

Ejercicio 1

Dibujar el circuito resultante de la síntesis del siguiente código en SystemVerilog:

```
Module ejercicio1
(input logic clk, rst,
input logic [7:0] a, b,
output logic [7:0] y,
output logic [3:0] z);

    logic [7:0] x;

    assign x = ~(a&b);

    always_ff @(posedge clk, negedge rst)
    begin
        if (~rst) begin
            y <= '0;
            z <= '0;

        end else begin
            y <= x ^ y;
            z <= x[7:4];
        end
    end

end
endmodule
```

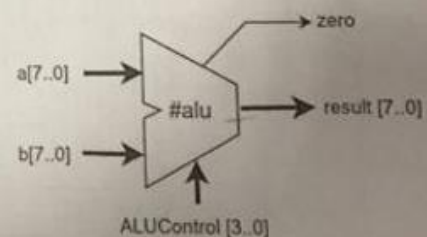
Ejercicio 2

A continuación se describe en SystemVerilog el test bench del módulo "alu". Completar la tabla con los valores EN HEXADECIMAL que toman todas las señales de entrada y salida de la ALU en los tiempos indicados, considerando que "#1" equivale a un tiempo de 1ps.

```
module testbench();
    logic [7:0] a, b, result;
    logic [3:0] ALUControl;
    logic zero;

    alu dut (a, b, ALUControl, result, zero);

    always begin
        a = '0; #10; a = 8'b111; #20;
    end
    always begin
        b = 8'b1010; #10; b = 8'b1111; #5;
    end
    initial begin
        ALUControl = '0; #10;
        ALUControl = 4'b0001; #0.015ns;
        ALUControl = 4'b0111; #10; $stop;
    end
end
endmodule
```

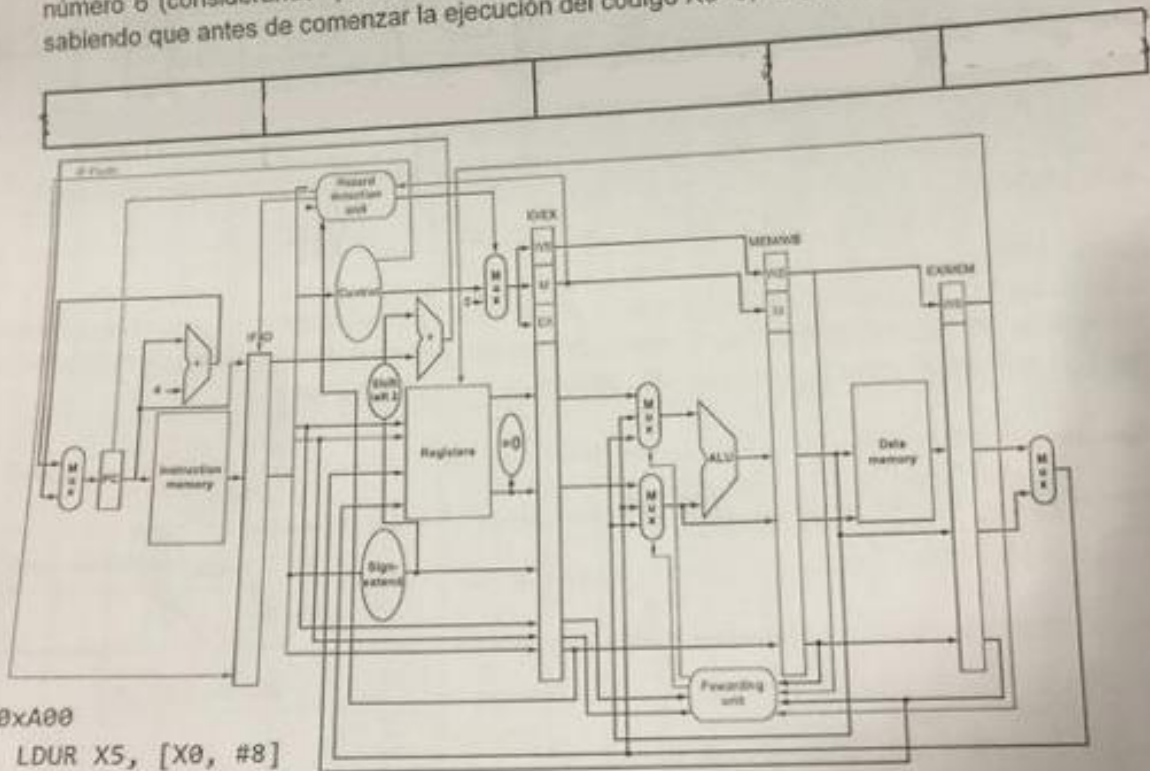


ALU control lines	result
0000	a AND b
0001	a OR b
0010	add (a+b)
0110	sub (a-b)
0111	pass input b

Señal/Tiempo	3ps	8ps	13ps	18ps	23ps	28ps	33ps
a							
b							
ALUControl							
result							
zero							

Ejercicio 3

El segmento de código dado se ejecuta en el procesador LEGv8 con pipeline optimizado para saltos de la figura (con *forwarding stall*). Analizando el estado del procesador en el ciclo de clock número 6 (considerando que el fetch de la primera instrucción se realiza en el ciclo número 1), y sabiendo que antes de comenzar la ejecución del código $X0=0$, responder:



```
.org 0xA00
LDUR X5, [X0, #8]
CBZ X0, label
STUR X3, [X0, #8]
AND X3, X4, X1
label: STUR X3, [X0, #16]
ADD X0, X0, X16
SUB X4, X4, X1
```

- Completar la tabla que está sobre la figura con la instrucción que se está ejecutando en cada etapa.
- ¿Qué valor hay a la salida del PC? _____
- ¿Qué instrucción se encuentra en la etapa de decodificación? _____

- d) ¿Qué operación realiza la ALU? _____
- e) Completar el siguiente cuadro con los valores que tienen las señales de control:

RegWrite	MemRead	MemWrite

Ejercicio 4

Asumiendo que las etapas individuales del pipeline de un procesador tienen las siguientes latencias:

	IF	ID	EX	MEM	WB
Latencia por etapa	100 ns	80 ns	120 ns	150 ns	100 ns

- a) Completar:

Tipo de procesador	Máxima frecuencia de reloj [Hz]	Latencia de una instrucción [ns]
Sin pipeline		
Con pipeline		

- b) ¿Cuántas instrucciones seguidas se deben ejecutar para que el rendimiento del procesador con pipeline sea mejor que el mismo sin pipeline? _____


Ejercicio 5

Considere que la siguiente sección de código ISR que está presente en el vector de excepciones de un microprocesador LEGv8 (ISA completa), similar al analizado en la Guía 3, con la única diferencia en la codificación del *Exception Syndrome Register (ESR)*, la cual se describe a continuación:

Tipo de Excepción	Valor ESR	
OpCode Invalido	0b0001	System Exception
Reservado	0b0010 - 0b0111	
IRQ CH0	0b1000	External Interrupt
IRQ CH1	0b1001	
...	...	
IRQ CH7	0b1111	

Address	Label	Assembly
0x000..0800:	exc_vector:	<pre> mrs x9, s2_0_c2_c0_0 subis xzr, x9, 0x01 b.ne irq //if Not Equal movz x10, #0x8B1F, lsl #16 movk x10, #0x03FF, lsl #0 mrs x11, s2_0_c1_c0_0 sturw x10, [x11, #0] b end </pre>
	irq:	<pre> subis xzr, x9, 0x07 b.ls end //if lower or Same bl nxt </pre>
	nxt:	<pre> lsl x9, x9, #3 add x0, x30, x9 br x0 </pre>
	end:	eret

a) Se pide completar los campos vacíos de la siguiente tabla (en formato HEXADECIMAL) con las direcciones de memoria y contenido de los registros solicitados en cada caso:

Valor del PC al momento de producirse la Excepción		Tipo de Excepción	Valor de los registros de excepción al ejecutarse la ISR			Dirección destino al finalizar la ejecución de la ISR
Address	Instrucción		ESR	ELR	ERR	
0x0...0100	B #4	Memory SegFault	0x03			
0x0...012C		InvOpCode				
0x0...01F0	ADD x0, x1, x1	IRQ CH3				

b) Determinar cuál de las siguientes afirmaciones es V o F:

- Ante una **excepción por OpCode Invalido**, se reemplaza el contenido de la memoria en la que está alojada la instrucción inválida y la siguiente, con dos instrucciones válidas.
- Ante una **excepción no documentada (Reserved)**, el código queda atrapado en un lazo infinito.
- Esta arquitectura del manejo de excepciones corresponde a un esquema de **VECTOR ÚNICO**
- La **ISR** implementa un **SOFTWARE POLLING** para la excepciones de interrupciones de E/S (IRQ) indexando los procedimientos de cada canal (CH) mediante el contenido del registro ESR

Ejercicio 6
Para la siguiente secuencia de código:

Memoria:

[illegible]

- Respuestas puntos b y c en la siguiente página:

