Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2343 — Arquitectura de Computadores (2020-2)

Ayudantía 3

Almacenamiento y Programabilidad

1 Precalentamiento

Ejercicios Rápidos:

1. **Endianness:** Al leer 2 palabras contiguas de 1 byte de una memoria en orden big endian, se obtiene el número 272. ¿Qué número se obtendría al leer estas palabras en orden little endian?

   Solución:

   $272 = 0x0110$

   Ahora si consideramos little endian, lo leemos: $0x1001$ (damos vuelta el orden de los bytes o palabras), lo que representa:

   $0x1001 = 1 * 16^3 + 1 * 16^0 = 4096 + 1 = 4097$.

2. **Almacenamiento:** ¿Cuál es la diferencia entre un latch y un flip-flop?

   Solución:

   Un latch puede realizar cambios de estado mienras se encuentra con la señal de control en 1, mientras que el flip-flop sólo lo hace cuando está en uno de los dos flancos (subida o bajada).

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2343 — Arquitectura de Computadores (2020-2)

Ayudantía 3

Almacenamiento y Programabilidad

2 Almacenamiento

Ejercicio 1:

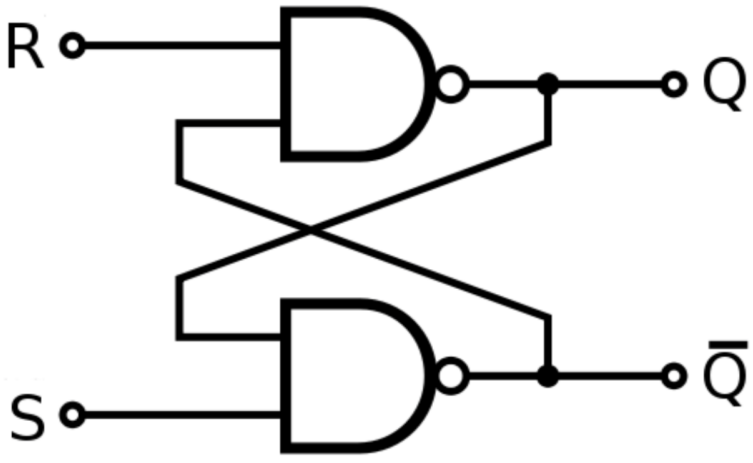
Modifique un latch tipo RS agregando una señal de control C, tal que los cambios en el estado del latch sólo se realicen cuando $C = 1$.

Solución:

Recordemos que la tabla de verdad de un latch tipo RS es:

S	R	Q^{t+1}
0	0	-
0	1	1
1	0	0
1	1	Q^t

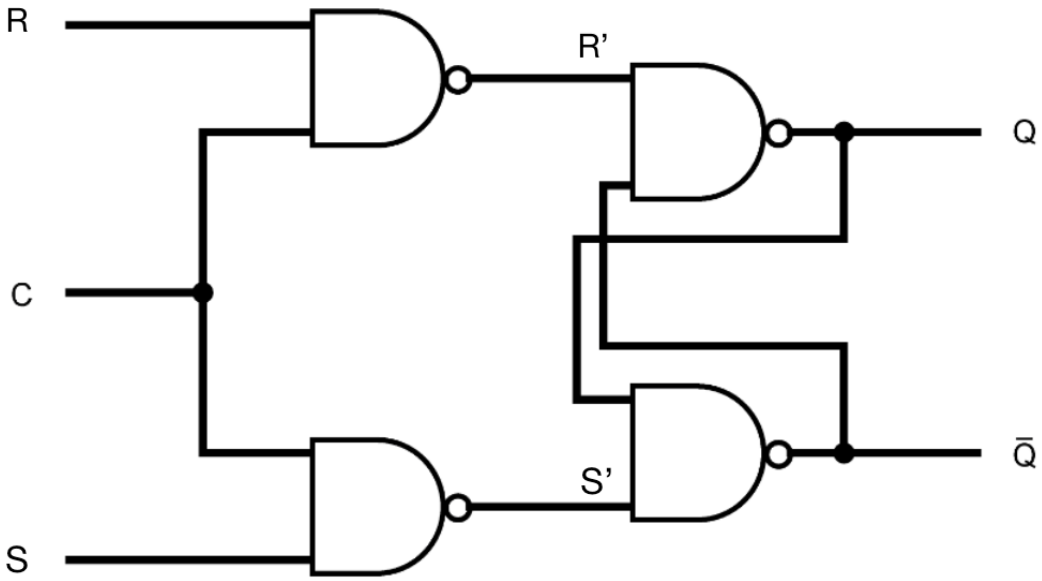
Y el circuito se ve así:



Ahora, si queremos agregar una señal de control C, tal que sólo se realicen cuando esta valga 1, lo común sería pensar agregar dos compuertas AND, una para cada entrada (R y S) para que al unirlas con la señal de control C, obtengamos las entradas del latch RS original, que consideraremos como R' y S'. El problema ocurre cuando buscamos que sólo haya un caso del indeterminado (cuando $R' = 0$ y $S' = 0$) ya que el AND tiene 3 casos donde el resultado es 0, por lo que la intuición nos dice que necesitaremos una compuerta NAND, para que solo haya un caso donde el resultado sea 0 (cuando ambas entradas son 1).
Dado que ya hemos identificado que las compuertas en las que incluiremos la señal de control deben ser NAND, podemos armar nuestra tabla de verdad, para cumplir con el comportamiento del latch RS:

R	S	C	R'	S'	Q^t
1	1	1	0	0	-
1	0	1	0	1	1
0	1	1	1	0	0
0	0	1	1	1	Q^t
0	1	0	1	1	Q^t
0	0	0	1	1	Q^t
1	0	0	1	1	Q^t
1	1	0	1	1	Q^t

A partir de esta tabla, podemos armar nuestro circuito, donde las entradas R y S originales del latch ahora vienen del output de un NAND entre la variable respectiva y la señal de control C.
De esta forma el diagrama queda:



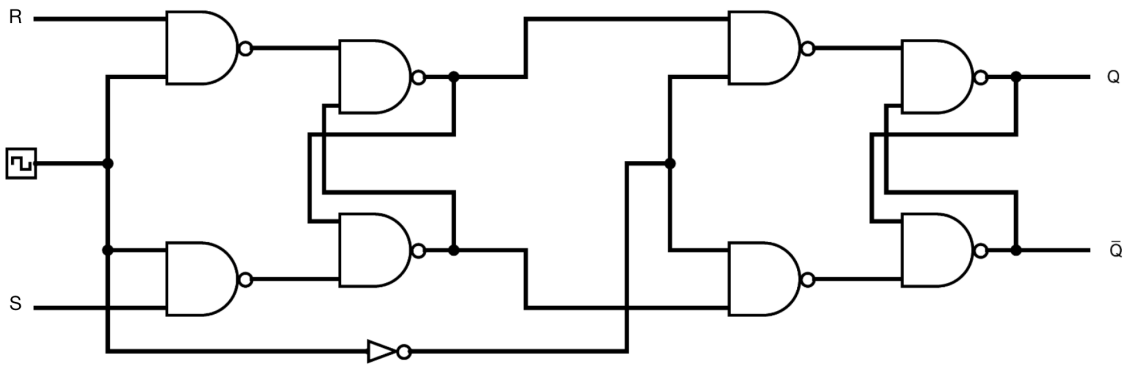
Ejercicio Propuesto:

Diseñe un flip-flop tipo RS, dado por la siguiente tabla de verdad:

S	R	C	Q	$\bar{Q}$
0	0	$\uparrow$	Q	$\bar{Q}$
0	1	$\uparrow$	0	1
1	0	$\uparrow$	1	0
1	1	$\uparrow$	-	-
X	X	0,1,$\downarrow$	Q	$\bar{Q}$

Solución:

La idea acá es seguir la lógica que se hizo en el ejercicio pasado, pero considerando que al tratarse de un flip-flop, estaremos trabajando con un clock en vez de una señal de control, y al ser un flip-flop debemos incluir dos veces el circuito del latch creado (uno con la señal del clock y otro con la negación de ésta).
El circuito se muestra a continuación:



Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2343 — Arquitectura de Computadores (2020-2)

Ayudantía 3

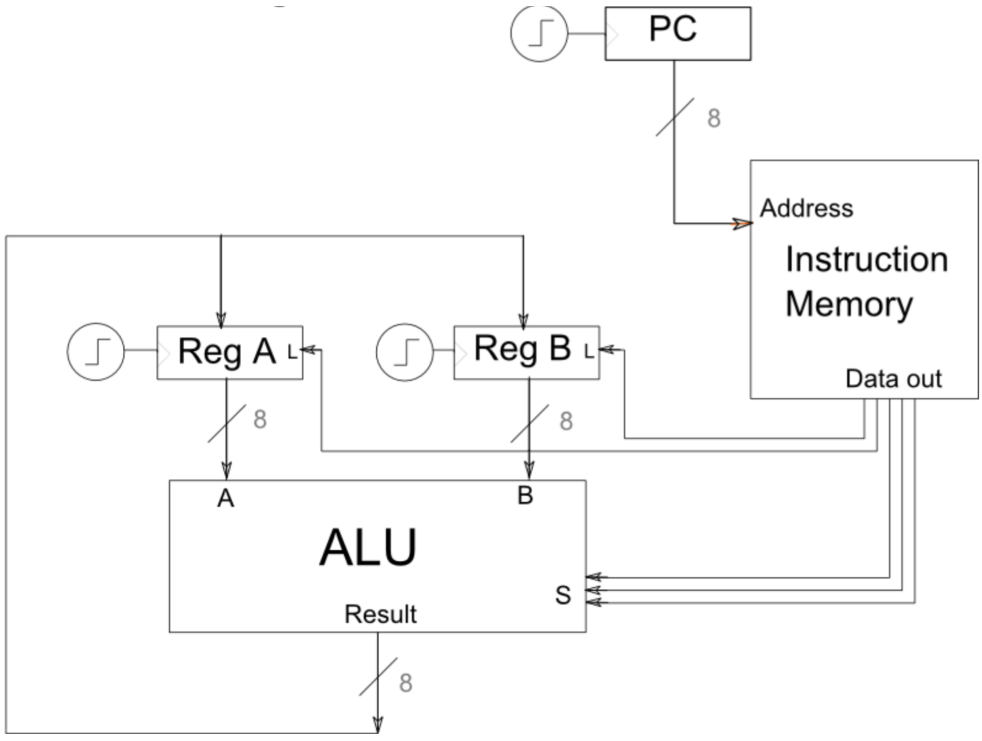
Almacenamiento y Programabilidad

3 Programabilidad

Ejercicio 1: I2 2012/1

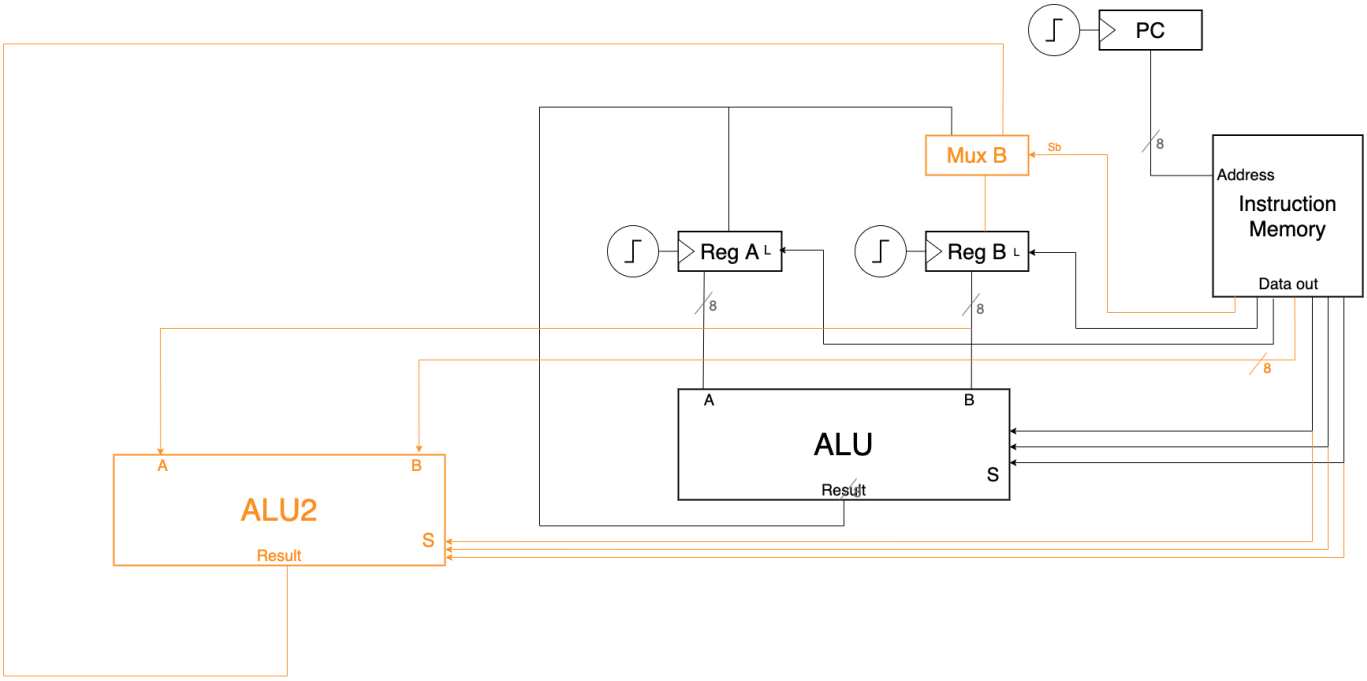
Detalle las modificaciones que le haría a la microarquitectura del computador básico para permitirle la ejecución de dos operaciones aritméticas-lógicas iguales, pero con distintos argumentos, de manera simultánea, i.e., el proceso debe tomar sólo un ciclo del clock.

(Se incluye un diagrama del computador básico visto hasta ahora para trabajar en base a éste.)



Solución:

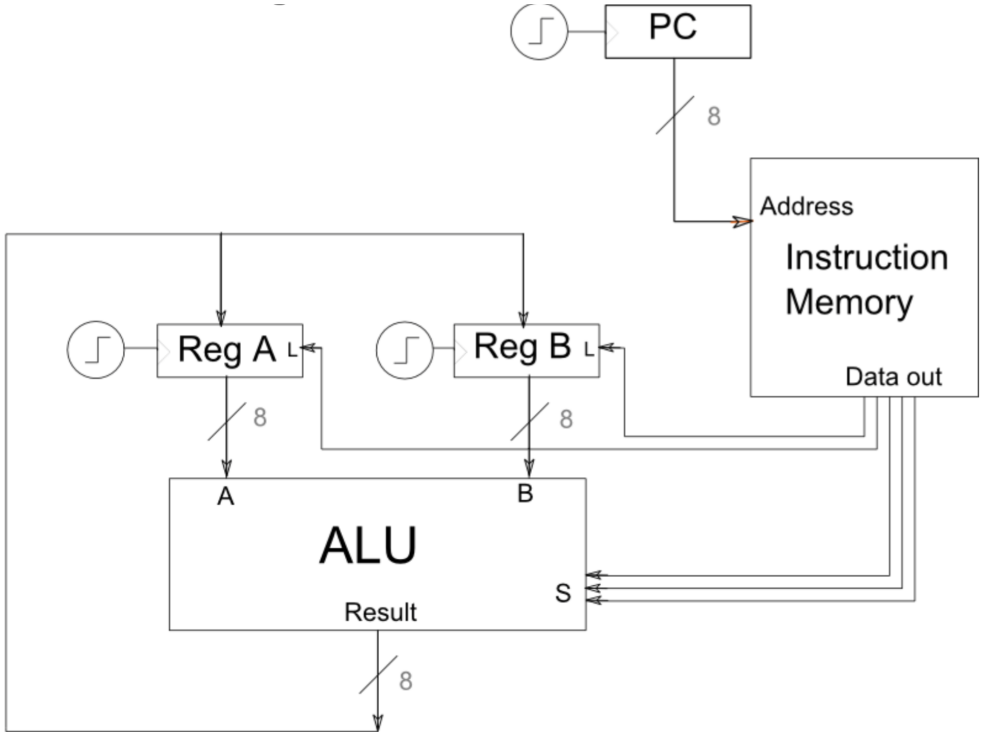
La modificación presentada a continuación agrega una ALU adicional, que obedece a las mismas señales de control S, logrando ejecutar 2 operaciones en paralelo (en el mismo ciclo del reloj). Para cumplir el tema de los argumentos distintos, en la $ALU2$, el input A sale a partir del valor guardado en el registro B y el input B sale de un literal o constante que viene determinado por la instrucción (sale de la instruction memory). Se decidió que el resultado fuese a parar al registro B, pero dado que tiene dos posibles valores (el de la ALU y el de la $ALU2$, se agrega un $MuxB$ con la señal de control s_b, que determina que valor se guarda en el registro.



Ejercicio 2: I1 2015/2

Modifique el diagrama de la máquina programable, de manera que soporte la instrucción GO TO **Dir**, que fuerza que la siguiente instrucción a ejecutarse sea la ubicada en la dirección **Dir**.

(Se incluye un diagrama del computador básico visto hasta ahora para trabajar en base a éste.)



Solución:

La modificación presentada a continuación utiliza el concepto de literal que se ocupó en la pregunta anterior. Ahora el literal no representa un parámetro para realizar una operación en la *ALU* sino que representa la dirección de la siguiente instrucción que se busca ejecutar (**Dir**). Ahora, dado que el *PC* continuamente sumaba uno para recorrer las instrucciones de forma secuencial, vamos a tener que agregar una nueva señal de control *LoadPC*, para definir cuando aceptaremos el parametro del literal y cuando no.