

Esta tarea debe de realizarse en equipos de hasta 4 estudiantes.

Los archivos que componen el proyecto deben de entregarse digitalmente mediante mediación virtual a más tardar el viernes 22 de Julio a las 11:59 pm. Cada hora de entregar tardía restara 1 punto a la calificación final de la tarea.

La presentación (demostración) del proyecto debe de realizar el martes 19 de Julio en horas de clase. Cada equipo tendrá un tiempo máximo de 30 minutos.

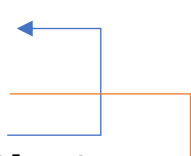
Esta tarea deberá realizarse en el lenguaje de descripción de hardware Verilog. La implementación deberá hacerse a nivel RTL (Register Transfer Level). No se penalizará el uso de lenguaje a nivel de compuertas, pero no se recomienda.

El proyecto consiste en completar la implementación del procesador que se comenzó en la Tarea 1, interconectando los componentes que fueron desarrollados en dicha tarea como también completando cualquier funcionalidad faltante para la ejecución de un programa de prueba mínimo.

El programa de pruebas deberá calcular el valor de 12 factorial (12!) (de manera ascendente, esto es, multiplicando $1 \times 2 \times 3 \times \dots \times 12$ y no $12 \times 11 \times \dots \times 1$) y almacenar el resultado en la posición de memoria 0x8000.

A manera de ejemplo, el siguiente programa ejecuta la suma de los 100 primeros números enteros de manera descendiente y lo almacena en la posición 0x1000

Dirección	Código ensamblador/simbólico	Código máquina (HEX)
0x0000	LD R0, 100	0x09000064
0x0001	LD R1, 1	0x09010001
0x0002	LD R2, 0	0x09020000
0x0003	ADD R2, R0	0xA6020000
0x0004	SUB R0, R1	0xB2000001
0x0005	JZ 0x0007	0xF1000007
0x0006	JMP 0x0003	0xD1000003
0x0007	STR [0x1000], R2	0x0B021000



El programa de pruebas deberá estar precargando en archivo verilog que implementa la memoria o puede cargarse haciendo uso de la función **readmemh()**.

Alguno de los opcodes/instrucciones implementadas en la Tarea 1 pueden ser necesarios para el programa de pruebas (pero no necesariamente todos). Se debe de mantener el mismo formato para las instrucciones e implementar al menos los opcodes faltantes para la ejecución del programa de prueba.

Forma General																																		
	3	3	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8
	Opcode				Dir				Operando A								Operando B																	

Entregables:

1. Todos los archivos verilog (.v y .vh) necesarios para “compilar” el CPU, su memoria y su testbench.
2. Documentación externa:
 - a. Tabla con las instrucciones que fueron implementadas (ADD, SUB, MUL, LOAD, ...). Debe de seguirse el formato usado en el enunciado de la Tarea 1.
 - b. Resultados de la ejecución del programa (Puede ser una captura de pantalla del gtwave, donde se detallen momentos importantes de la ejecución. Entre ellos debe estar la operación de STORE final donde se almacene el resultado)
 - c. Se debe de detallar cualquier problema encontrado en el proceso y principalmente cualquier problema no resuelto
3. Presentación tipo demostración (no tiene que ser en vivo. Puede utilizar una presentación con capturas de pantalla) donde se muestre la ejecución del programa. También se deberá abordar la sección de problemas encontrados que forman parte de la documentación externa.

Calificación:

1. Implementación del CPU en verilog [--30%--]
2. Los archivos .v que componen al sistema de CPU + Memoria compilan [--10%--]
3. Documentación externa [--10%--]
4. La ejecución del programa de prueba finaliza correctamente y muestra el resultado correcto [--20%--]
5. Presentación/Demostración del proyecto [--30%--]