de 8 bits conectada a través del modelo de Von Neumann.

Para su realización, los estudiantes deberán descargar del sitio EVA del curso una implementación *incompleta* de una CPU y realizar las modificaciones indicadas más adelante en la letra para asegurar su correcto funcionamiento. La CPU sobre la que se trabajará tiene las siguientes características:

- Bus de datos de 8 bits y bus de direcciones de 16 bits.
- 16 registros de propósito general, de 8 bits, de nombres Reg0, Reg1, Reg2..., Reg15. IR de 24 bits y PC de 16 bits.
- Entradas: reloj (**CLK**), **reset** activo por nivel bajo, buses de datos, dirección y control.
- Registro de banderas (**Zero, Negativo, Carry, Overflow**)
- Implementa un set de instrucciones de largo variable, múltiplo de 8 bits.

Set de Instrucciones

Instrucción	Descripción
`NOP`	No hace nada
`MOVI Reg1,INM8`	Carga el inmediato INM8 en Reg1.
`MOVR Reg1, Reg2`	Carga en Reg1 el contenido de Reg2
`MOVF Reg1`	Carga el registro FLAGS en la parte baja de Reg1
`LOAD Reg1, Reg2, Reg3`	Carga el contenido de memoria apuntado por la dirección Reg2Reg3 en Reg1
`SAVE Reg1, Reg2, Reg3`	Guarda el registro Reg1 en la dirección de memoria apuntada por Reg2Reg3
`OPER Reg1, Reg2`	*Realiza la operación de ALU especificada en OPER, entre Reg1 y Reg2, y guarda el resultado en Reg1. Las operaciones de ALU disponibles son: ADD, SUB, AND, OR, XOR, NOT, SHL y SHR. ADD y SUB afectan las banderas C y V.*
`OPERI Reg1, INM8`	*Realiza la operación de ALU especificada en OPERI, entre Reg1 y INM8, y guarda el resultado en Reg1.*
`MUL16 Reg1, Reg2, Reg3, Reg4`	Realiza la multiplicación entre los registros Reg3 y Reg4 y guarda el resultado en la pareja de registros Reg1 (byte alto) y Reg2 (byte bajo)
`JMP INM16`	Realiza un salto incondicional a la dirección INM16.
`JZ INM16`	Realiza un salto a la dirección INM16 si la bandera de Z vale 1.
`JC INM16`	Realiza un salto a la dirección INM16 si la bandera de C vale 1.

Nota: Además de lo mencionado, las operaciones ADD, SUB, AND, OR, XOR, NOT, SHR, SHL y MUL16 afectan las banderas Z y N.

El formato de instrucción que se va a utilizar es el siguiente:

Instrucción	Largo	Byte1	Byte2	Byte3
NOP	1 byte	00000000	-	-
MOVI	2 bytes	0001Reg1	INM8	-
MOVR	2 bytes	0010Reg1	Reg20000	-
MOVF	1 byte	0011Reg1	-	-
LOAD	2 bytes	0100Reg1	Reg2Reg3	-

SAVE	2 bytes	0101Reg1	Reg2Reg3	-
OPER	2 bytes	01100**OPER**	Reg1Reg2	
OPERI	3 bytes	01110**OPER**	Reg10000	INM8
MUL16	3 bytes	1000Reg1	Reg2Reg3	Reg40000
JMP	3 bytes	10010000	INM(L)	INM(H)
JZ	3 bytes	10100000	INM(L)	INM(H)
JC	3 bytes	10110000	INM(L)	INM(H)

La codificación de OPER es la siguiente:

ADD	000	SUB	001
AND	010	OR	011
XOR	100	NOT	101
SHL	110	SHR	111

Observaciones

- Pese a tener una capacidad de direccionamiento de 2^{16} palabras, físicamente la CPU se conecta a una memoria de 256 palabras.
- El orden de las flag es ZNCV
- La instrucción NOT ignora el segundo parámetro de la instrucción.
- El largo del ciclo de instrucción a implementar **debe** seguir la filosofía CISC. Es decir que las instrucciones **deben** tener un largo de ciclo variable en función de la complejidad de la misma.
- Se proveerá un conjunto de datos de prueba, el cual funcionará como guía. Las pruebas se darán en el formato de editor de ondas del MAX+Plus II (.scf). Debido a que el largo del ciclo de instrucción depende de la solución, y que en la simulación figuran valores

definidos para señales que en algún tiempo son *don't care*, es esperable que algunos tiempos puedan diferir con las pruebas provistas, y aún así ser una solución correcta.

Se Pide

1) Implementar con MAX+Plus II el circuito 8bit_shl.gdf y 8bit_shr.gdf, que dado dos números de 8 bits, realizan el corrimiento hacia la izquierda (o derecha) del primer operando, tantos lugares como indique el segundo operando.
2) Implementar con MAX+PLUS II el circuito 8bit_mul.gdf, que dado dos números de 8 bits representados en complemento a 2, los multiplica y devuelve el resultado en complemento a 2 de 16 bits. Dicho circuito es utilizado en la ALU. El símbolo generado (8bit_mul.sym) debe ser idéntico al entregado por lo docentes.
3) Construir con MAX+PLUS II los circuitos 'unidad de control' y 'registros internos' de la CPU, de modo que la CPU implemente el set de instrucciones indicado anteriormente. Se deberá implementar un ciclo de instrucción de largo variable. Los símbolos generados (control_unit.sym e internal_registers.sym) deberán ser idénticos a los entregados por los docentes.
4) Implementar un módulo de prueba de la CPU que ejecute un programa que calcule la suma en complemento a 2 de las posiciones de memoria entre 0xF0 y 0xFF. Si se produce overflow, se deberá escribir 0x01 en la posición de memoria 0xEE, en caso contrario, se deberá escribir el resultado en la posición 0xEF.

Nota: para las partes 1), 2) y 3), no se deben modificar ninguno de los módulos entregados por los docentes.

Forma de entrega y grupos de trabajo

El trabajo debe realizarse en grupo de exactamente cuatro personas y cada grupo debe entregar un único archivo (laboratorio.zip) que contenga en su interior:

- Un directorio con el proyecto Max+Plus II que contenga la CPU entregada por los docentes más los archivos .gdf y .sym necesarios para la implementación de las partes 1), 2), y 3). El proyecto debe incluir un archivo del editor de ondas con los casos de prueba más la prueba del programa pedido en la parte 4). Además se deberá entregar un archivo de inicialización de memoria (ram.mif) que implemente el programa pedido
- Documentación escrita que describa la implementación de los módulos y del programa. La entrega se realizará a través del sitio EVA del curso.

Los trabajos deberán ser entregados indefectiblemente antes del domingo 14 de octubre a las 23.30 horas. No se aceptará ningún trabajo pasada la citada fecha. En particular, no se aceptarán trabajos enviados por mail a los docentes del curso, ni entregados en medios magnéticos en el instituto. El sistema de entregas soporta múltiples entregas por grupo. Pruebe de realizar una entrega vacía con tiempo, a los efectos de verificar que su sistema le permite entregar correctamente.