UNIVERSIDAD DE GUANAJUATO

División de Ingenierías
Campus Irapuato-Salamanca

MICROCONTROLADORES Y MICROPROCESADORES

PROYECTO FINAL.
"Incubadora"

Alumnos:
HERNÁNDEZ CAMACHO JOSÉ LUIS
RIVERA MARTÍNEZ JUAN CARLOS

Profesor:
Dr. GUSTAVO CERDA VILLAFAÑA

Objetivo

Desarrollar una incubadora que lleve un registro de conteo de días, horas, minutos y segundos, lectura de datos de un sensor de temperatura y humedad y de acuerdo con todo esto poder manipular un servomotor para el proceso de agitado de los huevos, realizar el correcto calentamiento del huevo y mantener una humedad aceptable, por último, guardar los datos en la memoria EEPROM y poder realizar una interfaz dentro de una pantalla LCD.

Introducción

La automatización ha llegado a revolucionar muchos procesos y subprocesos en la industria así facilitando las actividades del ser humano, tanto como en el sector primario como en el secundario y terciario, en este proyecto se pretende facilitar el proceso de incubación de huevos de gallina llevando un conteo de horas de manera precisa con el que se registrará en la memoria del microprocesador el tiempo de incubación recorrido y en caso de cortes de energía se continue donde se quedó. Del mismo modo, usó y cuando mantener una temperatura y humedad adecuada además de un tiempo exacto para poder rotar los huevos y que de este modo tengan una incubación ideal.

Esto tiene el fin de optimizar una incubadora casera mediante la programación de una herramienta que pueda ayudar a controlar la temperatura a la que deben estar así como su humedad, controlando en todo momento la gestación de estos huevos hasta su nacimiento, con el cual podemos implementar el microcontrolador PIC 18F45K50, utilizar una pantalla LCD para poder desplegar un menú interactivo con botones externos y un sensor de temperatura y humedad para poder leer los datos del exterior, junto con un servo motor que podrá ayudarnos a rotar la cama de huevos, todo lo anterior se pretende diseñar en una plantilla de experimentos con el objetivo de presentar un prototipo funcional de este proyecto al finalizar.

Material

- PIC 18F45K50
- Servo motor de 1.8 kgF.cm
- 3 botón pulsador
- Sensor de temperatura y humedad DHT11
- Resistencias de 330 Ω y 4.7 kΩ,
- Pantalla LCD 16x2
- LED
- Motor DC (ventilador)
- Driver DRV8833
- Foco de filamento 127v
- Transistor 2N2222A
- Diodo 1N4004G
- 1 relé de bobina 5VDC

Desarrollo

Para comenzar el desarrollo del proyecto se plantean los objetivos principales de este entre los cuales hay que llevar a cabo algunas consideraciones pues el proceso de incubación es un proceso totalmente automático que para la incubación de huevos de gallina es de 21 días durante los cuales la incubadora realizará todo de manera automática sin intervención humana y solo se requerirá el suministro de agua a un recipiente en el interior de la incubadora para mantener condiciones de humedad aceptables, pues es necesario controlar el ambiente dentro de la incubadora para un correcto proceso de incubación.

El proceso debe durar 21 días en los cuales los primeros 18 días se requiere que la incubación cuente con una humedad del 55 al 60% y en los últimos días hasta un 65%. Del mismo modo la temperatura dentro de la incubadora se debe mantener en alrededor de 37.7ºC para una correcta incubación. Así mismo, se requiere girar los huevos cada hora y según varios estudios los ángulos más empleados son de 45º por lo que se estará girando entre 45º, -45º y 0º cambiando de ángulo cada hora de funcionamiento. Para mantener la humedad aceptable dentro de la incubadora se empleará un pequeño ventilador el cual se encargará de sacar el aire de su interior. El proceso de calentamiento dentro de la incubadora se llevara a través del uso de un foco de filamento que genere calor para mantener la temperatura aceptable en el interior, este foco debe pasar por la activación de un relé, pues su alimentación es de 127VCA y el microcontrolador solo proporcionará la señal de activación que se envía a través de un transistor 2N2222A el cual dará paso a la activación del relé, esto con el fin de que el transistor consuma las altas corrientes que se generan al apagar la bobina interna del relé y que pueden llegar a dañar el microcontrolador.

Los datos de temperatura y humedad son adquiridos a través de un sensor DHT11 el cual proporciona una salida digital con los datos de las mediciones y su lectura es relativamente sencilla como se muestra a continuación, pues solo requiere de enviar un par de pulsos al sensor y posteriormente realizar la lectura del mismo puerto con la respuesta del sensor.

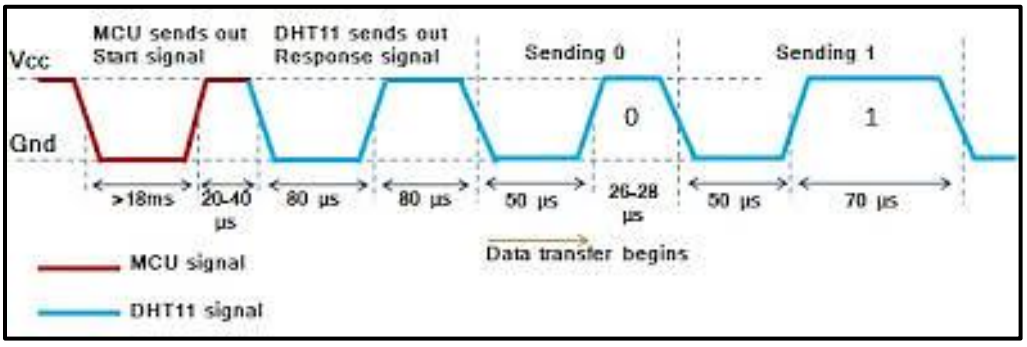


Figura 1 – Pulsos de respuesta DHT11

Por ultimo una de las consideraciones tomadas es cuando la energía eléctrica se corta pues si no se programa algo para estas situaciones el proceso perderá la cuenta del tiempo que lleva en incubación y podría llegar a ocasionar problemas, por ello se emplea la memoria EEPROM del microcontrolador en la cual se mandan a guardar los datos del día, hora y minuto en el cual se encuentra el ciclo de incubación, así como una bandera indicadora la

cual permitirá saber al microcontrolador al momento de encenderlo si se encontraba realizando un ciclo de incubación y entonces cargara los datos correspondientes al ciclo de incubación en el que se encontraba.

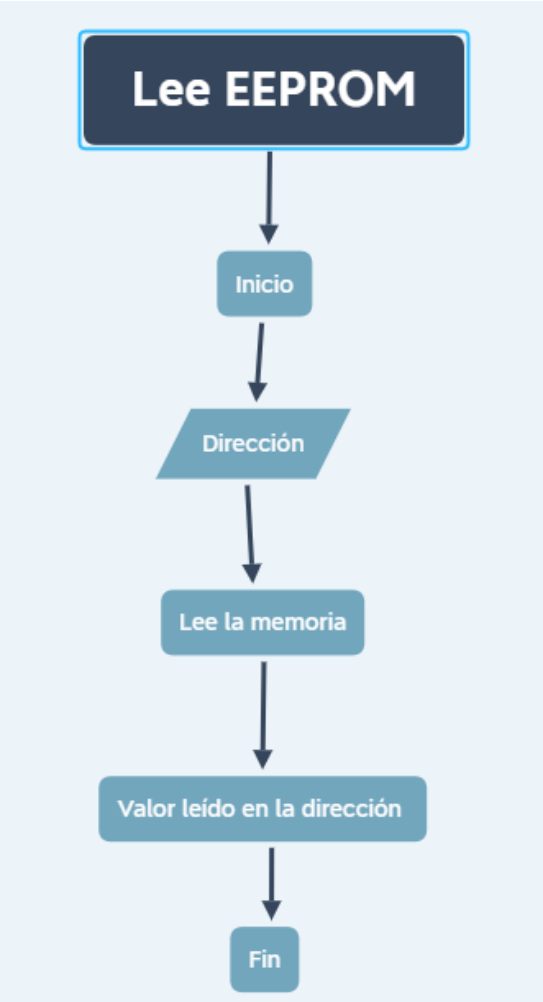
DIRECCIÓN DE MEMORIA EEPROM	DATO
0	BANDERA
1	Minutos
2	Horas
3	Días

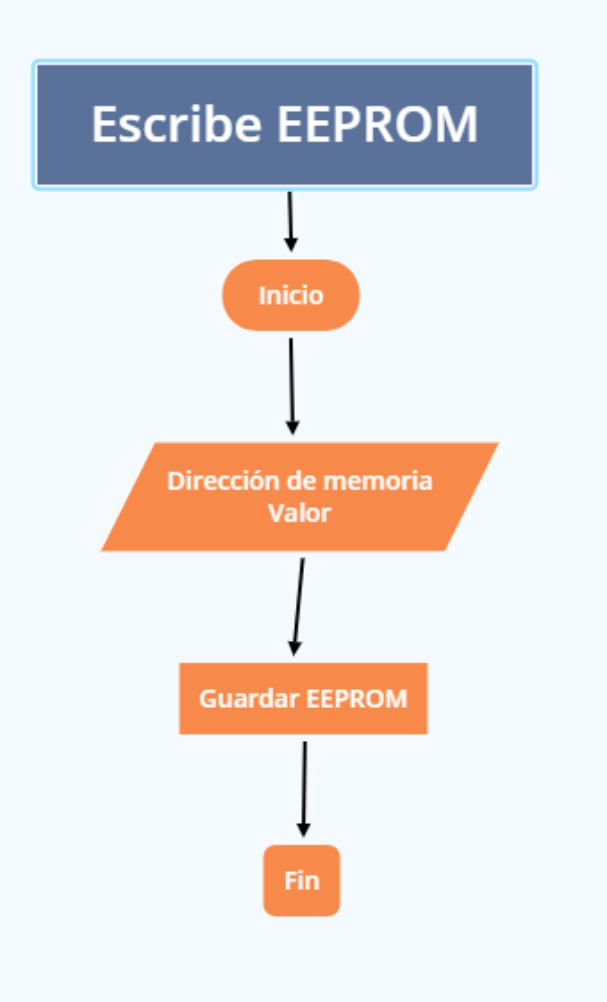
La tabla de arriba muestra como los datos se distribuyeron en la memoria EEPROM. Por último, para almacenar los datos se toma en cuenta que la memoria tiene un máximo de 100,000 escrituras por lo que se debe optimizar al máximo esta parte y es por ello que se tomó la decisión de realizar la escritura de la memoria EEPROM cada hora de funcionamiento del código guardando así el día, la hora y los minutos en que se encuentra el ciclo de incubación y que estas se puedan recuperar si hay un corte en la energía eléctrica durante el ciclo. De este modo tomando en consideración un 80% de utilidad, la vida útil esperada de la incubadora solo considerando el número de escrituras máximas de la EEPROM se esperan 9 años de funcionamiento.

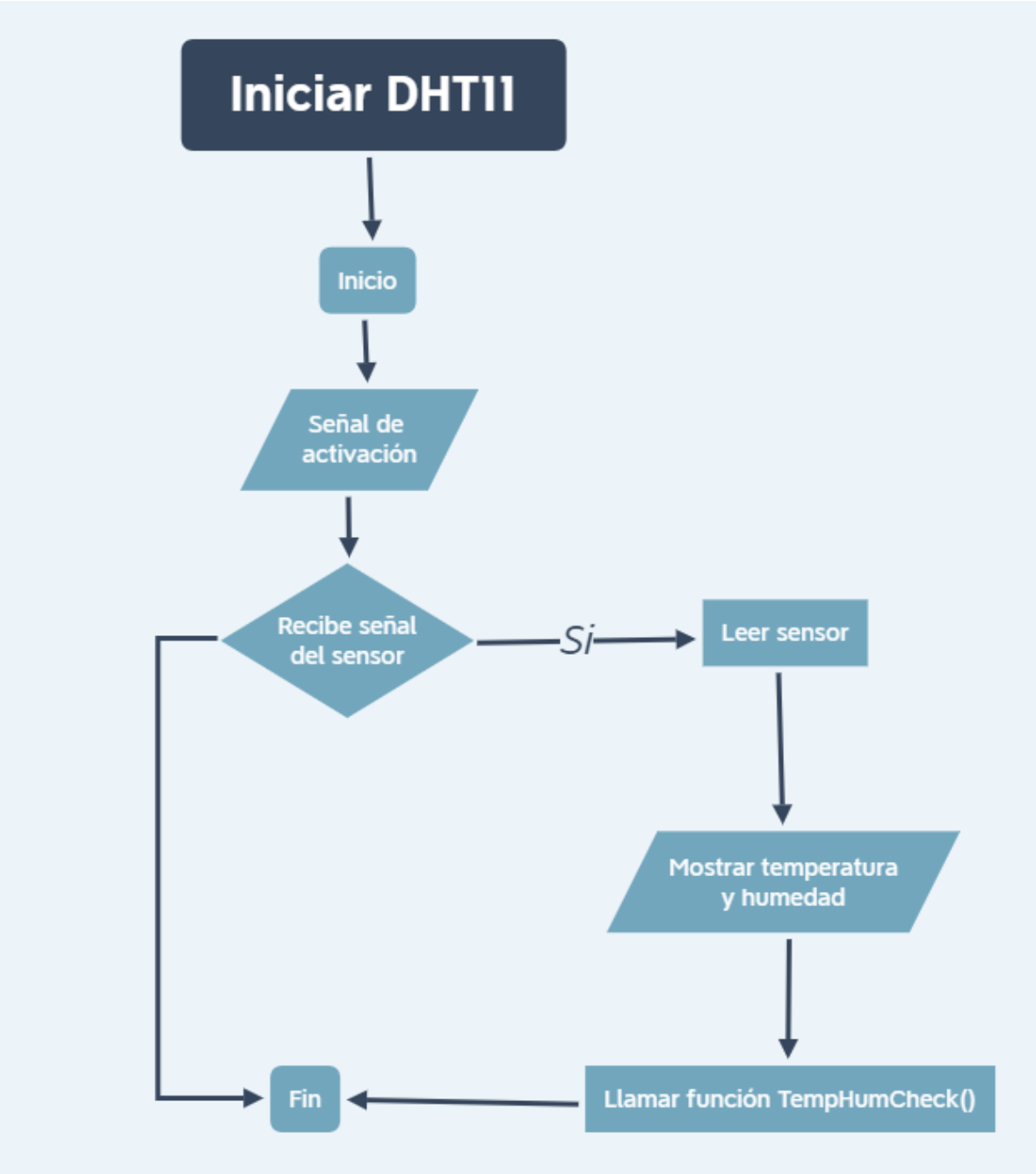
Así mismo, se utilizaron tres botones con la finalidad de poder iniciar, pausar y detener el ciclo de incubación, estos se ubicarán en la rutina principal del microcontrolador (main) ya que las interrupciones se dejarán para llevar a cabo una cuenta del tiempo del ciclo de incubación y el accionamiento del servomotor.

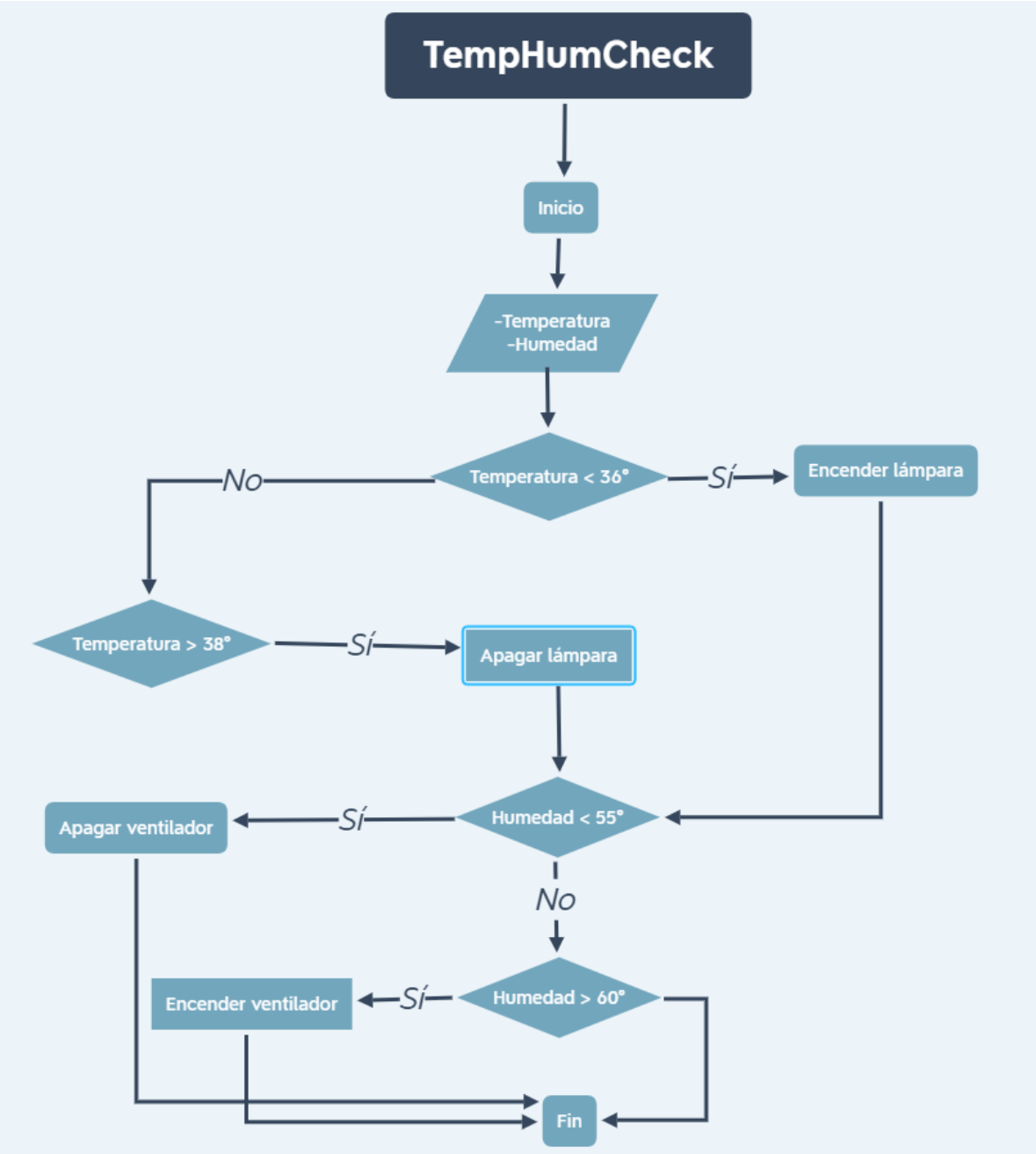
Ahora bien, para el proceso de la programación de las consideraciones anteriores fue necesaria la realización de los siguientes diagramas de flujo en los cuales basamos la programación del microcontrolador en su totalidad para lograr cumplir con los objetivos planteados.

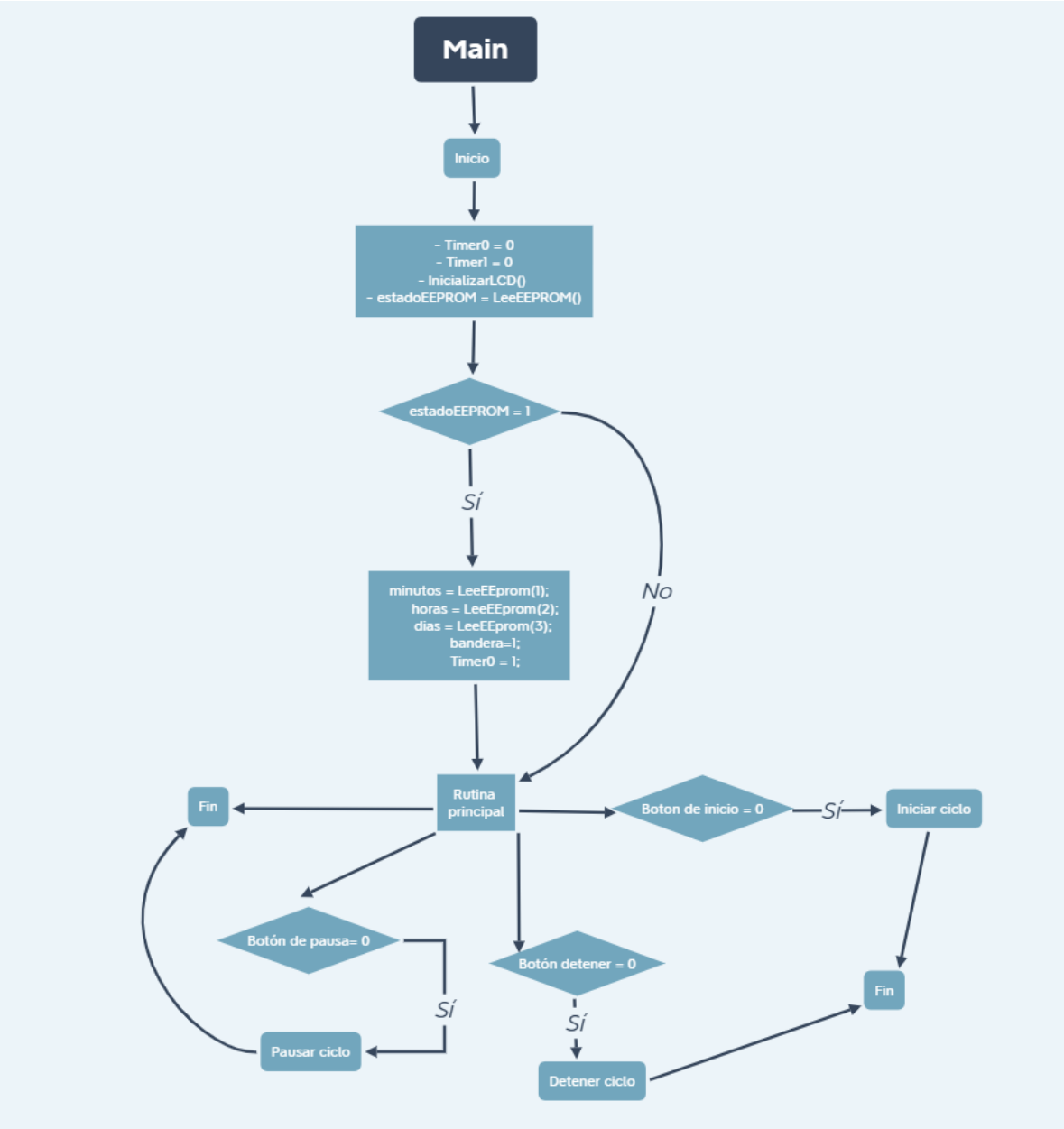
Diagramas de flujo

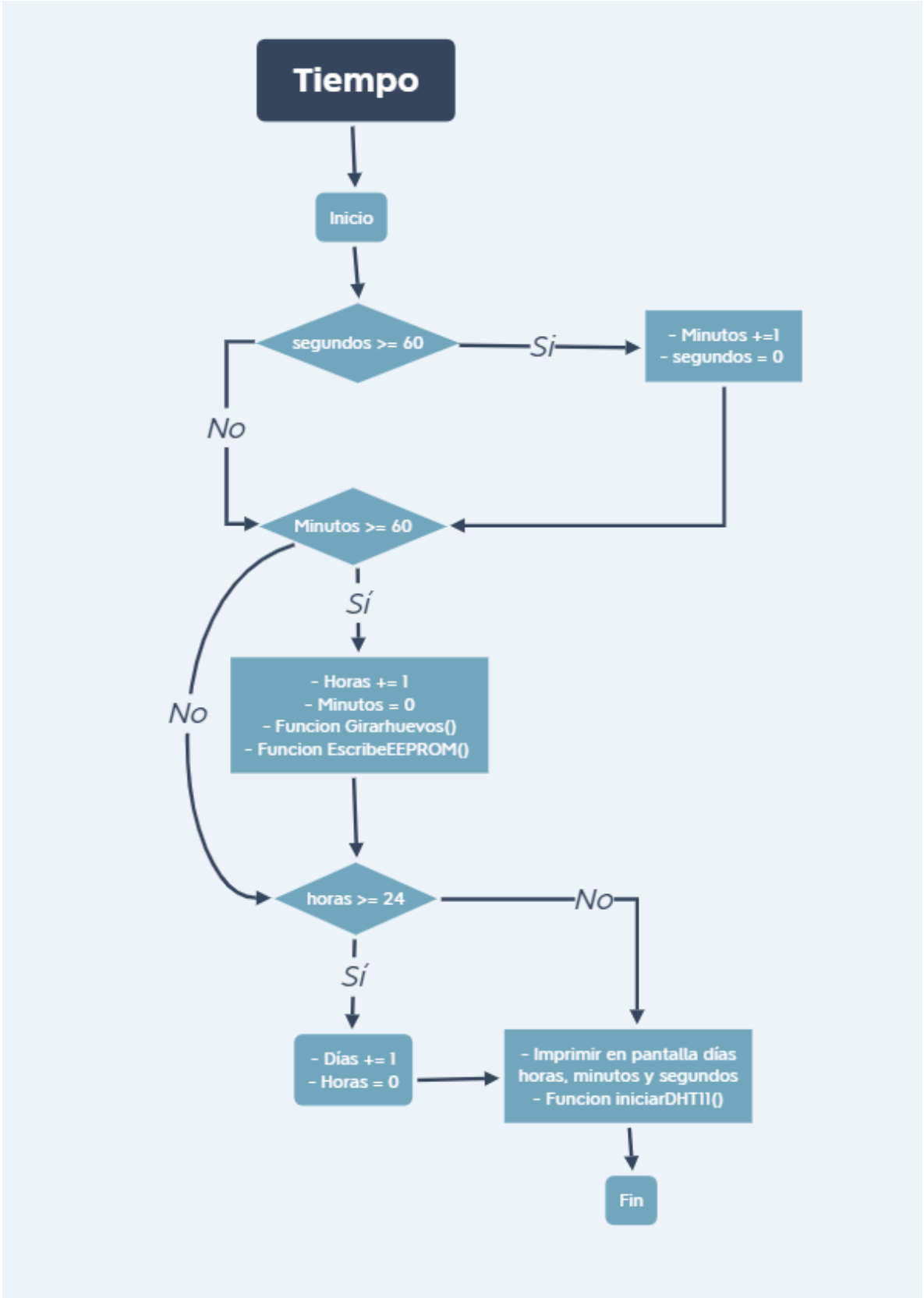


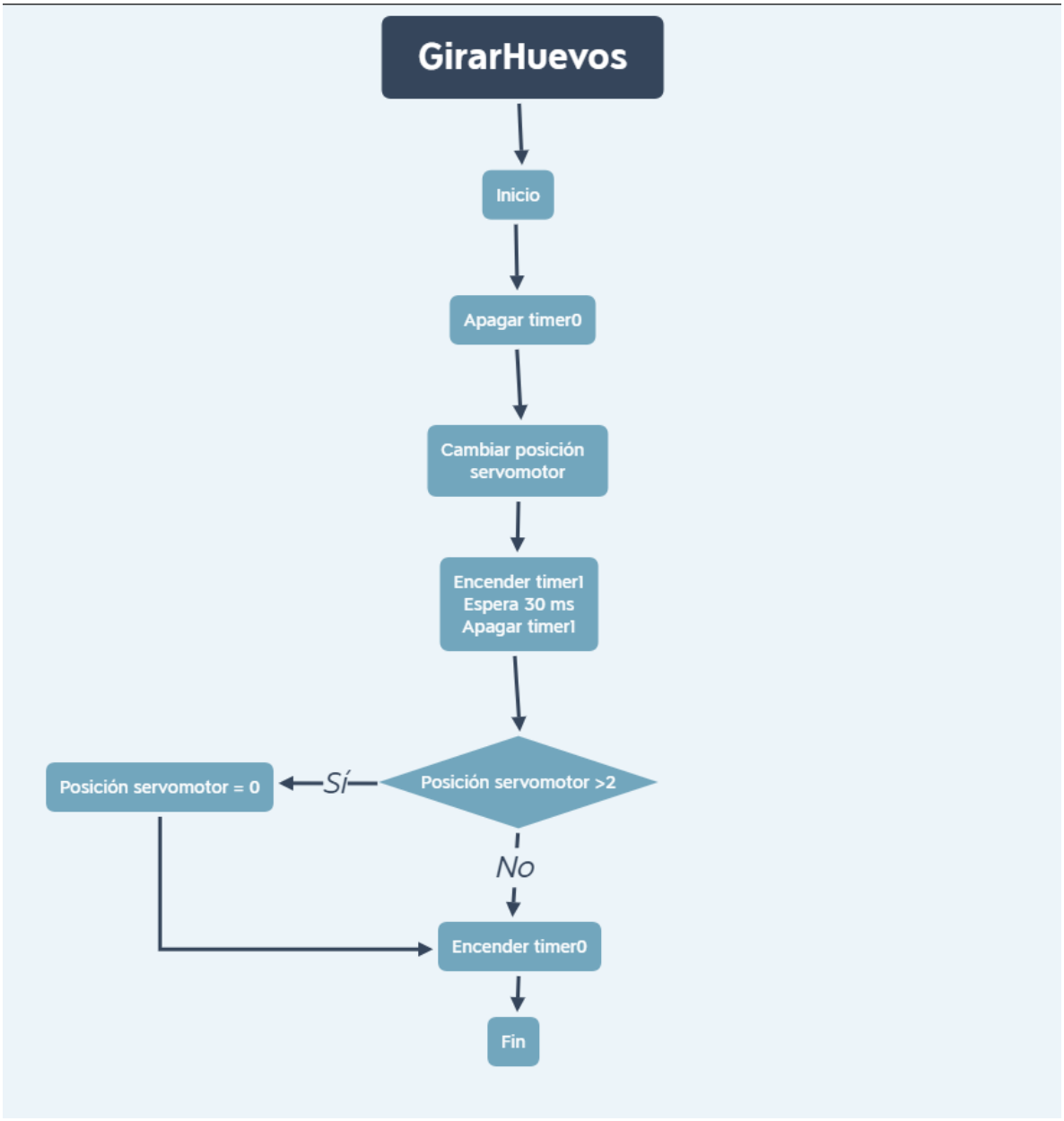


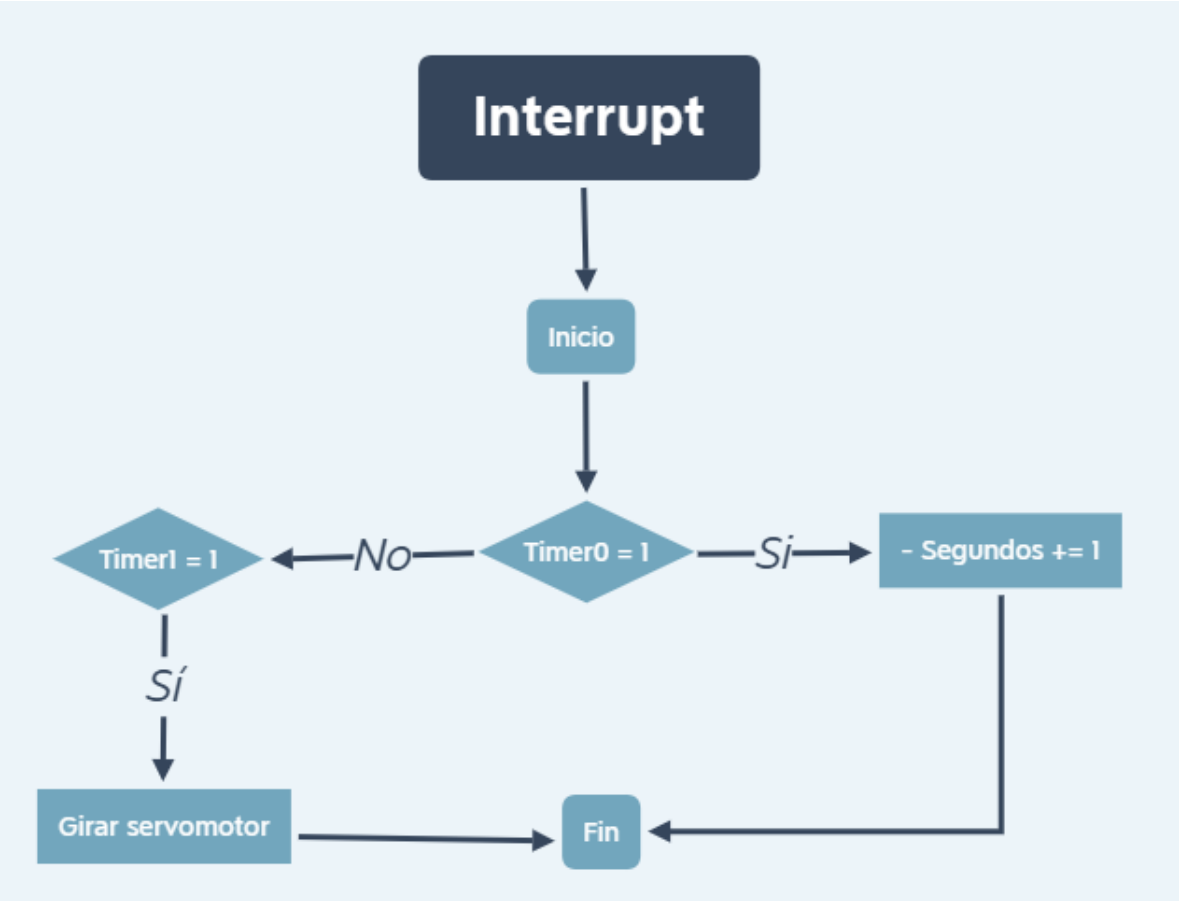

Iniciar DHT11
Inicio
Señal de activación
Recibe señal del sensor
Si
Leer sensor
Mostrar temperatura y humedad
Llamar función TempHumCheck()
Fin


TempHumCheck
Inicio
-Temperatura
-Humedad
Temperatura < 36°
No
Sí
Encender lámpara
Temperatura > 38°
Sí
Apagar lámpara
Apagar ventilador
Sí
Humedad < 55°
No
Encender ventilador
Sí
Humedad > 60°
Fin


Main
Inicio
- Timer0 = 0
- Timer1 = 0
- InicializarLCD()
- estadoEEPROM = LeeEEPROM()
estadoEEPROM = 1
Sí
No
minutos = LeeEEprom(1);
horas = LeeEEprom(2);
dias = LeeEEprom(3);
bandera=1;
Timer0 = 1;
Rutina principal
Fin
Boton de inicio = 0
Sí
Iniciar ciclo
Botón de pausa= 0
Botón detener = 0
Sí
Sí
Pausar ciclo
Detener ciclo
Fin


Tiempo
Inicio
segundos >= 60
Si
- Minutos +=1
- segundos = 0
No
Minutos >= 60
Sí
- Horas += 1
- Minutos = 0
- Funcion Girarhuevos()
- Funcion EscribeEEPROM()
No
horas >= 24
No
Sí
- Días += 1
- Horas = 0
- Imprimir en pantalla días horas, minutos y segundos
- Funcion iniciarDHT11()
Fin


GirarHuevos
Inicio
Apagar timer0
Cambiar posición servomotor
Encender timer1
Espera 30 ms
Apagar timer1
Posición servomotor >2
Sí
Posición servomotor = 0
No
Encender timer0
Fin



Basándonos entonces en los diagramas de flujo mostrados anteriormente se realizó la programación necesaria en XC8 sobre el microcontrolador PIC18F45K50 del cual se emplearon los siguientes recursos.

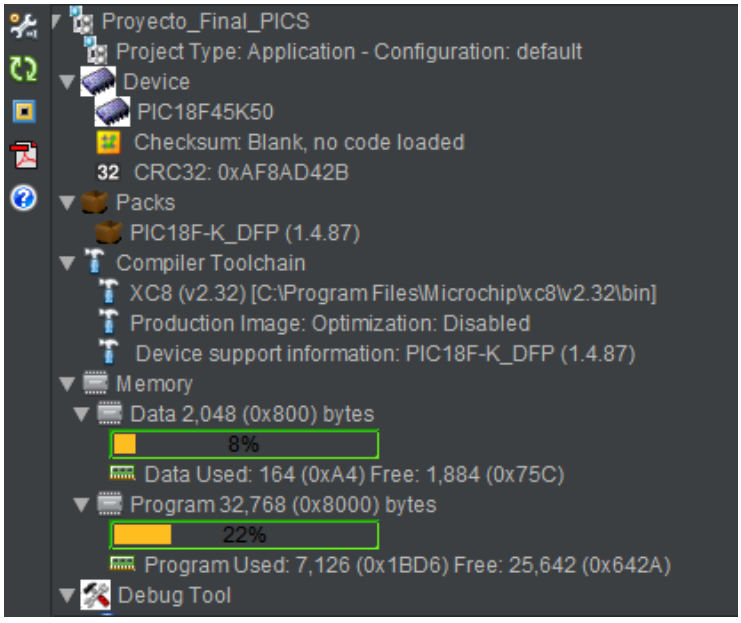


Figura 2 – Memoria utilizada en el microcontrolador.

Resultados

Inicialmente se probó el código en el simulador PROTEUS con el fin de verificar que todo estuviese funcionando de manera correcta y que no se tuvieran errores en la programación, de este modo se verifico que todo se encontraba funcionando de manera correcta

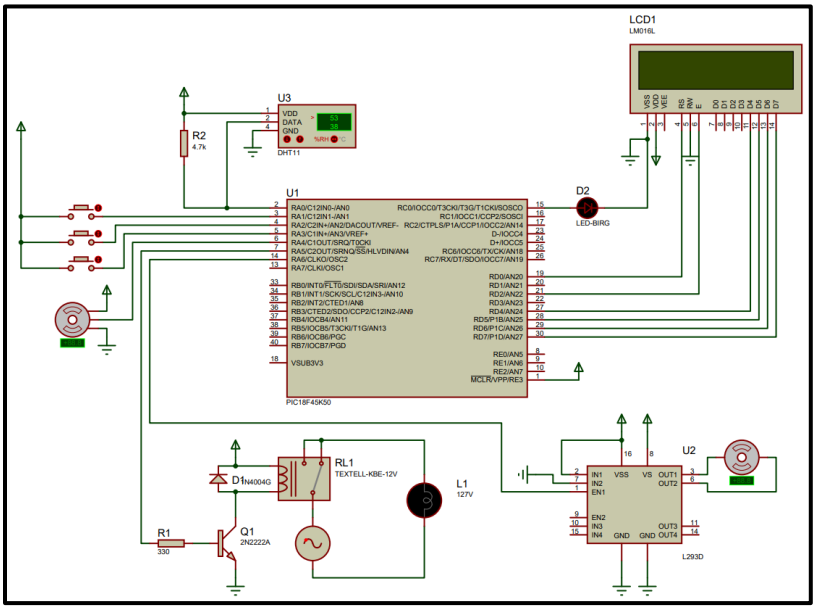


Figura 3 – Simulación del circuito en Proteus.

Se cargó el código al microcontrolador y se realizó un prototipo de la incubadora el cual se muestra a continuación:

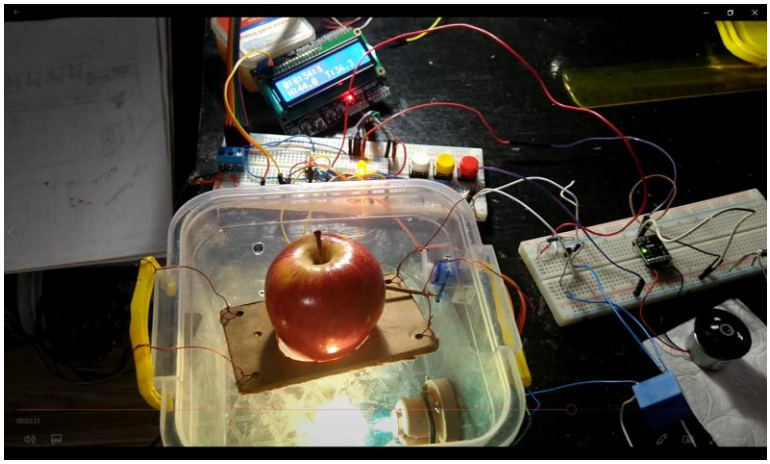


Figura 4 – Circuito físico.

En la imagen anterior se muestra un pequeño prototipo de la incubadora en el cual se pueden observar los componentes empleados en esta y que se encuentra funcionando. Para una mejor visualización de ello se adjuntó un video junto a este archivo.

Anexo del código

```
/*
 * File:   proyectoFinal.c INCUBADORA
 * Author: Hernandez Camacho Jose Luis
 *         Rivera Martinez Juan Carlos
 *
 * Created on 16 de octubre de 2021, 03:58 PM
 */
#pragma config FOSC = INTOSCIO // Oscillator Selection (Internal oscillator)
#pragma config WDTEN = OFF // Watchdog Timer Enable bits (WDT disabled in hardware
(SWDTENignored))
#pragma config MCLRE = ON // Master Clear Reset Pin Enable (MCLR pin enabled; RE3 input
disabled)
#pragma config LVP = OFF // Single-Supply ICSP Enable bit (Single-Supply ICSP disabled)
#pragma config ICPRT = OFF // Dedicated In-Circuit Debug/Programming Port Enable (ICPORT
disabled)
#include <xc.h>
#include <stdio.h>
#include <pic18f45k50.h>

#define _XTAL_FREQ 8000000// Frecuencia por default
char dias=0,horas=0,minutos=0,segundos = 0;
char servoPos = 0;
char estadoEEPROM;

//Bits de control
// T1CONbits.TMR1ON = 0;
// T0CONbits.TMR0ON = 0;
//  Empieza LCD //
void putch(char data) {     //la manda a llamar printf como 8 bits
   char Activa;
   Activa = data & 0xF0;   //deja los 4 bits mas significativos xxxx0000
   LATD = Activa | 0x05; //0bxxxx0101  con x05 los 4 bits mas significativos no cambian
                  //cuando ponen el pin 0001 significa que mandan un dato
                  //si ponen 0000 mandan un comando
                  //0101 el bit 2 sirve para activar el Enable
   __delay_us(10);
   LATD = Activa | 0x01; //0bxxxx0001
   __delay_ms(1);
   Activa = data << 4;
   LATD = Activa | 0x05;
   __delay_us(10);
   LATD = Activa | 0x01;
}
```

```
void putcm(char data) { //Putcm es para mandar comandos
   char Activa;
   Activa = data & 0xF0;
   LATD = Activa | 0x04;
   __delay_us(10);
   LATD = Activa;
   __delay_ms(1);
   Activa = data << 4;
   LATD = Activa | 0x04;
   __delay_us(10);
   LATD = Activa;
}

void InicializaLCD(void){
__delay_ms(30);
putcm(0x02);    // Inicializa en modo 4 bits
__delay_ms(1);

putcm(0x28);    // Inicializa en 2 líneas 5x7
__delay_ms(1);

putcm(0x2C);
__delay_ms(1);

putcm(0x0C);
__delay_ms(1);

putcm(0x06);
__delay_ms(1);

putcm(0x80); //Posiciona el curson en 1,1
__delay_ms(1);
}
//  TERMINA LCD //


void EscribeEEprom (char Direccion, char Valor){
   EEADR = Direccion;
   EEDATA = Valor;

   EECON1bits.WREN = 1;
   EECON2 = 0x55;
   EECON2 = 0xAA;
   EECON1bits.WR=1;
   while(EECON1bits.WR);
   EECON1bits.WREN = 0;
}
```

```
char LeeEEprom (char Direccion){
   EEADR = Direccion;
   EECON1bits.RD = 1;
   while (EECON1bits.RD); //Espera a que termine de escibir
   return EEDATA;
}

void tempHumCheck(char chumEnt,char chumDec, char ctempEnt, char ctempDec){
   //Rango de temperatura adecuado para incubacion max 38º min 36º
   //Rango de humedad adecuada para incubacion max 55 min 60 ultimos tres dias a 65

   if(ctempEnt+(ctempDec/100) < 36.0){//Encender lampara
      LATAbits.LA5 = 1;
   }else if(ctempEnt+(ctempDec/100) >38.0)//Apagar lampara
      LATAbits.LA5 =0;

   if((chumEnt+(chumDec/100) > 60.0) && (dias<=18) ){ //Encender ventilador para los primeros 18
dias
      LATAbits.LA6 = 1;
   }else if((chumEnt+(chumDec/100) < 55.0) && (dias<=18)){
      LATAbits.LA6 =0;
   }else if((chumEnt+(chumDec/100) < 63.0) && (dias>18)){ // Encender ventilador para los ultimos 3
dias
      LATAbits.LA6=0;
   }else if((chumEnt+(chumDec/100) > 65.0) && (dias>18)){
      LATAbits.LA6=1;
   }
}

void Configuracion(void){
   TRISD=0x00;
   ANSELD=0x00;
   TRISA=0b00001110; //RA0 como salida
   ANSELA=0x00;  //RA0 como digital
   TRISB=0x00; //canal B son salidas digitales
   ANSELB=0x00;
   TRISC=0x00;
   ANSELC=0x00;
   T0CON = 0b10000111; //prescaler 1:256
   INTCON = 0b10100000;//Habilitar interrupciones TMR0
   RCONbits.IPEN=0; //Desabilita interrupciones
   OSCCON = 0b01100011;

   T1CON = 0b00110111;  //timer FOSC, prescale=8, secon oscilator off, 16bits, enable timer1
   //20ms  -   0°
   TMR1L=0x77;
   TMR1H=0xEC;
```

```
PIE1=0b00000001;
   //PIR1=0b00000000; // BORRAR REGISTRO

   // Para la escritura de la EEPROM
   EECON1bits.EEPGD = 0; //Selecciona memoria de datos
   EECON1bits.CFGS = 0; // Selecciona memoria EEPROM

}

void iniciarDHT11(void){ // Funcion para la conversion A/D
   char i;
   char humEnt=0, humDec=0, tempEnt=0, tempDec=0, checksum=0;


   TRISAbits.RA0 = 0;
   LATAbits.LA0 = 0; // Pulso bajo por 20ms
   __delay_ms(20);
   LATAbits.LA0 = 1; //Pulso alto por 20us
   __delay_us(20);
   TRISAbits.RA0 = 1; //Pin como entrada
  //Señal de respuesta

   __delay_us(40);
   if(PORTAbits.RA0 == 0 ){
      __delay_us(60);

      if(PORTAbits.RA0 == 1){
         __delay_us(75);

   //while(!PORTAbits.RA0);
   //while(PORTAbits.RA0);

        //Lectura de humedad entera

         for(i=0;i<8;i++){
            while(!PORTAbits.RA0);
            __delay_us(30);
            if(PORTAbits.RA0){
               humEnt = humEnt<<1 | 1;
            }else{
               humEnt = humEnt<<1;

            }
            while(PORTAbits.RA0);
         }
```

```
//Lectura de humedad decimal
for(i=0;i<8;i++){
   while(!PORTAbits.RA0);
    __delay_us(30);
    if(PORTAbits.RA0){
       humDec = humDec<<1 | 1;
    }else{
       humDec = humDec<<1;
    }
    while(PORTAbits.RA0);
 }
//Lectura de temperatura entera
for(i=0;i<8;i++){
    while(!PORTAbits.RA0);
    __delay_us(30);
    if(PORTAbits.RA0){
       tempEnt = tempEnt<<1 | 1;
    }else{
       tempEnt = tempEnt<<1;

    }
    while(PORTAbits.RA0);
 }
//Lectura de temperatura decimal
for(i=0;i<8;i++){
    while(!PORTAbits.RA0);
    __delay_us(30);
    if(PORTAbits.RA0){
       tempDec = tempDec<<1 | 1;
    }else{
       tempDec = tempDec<<1;
    }
    while(PORTAbits.RA0);
 }
//Checksum
for(i=0;i<8;i++){
    while(!PORTAbits.RA0);
    __delay_us(30);
    if(PORTAbits.RA0){
       checksum = checksum<<1 | 1;
    }else{
       checksum = checksum<<1;
    }
    while(PORTAbits.RA0);
 }
```

```
if(checksum == (humEnt+humDec+tempEnt+tempDec)){
            putcm(0xC0);
            __delay_ms(1);
            printf("H:%d.%d  T:%d.%d",humEnt,humDec,tempEnt,tempDec);
            tempHumCheck(humEnt, humDec, tempEnt, tempDec);
         }
      }
   } // fIN DE LOS IFS DE COMPROBACION
}

void girarHuevos(void){
   //Activar el TMR1 y desactivar TMR0

   T0CONbits.TMR0ON = 0;
   servoPos +=1;

   T1CONbits.TMR1ON = 1;
   __delay_ms(30);
   T1CONbits.TMR1ON = 0;
   if(servoPos > 2){
      servoPos = 0;
   }
   T0CONbits.TMR0ON = 1;
}
void tiempo(void){
   if(segundos >= 60){
      putcm(0x01);
      __delay_ms(5);
      minutos += 1;
      segundos = 0;
      if(minutos >=60){
         horas += 1;
         minutos = 0;
         //Rutina para volteo de huevos
         girarHuevos();
         EscribeEEprom(1,minutos);
         EscribeEEprom(2,horas);
         EscribeEEprom(3,dias);
         if(horas >= 24){
            dias += 1;
            horas = 0;
         }
      }
   }
```

```
   putcm(0x80);    //Para poner el cursor en la posicion 1
   __delay_ms(1);
   printf("%d:%d:%d:%d",dias,horas,minutos,segundos);

   iniciarDHT11();

}
void __interrupt(high_priority) Inter(void){
   if(INTCONbits.TMR0IF == 1){
      segundos+=1;
      INTCONbits.TMR0IF=0;
      TMR0L = 0x7C;
      TMR0H = 0xE1;
   }
   if(PIR1bits.TMR1IF==1){

      if(servoPos==0){  //Cuando hay un 1
         LATAbits.LATA4=1;
         __delay_ms(1.25);
         LATAbits.LATA4=0;
         PIR1bits.TMR1IF=0;
         //Para -45º
         TMR1L=0xAF;
         TMR1H=0xED;
      }else if(servoPos == 1){
         LATAbits.LATA4=1;
         __delay_ms(1.5);
         LATAbits.LATA4=0;
         PIR1bits.TMR1IF=0;
         //Para 0º
         TMR1L=0x71;
         TMR1H=0xED;
      }else{
         LATAbits.LATA4=1;
         __delay_ms(1.75);
         LATAbits.LATA4=0;
         PIR1bits.TMR1IF=0;
         //Para 45º
         TMR1L=0x2C;
         TMR1H=0xEE;
         }

   }
}
```

```
void main(void) {
   __delay_ms(500);
   Configuracion();
   T0CONbits.TMR0ON = 0;
   T1CONbits.TMR1ON = 0;
   InicializaLCD();


   TMR0L = 0x7C;
   TMR0H = 0xE1;
   TMR1L=0x77;
   TMR1H=0xEC;
   char bandera=0;

   estadoEEPROM = LeeEEprom(0); // Si es 1 habia una rutina en proceso
   if(estadoEEPROM == 1){
      minutos = LeeEEprom(1);
      horas = LeeEEprom(2);
      dias = LeeEEprom(3);
      bandera=1;
      T0CONbits.TMR0ON = 1;
   }
   while(1){
      //Menu de opciones para el sistema
      LATCbits.LC0^=1;
      //tiempo();
      if((PORTAbits.RA1 == 0) && (bandera != 1) ){ // Se presiona INICIO
         //while(PORTAbits.RA1 == 0);

         bandera = 1;
         putcm(0x01);
         __delay_ms(5);
         putcm(0x80);
         __delay_ms(1);
         printf(" INICIANDO ");
         putcm(0xC0);
         __delay_ms(1);
         printf("INCUVACION");
         __delay_ms(1000);
         putcm(0x01);
         __delay_ms(5);
         //RCONbits.IPEN=1;
         T0CONbits.TMR0ON = 1;
         estadoEEPROM = LeeEEprom(0);
         EscribeEEprom(0,1);
```

```
if(estadoEEPROM == 0){
            EscribeEEprom(0,1);
            EscribeEEprom(1,0);
            EscribeEEprom(2,0);
            EscribeEEprom(3,0);
          }

      }else if((PORTAbits.RA2 == 0) && (bandera == 1)){ //Se presiona PAUSA
          //while(PORTAbits.RA2 == 1);
          bandera = 2;
          //RCONbits.IPEN = 0;
          T0CONbits.TMR0ON = 0;
          putcm(0x01);
          __delay_ms(5);
          EscribeEEprom(1,minutos);
          EscribeEEprom(2,horas);
          EscribeEEprom(3,dias);


      }else if(PORTAbits.RA3 == 0  && (bandera == 1 || bandera == 2)){ // Se presiona DETENER
          //while(PORTAbits.RA3 == 1);
          bandera = 3;
          T0CONbits.TMR0ON = 0;
          //limpiar tiempos
          horas =0;
          minutos=0;
          dias=0;
          segundos=0;
          EscribeEEprom(0,0);
          EscribeEEprom(1,0);
          EscribeEEprom(2,0);
          EscribeEEprom(3,0);

      }

      switch(bandera){
          case 0: //sin iniciar
              //putcm(0x01);
              //__delay_ms(1);
              putcm(0x80);
              __delay_ms(1);
              printf("Presiona INICIO ");
              putcm(0xC0);
              __delay_ms(1);
              printf("Para comenzar   ");
              break;
```

```
case 1: // Ciclo activo
            tiempo();
            if(dias == 21 ){
               T0CONbits.TMR0ON = 0;//Apaga timer
               bandera=0;
               putcm(0x01);
               __delay_ms(5);
               putcm(0x80);
               __delay_ms(1);
               printf("INCUBACION");
               putcm(0xC0);
               __delay_ms(1);
               printf("FINALIZADA");
               //Limpiar EEPROM
               EscribeEEprom(0,0);
               dias=0;
               horas=0;
               minutos=0;
               segundos=0;
               while(PORTAbits.RA3 == 0);


            }
            break;
         case 2://Ciclo en pausa
            putcm(0x80);
            __delay_ms(1);
            printf(" Ciclo en Pausa ");
            putcm(0xC0);
            __delay_ms(1);
            printf("Pulsa INICIO");

            break;
         case 3:
            putcm(0x01);
            __delay_ms(5);
            putcm(0x80);
            __delay_ms(1);
            printf("Ciclo Cancelado");
            putcm(0xC0);
            __delay_ms(1);
            printf("Saliendo");
            __delay_ms(2000);
            bandera = 0;

            break;
}
```

```
      __delay_ms(1000);
   }

   return;
}
```

Conclusiones

Hernández Camacho José Luis

En la realización de este proyecto se aplicaron todos los conocimientos adquiridos en la materia, pues cada parte de la programación fue requiriendo de diversos recursos que se adquirieron conforme se avanzó en la materia. El uso de los temporizadores fue clave para la realización del reloj principal de la incubadora con el timer0 y del movimiento del servomotor pues con el timer1 se estuvieron enviando las señales para girar al servomotor a posiciones específicas. La lectura y escritura en puertos fue de suma importancia para leer el sensor DHT11 que inicialmente se planeaba usar un sensor de temperatura analógico y su lectura fuese sido bastante rápida, sin embargo, al usar el DHT11 nos encontramos con que era necesario aumentar la velocidad del oscilador interno para poder realizar la lectura de los datos digitales de manera correcta. Del mismo modo, la escritura en la memoria EEPROM fue sumamente interesante pues no habíamos empleado tanto esta función y fue hasta ahora que nos vimos en la necesidad de escribir varios datos en la memoria cuando le dimos una gran utilidad a las funciones de escritura y lectura de esta. Por otro lado, el incluir el foco de 127VCA fue una parte bastante interesante pues trabajamos con nuevos componentes como el relevador y empleamos técnicas de conmutación con transistores, datos que se han visto en otras materias y que ahora son útiles para accionar componentes que requieren de un mayor potencial que los 5VDC con los que hemos estado trabajando para el microcontrolador. Sin duda alguna el proyecto fue bastante productivo para reafirmar los conocimientos y además se aprendieron un montón de cosas nuevas durante todo el desarrollo de este, así como del trabajo en equipo que se estuvo llevando durante todo su desarrollo que fue clave para cumplir con los objetivos del proyecto.

Rivera Martínez Juan Carlos

Este proyecto fue un desafío en el que pudimos poner a prueba todos los conocimientos que adquirimos durante el curso así como lo que son interrupciones, servomotor, timer0 y timer1, botones, memoria EEPROM y me siento satisfecho de poder incorporar todo lo aprendido en este proyecto, realmente fue una experiencia de estrés y gratitud durante el proceso más que nada al ver que los objetivos se iban cumpliendo conforme avanzábamos en la programación, se logró el objetivo principal del proyecto el cuál fue tener una incubadora funcional a pesar de ser un prototipo pero que tiene potencial para poder ser un sistema más grande, la parte más complicada para mí fue la programación del servomotor, también tuvimos ciertos problemas con la programación del timer0, esto debido a que teníamos una frecuencia de 1 MHz entonces todos los procesos que tenía que realizar el microcontrolador necesitaban de mucha más frecuencia para poder trabajar de manera fluida y nos dimos cuenta al ver que la pantalla LCD no reaccionaba de manera correcta al mostrar el conteo de segundos, pudimos reparar este error y cuenta de manera precisa el tiempo. Con este proyecto puedo llevarme de aprendizaje que un buen trabajo en equipo y coordinación logran cumplir cualquier meta planteada, estoy contento con los resultados y el trayecto por el que pasamos en este proyecto.

Bibliografía

- MICROCHIP Datasheet PIC18F45K50
- SGS-THOMSON Datasheet L293D
- DATASHET DEL DHT11
- Ibrahim, D. (2008). Advanced PIC Microcontroller Projects in C: from USB to RTOS with the PIC18F. In *Book* (Vol. 53, Issue 9).