
Tecnológico Nacional de México

Instituto Tecnológico de Tijuana

Su dirección Académica

Departamento de Sistemas y Computación

Enero – junio 2017

Serie SC8A

Materia:

Sistemas Programables

Unidad 2

Título:

Práctica 5

Maestro:

Mitre Padilla Luis Alberto

Alumno:

Álvarez Corral Miguel Ángel 13211384

Espinoza Covarrubias Alejandro 13211465

Fecha:

Mayo 9 del 2017

## Introducción

En esta práctica se requiere con la ayuda de Arduino, un dip switch y una aplicación en C#, controlar el encendido y apagado de cuatro diodos leds, de manera que muestren los números del 0 al 15 en forma binaria, ya se colocándolos de manera manual con el dip switch o enviando el número desde la aplicación en C#, de igual manera se utilizara un módulo bluetooth HC-05 para que de esta manera se envíen y reciban los datos. En Arduino UNO se deberá cargar el código desarrollado para que este cumpla el funcionamiento requerido.

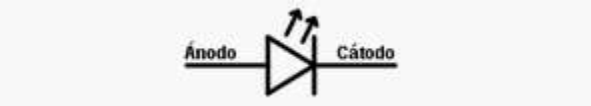
## Marco teórico

Esta práctica será realizada con los siguientes componentes:

- Protoboard
- Diodo LED
- Dip switch
- Resistencias 220 Ohms
- Arduino UNO
- Modulo bluetooth HC-05

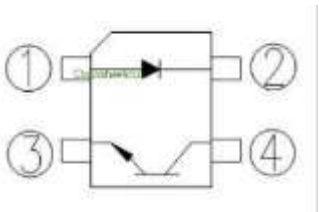
**Protboard**: es una especia de tablero con orificios, en la cual se pueden insertar componentes electrónicos y cables para armar circuitos. Como su nombre lo indica, esta tableta sirve para experimentar con circuitos electrónicos, con lo que se asegura un buen funcionamiento del mismo.

**Diodo LED**: El LED (Light-Emitting Diode: Diodo Emisor de Luz), es un dispositivo semiconductor que emite luz incoherente de espectro reducido cuando se polariza de forma directa la unión PN en la cual circula por él una corriente eléctrica. Su símbolo electrónico es:




Dip switch: Un DIP se trata de un conjunto de interruptores eléctricos que se presenta en un formato encapsulado (en lo que se denomina dual in-line-Package), la totalidad del paquete de interruptores se puede también referir como interruptor dip en singular.


**ITR 8102:** consiste en un diodo emisor infrarrojo y un Fototransistor NPN de silicio encerrado uno frente a otro convergiendo por medio de un eje óptico. Su representación en diagrama es:



**Resistencia:** se le denomina resistencia eléctrica a la oposición al flujo de electrones al moverse a través de un conductor. La unidad de resistencia en el Sistema Internacional es el ohmio, que se representa con la letra griega omega (Ω), en honor al físico alemán Georg Simon Ohm, quien descubrió el principio que ahora lleva su nombre. Su símbolo eléctrico es:

**Arduino UNO:** El Arduino Uno es una placa electrónica basada en el ATmega328 (ficha técnica). Cuenta con 14 pines digitales de entrada / salida (de los cuales 6 se pueden utilizar como salidas PWM), 6 entradas analógicas, un 16 MHz resonador de cerámica, de una conexión USB, un conector de alimentación, una cabecera ICSP, y un botón de reinicio.


**Modulo bluetooth HC-05:** EL modulo Bluetooth HC-05 viene configurado de fábrica como Esclavo, pero se puede cambiar para que trabaje como maestro, además al igual que el hc-06, se puede cambiar el nombre, código de vinculación velocidad y otros parámetros más.

¿Qué es un dispositivo bluetooth maestro y dispositivo esclavo?

- Modulo bluetooth hc-05 como esclavo:

Cuando está configurado de esta forma, se comporta similar a un HC-06, espera que un dispositivo bluetooth maestro se conecte a este, generalmente se utiliza cuando se necesita comunicarse con una PC o Celular, pues estos se comportan como dispositivos maestros.

- Modulo bluetooth hc-05 como Maestro:

En este modo, EL HC-05 es el que inicia la conexión. Un dispositivo maestro solo se puede conectarse con un dispositivo esclavo. Generalmente se utiliza este modo para comunicarse entre módulos bluetooth. Pero es necesario antes especificar con que dispositivo se tiene que comunicar.

El módulo HC-05 viene por defecto configurado de la siguiente forma:

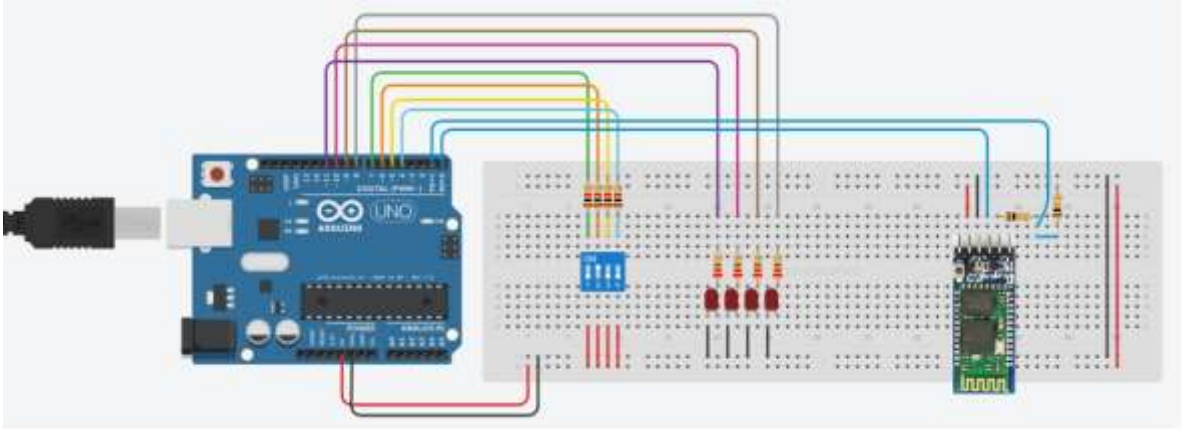
- Modo o role: Esclavo
- Nombre por defeco: HC-05

- Código de emparejamiento por defecto: 1234
- La velocidad por defecto (baud rate): 9600

EL Modulo HC-05 tiene 4 estados los cuales es importante conocer:

1. Estado Desconectado:
   1.1. Entra a este estado tan pronto alimentas el modulo, y cuando no se ha establecido una conexión bluetooth con ningún otro dispositivo.
   1.2. EL LED del módulo en este estado parpadea rápidamente
   1.3. En este estado a diferencia del HC-06, el HC-05 no puede interpretar los comandos AT.
2. Estado Conectado o de comunicación:
   2.1. Entra a este estado cuando se establece una conexión con otro dispositivo bluetooth.
   2.2. El LED hace un doble parpadeo.
   2.3. Todos los datos que se ingresen al HC-05 por el Pin RX se trasmiten por bluetooth al dispositivo conectado, y los datos recibidos se devuelven por el pin TX. La comunicación es transparente.
3. Modo AT 1:
   3.1. Para entrar a este estado después de conectar y alimentar el modulo es necesario presionar el botón del HC-05.
   3.2. En este estado, podemos enviar comandos AT, pero a la misma velocidad con el que está configurado.
   3.3. EL LED del módulo en este estado parpadea rápidamente igual que en el estado desconectado.
4. Modo AT 2:
   4.1. Para entrar a este estado es necesario tener presionado el botón al momento de alimentar el modulo, es decir el modulo debe encender con el botón presionado, después de haber encendido se puede soltar y permanecerá en este estado.
   4.2. En este estado, para enviar comandos AT es necesario hacerlo a la velocidad de 38400 baudios, esto es muy útil cuando nos olvidamos la velocidad con la que hemos dejado configurado nuestro modulo.
   4.3. EL LED del módulo en este estado parpadea lentamente.

**Diagrama de Circuit.io**



**Código de la práctica**

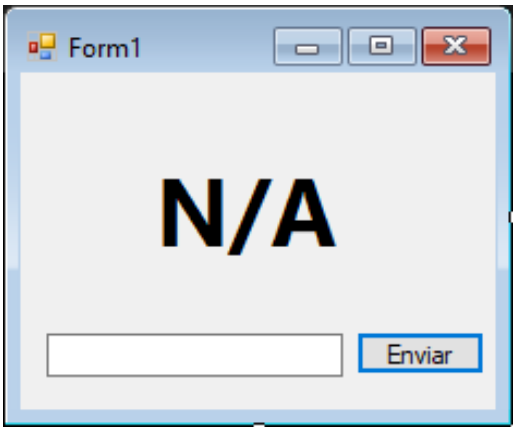
```
/*Circuito:
 * En los pines 4 - 7 se conecta el dip switch donde se tomaran las entradas.
 * En los pines 8 - 11 se conectan los leds.
 * El dip switch se conectada con un pin a la alimentación positiva y el otro
 * al pin del arduino junto con una resistencia a tierra.
 */
#include <SoftwareSerial.h> //Software Serial Port
#define RxD 0
#define TxD 1

byte state = 0;
void setup(){
  DDRD = B00001110; //Configuramos los pines 0-3 como salidas y los pines 4 - 7 como entradas.
  DDRB = B11111111; //Configuramos los pines 8-13 como salidas.
  pinMode(RxD, INPUT);
  pinMode(TxD, OUTPUT);
```

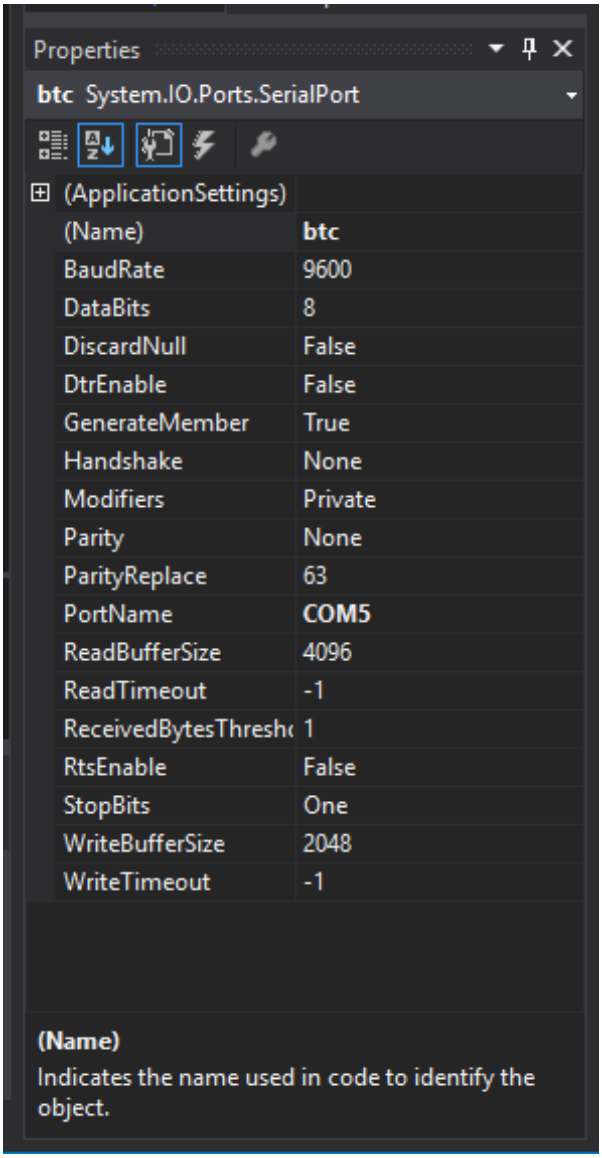
```
  Serial.begin(9600);
}

void loop(){
  if(state != (PIND & B11110001)) //Revisamos si la lectura del dip switch es diferente de la última vez
  {
    state = PIND & B11110001; //Guardamos el estado de la lectura actual
    PORTB = state >> 4; //Imprimimos el resultado en el puerto B (pines 8 - 11)
    Serial.println(PORTB);
  }
  delay(100);
  if(Serial.available()) //Si tenemos una lectura en el puerto serial.
  {
    char i = Serial.read(); //Tomamos el carácter enviado por el usuario
    byte n = (i-48); //Asumimos que es un número y restamos  48 para convertir de ascii a numérico
    if(Serial.available())
    { //Si tenemos otro número (en caso de números de dos dígitos)
      i = Serial.read();
      n = n * 10 + (i-48); //Agregamos el último digito
    }
    PORTB = n; //Imprimimos el valor del numero en los leds.
  }
}
```

## Interfaz de la aplicación en C#




## Configuración de puerto en C#



**Código de la aplicación en C#**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO.Ports;

namespace Practica_4_Unidad_2
{
    public partial class Form1 : Form
    {
        int i;
        public Form1()
        {
            InitializeComponent();
            btc.Open();
            timer1.Enabled = true;
        }

        private void btn_Send_Click(object sender, EventArgs e)
```

```
{
    try
    {
      i = int.Parse(txt_Input.Text);

       if (i < 0 || i > 15)
       {
          MessageBox.Show("Ingrese numeros dentro del rango 0 - 15.");
          return;
       }
       btc.Write(txt_Input.Text);
       lbl_Output.Text = txt_Input.Text;
    }
    catch (System.Exception)
    {
       MessageBox.Show("Entrada invalida, intente nuevamente con numeros enteros en el rango 0 - 15");
    }
}

private void timer1_Tick(object sender, EventArgs e)
{
    if (btc.BytesToRead > 0)
       lbl_Output.Text = "";

    while (btc.BytesToRead > 0)
```

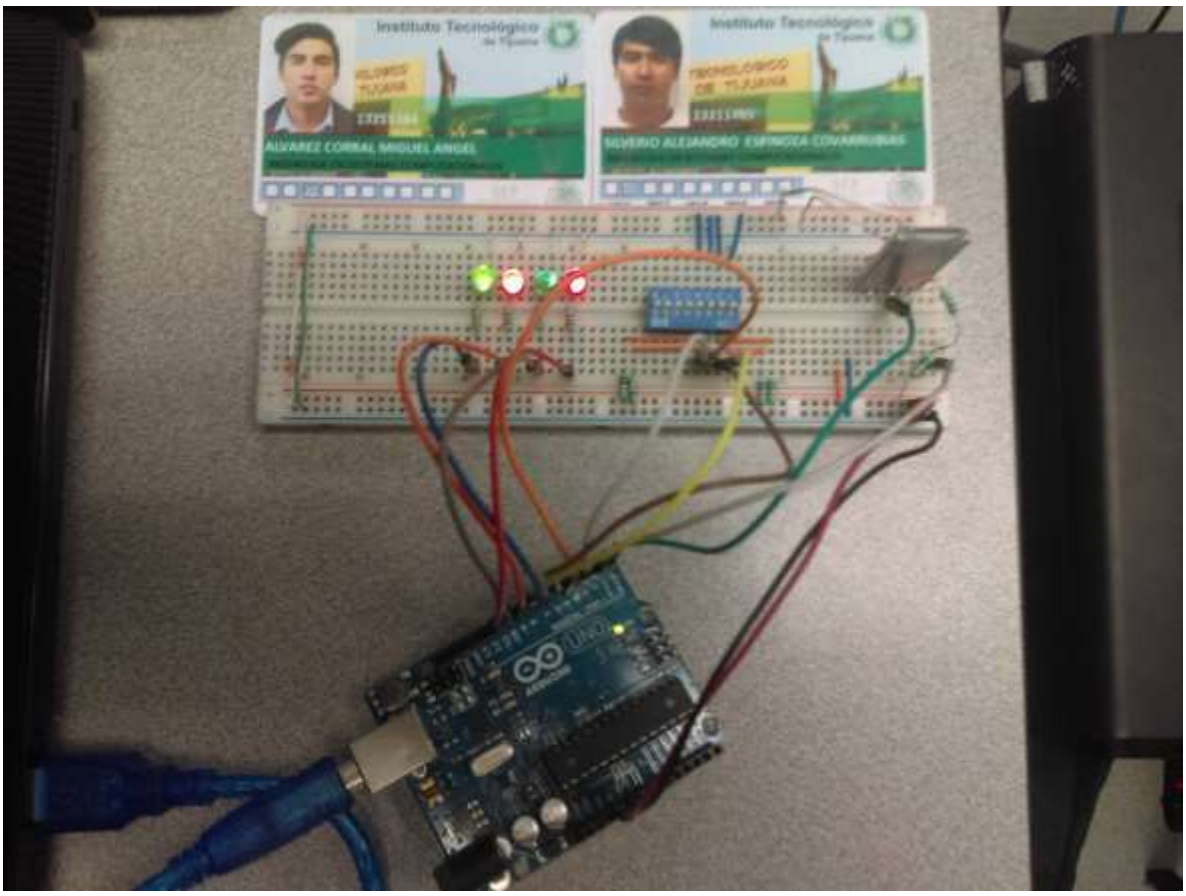
```
            {
                lbl_Output.Text += btc.ReadLine();
            }
        }

        private void Form1_FormClosed(object sender, FormClosedEventArgs e)
        {
            btc.Close();
        }

        private void txt_Input_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
```

## Diagrama real



## Conclusión

Utilizando ArduinoUNO, un protoboard, un módulo bluetooth HC-05 y una aplicación en C# se pudo lograr que la práctica funcionara de la manera requerida. Se aprendió a realizar una comunicación entre un Arduino, una aplicación en C# y el modulo bluetooth, al igual que esto no solo se puede hacer con un solo lenguaje de programación ya que se pueden implementar otros lenguajes como Python. Es importante conocer los componentes que tienen polaridad y los que no, ya que esto es un factor importante al momento de conectar los componentes. También se desarrolló un código el cual sería necesario para poder realizar el funcionamiento adecuado de los componentes antes mencionados. Es una forma útil de utilizar tanto Arduino, el modulo bluetooth, como C#, ya que no solo se pueden implementar aplicaciones como las de esta práctica sino que también muchas otras.