Arquitectura de Computadores
543.426

Ayudante: Antonio Saavedra

Ayudantía No. 3
7 de mayo de 2017

Problema 1

Se desea agregar la instrucción 'swap condicional' al procesador uniciclo:

cswap $rt, Imm16($rs)

Esta instrucción lee una palabra de memoria desde la dirección dada por la suma del registro $rs y la constante inmediata de 16 bits Imm16 (extendida en signo a 32 bits), y la almacena en el registro $rt. Adicionalmente, y sólo si el valor leído de memoria es distinto de cero, la instrucción almacena el valor original del registro $rt en la dirección de memoria Imm16($rs). En otras palabras, la descripción RTL de la instrucción es:

```
R[rt] <- Mem[R[rs] + Sext(Imm16)];
If (Mem[R[rs] + Sext(Imm16)] != 0)
  Mem[R[rs] + Sext(Imm16)] <- R[rt]
```

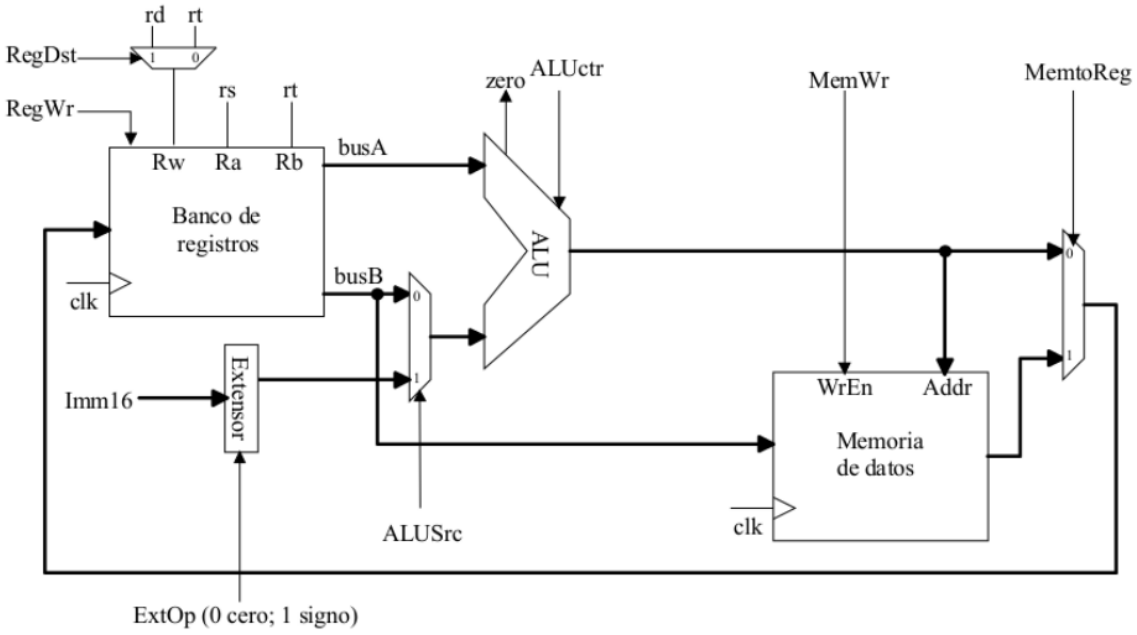

Figura 1: unicesador Uniciclo

	R-type	ori	andi	lw	sw	beq
RegDst	1	0	0	0	x	x
ALUSrc	0	1	1	1	1	0
MemtoReg	0	0	0	1	x	x
RegWr	1	1	1	1	0	0
MemWr	0	0	0	0	1	0
nPC_sel	0	0	0	0	0	1
ExtOp	x	0	0	1	1	x
ALUOp	func	or	and	add	add	sub

Problema 2

Se desea agregar la instrucción 'test and sub' al procesador uniciclo, la cual es útil en la implementación de semáforos para sincronización de procesos en sistemas operativos. La nstrucción utiliza el formato-I, de manera que especifica un registro $rs, un registro $rt y una constante Imm16. La instrucción lee un dato almacena o en memoria en la dirección indicada por el registro $rs, y lo almacena en el registro $rt. Además, si el dato leído es mayor o igual que la constante Imm16, la instrucción resta a este número la constante Imm16, y almacena el resultado en la misma dirección de memoria. La descripción RTL es:

```
R[rt] <- M[rs]
if M[rs] >= Sext(Imm16)
        M[rs] <- M[rs] - Sext(Imm16)
```

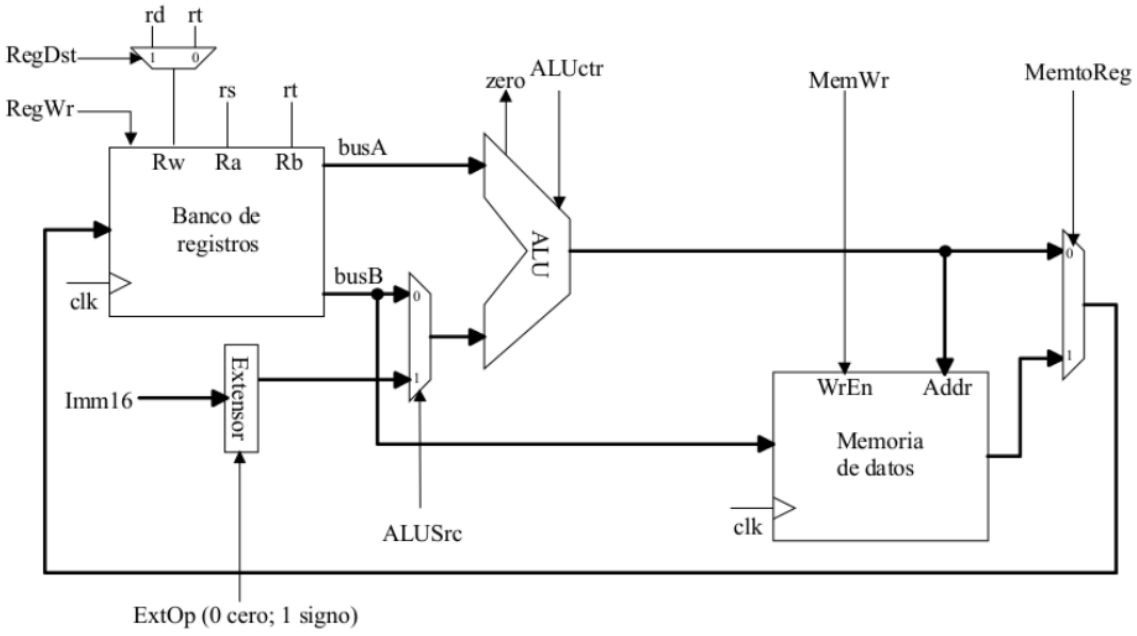

Figura 2: Procesador Uniciclo

	R-type	ori	andi	lw	sw	beq
RegDst	1	0	0	0	x	x
ALUSrc	0	1	1	1	1	0
MemtoReg	0	0	0	1	x	x
RegWr	1	1	1	1	0	0
MemWr	0	0	0	0	1	0
nPC_sel	0	0	0	0	0	1
ExtOp	x	0	0	1	1	x
ALUOp	func	or	and	add	add	sub

Problema 3

Se desea agregar al procesador multiciclo MIPS una instrucción que calcula el máximo entre dos números de 32 bits en complemento a 2 almacenados en memoria, y almacena este máximo en una de las dos direcciones. La instrucción (MMAX) utiliza el formato-I, construyendo las dos direcciones de memoria con los registros rs y rt, y la misma constante de 16 bits de desplazamiento. La descripción RTL lógica es:

```
M[rt + Sx(Imm16)] ← MAX ( M[rs + Sx(Imm16)], M[rt + Sx(Imm16)] )
```

Donde rs y rt son los campos correspondientes de la instrucción, Imm16 es la constante de 16 bits incluida en la instrucción, y Sx denota extensión en signo a 32 bits. Agregue la instrucción MMAX al procesador MIPS multiciclo. Minimice la cantidad de ciclos de ejecución de la instrucción y el hardware extra agregado. Puede agregar multiplexores y cables, pero no registros ni sumadores. Recuerde que el bit más significativo de un número de 32 bits vale 1 cuando el número es negativo. Por simplicidad puede asumir que este bit corresponde siempre al signo del resultado de la ALU incluso en condiciones de overflow. Más realistamente, para incluir las condiciones de overflow de resta se calcula el signo como el EXOR entre el bit de carry de la ALU y el bit más significativo del resultado (pero puede ignorar esta situación en este ejercicio).

a) Escriba una descripción RTL física (ciclo a ciclo) para su implementación.

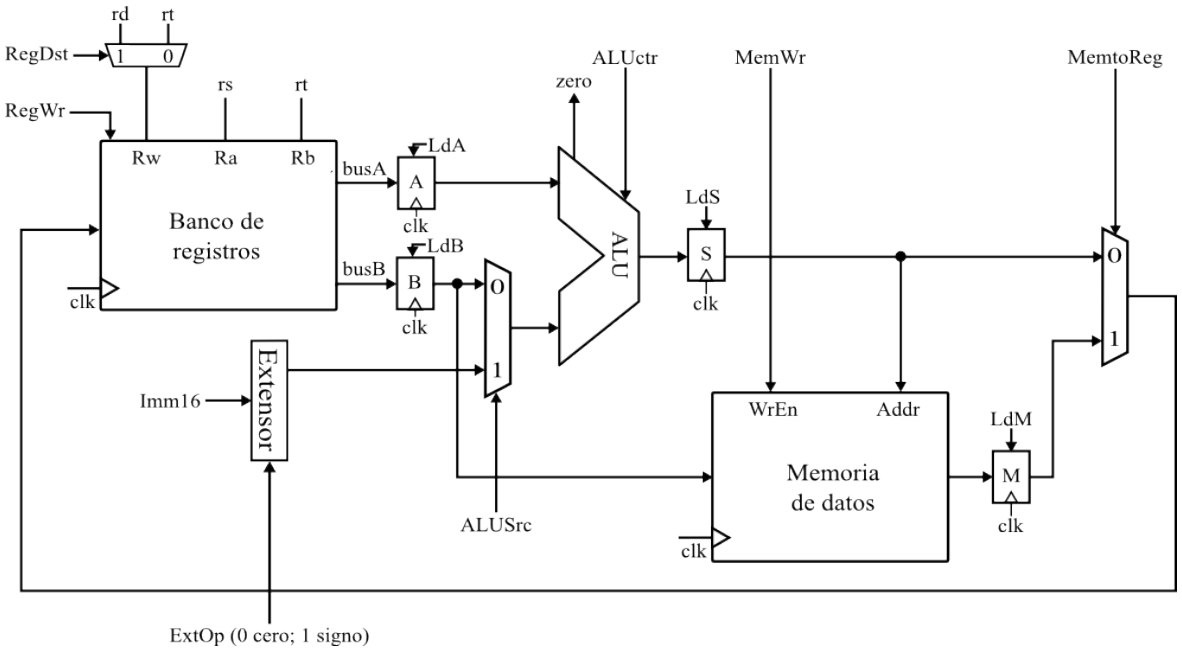

Figura 3: Procesador Multiciclo

	(1)	(2)	(3)	(4)	(5)
LdIR					
RegDst					
RegWr					
LdA					
LdB					
ExtOp					
ALUSrc					
ALUCtr					
LdS					
MemWr					
LdM					
MemtoReg					

Cuadro 1: Señales de control por ciclo de instrucción