Práctica 4
Procesador: arquitectura, camino de datos y control

Nombre y Apellidos	
Nombre y Apellidos	

Número de grupo de laboratorio	

Preguntas

1 Traduzca las siguientes instrucciones RISC-V a lenguaje ensamblador:

lenguaje máquina	lenguaje ensamblador
FF85AF03	
00C735B3	
10000297	

lenguaje máquina	lenguaje ensamblador
00E780A3	
F7DFF0EF	
01D09463	

2 Un tipo numérico de datos en un lenguaje de alto nivel tiene un rango de valores limitado. En consecuencia, al efectuar operaciones algebraicas es posible que el resultado no se pueda representar (desbordamiento).

Suponga la operación de suma de los números naturales x e y, con un rango de valores 0 <= x, y < M, donde M-1 es el máximo valor que se puede representar. Por tanto, la operación x + y produce debordamiento cuando x > (M-1) - y.

Escriba una secuencia de instrucciones en lenguaje C que detecte desbordamiento sin efectuar la suma. Suponga que el valor máximo de un número natural está especificado por la constante MAX. En las operaciones que se especifiquen no debe producirse desbordamiento en ningún caso.

secuencia de instrucciones C
int x, y; int desb; // 1 indica desbordamiento

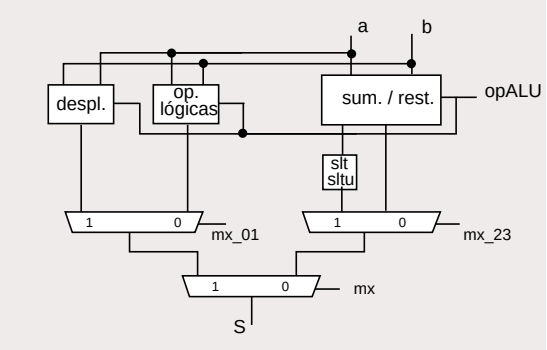
Suponga que los operandos se representan con vectores de 32 bits (M = MAX+1 = 2^{32}). Escriba una secuencia de instrucciones RISC-V en lenguaje ensamblador que detecte desbordamiento sin efectuar la suma. Tenga en cuenta que las operaciones de la secuencia no deben producir desbordamiento en ningún caso. El número de instrucciones debe ser el mínimo. Suponga que los operandos x e y están almacenados en los registros x10 y x11 respectivamente. El resultado se almacena en el registro x9 (el valor 1 indica desbordamiento).

secuencia de instrucciones ensamblador

3 Suponga que el procesador interpreta una instrucción de secuenciamiento condicional que cumple la condición. Indique las instrucciones de secuenciamiento que pueden generar los valores de salida del módulo EVAL (“Secuenciamiento condicional relativo“ en la página 288). Marque con X cualquier combinación que no se pueda producir.

ig	me	meu	instruccion secuenciamiento condicional
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

4 En la figura se muestran las 3 unidades funcionales de la ALU (página 292) y el árbol de multiplexores de selección: a) desplazamiento lógico o aritmético (a la derecha o a la izquierda), b) operación lógica, y c) sumador algebraico y comparador de menor (enteros o naturales). El módulo slt/sltu formatea (añade ceros a la izquierda) la salida de condición del sumador.

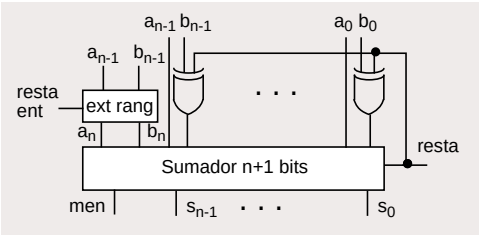


La ubicación de los ficheros relacionados con este proyecto (ALU.qpf) se indica en la página 295. Las dos primeras unidades funcionales y el formateador ya están diseñados (ficheros despla.vhd, logica.vhd y slt.vhd respectivamente). El fichero ALU.vhd contiene la descripción estructural de la ALU, excepto la selección de las salidas de las 3 unidades y el formateador. Esta seleccción se efectua mediante 2 niveles de multiplexores.

Deduzca las expresiones lógicas de las 3 señales de selección de los multiplexores en función de la señal de control opALU (“Señales de control de la ALU“ en la página 329). Inclúyalas en cuerpo de la arquitectura.

nivel 1	mx_01	
	mx_23	
nivel 2	mx	

El fichero sumalg.vhd contiene la interface del sumador algebraico y comparación de menor. Se utiliza un sumador de vectores de n+1 bits, puertas xor y lógica para extender el rango de los vectores a sumar. La salida men se activa cuando se cumple la condición a<b. Recuerde que los operandos se pueden interpretar como enteros o naturales. Esta salida se formatea (añadiendo ceros a la izquierda) en la unidad slt/sltu, que también está diseñada (fichero slt.vhd).



Analice el fichero sumalg.vhd. Deduzca las sentencias de asignación concurrente de las señales resta y ent (en funcion de opALU) y del bit más significativo de los vectores de entrada del sumador de n+1 bits (an, bn). Escriba las sentencias de asignación de las salidas de la unidad (en función del vector resultado de la suma).

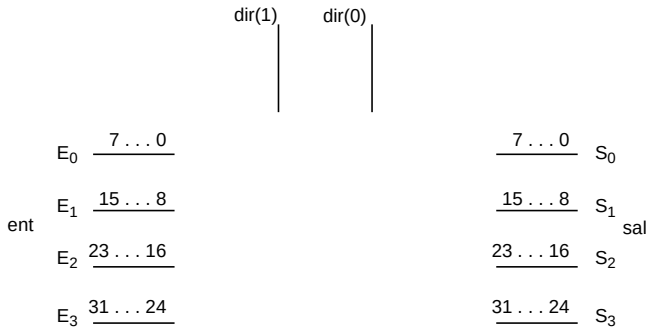
	resta		an	
	ent		bn	
salidas	men			
	s			

Compruebe el diseño efectuado. El programa de prueba suministrado (prueba_ALU.vhd) compara, para un conjunto reducido de vectores, las salidas de la alu original y la diseñada.

5 El circuito FMTL (fichero FMTL.vhd) formatea el dato leído de la memoria que se utiliza para actualizar el banco de registros (“Segundo nivel: organización de la memoria de datos“ en la página 295).

Analice el componente “alinear” y muestre un esquema del circuito. Como ayuda, utilice el visor RTL de quartus. Rellene la tabla que relaciona los datos de entrada y salida del módulo en función de los dos bits menos significativos de la dirección. Utilice la nomenclatura E_i y S_i para indicar el byte i del dato de entrada y de salida.

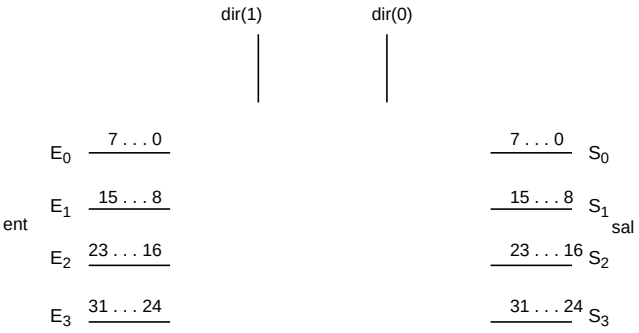
dir	sal			
1..0	S_3	S_2	S_1	S_0
0 0				E_0
0 1				E_1
1 0				E_2
1 1				E_3



6 El circuito FMTE formatea el dato leído del banco de registros para actualizar los bancos de la memoria de datos (“Segundo nivel: organización de la memoria de datos“ en la página 295).

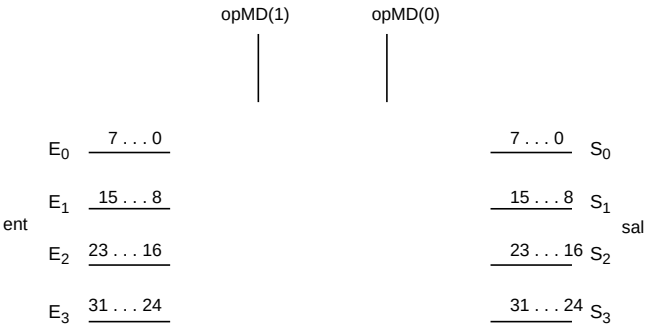
En el diseño base (fichero FMTE.vhd), el circuito consta de 2 módulos: a) alineamiento de datos (fichero alinearE.vhd), y b) selección de los bancos a actualizar (fichero sel_byte.vhd). El módulo alinearE utiliza los 2 bits menos significativos de la dirección.

Analice la descripción de la arquitectura del módulo alinearE y dibuje un esquema del circuito. Utilice el visor RTL de quartus. Rellene la tabla que relaciona los datos de entrada y salida del formateador en función de los dos bits menos significativos de la dirección.



dir	sal			
1..0	S_3	S_2	S_1	S_0
0 0				E_0
0 1				E_0
1 0				E_0
1 1				E_0

7 Proponga un diseño alternativo del módulo alinearE que utilice únicamente los 2 bits menos significativos de la señal de control opMD para formatear el dato de escritura a memoria (“Figura 4.107“ en la página 332). La ubicación de los ficheros relacionados con este proyecto (ALINEAE.qpf) se indica en la página 298. En primer lugar rellene la tabla que relaciona los datos de entrada y salida del formateador en función de los dos bits menos significativos de la dirección. Minimice el número de niveles de selección y el número de multiplexores de 2 entradas.



opMD 1..0	sal S_3	S_2	S_1	S_0
0 0				E_0
0 1				E_0
1 0				E_0
1 1	X	X	X	X

Compruebe el diseño efectuado. Para ello modifique el programa de prueba.

8 Obtenga las siguientes métricas cuando el procesador base ejecuta el programa euclides: instrucciones ejecutadas, aritmético-lógicas, load, store, secuenciamiento condicional e incondicional (“Simulación de un programa concreto“ en la página 306). Para ello, añada un proceso al programa de prueba (prueba_proc_MD_MI.vhd). Este proceso debe utilizar únicamente las señales de control opALU (“Figura 4.99“ en la página 330), opMD (“Figura 4.107“ en la página 332) y opSEC (“Figura 4.110“ en la página 333) para determinar el tipo de instrucción interpretada en cada ciclo. Adjunte el código vhdl del proceso.

Instrucciones ejecutadas		Instrucciones aritmético-lógicas	
Instruciones Load		Instrucciones Store	
Instrucciones secuenciamiento condicional		Instrucciones secuenciamiento incondicional	