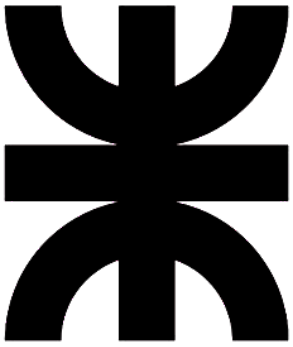
UNIVERSIDAD TECNOLÓGICA NACIONAL



FACULTAD REGIONAL PARANÁ

CARRERA: Ingeniería Electrónica
CÁTEDRA: Técnicas Digitales II

Trabajo Práctico N°5
Control de un motor de corriente continua

ALUMNOS:
Battaglia Carlo
Escobar Gabriel

Paraná, 16 de noviembre de 2022

1. Desarrollo

Para este circuito se optó por utilizar un motor de corriente continua de $12[V]$ de bajo consumo ($600[mA]$). Para comandar la velocidad de este motor se utiliza un Arduino UNO. Además, se añade una etapa de potencia que permite vincular el Arduino con el motor. En esta etapa se dispone de un transistor $TIP122$ (Darlington) el cual permite la alimentación del motor siendo controlado por el Arduino.

Debido a que es un motor de corriente continua se podrá regular su velocidad acorde a la tensión que se le aplique entre sus terminales de alimentación siendo una relación directa (entre tensión y velocidad).

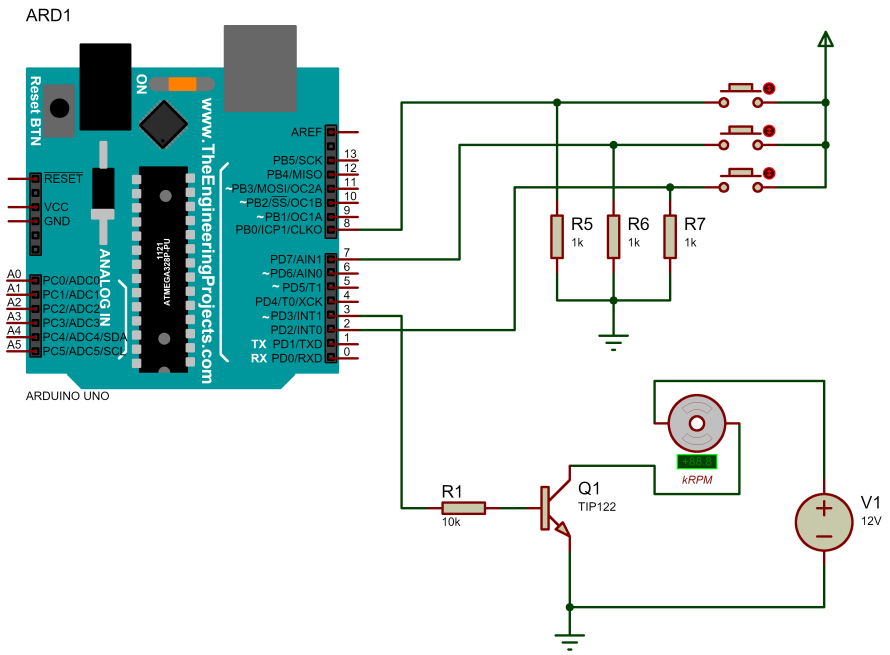
El Arduino a través del pin digital 3 enviara una señal PWM la cual, a través de una resistencia, se inyecta a la base del Darlington. La señal permite regular la velocidad de giro que adquiera el motor. Las velocidades que podrá desarrollar son discretas con saltos de un 10 % respecto a la velocidad total que pudiera desarrollar, dando un total de 9 velocidades distintas.

El terminal negativo del motor se conecta al colector del transistor y el terminal positivo a Vcc ($12[V]$). De esta manera se consigue que en el momento que se sature el Darlington, el motor tome la referencia de GND a través del emisor.

Se dispone de un pulsador para encender/apagar el motor el cual está conectado a una entrada de interrupción del Arduino a través del pin 2. Al encenderse el motor, se establece automáticamente una velocidad de giro del 50 %. Además, el circuito cuenta con dos pulsadores mas que tienen el propósito de aumentar y reducir la velocidad del motor, ambos lo harán a un 10 % de la velocidad. El pulsador utilizado para aumentar la velocidad se lo conecta como entrada al pin 7 y el pulsador para disminuir la velocidad al pin 8. El aumento o la disminución de la velocidad debe responder únicamente cuando el motor se encuentre encendido no en caso contrario. Además, la velocidad máxima que podrá conseguir el motor se producirá al 90 % de la tensión de alimentación no debiendo permitir que la misma aumente si se sigue presionando el pulsador para aumentarla. De manera similar, la velocidad mínima que podrá conseguir el motor se producirá al 10 % de la tensión de alimentación no debiendo permitir que la misma disminuya si se sigue presionando el pulsador para este fin.

Por último, se utiliza el led del Arduino para indicar el funcionamiento del motor o para indicar el modo *Stand By*.

2. Circuito implementado



3. 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 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;
```

Parámetros de configuración de entradas, salidas y de la interrupción.

```
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();
}
```

Función loop.

```
void loop()
{
    if(state == M_RUNNING)
    {
        if(millis() - coolDown > COOLDOWN_MS)
            if(digitalRead(pinVelUp) && (vel < VEL_MAX))
            {
                vel += VEL_STEP;
                coolDown = millis();
            }else if(digitalRead(pinVelDown) && (vel >
   VEL_MIN))
            {
                vel -= VEL_STEP;
                coolDown = millis();
            }
    }else
    {
        vel = 0;
        digitalWrite(LED_BUILTIN,!digitalRead(LED_BUILTIN));
        delay(200);
    }
    analogWrite(pinPWM,vel);
}
```

El primer condicional *if* corrobora que el motor este encendido.

El segundo condicional se lo utiliza con dos propósitos, uno de ellos es para evitar pulsaciones sucesibas (rebote) y el otro para permitir que se pueda mantener pulsado el botón para aumentar o disminuir la velocidad.

El tercer y cuarto condicional son para aumentar o disminuir la velocidad del motor respectivamente. Una vez dentro del condicional se establece la nueva velocidad.

El *else* implica que no se encuentra presionado ningún pulsador por lo cual establece la velocidad en cero y cambia el estado del led de status junto

con un pequeño delay. Si no se está presionando ningún botón durante un tiempo, se produce el parpadeo de este led de status por lo que determina el estado de *Stand By*.

Al final de cada bucle se escribe el valor de la velocidad en el pin utilizado para el PWM.

Por último, la función asociada a la interrupción.

```
void switchMotorOnOff()
{
    if(millis() - coolDown > COOLDOWN_MS)
    {
        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.