Anexo de la Práctica No. 2 Ensamblador, Interrupciones & Temporizadores: Códigos empleados

2.1. Ensamblador

Determinante

1. Escribir un código en C que obtenga el determinante, a = 10, b = 20, c = 30.

```
int discriminante(int a, int b, int c){
  return b*b-4*a*c;
}
int main(){
  printf("Discriminante con a = 10, b = 20 y c = 30          ");
  int discriminante1 = discriminante(10,20,30);
  printf("Resultado: %d",discriminante1);
  return 0;
}
```

2. Escribir en ensamblador el mismo código que el punto anterior

```
int var_out1,var_out2;
void loop() {
 delay(2000);
 asm volatile(
  "ldi r16, 0x0A \n\t"
  "ldi r17, 0X14 \n\t"
  "ldi r18, 0x1E \n\t"
  "ldi r19, 0x04 \n\t"
  "mul r17, r17 \n\t"
  "movw r26, r0 \n\t"
  "lsl r16 \n\t"
  "lsl r18 \n\t"
  "mul r18, r18 \n\t"
  "movw r28, r0 \n\t"
  "sub r26, r28 \n\t"
  "sbc r27, r29 \n\t"
  "mov %0, r26 \n\t"
  "mov %1, r27 \n\t"
  :"=r" (var_out1), "=r" (var_out2)
  :
  );
  Serial.println(var_out1);
  Serial.println(var_out2);
 delay(2000);
 }
```

Promedio

1. Escribir un código en C que obtenga el promedio de números

```
int main()
```

```
{
  printf("Promedio de n numeros\n");
  int nums[3] = {40,60,110};
  int sum = 0;
  int size = sizeof(nums)/sizeof(int);
  for(int i = 0; i< size; i++){
            sum = sum + nums[i];
  }
  printf("\n%d",sum/size);
  return 0;
}
```

2. Escribir en ensamblador el mismo código que el punto anterior

```
  int var_out1;
void loop() {
 int var_out = 0;
 asm volatile (
  "ldi r16, 0x07 \n\t"
  "ldi r17, 0x08 \n\t"
  "ldi r18, 0x01 \n\t"
  "ldi r19, 0xa  \n\t"
  "ldi r20, 0xf  \n\t"
  "ldi r21, 0x0  \n\t"
  "ldi r22, 0x05 \n\t"
  "add r16, r17 \n\t"
  "adc r16, r18 \n\t"
  "adc r16, r19 \n\t"
  "adc r16, r20 \n\t"
  "again: \n\t"
  "sbc r16, r22 \n\t"
  "inc r21 \n\t"
  "cp r16, r22 \n\t"
  "brpl again \n\t"
  "mov %0, r21 \n\t"
  : "=r" (var_out)
 );
 Serial.println(var_out1);
 delay(2000);
}
```

Entradas y salidas

1. Conectar un LED y hacerlo parpadear utilizando ensamblador
2. Conectar un botón para cambiar el estado del LED utilizando ensamblador

```
void loop() {
 asm volatile(
  "cbi %0, %1 \n\t" \
  "sbis %2, %3 \n\t" \
  "sbi %0, %1 \n\t" \
```

```
          :  : "I" (_SFR_IO_ADDR(PORTB)), "I" (PORTB5), "I" (_SFR_IO_ADDR(PINB)), "I"
(PINB5)  :
           );
        }
```

2.2. Interrupciones

1. Conectar 1 LED y hacerlo parpadear a 2Hz utilizando la función delay.

2. Conectar un botón e implementar una función que incrementa un contador cada vez que se presiona el botón.

```
int port10 = 10;
int contador = 0;
int button = 30;
int valorBoton;
//int A8 = 8;
void setup() {
  Serial.begin(9600);
  pinMode(port10,OUTPUT);
  pinMode(button,INPUT);
  pinMode(A8,INPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(port10,HIGH);
  delay(500);
  digitalWrite(port10,LOW);
  delay(500);
  contador += aumentar();
  Serial.println("cont:");
  Serial.println(contador);
  Serial.println("Analogo: ");
  Serial.println(analogRead(A8));
}
int aumentar(){
  valorBoton = digitalRead(button);
  Serial.println("valorBtn:");
  Serial.println(valorBoton);
  if(valorBoton == 1){
    return 1;
  }
  return 0;
}
```

Sensor infrarrojo

1. Reemplazar el botón por un LED infrarrojo y un fotodiodo, cada vez que hay un cambio en el estado del fotodiodo incrementar el contador.

```
int port10 = 10;
int contador = 0;//Conectar un botón e implementar una función que incrementa un contador cada vez que se presiona el botón.
int button = 30;
int valorBoton;

//int A8 = 8;


void setup() {
  Serial.begin(9600);
  pinMode(port10,OUTPUT);
  pinMode(button,INPUT);
  pinMode(A8,INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(port10,HIGH);
  delay(500);
  digitalWrite(port10,LOW);
  delay(500);
  contador += aumentar();
  Serial.println("cont:");
  Serial.println(contador);
  Serial.println("Analogo: ");
  Serial.println(analogRead(A8));
}

int aumentar(){
  valorBoton = digitalRead(button);
  Serial.println("valorBtn:");
  Serial.println(valorBoton);
  if(valorBoton == 1){
   return 1;
  }
  return 0;

}
```

2. Asociar la función a una interrupción

```
int cont = 0;
int aux1 = 0;
int aux2 = 0;

int pin10 = 10;
int led = 12;
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(pin10, INPUT);
  pinMode(led, OUTPUT);

  cli();
  DDRD &= ~(1 << DDD1);
  PORTD |= (1 << PORTD1);
  EICRA |= (1 << ISC10);
  EIMSK |= (1 << INT1);
  sei();
}

ISR(INT1_vect)
{
  digitalWrite(led, HIGH);
  aux1= digitalRead(pin10);

  if (aux1 != aux2) {
   cont++;
   if (aux1== HIGH){
            digitalWrite(led, HIGH);
   }
   }
  aux2 = aux1;
  Serial.print(cont/2);
}
void loop() {
 }
```

2.3. Temporizadores

Modos de temporizadores

Utilizando temporizadores, hacer parpadear 1 LED utilizando el modo normal a 2Hz

```
int led = 10;
int prende = 1;

//Interrupción en modo normal
void setup() {
  cli();
  TCCR1B = 0; TCCR1A = 0;
  TCCR1B |= (1 << CS12);
  TCNT1 = 34286;
  //TCNT1 = 3036;
  TIMSK1 |= (1 << TOIE1);
  sei();
  pinMode(led, OUTPUT);
```

```
}
ISR(TIMER1_OVF_vect) {
   TCNT1 = 34286;
   if(prende == 0){
     digitalWrite(led,HIGH);
     prende = 1;
   } else {
     digitalWrite(led, LOW);
     prende = 0;
   }
}
```

Utilizando temporizadores, hacer parpadear 1 LED utilizando el modo CTC a 4Hz

```
int led = 10;
int prende = 1;
void setup(){
  cli();
   TCCR1B = 0; TCCR1A = 0;
   TCCR1B |= (1 << CS12);//habilitar temporizadores
   TCCR1B |= (1 << WGM12);
   OCR1AH = 0x3D; OCR1AL = 0x09;
   TIMSK1 |= (1 << OCIE1A);
   sei();
   pinMode(led, OUTPUT);
 }
 ISR(TIMER1_COMPA_vect){
   //OCR1AH = 0x3D; OCR1AL = 0x09;
   if(prende == 0){
     digitalWrite(led,HIGH);
     prende = 1;
   } else {
     digitalWrite(led, LOW);
     prende = 0;
   }
}
```

Semáforo

```
int rojo = 10;
int amarillo = 11;
int verde = 12;
void setup() {
  pinMode(rojo,OUTPUT);
  pinMode(amarillo,OUTPUT);
  pinMode(verde,OUTPUT);
}
```

```
void loop() {
  /**
   * El rojo dura 10 segundos
   * El verde dura 15 segundos
   * El amarillo se enciende los ´ultimos 3 segundos del verde
   *
   * Al apagarse el amarillo se prende el rojo y al apagarse el rojo se prende el verde
   */
  digitalWrite(rojo,  HIGH);
  delay(10000);
  digitalWrite(rojo,  LOW);
  digitalWrite(verde,  HIGH);
  delay(12000);
  digitalWrite(amarillo,  HIGH);
  delay(3000);
  digitalWrite(verde,  LOW);
  digitalWrite(amarillo,  LOW);
}
```