Manipulación de bits en Arduino

Operaciones con bits

Operador	Función	Resultado	Ejemplo
&	And binario	1 si ambos bits son 1; de lo contrario resulta 0	3 & 1 igual a 1
|	Or binario	1 si al menos uno de los bits es 1, de lo contrario resulta 0	3 | 1 igual a 3
^	Or exclusivo binario	0 si los dos bits son iguales; de lo contrario resulta 1	3 ^ 1 igual a 2
~	Negación binaria	Invierte cada bit	~0011 es igual a ~1100
>>	Corrimiento a la derecha	Recorre los bits a la derecha el número de posiciones indicado	6 >> 2 es igual a 1 (110 >> 2 = 001
<<	Corrimiento a la izquierda	Recorre los bits a la izquierda el número de posiciones indicado	6 << 2 es igual a 0 (110 << 2 = 000)

```
/*
* bits sketch: Ejercicio para demostrar los operadores de bits
*/
void setup() {
  Serial.begin(9600);
}

void loop(){
  Serial.print("3 & 1 es igual a ");        // Operación And (3 And 1)
  Serial.print(3 & 1);                     // imprime resultado en base 10 ( decimal)
  Serial.print(" decimal, o en binario es: ");
  Serial.println(3 & 1 , BIN);             // imprime el resultado en base 2 (binario)

  Serial.print("3 | 1 es igual a ");       // Operación Or (3 or 1)
  Serial.print(3 | 1 );                    // imprime resultado en base 10 ( decimal)
  Serial.print(" decimal, o en binario es: ");
  Serial.println(3 | 1 , BIN);             // imprime el resultado en base 2 (binario)

  Serial.print("3 ^ 1 es igual a ");       // Operación Xor (3 Xor 1)
  Serial.print(3 ^ 1);                     // imprime resultado en base 10 ( decimal)
  Serial.print(" decimal, o en binario es: ");
  Serial.println(3 ^ 1 , BIN);             // imprime el resultado en base 2 (binario)

  byte byteVal = 1;
  int intVal = 1;
  byteVal = ~byteVal;
  intVal = ~intVal;                        // se aplica la operación de negación binaria
```

```
  Serial.print("~byteVal (~1) es igual a ");  // negación de un valor de 8 bits
  Serial.print(byteVal);                     // imprime resultado en base 10 ( decimal)
  Serial.print(" decimal, o en binario es: ");
  Serial.println(byteVal, BIN);              // imprime el resultado en base 2 (binario)

  Serial.print("~intVal (~1) es igual a ");    // negación de un valor de 16 bits
  Serial.print(intVal);                      // imprime resultado en base 10 ( decimal)
  Serial.print(" decimal, o en binario es: ");
  Serial.println(intVal, BIN);               // imprime el resultado en base 2 (binario)

  Serial.print("aplicando la operación ");
  Serial.print(byteVal, BIN);                // imprime el valor en base 2 (binario)
  byteVal = byteVal >> 2;
  Serial.print(" >> 2 es igual a ");
  Serial.println(byteVal, BIN);              // imprime el resultado en base 2 (binario)

  byteVal =1;
  Serial.print(" aplicando la operación ");
  Serial.print(byteVal, BIN);                // imprime el resultado en base 2 (binario)
  Serial.print(" << 2 es igual a");
  byteVal = byteVal << 2;
  Serial.println(byteVal, BIN);              // imprime el resultado en base 2 (binario)
}
```

Escribiendo y Leyendo bits

bitSet(x, bitPosition) – Escribe 1 en el bit de la posición indicada (bitPosition) de la variable x.

bitClear(x, bitPosition) – Escribe 0 en el bit de la posición indicada (bitPosition) de la variable x.

bitRead(x, bitPosition) – Regresa el valor (0 ó 1) que tiene el bit de la posición indicada (bitPosition) de la variable x.

bitWrite(x, bitPosition, value) – Escribe el valor dado (value) en el bit de la posición indicada (bitPosition) de la variable x.

bit(bitPosition) – Regresa el valor asociado a la posición del bit indicada (bitPosition), por ejemplo: bit(0) es 1, bit(1) es 2, bit(2) es 4, etc.

Escribiendo y Leyendo bits

x << 1 equivale a x * 2 (Ej. 00000001 << 1 = 00000010)

x << 2 equivale a x * 4 (Ej. 00000001 << 2 = 00000100)

x << 3 equivale a x * 8 (Ej. 00000001 << 4 = 00001000)

x >> 1 equivale a x / 2 (Ej. 00001000 >> 1 = 00000100)

x >> 2 equivale a x / 4 (Ej. 00001000 >> 2 = 00000010)

x >> 3 equivale a x / 8 (Ej. 00001000 >> 4 = 00000001)

Manipulando grupos de bits

lowByte(int i) – Extrae el byte menos significativo

highByte(int i) – Extrae el byte más significativo

¿Para un long?

```
/*sketch Extrae_bytes_y_words
 * Para demostrar como se pueden extraer bytes y words de enteros
 */
#define _highWord(w) ((w) >> 16)
#define _lowWord(w) ((w) & 0xffff)

int intValue = 258; // 258 en hexadecimal es 0x102
long longValue = 0x1020304;
int repite = 1;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
   if (repite > 0) {
   int loWord,hiWord;
   byte loByte, hiByte;
   hiByte = highByte(intValue);
   loByte = lowByte(intValue);
   Serial.print("El valor int decimal : ");
   Serial.println(intValue,DEC);
   Serial.print("cuyo valor en hexadecimal es: ");
   Serial.println(intValue,HEX);
   Serial.print("Su byte más significativo en decimal es : ");
   Serial.println(hiByte,DEC);
   Serial.print("Su byte menos significativo en decimal es : ");
   Serial.println(loByte,DEC);
```

CONTINUACIÓN….

```
  Serial.print("Su byte más significativo en hexadecimal es : ");
  Serial.println(hiByte,HEX);
  Serial.print("Su byte menos significativo en hexadecimal es : ");
  Serial.println(loByte,HEX);
  Serial.println();

  loWord = _lowWord(longValue);
  hiWord = _highWord(longValue);
  Serial.print("El valor long decimal : ");
  Serial.println(longValue,DEC);
  Serial.print("cuyo valor hexadecimal es: ");
  Serial.println(longValue,HEX);
  Serial.print("Su word más significativo en decimal es : ");
  Serial.println(hiWord,DEC);
  Serial.print("Su word menos significativo en decimal es : ");
  Serial.println(loWord,DEC);
  Serial.print("Su word más significativo en hexadecimal es : ");
  Serial.println(hiWord,HEX);
  Serial.print("Su word menos significativo en hexadecimal es : ");
  Serial.println(loWord,HEX);

  repite--;
  delay(10000);
  }
}
```

Manipulando grupos de bits

int word(byte hi, byte low) – crea un entero (int) a partir de dos bytes

```
/*sketch Crea_int_de_bytes
 * Para demostrar como se pueden crear word o enteros desde bytes
 */
int intValue = 0x102; // 258
void setup()
{ Serial.begin(9600);
}
void loop()
{ int loWord;
  byte loByte, hiByte;
  hiByte = highByte(intValue);
  loByte = lowByte(intValue);
  Serial.print("El entero : ");
  Serial.println(intValue,DEC);
  Serial.print("está compuesto por los dos bytes ");
  Serial.print(loByte, DEC);
  Serial.print( " y ");
  Serial.println(hiByte,DEC);

  loWord = word(hiByte, loByte);  // convierte los bytes en un entero (word)
  Serial.print("el entero creado a partir de los bytes es: ");
  Serial.println( loWord, DEC);
  delay(10000);
}
```

```
/*sketch Crea_long_de_words
 * Para demostrar como se pueden crear long a partir de words
 */
#define makeLong(hi, low) (((long) hi) << 16 | (low))
#define highWord(w) ((w) >> 16)
#define lowWord(w) ((w) & 0xffff)
// declara un valor para probar
long longValue = 0x1020304; // en decimal: 16909060
// en binario : 00000001 00000010 00000011 00000100
void setup()
{ Serial.begin(9600);
}
void loop()
{ int loWord,hiWord;
  Serial.print("El valor long es: ");
  Serial.println(longValue,DEC);             // imprime 16909060
  loWord = lowWord(longValue);          // extrae dos words de longValue
  hiWord = highWord(longValue);
  Serial.print("El valor long se compone de los siguientes words ");
  Serial.println(loWord,DEC);   //imprime 772 (16 bits menos significativos)

  Serial.println(hiWord,DEC);    //imprime 258 (16 bits más significativos)
  Serial.print("Al crear un long a partir de ellos obtenemos : ");
  longValue = makeLong( hiWord, loWord);        // crea un long a partir de los dos words
  Serial.println(longValue,DEC); // debe imprimir 16909060

  delay(10000);
}
```

Funciones para manipular Números

floor(x)

ceil(x)

sin(rad)

cos(rad)

tan(rad)

PI

Funciones para manipular Números

random(max)

random(min, max)

randomseed(seed) — randomseed(aralogRead(pin))

Funciones para manipular Números

constrain(x, min, max) -- myConstrainedX= constrain(x, min, max)

map(value, fromLow, fromHigh, toLow, toHigh)

min(x,y)

max(x,y)

pow(x,y)

sqrt(x)