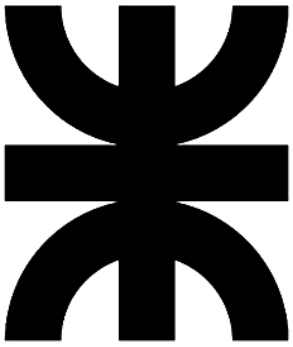
UNIVERSIDAD TECNOLÓGICA NACIONAL



FACULTAD REGIONAL PARANÁ

CARRERA: Ingeniería Electrónica
CÁTEDRA: Técnicas Digitales II

Trabajo Práctico N°6
Control desde la PC del TP Nro 5

ALUMNOS:
Battaglia Carlo
Escobar Gabriel

Paraná, 17 de noviembre de 2022

1. Circuito implementado

Mismo circuito que en el TP N°5.

2. Código

Definición y declaración de variables.

```
#define VEL_MAX 230
#define VEL_MIN 26
#define VEL_STEP 26
#define M_STOPPED 0
#define M_RUNNING 1
#define M_MANUAL 0
#define M_REMOTE 1
#define COOLDOWN_MS 100

int pinPWM = 3;
int pinOnOff = 2;
int pinVelUp = 7;
int pinVelDown = 8;
unsigned long coolDown;
byte vel = 128;
bool state = M_STOPPED;
bool mode = M_MANUAL;
bool stringComplete;
String inputString;
```

Parámetros de configuración de entradas, salidas, de la interrupción y se inicializa la comunicación serial a 9600 baudios.

```
void setup()
{
    pinMode(LED_BUILTIN, OUTPUT);
    pinMode(pinPWM, OUTPUT);
    pinMode(pinOnOff, INPUT);
    pinMode(pinVelUp, INPUT);
    pinMode(pinVelDown, INPUT);
    attachInterrupt(digitalPinToInterrupt(pinOnOff),
   switchMotorOnOff, RISING);
    coolDown = millis();

    Serial.begin(9600);
    inputString.reserve(200);
}
```

Función loop.

```
void loop()
{
    if(state == M_RUNNING)
    {
        if((millis() - coolDown > COOLDOWN_MS) && (mode ==
    M_MANUAL))
        {
            if(digitalRead(pinVelUp)) motorVelUp();
            else if(digitalRead(pinVelDown)) motorVelDown();
        }
    }else
    {
        vel = 0;
        digitalWrite(LED_BUILTIN,!digitalRead(LED_BUILTIN));
        delay(200);
    }
    analogWrite(pinPWM,vel);
}
```

Lo primero que se verifica en esta función es que el motor se encuentre encendido. Luego, se corrobora que este activo el modo de funcionamiento manual. Si así lo es, se llamará a la función para aumentar o para disminuir la velocidad acorde al botón que se haya pulsado. Si no se presionó ningún botón, se establece la velocidad en cero. Finalmente se escribe en el pin de PWM la nueva velocidad que tendrá el motor. Función asociada a la interrupción.

```
  void switchMotorOnOff()
{
    if((millis() - coolDown > COOLDOWN_MS) && (mode ==
    M_MANUAL))
    {
        state = !state;
        vel = 128;
        coolDown = millis();
        digitalWrite(LED_BUILTIN,HIGH);
    }
}
```

Esta función cambia el estado de encendido a apagado del motor o viceversa, establece la velocidad de funcionamiento al 50 % y enciende el led de status siempre y cuando el modo de funcionamiento se encuentre en MANUAL. Funciones asociadas a aumentar y disminuir la velocidad en un 10 % respectivamente.

```
void motorVelUp()
{
    if(vel < VEL_MAX) vel += VEL_STEP;
    coolDown = millis();
}
void motorVelDown()
{
    if(vel > VEL_MIN) vel -= VEL_STEP;
    coolDown = millis();
}
```

Función para establecer el valor de la velocidad del motor cuando se encuentre activo el funcionamiento en modo REMOTO.

```
bool setMotorVel(float velPercent)
{
    bool ret = true;
    if((velPercent <= 100) && (velPercent >= 0)) vel =
   velPercent*255/100;
    else ret = false;
    return ret;
}
```

Función que se ejecuta al final de la función loop cuando haya datos disponibles en el puerto serial.

```
void serialEvent()
{
    while (Serial.available())
    {
        char code, inChar = (char)Serial.read();
        float velPercent;
        String ret = "$TD2,OK*";
        inputString += inChar;
        if (inChar == '\n')
        {
            code = inputString.charAt(5);
            if(inputString.substring(0,5).equals("$TD2,") &&
   (inputString.length() <= 12) && inputString.endsWith("*\n"
   ) && (code == 'a' || mode == M_REMOTE))
            {
                switch (code)
                {
                    case 'a':
                        mode = M_REMOTE;
                    break;
                    case 'b':
                        mode = M_MANUAL;
                    break;
                    case 'c':
```

```
23                          state = M_RUNNING;
24                          vel = 128;
25                          digitalWrite(LED_BUILTIN,HIGH);
26                      break;
27                      case 'd':
28                          state = M_STOPPED;
29                          vel = 128;
30                      break;
31                      case 'e':
32                          motorVelUp();
33                      break;
34                      case 'f':
35                          motorVelDown();
36                      break;
37                      case 'g':
38                          velPercent = inputString.substring(7,
    inputString.indexOf('*')).toFloat();
39                          if(!setMotorVel(velPercent)) ret = "
    $TD2,Err*";
40                      break;
41                      default:
42                          ret = "$TD2,Err*";
43                      break;
44                  }
45              }else ret = "$TD2,Err*";
46              Serial.println(ret);
47              inputString = "";
48          }
49      }
50  }
```

Dentro de esta función se ejecuta un bucle mientras existan datos en el puerto serial.

En cada iteración del bucle, se guarda cada caracter leído en el puerto en una variable tipo “char” que luego se acumula en otra de tipo “String”. Esta última variable contendrá el mensaje completo que se envió al puerto serial.

Si el ultimo carácter leído en el puerto es un salto de línea, se ingresa al condicional “if”.

Como la información útil viene en la quinta posición del mensaje enviado, tomamos este caracter que nos determinara la acción a realizar dependiendo de su contenido. El contenido del mismo, para un correcto funcionamiento, debería ser a, b, c, d, e, f o g.

Mediante la segunda condición se comprueba que el largo del mensaje recibido no supere el tamaño del mayor posible mensaje a recibir, ya que si fuera así el mensaje quedaría descartado.

La tercera condición corrobora que el mensaje recibido finalice con un

salto de línea.

Finalmente, la última condición verifica que el modo actual de funcionamiento sea REMOTO.

Una vez cumplida todas las condiciones mencionadas con anterioridad, se ingresa a un condicional “switch”.

Dependiendo del carácter leído realizara la tarea correspondiente.

Se guarda el resultado de la operación en una variable “String”, dependiendo del éxito o no de la misma, con un mensaje que lo identifica.

Finalmente, se envía al puerto serie este resultado y se limpia el contenido de la “String” mensaje.