

Organización y Arquitectura de Computadoras

Practica 4

Arrieta Mancera Luis Sebastian, Martínez Hernández Zuriel Enrique

Procedimiento:

6. Ejercicios

1. Desarrolla un sumador completo de 1 bit.

= Sumador de 1 bit =

Sabemos que la suma de 2 bits es:

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

↓ Losuma
Acarreo

Hagamos los circuitos de la tabla anterior

- Notemos que la única vez que el acarreo es 1 es cuando $A=1$ y $B=1$ esto es equivalente a la compuerta AND.

$C = AB$

circuito

AND

- En la suma el resultado es uno cuando tenemos

$S = \bar{A}B + A\bar{B}$

circuito

NOT AND AND OR

Figura 1

Hay ocasiones en las que es necesario tomar en cuenta el acarreo de entrada y no solo el de salida, por ejemplo si realizamos la suma

$$\begin{array}{r}
 11111 \\
 + 1001101 \\
 1111000011
 \end{array}$$

Tenemos que sumar

$$\begin{array}{r}
 1 \\
 + 1 \\
 \hline
 11
 \end{array}$$

Lo que nos da como resultado el 3 en binario

Esto es hacer una tabla de verdad con tres variables A , B y C_{input} (Acarreo de entrada)

A	B	C _{input}	C _{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Hagamos los circuitos de la suma y del acarreo usando mapas de Karnaugh

C_{out} (Usamos minterminos i.e. Consideramos las 1s)

A \ B C _{in}	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Suma (S) (Usamos minterminos)

A \ B C _{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$C_{out} = AC_{in} + AB + BC_{in}$$

$$S = A\bar{B}\bar{C}_{in} + \bar{A}\bar{B}C_{in} + ABC_{in} + \bar{A}B\bar{C}_{in}$$

$$\Rightarrow \text{Esto también es } (A \oplus B) \oplus C$$

↓
XOR

Figura 2

Realizamos los circuitos anteriores en logisim. Estos últimos no los realizamos en el procedimiento ya que son muy extensos, mejor los pasamos de una vez a logisim.

1. Preguntas

- ¿Qué operaciones aritméticas y lógicas son básicas para un procesador? Justifica tu respuesta. Según las notas del curso página 80, las instrucciones aritmético lógicas que realiza una ALU son

- Aritméticas
 - suma
 - resta
 - multiplicación
 - corrimiento
 - inversión de signo
 - residuo
 - comparaciones
- Lógicas
 - conjunción
 - disyunción
 - disyunción exclusiva
 - negación

- El diseño utilizado para realizar la adición resulta ser ineficiente , ¿por qué? ¿Qué tipo de sumador resulta ser más eficiente?.

Por lo que explicó Ximena en clase, no resulta ser muy eficiente el sumador que realizamos, puesto que se tiene que pasar por una cantidad muy grande de compuertas para realizar la adición de numero con una cantidad de bits muy grande. Por ello actualmente se usan algunas heurísticas para reducir el número de operaciones realizadas por los procesadores.

- ¿Cuántas operaciones más podemos agregar al diseño de esta ALU? ¿Qué tendríamos que modificar para realizar más operaciones?

Como anteriormente se menciona en la pregunta una, las operaciones que nos faltarían implementar en nuestra ALU son

- corrimiento
- inversión de signo
- residuo
- comparaciones ($A < B$, $A <= B$, $A > B$, $A >= B$)
- disyunción exclusiva
- negación

Para realizar las operaciones anteriores tendríamos que realizar cada circuito y agregarlo a nuestra ALU.