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

IIC2343 - Arquitectura de Computadores (II/2019)

Entrega 3

Entrega: Lunes 28 de Octubre | 10:59:59 a.m.

Requisitos

- Esta entrega es de carácter grupal. Cualquier tipo de falta a la honestidad académica será sancionada con la **reprobación** del curso con la nota mínima.
- Los nombre de archivos y el cómo deben ser ejecutados son parte del formato, no respetarlo será penalizado.
- El programa de la **placa** deberá ser realizado en `VHDL`.
- La **documentación** deberá ser realizada en un archivo `Markdown` y subirlo junto a su tarea, de nombre README.md, en el mismo repositorio.
- Esta entrega deberá ser subida a su repositorio grupal de *GitHub* antes de la fecha y hora dada.

Objetivo

Para esta entrega tendrán que diseñar y armar su propio computador básico de 16 bits, el que utilizarán posteriormente para ejecutar programas hechos en lenguaje *assembly*. Lo que deberán armar en Vivado es su propia CPU, la cual deberá cumplir con una serie de requisitos descritos más abajo, dentro de los cuales se encuentra soportar una lista determinada de instrucciones en *assembler*.

Adicionalmente, deberán crear un ejecutable o un archivo `Python3` que escriba un programa en la ROM basado en un archivo escrito en lenguaje assembler del computador. Para mayor información, revise el enunciado de la entrega del Assembler.

Especificaciones CPU

Sin entrar todavía en detalles, el computador que deberán armar debe contar con:

- **Todos los componentes de la entrega anterior** con sus input y output respectivos
- Un **registro de *stack*** de 16 *bits* (registro `SP`) con opcion de incrementar o decrementar en 1.
- Un **multiplexor pc** (MUX pc) que seleccionen entre dos entradas, cada una de 16 *bits*.
- Un **multiplexor datain** (MUX datain) que selecciona el dato de entrada de la *Main Memory,* ambos de 16 *bits*.
- Un **multiplexor s** (MUX s) que seleccionen entre la dirección a que apuntará la *Main Memory*, ambos de 12 *bits*.
- Un ***Adder*** que dados dos valores de 12 bits da el resultado de su suma de ambas entradas; y su salida también es de 16 bits *bits*.

En la Figura 1 se muestra el diagrama del computador básico recién descrito, donde se pueden apreciar cada uno de los bloques mencionados junto con las interconexiones pertinentes (además de la indicación del tamaño de cada bus). Cabe mencionar que todos los bloques síncronos de la Figura 1 son de flanco de subida.

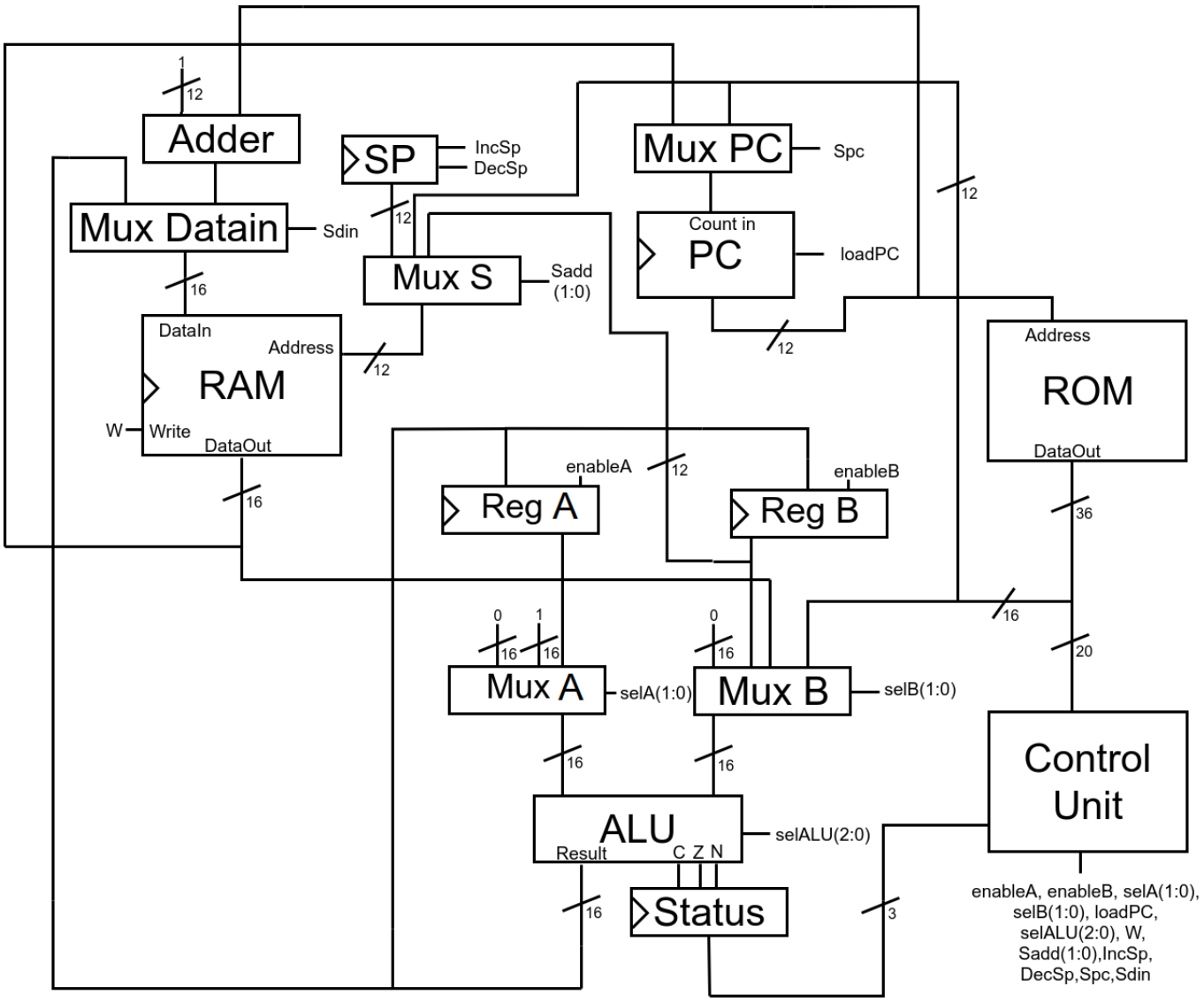

Figura 1: Computador básico.

A continuación, la lista de instrucciones que su computador básico deberá soportar:

Entrega 2

MOV	A,B	guarda B en A
	B,A	guarda A en B
	A,Lit	guarda un literal en A
	B,Lit	guarda un literal en B
	A,(Dir)	guarda Mem[Dir] en A
	B,(Dir)	guarda Mem[Dir] en B
	(Dir),A	guarda A en Mem[Dir]
	(Dir),B	guarda B en Mem[Dir]
ADD SUB AND OR XOR	A,B	guarda A op B en A
	B,A	guarda A op B en B
	A,Lit	guarda A op literal en A
	B,Lit	guarda A op literal en B
	A,(Dir)	guarda A op Mem[Dir] en A
	B,(Dir)	guarda A op Mem[Dir] en B
	(Dir)	guarda A op B en Mem[Dir]
NOT SHL SHR	A	guarda op A en A
	B,A	guarda op A en B
	(Dir),A	guarda op A en Mem[Dir]
INC	A	incrementa A en una unidad
	B	incrementa B en una unidad
	(Dir)	incrementa Mem[Dir] en una unidad
DEC	A	decrementa A en una unidad
CMP	A,B	hace A-B
	A,Lit	hace A-Lit
	A,(Dir)	hace A-Mem[Dir]
JMP	Ins	carga Ins en PC
JEQ	Ins	carga Ins en PC si en el status Z = 1
JNE	Ins	carga Ins en PC si en el status Z = 0
NOP		no hace cambios

Entrega 3

MOV	A,(B)	guarda Mem[B] en A
	B,(B)	guarda Mem[B] en B
	(B),A	guarda A en Mem[B]
	(B),Lit	guarda Lit en Mem[B]
ADD SUB AND OR XOR	A,(B)	guarda A op Mem[B] en A
	B,(B)	guarda A op Mem[B] en B
NOT SHL SHR	(B),A	guarda op A en Mem[B]
INC	(B)	incrementa Mem[B] en una unidad
CMP	A,(B)	hace A-Mem[B]
JGT	Ins	carga Ins en PC si en el status N = 0 y Z = 0
JGE	Ins	carga Ins en PC si en el status N = 0
JLT	Ins	carga Ins en PC si en el status N = 1
JLE	Ins	carga Ins en PC Ins si en el status N = 1 o Z = 1
JCR	Ins	carga Ins en PC Ins si en el status C = 1
PUSH	A	guarda A en Mem[SP] y decrementa SP
	B	guarda B en Mem[SP] y decrementa SP
POP	A	incrementa SP y luego guarda Mem[SP] en A
	B	incrementa SP y luego guarda Mem[SP] en B
CALL	Ins	guarda PC+1 en Mem[SP], carga Ins en PC y decrementa SP
RET		incrementa SP y luego carga Mem[SP] en PC

Tablas de verdad componentes

A continuación se mostrará la tabla de verdad de algunos de los componentes del computador de esta entrega.

- `SP`

Entradas			Salidas
clock	up	down	dataout
Flanco Subida	*	*	dataout
Flanco Subida	1	0	dataout + 1
Flanco Subida	0	1	dataout - 1

Figura 2: Tabla de verdad de SP.

NOTA: El SP debe partir en `"111111111111"`.

- `Adder`

Entradas		Salidas
a(11:0)	b(11:0)	result(15:0)
*	*	a + b

Figura 3: Tabla de verdad de Adder.

Bitácora de proyecto

Adicionalmente, para esta entrega su grupo deberá actualizar su bitácora **en formato Markdown** que hable de los siguientes puntos:

- Explicación del funcionamiento de cada componente de su implementación del computador básico y con especial detalle en lo que se refiere la unidad de control (arquitectura interna y señales).
- La especificación de la estructura de las instrucciones de su CPU, es decir, la codificacíon de cada uno de los bits de su palabra de instrucción.
- El trabajo realizado por cada uno de los miembros de su grupo. Se debe indicar específicamente que hizo cada integrante del grupo y deben explicar que fue lo mas difícil para el grupo en la entrega.
- Especificaciones sobre el uso de su Assembler.
- Una tabla o referencia donde se indique que instrucciones son soportadas por su implementación de acuerdo a la especificación dada anteriormente.

Requerimientos

- Soportar las instrucciones de la entrega pasada.
- Extender la arquitectura necesaria, de acuerdo al enunciado para soportar las nuevas instrucciones.
- Crea un ejecutable para escribir la ROM con un archivo Assembler.
- **Incluir el README.md, el Informe y los archivos correspondientes con lo solicitado.**

Entrega

La entrega se realizará a través de GitHub. El repositorio debe contener una carpeta con su proyecto de Vivado, el ensamblador y los archivos `.bit`. En el caso de la carpeta del proyecto, deben subir solo la carpeta `.srcs`, el archivo `.xpr` y el archivo `Basys3.xdc` (las carpetas y archivos restantes se recomienda evitar su subida configurando el archivo `.gitignore`). Para el *assembler*, deberá subir su ejecutable o archivo `Python3` solicitado, el cual debe estar en su propia carpeta. Finalmente, los archivos `.bit` deben estar dentro de su propio directorio y tener el nombre correspondiente al algoritmo en assembly desde el que se creo.

En resumen, dentro de su repositorio deben tener tres carpetas, la del proyecto de Vivado, la del *assembler* y la de los archivos `.bit`. Documentos como la bitácora y el `README`, debe estar en el repositorio, en el directorio raíz.

Evaluación

La evaluación de su entrega se enfocará en los siguientes puntos:

- El uso correcto de Vivado y escritura en VHDL, es decir, que los archivos se desplieguen correctamente al abrir el archivo .xpr y que estos sean capaces de compilar el .bit
- El correcto funcionamiento de su ensamblador para la escritura de la ROM.
- El correcto funcionamiento de los archivos .bit solicitados.

- **Una semana antes se les va entregar una serie de algoritmos en assembly para los cuales deberán generar y entregar los .bit**
- El contenido de la bitácora. Este debe explicar con claridad los elementos que se pidieron.
- **Evaluación de Pares.** Deberán contestar un form semanalmente respecto al trabajo que han realizado juntos. Deberán indicar si han trabajado como grupo y quienes han trabajado. No contestar este form significará un descuento en su nota.

Integridad académica

Los alumnos de la Escuela de Ingeniería de la Pontificia Universidad Católica de Chile deben mantener un comportamiento acorde a la Declaración de Principios de la Universidad. En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un Procedimiento Sumario. Es responsabilidad de cada alumno conocer y respetar el documento sobre Integridad Académica publicado por la Dirección de Docencia de la Escuela de Ingeniería.

Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica. Todo trabajo presentado por un alumno para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno, sin apoyo en material de terceros. Por "trabajo" se entiende en general las interrogaciones escritas, las tareas de programación u otras, los trabajos de laboratorio, los proyectos, el examen, entre otros. Si un alumno copia un trabajo, obtendrá nota final 1,1 en el curso y se solicitará a la Dirección de Docencia de la Escuela de Ingeniería que no le permita retirar el curso de la carga académica semestral. Por "copia" se entiende incluir en el trabajo presentado como propio partes hechas por otra persona.

Obviamente, está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la referencia correspondiente.

Lo anterior se entiende como complemento al Reglamento del Alumno de la Pontificia Universidad Católica de Chile. Por ello, es posible pedir a la Universidad la aplicación de sanciones adicionales especificadas en dicho reglamento.