**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FIEC**

**Laboratorio de Microcontroladores**

**Proyecto #2**

**Dispensador de alimentos para mascotas**

**Nombre:**

**Veronica Pozo Pera**

**John Colina**

**Paralelo: # 2**

**Grupo: # 6**

**Fecha de presentación:**

**23/02/2016**

**2015 – 2 TÉRMINO**

1. **Enunciado del proyecto**

Se desea implementar un sistema con motor para permitir o impedir el paso de alimento a una bandeja, esto debe ser programado a través de LCD - teclado y a su vez de forma inalámbrica de la siguiente manera:

1.- Se debe programar cada qué tiempo cae el alimento en la bandeja (segundos, minutos, horas).

2.- Se debe programar por cuánto tiempo permanece abierta la compuerta a la bandeja (segundos, minutos).

Estos dos deben funcionar en forma cíclica al menos que el usuario active el modo manual el cual pedirá estos dos tiempos pero solo lo hará una vez.

1. **Diagrama de Bloques**
2. **Diagrama de Flujo funcional del programa principal y de las subrutinas**

*Programa principal*

*Subrutinas*

1. **Descripción del algoritmo o estrategia utilizada**
2. Definicion de conexiones para Keypad
3. Definicion de conexiones para LCD
4. Empieza el programa
5. Leer de EEPROM:
   1. Tamaño de porción
   2. Numero horas
   3. Numero minutos
   4. Numero segundos
6. Configuración de parametros
   1. CP deshabilitado
   2. WDT deshabilitado
7. Inicialiazacion de PWM
8. Inicializacion de UART
9. Inicializacion de Keypad
10. Inicializacion de LCD
11. Presionar una tecla para configurar
12. Seleccionar el tamaño (A, B, C, D)
13. Establecer el tiempo
    1. Numero horas
    2. Numero minutos
    3. Numero Segundos
14. Guardar en la EEPORM el tamaño, horas, minutos, segundos.

INTERRUPCION

1. Verifica el tiempo que ha pasado comparándolo con el tiempo configurado por el usuario
2. Repone el TMR0 en .130
3. Limpio la bandera del TMR0
4. **Listado del programa fuente en lenguaje ensamblador con comentarios en las líneas de código que considere fundamentales**

#include "servir.c"

void interrupt();

unsigned short segundos, minutos, horas;

unsigned int counter;

unsigned short kp, hora, minuto, seg, dig, tama;

char txth,txtm,txts;

// Keypad module connections

char keypadPort at PORTD;

// End Keypad module connections

// LCD module connections

sbit LCD\_RS at RA4\_bit;

sbit LCD\_EN at RA5\_bit;

sbit LCD\_D4 at RA0\_bit;

sbit LCD\_D5 at RA1\_bit;

sbit LCD\_D6 at RA2\_bit;

sbit LCD\_D7 at RA3\_bit;

sbit LCD\_RS\_Direction at TRISA4\_bit;

sbit LCD\_EN\_Direction at TRISA5\_bit;

sbit LCD\_D4\_Direction at TRISA0\_bit;

sbit LCD\_D5\_Direction at TRISA1\_bit;

sbit LCD\_D6\_Direction at TRISA2\_bit;

sbit LCD\_D7\_Direction at TRISA3\_bit;

// End LCD module connections

void main() {

tama = EEPROM\_read(0x01);

hora = EEPROM\_read(0x02);

minuto = EEPROM\_read(0x03);

seg = EEPROM\_read(0x04);

OPTION\_REG = 0x83; // Set timer TMR0

TMR0 = 131;

INTCON = 0xA0; // Disable interrupt PEIE,INTE,RBIE,T0IE

PORTA = 0; // Turn off both displays

TRISA = 0; // All port A pins are configured as outputs

PORTb = 0; // Turn off both displays

TRISb = 0;

C1ON\_bit = 0; // Disable comparators

C2ON\_bit = 0;

PWM1\_Init(500);

ANSEL = 0;

ANSELH = 0;

UART1\_Init(19200);

Keypad\_Init(); // Initialize Keypad

Lcd\_Init(); // Initialize Lcd

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

counter = 0;

segundos = 0;

minutos = 0;

horas = 0;

kp = 0;

do{

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1, 1, "Hola ");

Lcd\_Out(2, 1, "Pres una tecla");

do {

kp =Keypad\_Key\_Click(); // Store key code in kp variable

if (UART1\_Data\_Ready()){

kp = 1;

}

}while (!kp);

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1, 1, "Elija tamaño");

delay();

tama = qtama ();

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1, 1, "Cada cuanto"); // Write message text on Lcd

Lcd\_Out(2, 1, "alimentar mascot");

delay();

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1, 1, "Numero de horas"); // Write message text on Lcd

hora = obtenerNumero (24);

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1, 1, " # de minutos"); // Write message text on Lcd

minuto = obtenerNumero (60);

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1, 1, "# de segundos"); // Write message text on Lcd

seg = obtenerNumero (60);

EEPROM\_Write(0x01,tama);

EEPROM\_Write(0x02,hora);

EEPROM\_Write(0x03,minuto);

EEPROM\_Write(0x04,seg);

counter = 0;

segundos = 0;

minutos = 0;

horas = 0;

}while (1);

}

void interrupt() {

if (INTCON.TMR0IF) {

counter++;

if (counter == 1000) {

counter =0;

segundos++;

portb++;

if (segundos == 60) {

segundos = 0;

minutos++;

if (minutos == 60) {

minutos = 0;

horas++;

}

}

}

if (segundos==seg && minutos==minuto && horas==hora) {

alimentar(tama);

counter = 0;

segundos = 0;

minutos = 0;

horas = 0;

}

}

INTCON.TMR0IF = 0;

TMR0 = 130;

}

void alimentar(unsigned short tamanio);

unsigned short qtama (unsigned short kp);

unsigned short teclado (unsigned short kp);

void delay();

unsigned short qtama () {

unsigned short kp;

qtama:

do {

kp = Keypad\_Key\_Click();

if (UART1\_Data\_Ready()){

kp = UART1\_Read();

goto bluetooth1;

}

}while (!kp);

kp = teclado(kp);

bluetooth1:

if(kp == 'A' ||kp== 'B' ||kp== 'C'||kp== 'D'){

Lcd\_chr(2, 4, kp);

do {

kp = Keypad\_Key\_Click(); // Store key code in kp variable

if (UART1\_Data\_Ready()){

kp = UART1\_Read();

goto bluetooth2;

}

}while (!kp);

kp = teclado(kp);

bluetooth2:

if(kp == '#'){

goto qtama;

}

if(kp == '\*'){

Lcd\_Chr (2,6,'G'); Lcd\_Chr (2,7,'u'); Lcd\_Chr (2,8,'a'); Lcd\_Chr (2,9,'r');Lcd\_Chr (2,10,'d');Lcd\_Chr (2,11,'a'); Lcd\_Chr (2,12,'d'); Lcd\_Chr (2,13,'o');

}

}else {

goto qtama;

}

return (kp);

}

void alimentar(unsigned short tama) {

PWM1\_Set\_Duty(70);

PWM1\_Start(); // start PWM1

Delay\_ms(1000);

PWM1\_Stop();

if (tama = 65){

Delay\_ms(100);

}

else if (tama = 66){

Delay\_ms(150);

}

else if (tama = 67){

Delay\_ms(200);

}

else if (tama = 68){

Delay\_ms(250);

}

PWM1\_Set\_Duty(50);

PWM1\_Start();

Delay\_ms(1000);

PWM1\_Stop();

}

unsigned short obtenerNumero (unsigned short numeroMax) {

unsigned short dig,numero,kp;

volverObtenerNumero:

dig = 0;

numero=0;

Lcd\_chr(2, 5, ' '); Lcd\_chr(2, 4, ' '); Lcd\_chr(2, 3, ' '); Lcd\_chr(2, 2, ' '); Lcd\_chr(2, 1, ' ');

leerNumero:

do {

kp = Keypad\_Key\_Click(); // Store key code in kp variable

if (UART1\_Data\_Ready()){

kp = UART1\_Read();

goto bluetooth4;

}

}while (!kp);

kp = teclado(kp);

bluetooth4:

if(kp == 'A' || kp== 'B' || kp == 'C' ||kp == 'D'){

goto leerNumero;

}else if (kp>=48 && kp<=57) {

dig++;

Lcd\_chr\_Cp(kp);

numero = (kp-48)+(10\*numero);

}

if(kp == '#'){

goto volverObtenerNumero;

}else if (kp == '\*' ){

Lcd\_Chr (2,6,'G'); Lcd\_Chr (2,7,'u'); Lcd\_Chr (2,8,'a'); Lcd\_Chr (2,9,'r');Lcd\_Chr (2,10,'d');Lcd\_Chr (2,11,'a'); Lcd\_Chr (2,12,'d'); Lcd\_Chr (2,13,'o');

delay();

}else if (numero >= numeroMax){

//Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_chr(2, 1,'M'); Lcd\_chr(2, 2,'a'); Lcd\_chr(2, 3,'x'); Lcd\_chr(2, 5,'t');

delay();

delay();

goto volverObtenerNumero;

}else{

goto leerNumero;

}

return (numero);

}

unsigned short teclado (unsigned short kp) {

switch (kp) {

case 1: return 49; break; // 1

case 2: return 50; break; // 2

case 3: return 51; break; // 3

case 4: return 65; break; // A

case 5: return 52; break; // 4

case 6: return 53; break; // 5

case 7: return 54; break; // 6

case 8: return 66; break; // B

case 9: return 55; break; // 7

case 10: return 56; break; // 8

case 11: return 57; break; // 9

case 12: return 67; break; // C

case 13: return 42; break; // \*

case 14: return 48; break; // 0

case 15: return 35; break; // #

case 16: return 68; break; // D

}

}

void delay(){

delay\_ms(500);

}

1. **Copia impresa del circuito armado en PROTEUS para la simulación en el momento de su ejecución**
2. **Conclusiones**

Se puede concluir que en este proyecto hemos utilizado lo aprendido durante las todas prácticas siendo muy importante saber cómo utilizar interrupciones para que el microcontrolador se detenga y realice una función específica en la interrupción.

El Timer0 puede utilizarse como temporizador o como contadores, de acuerdo a como los configuremos. En este proyecto se usó como temporizador para poder verificar el tiempo transcurrido para alimenta a la mascota.

1. **Recomendaciones**

Es recomendable usar interrupciones cuando tenemos salir del programa cada cierto tiempo (en este caso, para alimentar a la mascota). En la interrupción por TMR0, podemos pausar el programa cada cierto tiempo y verificar que el tiempo transcurrido es igual al tiempo establecido para alimentar a la mascota.

Cuando se utilizan varios archivos para las funciones y en esas funciones vamos a mostrar texto en el LCD, es recomendable mostrar carácter por carácter y no enviar una cadena de caracteres porque puede dar un error de desbordamiento en la pila.