

Organización del Procesador

año 2016

Guía de ejercicios prácticos 2

Representación de Números Enteros y Racionales (Punto Flotante)

Requerimientos: Para ejercitar circuitos electrónicos se utilizará la herramienta *AutoDesk CIRCUIT*, la misma es ejecutada en plataforma web y es gratuita. Para poder utilizarla y almacenar los circuitos desarrollados, deberá entrar a la página de la herramienta <https://circuits.io> y crear una cuenta personal.

Además, para realizar algunos ejercicios de esta guía es necesario disponer del compilador de algunos de los lenguajes utilizados durante la materia Introducción a la Algorítmica (Pascal o C).

1. Escriba la representación (binaria) de los siguientes números, si el modo de representación interna fuera: (I) Signo y Magnitud, (II) Complemento a la base -1 y (III) Complemento a la base. Asuma el tamaño de palabra de 8 bits:
a) 31 c) -21 d) -40 e) -9 f) 12 g) 0 h) -0
2. ¿Cuál es el rango de representación de números en *complemento a la base*, si utilizamos un registro de: 16, 32 bits ?
3. Resuelva las siguientes operaciones suponiendo una A.L.U. de 5 bits que utilice la representación interna de complemento a la base. Indique además si el resultado es representable o no y en qué estado quedan los *flags*(indicadores) de *Carry Out* y *Overflow*.
a) $8 - 3$ b) $3 - 7$ c) $4 - 4$ d) $7 - 8$ e) $9 - 5$ f) $8 + 8$ g) $-8 - 8$ h) $10 + 10$
4. Resuelva las siguientes operaciones suponiendo una A.L.U. que utilice complemento a la base, con registros del mínimo tamaño necesario para realizar las operaciones:
a) $6 * 7$ b) $9 * 7$ c) $-5 * 4$ d) $-6 * 5$ e) $3 * -5$ f) $4 * -7$ g) $-2 * -3$
5. Para los incisos a), c), g) del Ejercicio 4 grafique las etapas cómo la resolvería una A.L.U. de 5 bits.
6. Resuelva los incisos b), d), f) del ejercicio 4 utilizando el algoritmo de *Booth*.
7. Resuelva las siguientes operaciones de números positivos, suponiendo una A.L.U. con la mínima cantidad de bits necesarios para representar los operandos en complemento a la base.
a) $15 / 3$ b) $20 / 3$ c) $17 / 2$ d) $1 / 2$
8. Para el inciso b) del Ejercicio 7 grafique las etapas cómo lo resolvería una A.L.U. con la mínima cantidad de bits necesarios para representar los operandos en complemento a la base, utilizando el algoritmo de división *con restauración* y *sin restauración*.
9. Análisis de Representación con Lenguajes de Programación de alto nivel:
Analice la representación de números enteros en declaraciones de variables en lenguajes de alto nivel. Para ello, mediante *aliasing* evalúe el valor de una variable entera mediante una secuencia de Bits.

Ayudas: El siguiente código Pascal muestra cómo declarar un arreglo compactado de booleanos (bits) en la línea 7, luego mediante la acción *PrintBits* (línea 14) podemos imprimir bit a bit el valor. Notar que el indicador para el compilador de la línea 3, indica que el arreglo de booleanos

se represente con un bit por posición en vez de 1 byte como normalmente lo realizaría sin él. Al generar un *aliasing* entre la dirección de memoria de una variable (@) y un puntero a un arreglo de bits, podemos observar cómo está representada la información a nivel de bits. Nota: recuerde que, a diferencia de cómo lo escribimos manualmente, los bits más significativos se encuentran a derecha.

objetivos:

- (a) Modifique la acción de visualización para separar de a 8 bits (1 Byte) con algún símbolo, por ejemplo |.
- (b) Ejecute el programa y analice la representación del número 2, ¿es la que esperaba ?).
- (c) Modifique el programa, asignando un valor entero negativo y luego ejecute, ¿ qué representación utiliza ?.
- (d) Modifique el programa incorporando más variables y de distinto rango de representación, como ShortInt, LongInt, Int64, analice los resultado. Ayuda: agrande el tamaño del arreglo para poder abrazar con el *aliasing* toda la representación de las diferentes variables. Notar que puede analizar la representación de dos variables contiguas si el tamaño del arreglo es lo suficientemente grande.
- (e) Incorpore una acción para poder modificar un bit en una posición del arreglo. Luego utilícela para modificar el valor de alguna de las variables a nivel de bits.

```

1  program IntegerTest;
2
3  { $BITPACKING ON }
4  Uses Math;
5
6  const cantBits = 32;
7  type TByteBits = bitpacked array[0..cantBits-1] of Boolean;
8  type TPByteBits = ^TByteBits;
9
10 var a:Integer;
11     p:TPByteBits;
12
13 (* Imprime una secuencia de bits (arreglo compactado de booleanos) *)
14 procedure PrintBits(p:TPByteBits);
15 var i:integer;
16 begin
17     for i:=0 to cantBits-1 do begin
18         if p^[i] then write('1')
19             else write('0');
20     end;
21 end;
22 begin (* Programa *)
23     a:=2;
24     p:=@a;
25     PrintBits(p);
26 end.
```

10. ★ Utilizando el simulador de circuitos y siguiendo el esquema de la Fig.1, construya un componente Full Adder como lo describe la Fig.2. Ayuda, conecte con voltaje positivo el pin (vcc) y el pin (gnd) de cada circuito integrado a (-). Alimentando con una fuente de poder entre 2 y 2.4 volts no es necesario incorporar resistencias. Recuerde que los circuitos integrados de AND XOR y OR binarios disponibles son cuádruples (cada uno contiene 4 compuertas) de las cuales sólo se utilizarán las necesarias, el resto de los pines quedarán sin conexiones. Para comprobar el correcto funcionamiento puede incorporar switches para simular las entradas (Sumando1, Sumando2 y Carry In) y Leds para mostrar las salidas (Resultado y Carry Out). (*Low-Hanging Fruit: si se entusiasmó puede duplicar el componente Full Adder interconectando los carry out - carry in del siguiente para construir una semi (no resta) Pascalina (electrónica :P).*)

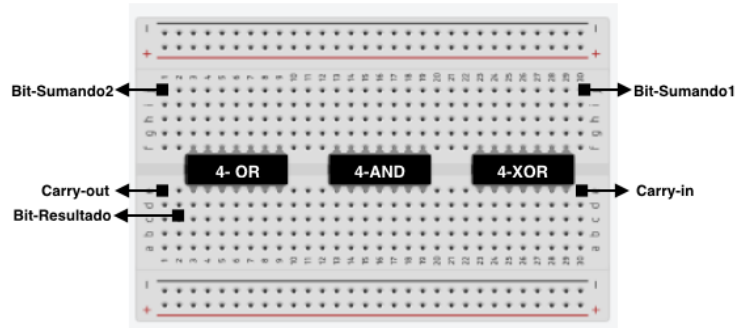


Figure 1: Esquema de componentes para implementar un circuito FullAdder

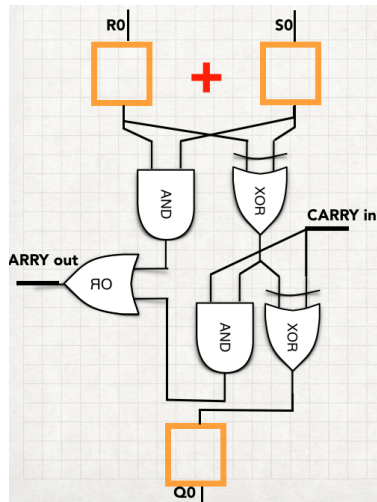


Figure 2: Circuito FullAdder

11. Suponiendo una representación (1 bit Signo, 4 bits exponente y 6 bits mantisa) represente los siguientes números racionales (*recuerde el bit de oculto y puede escribirlo como S EEEE MMMMMM*) :
a) 20 b) -2,25 c) 0,55 d) 0
12. Utilizando la representación IEEE para single, ¿qué número representa la siguiente configuración ? (hint: la posición de cada dígito está en la posición como indica IEEE):

0000000000000000000000001101001100000011

Recuerde que IEEE Single especifica la siguiente representación :

Elemento	bits	posición
Signo	1	[31]
Exponente	8	[30-23]
Mantisa	23	[22-0]

13. Utilizando la representación y valores del **ejercicio 11**, realice la suma de a) y c).