Organización y Arquitectura de Computadoras

Facultad de Ciencias, UNAM

José Ethan Ortega González: 316088327
Etzael Iván Sosa Hedding: 316259305

1. Un registro de desplazamiento es un circuito secuencial que desplaza a la izquierda o a la derecha la información contenida en el. Considerando el desplazamiento de 1 bit a la izquierda, ¿cómo se implementa dicho circuito? ¿Cómo podríamos simular su funcionamiento con las operaciones que se tienen en la ALU de 8 bits?

Solución:

Para implementarlo es muy sencillo, solo tenemos que conectar n flip-flops de tipo D, donde la entrada del primero puede elegirse y su salida es la entrada del siguiente flip-flop, y el reloj está conectado a todos en paralelo. Es más conocida como SIPO (Serial Input Parallel Output). Para poder simular el funcionamiento con nuestra ALU, simplemente necesitamos sumar y restar mediante el siguiente algoritmo:

> Sean $\{n, \ldots, 1\}$ los indices de los dígitos en el registro. Buscamos el 1 con mayor indice k y a todo el número le sumamos otro número cuyo único 1 está en la posición $k+1$, y luego restamos un número cuyo único número está en la posición k. Repetimos el algoritmo desde $k-1$ hasta 1. Esto nos da un desplazamiento a la izquierda introduciendo un 0, si queremos introducir un 1 hacemos exactamente lo mismo y le sumamos un 1 al final.

2. Considerando el formato de instrucción para nuestro circuito completo (ALU y Banco de Registros conectados) que se muestra en la Tabla 1 escribe las instrucciones necesarias para calcular las siguientes operaciones (Suponga que A, B y C se encuentran en los registros 1, 2 y 3 respectivamente):

- $2A + B - C$
- $(-A \& B)|C$

Solución:

- Las instrucciones son las siguientes:

```
01100000 00000000 // Sumamos dos veces a A y lo guardamos en A
01100100 01000000 // Sumamos a A y B y lo guardamos en B
11101001 10000000 // Restamos a B y C y lo guardamos en C
```

- Las instrucciones son las siguientes:

```
01000000 00000000 // Negamos las entradas de A y lo guardamos en A
00000100 01000000 // Hacemos un AND de bits con A y B y lo guardamos en B
00101001 10000000 // Hacemos un OR de bits con B y C y lo guardamos en C
```