Tarea 3

Lenguajes de Descripción de Hardware (HDL)

Reglas generales

En esta tarea deberá hacer uso de la plataforma online EDA Playground[1] para diseñar la estructura de una ALU simple y todas sus funciones, además de presentar un testbench que permita comprobar el funcionamiento del diseño. Se recomienda que utilice la configuración de la tabla 1 dentro de EDA Playground.

Además del diseño deberá confeccionar un informe que detalle el proceso de resolución de la tarea y que utilice tablas y figuras como apoyo. El informe debe ser un único documento en formato `.pdf` que incluya un título, resumen, introducción, desarrollo, resultados, análisis y conclusiones. Se recomienda el uso de LaTeX para la realización del informe, y organizar su trabajo en secciones y subsecciones para mantener el orden del documento.

Languages & Libraries

Testbench + Design	SystemVerilog/Verilog	**UVM/OVM**	None
Other Libraries	None	**Enable TL-Verilog**	No
Enable Easier UVM	No	**Enable VUnit**	No

Tools & Simulators

Compiler	Aldec Riviera Pro 2020.04	**Compile Options**	`-timescale 1ns/1ns`
Run Options	`+access+r`	**Run Time**	10 ms
Use run.do Tcl file	No	**Use run.bash shell script**	No
Open EPWave after run	Si (para ver resultados)	**Download files after run**	No

Tabla 1: Configuración recomendada para EDA Playground

Enunciado

La mejora a la máquina para predecir el resultado del torneo dejó impresionados tanto a los managers como a los competidores, y ahora una gran empresa de tecnología le está ofreciendo la oportunidad de comercializar su producto para aplicarse en otras competencias. Temiendo que alguien pueda darle un mal uso a la máquina que construyó decide anotar todos los detalles necesarios y luego destruir el único ejemplar, para después utilizar un lenguaje de descripción de hardware y escribir formalmente la estructura de la máquina.

El único problema que encuentra con esta idea es que no está seguro si el fabricante tiene los mismos componentes que utilizó al construir la máquina, por lo que comienza por diseñar un sistema que pueda calcular los valores desados, además de unas cuantas operaciones extras que añade como medida de seguridad en caso de que alguien quiera averiguar el funcionamiento de la máquina mediante ingeniería inversa, algo como si fueran señuelos que no participan realmente de la operación.

Las operaciones que podrá resolver el circuito se dividen en 4 categorías: aritméticas, lógicas, de transposición, y relacionadas al torneo. Teniendo en cuenta que la máquina toma dos valores en Complemento a 2 de entrada y entrega uno de salida, las operaciones son las que se muestran en la tabla 2.

Cabe recordar que las operaciones del torneo dependen de las ecuaciones previamente utilizadas, solo que esta vez deben ser calculadas en el momento usando los otros componentes diseñados ya que no se tiene claro quiénes serán los clientes finales. Dado que la máquina tiene solo 2 entradas y las ecuaciones tienen más de un parámetro, cuando se encuentre una instrucción del torneo se recibirán datos durante n ciclos en los que no se debe entregar una salida, y luego de recibir todos los datos se debe entregar 00000000 (0x00) en caso de ganar el primer competidor, y 11111111 (0xff) en otro caso. Más detalles al respecto se pueden ver en la tabla 3.

[1] Disponible en https://www.edaplayground.com/

Joaquín Montes joaquin.montes@sansano.usm.cl
Juan Pablo Varas juan.varash@sansano.usm.cl
Andrés Gómez andres.gomez@sansano.usm.cl

Operaciones aritméticas

A + B	Suma con Carry Look-ahead Adder (CLA)
A − B	Resta con Ripple-carry Adder (RCA)
A × B	Multiplicación
A / B	División entera (cuociente sin resto)

Operaciones lógicas

NOT A	Bitwise NOT
A OR B	Bitwise OR
A AND B	Bitwise AND
A XOR B	Bitwise XOR

Operaciones de transposición

A LSL B	Logical shift left
A LSR B	Logical shift right
A ASR B	Arithmetic shift right
A ROL B	Rotate left
A ROR B	Rotate right

Operaciones del torneo

Peso ligero	A y B son peso ligero
Peso pesado	A y B son peso pesado
Peso mixto	A y B son de pesos distintos

(a) Detalles por operación

Operación	Codificación
Suma	0000
Resta	0001
Multiplicación	0010
División entera	0011
Bitwise NOT	0100
Bitwise OR	0101
Bitwise AND	0110
Bitwise XOR	0111
Logical shift left	1000
Logical shift right	1001
Arithmetic shift right	1010
Rotate left	1011
Rotate right	1100
Peso ligero	1101
Peso pesado	1110
Peso mixto	1111

(b) Codificación de las operaciones

Tabla 2: Operaciones de la ALU simple

Para permitir que la máquina funcione de forma autónoma requiere de una ROM simple con la serie de instrucciones a ejecutar, y un Program Counter (PC) que itere sobre la ROM. Además decide agregar una serie de indicadores (flags) en la salida de la máquina para que un operador pueda saber qué está ocurriendo. Los detalles del diseño se pueden ver esbozados en la figura 1, y los indicadores a implementar en la tabla 4.

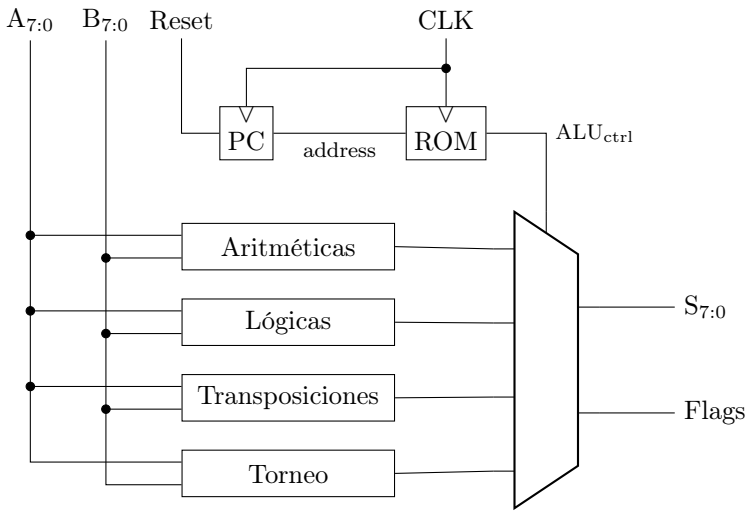

Figura 1: Diagrama simple del circuito a diseñar

Joaquín Montes joaquin.montes@sansano.usm.cl
Juan Pablo Varas juan.varash@sansano.usm.cl
Andrés Gómez andres.gomez@sansano.usm.cl

Operación	Ciclo	Parámetro	Resultado
Peso ligero	1	Estatura	-
	2	AGI	$\frac{\text{Estatura}}{\text{AGI}} + \frac{100}{\text{Estatura}} + \text{AGI}$
Peso pesado	1	Peso	-
	2	RES	$5 * \text{Peso} + 2 * \text{RES}$
Peso mixto	1	Estatura	-
	2	AGI	-
	3	Peso	-
	4	STR	-
	5	RES	$\frac{\text{Estatura}}{\text{AGI}} + 3 * \text{Peso} + \frac{\text{STR+AGI+RES}}{3}$

Tabla 3: Orden de lectura de parámetros para operaciones del torneo

Flag	Condición
A	Si se ejecuta una operación aritmética
L	Si se ejecuta una operación lógica
T	Si se ejecuta una operación de transposición
R	Si se ejecuta una operación del torneo
Z	Si el resultado es cero
N	Si el resultado es negativo
C	Si el resultado produce un carry
V	Si el resultado produce un overflow
E	Si el resultado produce un error (división por cero)

Tabla 4: Indicadores (Flags)

Joaquín Montes joaquin.montes@sansano.usm.cl
Juan Pablo Varas juan.varash@sansano.usm.cl
Andrés Gómez andres.gomez@sansano.usm.cl

Entrada y salida de datos

La entrada de datos se hará cargando la ROM diseñada con una serie de operaciones al iniciar la simulación del circuito, con hasta un máximo de 16 operaciones. Estas operaciones deben ser trabajadas una a una usando un reloj y el Program Counter. Luego se entregarán datos en dos registros de 8 bits, A y B. En el caso de las operaciones del torneo, la operación se repetirá la cantidad de veces necesaria para recibir todos los parámetros.

La salida de datos debe hacerse imprimiendo los valores que se obtienen tras cada ciclo del circuito a través de la consola/output de EDA Playground. Se debe incluir tanto el resultado de la operación como el valor de todas las flags. El resultado se entrega a través de un solo registro de 8 bits.

Datos de ejemplo

Para todos los siguientes ejemplos se usaron los datos de la figura 5b cargados en la ROM. Los inputs, el output, y las flags están codificadas como números hexadecimales. Las flags están ordenadas según la tabla 4.

ROM	Operación	Input A	Input B	Output S	Flags
0	Suma	7f	2f	ae	10a
1	Suma	ff	01	00	112
2	División entera	11	04	04	100
3	Bitwise NOT	55	--	aa	088
4	Peso ligero	64	60	--	--
5	Peso ligero	07	04	ff	028
6	Peso mixto	3c	53	--	--
7	Peso mixto	04	04	--	--
8	Peso mixto	0f	15	--	--
9	Peso mixto	06	01	--	--
10	Peso mixto	05	05	00	020
11	Multiplicación	03	05	0f	100
12	Arithmetic shift right	ff	02	ff	048
13	Logical shift right	ff	02	3f	040
14	Rotate left	aa	03	55	040
15	División entera	3b	00	00	101

(a) Datos de ejemplo

ROM

0	0
	0
	3
	4
4	d
	d
	f
	f
8	f
	f
	f
	2
12	a
	9
	b
	3

(b) Operaciones en hexadecimal

Tabla 5: Ejemplos de la ejecución de la tarea

Joaquín Montes joaquin.montes@sansano.usm.cl
Juan Pablo Varas juan.varash@sansano.usm.cl
Andrés Gómez andres.gomez@sansano.usm.cl

Consideraciones

- **La fecha de entrega para la tarea es el miércoles 16 de junio de 2022 a las 23:55 hrs.**
- **Se descontarán 25 puntos de la nota máxima por cada día o fracción de atraso, hasta un máximo de 50 puntos. Cualquier atraso sobre 48 horas se evaluará con nota 0.**
- La tarea debe realizarse en parejas (grupos de 2 personas). Ante cualquier sospecha de copia o trabajo colaborativo entre grupos se informará a las autoridades correspondientes y se evaluará con nota 0.
- La tarea se debe entregar via Aula en un solo archivo comprimido en formato `.zip` de nombre `T3_APELLIDOP1_APELLIDOP2.zip` en orden alfabético que incluya los siguientes archivos:
  - Un solo archivo `README.txt` con el nombre y ROL USM de cada integrante, además de cualquier aclaración que sea necesaria.
  - Dos archivos `.sv` con el diseño y el testbench.
  - Un solo archivo `.pdf` con el informe completo del desarrollo de la tarea.
- El informe debe contener las siguientes secciones, cada una ordenada y con toda la información necesaria:
  - Portada, incluyendo el nombre y ROL USM de los integrantes, además de un título descriptivo.
  - Resumen, donde describa brevemente el desarrollo y resultados de la tarea.
  - Introducción, dejando claro el objetivo de la tarea y cualquier algoritmo que utilice.
  - Desarrollo, explicando detalladamente la resolución de la tarea.
  - Resultados, con todos los valores que haya obtenido durante el desarrollo de la tarea.
  - Análisis, donde discuta los resultados de la sección anterior y cualquier complicación que haya tenido.
  - Conclusión, comentando el nivel de finalización de la tarea.
- El diseño del circuito de la tarea pondera por 40 % de la nota, mientras que el informe pondera por 60 %. En caso de no entregarse una de las dos partes, se evaluará la tarea completa con nota 0.
- Todas las preguntas respecto a la tarea deben hacerse a través del foro de consultas en Aula. **No se responderán dudas durante las 48 horas previas a la entrega**.

Joaquín Montes joaquin.montes@sansano.usm.cl
Juan Pablo Varas juan.varash@sansano.usm.cl
Andrés Gómez andres.gomez@sansano.usm.cl