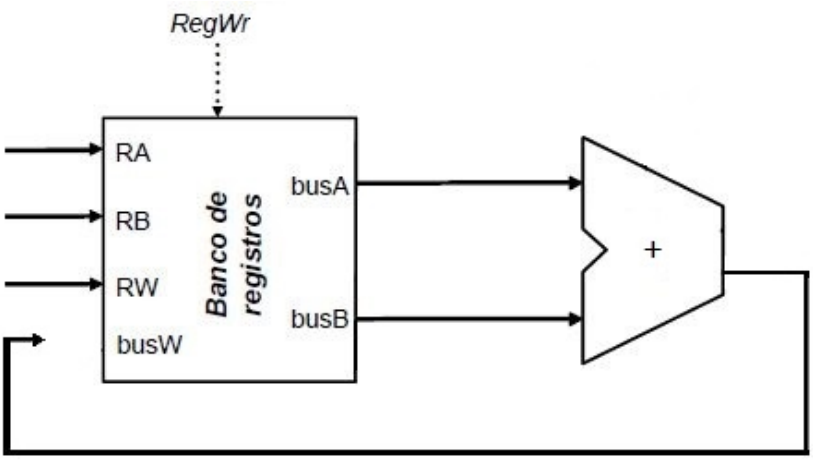
DISEÑO DE UN PROCESADOR CON BANCO DE REGISTROS

Práctica 1 (2ª sesión)



Arquitectura y Organización de Computadores 2
2º Grado Ingeniería Informática

Luis M. Ramos luisma@unizar.es
Alejandro Valero alvabre@unizar.es
José Luis Briz briz@unizar.es
Javier Resano jresano@unizar.es

Escuela de Ingeniería y Arquitectura
Universidad Zaragoza

Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

1 RESUMEN

En esta segunda sesión vamos a diseñar un procesador muy sencillo que trabaje con el banco de registros y un sumador. Lo programaremos para que realice una serie de escrituras y sumas. Una vez diseñado realizaremos un análisis temporal para averiguar la frecuencia máxima a la que puede funcionar.

***Al entrar al laboratorio** coloca el **trabajo previo del apartado 2.1** encima de la mesa, de forma visible. Además, debes disponer del **banco de registros** diseñado en la sesión anterior **finalizado y funcionando correctamente**. Antes de empezar a trabajar con Logisim conéctate a Moodle desde uno de los equipos del laboratorio y realiza el **control de asistencia a la sesión**.*

*La práctica finaliza cuando los procesadores funcionan correctamente y han sido entregados a través de **Moodle**. También hay que entregar los **resultados del apartado 3**.*

2 DISEÑO DE UN PROCESADOR CON BANCO DE REGISTROS

2.1 TRABAJO PREVIO: DISEÑO EN PAPEL

a) Dibuja sobre papel los siguientes componentes: un banco de registros **BR32**, una ROM, un contador, un sumador de 32 bits y un multiplexor. Interconecta los componentes, teniendo en cuenta que:
   - En cada dirección de la ROM se escribirá una instrucción.
   - El contador se utilizará para direccionar la ROM.
   - Con el sumador se calculará la suma de los dos registros leídos del **BR32**.
   - El multiplexor seleccionará el dato adecuado a escribir en el **BR32**.
   - El procesador debe soportar las siguientes instrucciones:

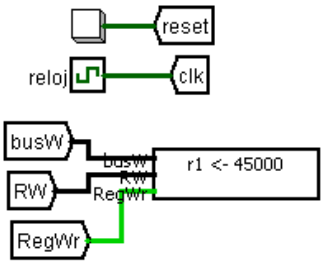
```
mov K,rd        ; SignExt(K) → BR(rd)          K cte de 16 bits
add ra,rb,rd    ; BR(ra) + BR(rb) → BR(rd)
```

b) Define el formato de instrucción y la codificación de las instrucciones.
c) Diseña la unidad de control del procesador, que calcule el valor de las señales de control (**RegWr** y bit selección MUX) a partir del código de operación de la instrucción.
d) Utilizando las instrucciones anteriores escribe un programa ensamblador (NIA.asm) que guarde en los registros r1 – r7 los valores obtenidos en webdiis.unizar.es/~luisma/aoc2/nia1.php a partir de tu NIA y calcule la suma de todos ellos en r0.

2.2 TRABAJO EN EL LABORATORIO: DISEÑO CON LOGISIM

a) Abre en Logisim el circuito de la práctica anterior (NIA-BR32test.circ). Borra el componente **BR32test** y los cables que lo conectaban a tu **BR32**. Descarga la librería Logisim **BR32test** (Proyecto / Descargar Librerías). Guárdalo en un fichero nuevo (NIA-mono.circ).
b) Coloca los componentes del apartado 2.1 e interconéctalos según tu diseño. Conecta el componente *Reloj* al contador y al **BR32**.
c) Traduce tu programa del apartado 2.1.d a hexadecimal y programa la ROM.

d) Para facilitar la depuración coloca en el extremo superior izquierdo del circuito:
   - un botón (carpeta *Input/Output*) conectado a la señal **reset**;
   - el componente *Reloj*;
   - descarga el fichero aoc2.jar de la carpeta Logisim de Moodle y guárdalo en la misma carpeta que tu fichero .circ. Carga la librería aoc2.jar (*Proyecto / Cargar Librería / Librería JAR*). Indica **com.cburch.aoc2.Components** como *Nombre de la Clase*. Coloca un componente *RTL1 Viewer* y conéctale las señales **busW**, **RW** y **RegWr**.
   - Un visualizador del estado, tal como sale del registro de estado



e) Ejecuta ciclo a ciclo hasta que funcione correctamente.
f) Finalmente entrega tu circuito a través del recurso Moodle "Entrega NIA-mono". Antes de entregar comprueba que has marcado **main** como circuito principal y que usas un único componente *Reloj*.

2.2.1 Diseño multiciclo

a) Guarda el diseño anterior en un fichero nuevo (NIA-multi.circ).
b) Modifica el diseño del procesador conectando dos registros a la entrada del sumador (**A** y **B**) y otro a la salida (**C**).
c) Dibuja el autómata de la unidad de control para el procesador multiciclo. Impleméntala usando un registro (estado) y una o dos ROM (función de transición y función de salida).
d) Ejecuta ciclo a ciclo hasta que funcione correctamente.
e) Finalmente entrega tu circuito a través del recurso Moodle "Entrega NIA-multi". Antes de entregar comprueba que has marcado **main** como circuito principal y que usas un único componente *Reloj*. Debes entregar también el autómata de la unidad de control NIA-multi.fsm (dibujado con la herramienta Qfsm disponible en Moodle).

3 ANÁLISIS TEMPORAL

a) Considera los siguientes retardos: d_{NOT} = 5 ps; $d_{OR2\text{-}4}$ = 20 ps; $d_{AND2\text{-}4}$ = 20 ps; d_{REG} = 50 ps; $tsetup_{REG}$ = 30 ps; d_{ROM} = 80 ps; d_{CONT} = 60 ps; d_{SUM} = 260 ps.
b) Identifica el camino crítico del procesador monociclo y su retardo ($dmax_{mono}$). ¿Cuál es su frecuencia máxima (GHz) de funcionamiento ($fmax_{mono}$)?
c) Identifica el camino crítico del procesador multiciclo y su retardo ($dmax_{multi}$). ¿Cuál es su frecuencia máxima de funcionamiento ($fmax_{multi}$)?
d) ¿Cuánto tiempo tarda el procesador monociclo en ejecutar tu programa (tex_{mono})?
e) ¿Cuánto tarda el procesador multiciclo (tex_{multi})?
f) ¿Cuál es la relación entre los tiempos de ambos procesadores (tex_{multi} / tex_{mono})?
g) Cuando tengas todos los cálculos realizados introdúcelos a través del recurso Moodle "Análisis temporal monociclo vs multiciclo".