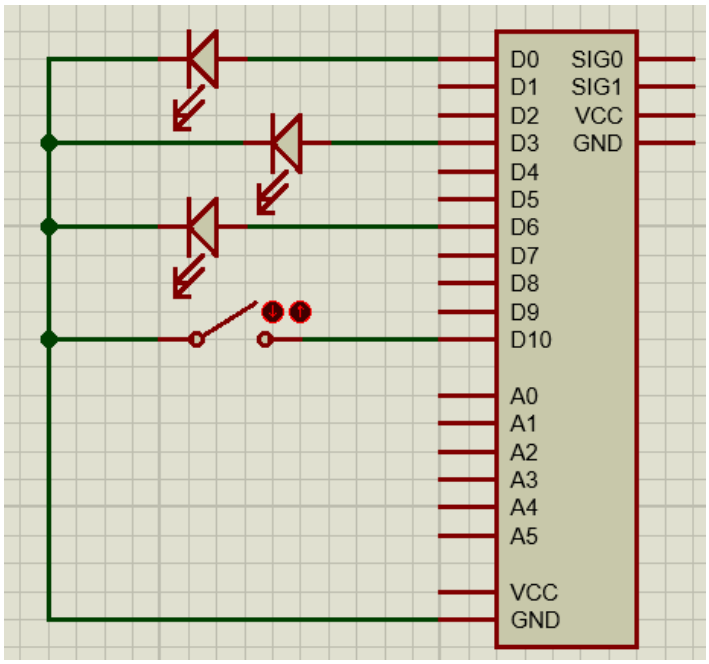
Ayudantía N°9

- ***Enunciado del problema 1:***

Se desea generar un código que alterne tres leds, de forma que apretando un pulsador éstos vayan prendiéndose y apagando el resto de los leds presentes. El pulsador debe ser corregido ya que la lectura no es limpia, por lo cual deberá ocupar procesos que le permitan solucionar este problema. Además tanto los tres leds como el pulsador están conectados al puerto D.



- ***Código 1:***

```
#define F_CPU 12000 // Frencuencia del microprocesador
#include <avr/io.h>
#include <avr/interrupt.h> //Libreria de las interrupciones

typedef struct{ // Aqui se define el tipo de estructura del proceso
    unsigned char IDTASK;
    long TIMER;
    long DELTA;
} PROC_STRUCT;
static volatile PROC_STRUCT Procesos [1]; // Crea los procesos, el número que va
entre [] debe llevar la cantidad de procesos a utilizar, en este caso 1.
ISR(TIMER0_OVF_vect){ //Va sumando al timer, en caso de ser mas de un proceso va
a variar el i por la cantidad de procesos
    for(unsigned char i=0; i<1;i++){
          Procesos [i].TIMER++;
    }
}
int main(void){
    unsigned char g=0; //Contador
    unsigned char sw1=0; //Contador de los debounce o de ''rebotes''
```

```
    DDRD |= _BV(PD0) | _BV(PD3)| _BV(PD6); // Define como salida esos pines
    PORTD |= _BV(PD10); //Define como entrada ese pin
    PORTD &= ~_BV(PD3) & ~_BV(PD6); // Cambia el estado lógico inicial a cero de
ambos pines
    TCCR0B |= _BV(CS01); // Pescaler 64 (De la libreria)
    TIMSK0 |= _BV(TOIE0); // Activa las interrupciones del timer (De la libreria)
    sei(); // Indica que vamos a ocupar las interrupciones globales (Los
procesos)
    Procesos[0].TIMER=0;
    Procesos[0].DELTA=10; // Valor para un boton

    while (1){
          if( Procesos[0].TIMER>=Procesos[0].DELTA){ //Cada vez que la tarea
timer llegue al valor de delta, se introducira a este ciclo
                Procesos[0].TIMER=0; //Hace llegar a cero el timer
                if (bit_is_clear(PIND,PD10)) { //Si el boton es apretado
                      if(sw1<250){
                            sw1++; //Contando la cantidad de debounce, sin que
pase los 250, ya que sw1 es un valor ''Unsigned char''
                      }
                      } else { //Si el boton se dejó de apretar
                      if(sw1>=30) { // Que al menos haya contado diez debounce,
valor 30 asignado por ''experiencia''
                            if (g==0){ //Va cambiando el valor de IDTASK en
caso de ser apretado el boton
                                  Procesos[0].IDTASK=1;
                                  g=1;
                                  }if (g==1){
                                  Procesos[0].IDTASK=2;
                                  g=2;
                                  } else {
                               Procesos[0].IDTASK=3;
                                  g=0;
                      } sw1=0; //El contador de debounce vuelve a cero
                }
          }
          if (Procesos[0].IDTASK==1){ // Empezado a variar lo que pase con los
led's dependiendo en que valor quedo IDTASK
                PORTD |= _BV(PD0); //Prende el primer led
                PORTD &= ~_BV(PD3) & ~_BV(PD6); //Apaga los demás led’s
          } if (Procesos[0].IDTASK==2){
                PORTD |= _BV(PD3); //Prende el segundo led
                PORTD &= ~_BV(PD0) & ~_BV(PD6); //Apaga los demás led's
          } if (Procesos[0].IDTASK==3){
                PORTD |= _BV(PD6); //Prende el tercer led
                PORTD &= ~_BV(PD3) & ~_BV(PD0); //Apaga los demás led's
          }
    }
}
```

- *Enunciado del problema 2:*

Se desea programar un botón que haga girar una polea que está conectada a un motor paso a paso, esta polea deberá mover una polea mayor la cual hará mover un brazo que regulará un paso de fluido por unos canales. Es decir que si se aprieta el botón el fluido deberá circular por un canal y se vuelve a apretar el botón este deberá fluir por el otro canal. Se adjunta una imagen del microprocesador y del sistema. Además debe considerarse las intermitencias en las señales, por lo cual debe arreglar el problema para que el equipo funcione correctamente. Las poleas solo se moverán en un sentido, por lo cual el brazo automáticamente se levantara para no realizar un tope con la puerta de cierre.

Datos:

Diámetro polea menor: 5 cm

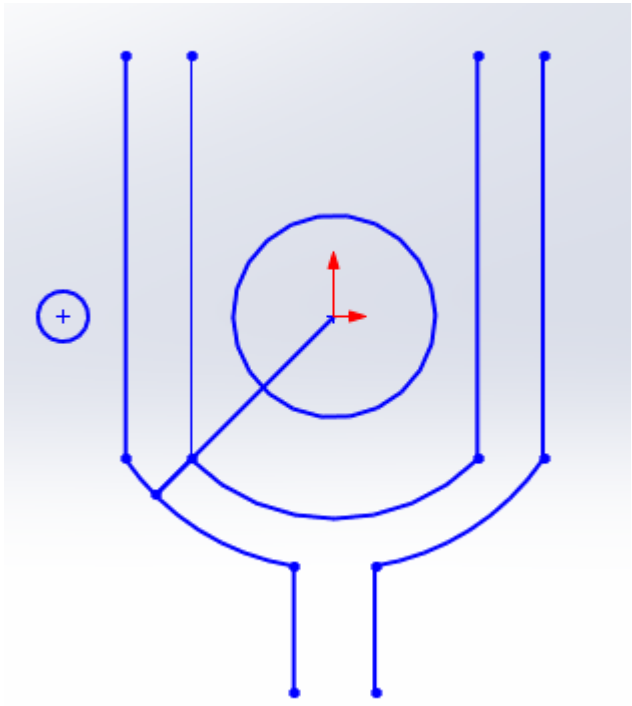
Diámetro polea mayor: 20 cm

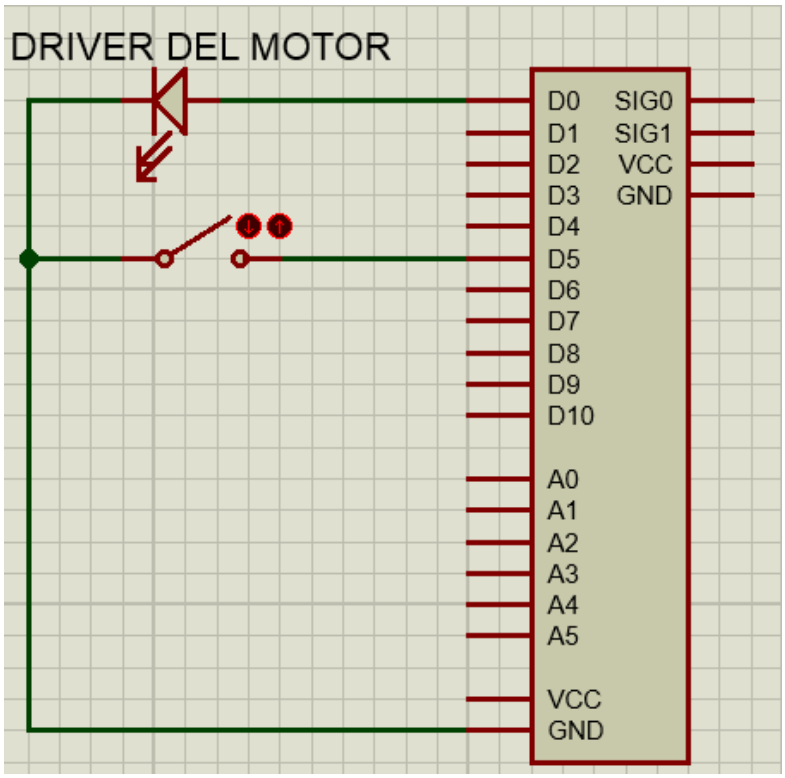
Largo total del brazo: 25 cm

Ángulo respecto al eje Y que debe situarse el brazo: 45°

Datos del motor: 200 (pasos/revolución)

Imágenes del problema:





- ***Código 1:***

```
#define F_CPU 12000 // Frencuencia del microprocesador

#include <avr/io.h>
#include <avr/interrupt.h> //Libreria de las interrupciones

typedef struct{
    int IDTASK;
    long TIMER;
    long DELTA;
} PROC_STRUCT;

static volatile PROC_STRUCT Procesos [2];
ISR(TIMER0_OVF_vect){
    for(unsigned char i=0; i<=1;i++){
           Procesos [i].TIMER++;
    }
}

int main(void){
    unsigned char sw1=0;
    DDRD |= _BV(PD0);
    PORTD |= _BV(PD5);
    TCCR0B |= _BV(CS01);
    TIMSK0 |= _BV(TOIE0);
    sei();

    Procesos[0].TIMER=0; //Procesos del boton
    Procesos[0].DELTA=10;
    Procesos[0].IDTASK=0;

    Procesos[1].TIMER=0; //Procesos del motor
    Procesos[1].DELTA=0;
    Procesos[1].IDTASK=0;

    while (1){
           if( Procesos[0].TIMER>=Procesos[0].DELTA){
                  Procesos[0].TIMER=0;
                  if (bit_is_clear(PIND,PD5)) {
                         if(sw1<250){
                                sw1++;
                         }
                         } else {
                         if(sw1>=30) { //Hasta aqui nada nuevo...
                                if (Procesos[0].IDTASK==3){ //Si idtask cambio su
número debido a que estuvo en el paso mas ''grande'' que cambie su valor a uno
para que vaya a realizar el paso mas ''pequeño''
                                       Procesos[0].IDTASK=1;
                                       } else (Procesos[0].IDTASK==2){ //Lo mismo
que el anterior pero al revés
                                       Procesos[0].IDTASK=0;
                                       }
                         sw1=0;
                  }
```

```
          }
    if(Procesos[0].IDTASK == 1){ //Para realizar el pequeño paso
            if(Procesos[1].TIMER >= Procesos[1].DELTA){
                  Procesos[1].TIMER=0;
                  if(Procesos[1].IDTASK <= 200*2){ // Realizar 200 pasos para
poder moverse 90°, por dos ya que el microcontrolador realiza dos ciclos por cada
paso
                      PORTD |= _BV(PD0); //Dejo funcionando el motor
                         Procesos[1].IDTASK++;
                  } else {
                         Procesos[1].IDTASK = 0;
                         Procesos[0].IDTASK = 2;
            }
        }
    } if(Procesos[0].IDTASK == 0){
         if(Procesos[1].TIMER >= Procesos[1].DELTA){
               Procesos[1].TIMER=0;
               if(Procesos[1].IDTASK <= 600*2){ //Lo mismo que lo anterior pero
debe recorrer 270°
                      PORTD |= _BV(PD0);
                      Procesos[1].IDTASK++;
           } else {
                      Procesos[1].IDTASK = 0;
                      Procesos[0].IDTASK = 3;
        }
    }
    } else {
     PORTD &= ~_BV(PD0); //el motor debe quedarse apagado si no se ha apretado el
boton
    }
  }
}
```