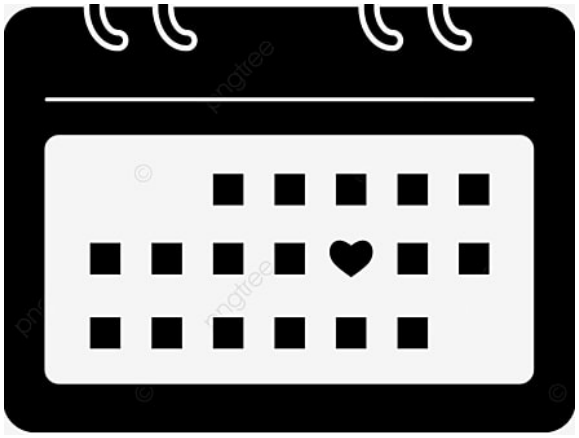
18 DE MAYO DE 2022

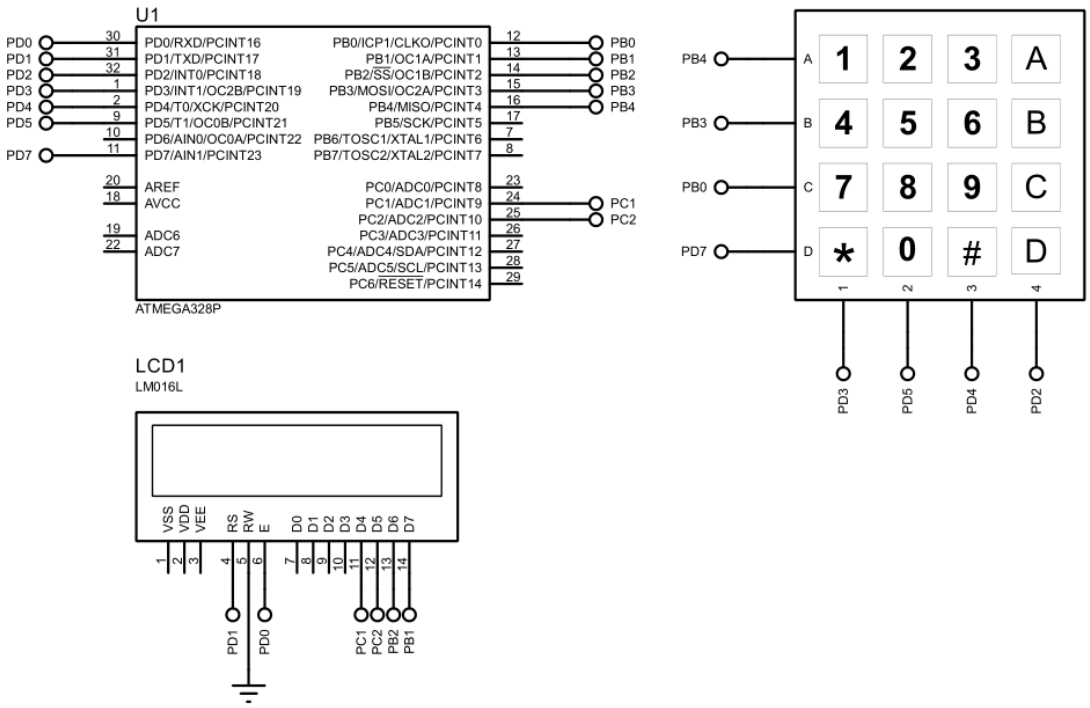


TP N°2 ENTREGABLE

BLANCO VALENTIN NICOLAS, PALADINO GABRIEL AGUSTIN
CIRCUITOS DIGITALES Y MICROCONTROLADORES

Interpretación:

Se debería realizar un proyecto en c de un reloj digital disponiendo de un LCD para mostrar los datos y un teclado matricial (4x4) para modificar los mismos. Las conexiones de estos periféricos se realizaron tal cual la figura 1.



(Figura 1: Conexiones LCD y teclado matricial al Atmega328P)

Además, para dicho proyecto se deberá contabilizar el tiempo de manera “exacta” por lo que se utilizará alguno de los timers internos del MCU para generar interrupciones periódicas. Primeramente, el LCD mostrará la hora y fecha. Por otro lado, cuando se está realizando la modificación de los datos no se debería realizar ninguna interfaz amigable, solamente realizar un parpadeo de los datos en dicho campo (año, mes, día, hora, minuto, segundo) y en segundo plano seguir contabilizando la hora sin mostrarla de manera tal que si se cancela la modificación la hora siga siendo correcta.

Resolución del problema:

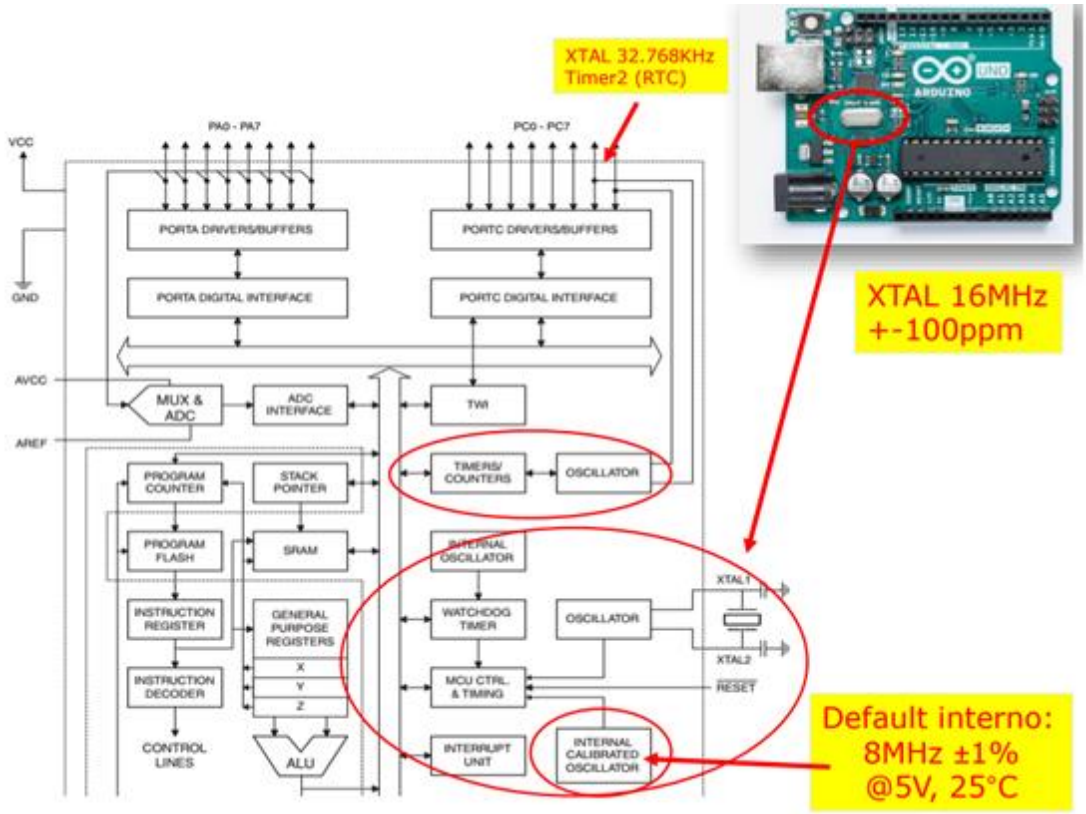
Para cada requerimiento se realizó lo siguiente:

Funcionalidad a:

Problema: En el estado por defecto se deberá mostrar en la primer línea del LCD un reloj funcionando con el formato HH:MM:SS (horas, minutos y segundos) y en la segunda línea la fecha en formato DD/MM/AA (día, mes y año). El valor inicial del reloj se establece en tiempo de compilación por ejemplo 23:59:59 de 31/12/21.

Este problema fue subdividido en dos grandes bloques para su explicación: por una parte, se encuentra la contabilización del tiempo y por otro la visualización del mismo.

Contabilización del tiempo: Antes de explicar cómo fue realizada la tarea se debe aclarar que se utilizó el oscilador externo de 16MHz que trae incorporado la placa Arduino uno (ver figura 2), dicho dato servirá para realizar los cálculos más adelante.



(Figura 2: Frecuencias de los osciladores disponibles en Arduino Uno)

Para contabilizar cada segundo se hizo uso de interrupciones, en específico de la interrupción disparada por el overflow_vector del timer 0 (_vector(16)) utilizando el modo normal de la misma (observar figura 3 para ver que TCCR0A debe ser 0x00) y el preescalador configurado en 1024 (observar figura 4 para ver que TCCR0B debe ser 0x05).

El timer fue configurado de esta manera para que no haya interrupciones muy frecuentemente y con tal de lograr una reducción del error. A partir de esta configuración decidimos por generar interrupciones periódicas cada 10 ms para ello asignamos en tcnt0 100 (se asigna cada vez que ocurre el desbordamiento y al iniciar).

Como dijimos con lo anterior se conseguía llegar a 10 ms por interrupción por lo que contabilizando internamente 100 veces se llegaría al segundo. En la parte inferior se puede observar cómo se procedió para la obtención de los valores descriptos anteriormente.

$$Se\ busca\ generar\ una\ interrupcion\ cada\ T = 10ms\ lo\ que\ nos\ da\ f = \frac{1}{T} = 100Hz.$$

$$Partiendo\ de\ estos\ 100Hz\ se\ consigue\ el\ registro\ de\ comparacion\ a\ partir\ de\ la\ siguiente\ formula:$$

$$RegComp = \frac{Frecuencia\ Clk}{Preescalador * Frecuencia\ Interrupcion} - 1 = \frac{16MHz}{1024 * 100} - 1 = 155.25$$

$Para\ asignar\ TCNT0\ se\ realiza: desbortamiento - RegComp = 255 - 155 = 100$

$Para\ calcular\ el\ error\ por\ cada\ interrupcion\ se\ utiliza:$

$$errorT = T - \frac{RegComp * Preescalador}{Frecuencia\ Clk} = 10\ ms - \frac{156 * 1024}{16MHz} = 0.016ms$$

$De\ lo\ anterior\ se\ calcula\ el\ Error\ por\ cada\ segundo\ multiplicando\ lo\ anterior\ por\ 100\ por\ lo\ tanto\ ErrorReloj = 1.6ms$

Table 14-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

(Figura 3: Tabla configuración modo de operación en timer)

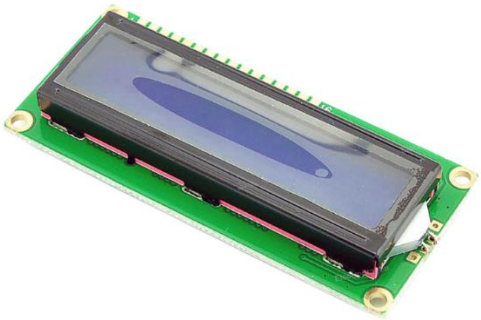
Table 14-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

(Figura 4: Tabla configuración preescaler en timer)

Visualización del tiempo: Para esta etapa se utiliza como periférico un display LCD de 16x2 (Figura 5) por lo que la pantalla cuenta con 2 filas y cada fila tiene la capacidad de mostrar 16 caracteres. Este se utiliza principalmente en proyectos de electrónica para mostrar y desplegar información lo que lo hace ideal para crear interfaz hombre-máquina para tener control o visualización de información.

Este periférico fue configurado para trabajar en modo 4 bits de datos (Bits D4 a D7 del LCD) y se configuro para enviar datos al LCD conectando R/W a tierra (R/W=0).



(Figura 5: Display LCD 16x2)

Para realizar dicha tarea se utilizaron la librería lcd.c y la cabecera lcd.h proporcionadas por la catedra. De estas se tomaron las funciones LCDGotoXY(Posx,Posy), LCDescribeDato(valor, tamaño), LCDSendchar(car) y LCDinit().

Primeramente posicionamos el cursor en la posición x=4 y=0 y procedimos a imprimir la hora actual con la función LCDescribeDato(valor, tamaño) donde para la variable valor se utilizaba la hora minuto y segundo y la variable tamaño era una constante igual a 2. Además, entre medio de cada campo se hizo una separación con ":" a partir de la función LCDSendchar(car). Luego nos posicionamos en x=4 y=1 e imprimimos la fecha actual de la misma manera que fue impresa la hora a diferencia que entre día, mes y año se separó con "/". A continuación, se puede observar el pseudocodigo de lo descripto anteriormente (Figura 6).

```
Iniciamos LCD;
Posicionamos el cursor en (4, 0);
Imprimimos hora;
Imprimimos caracter de separacion (:);
Imprimimos minutos;
Imprimimos caracter de separacion (:);
Imprimimos segundos;
Posicionamos el cursor en (4, 1);
Imprimimos dia;
Imprimimos caracter de separacion (/);
Imprimimos mes;
Imprimimos caracter de separacion (/);
Imprimimos año;
```

(Figura 6: Pseudocodigo impresión LCD)

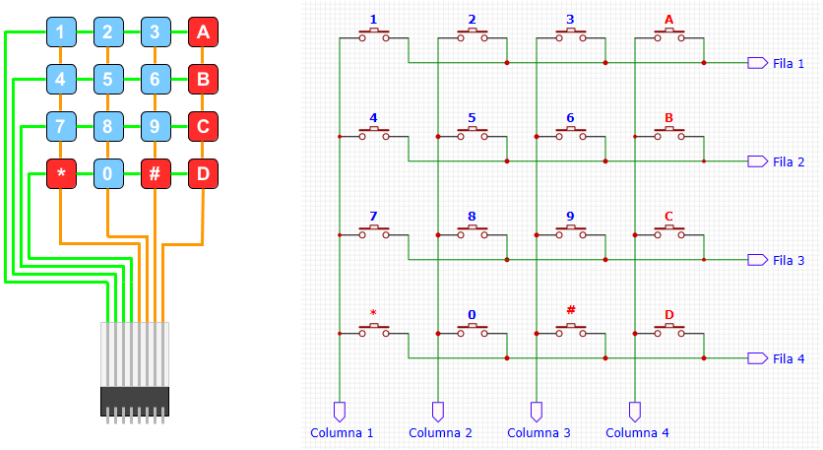
Funcionalidad B, C y D:

Problema b: Para modificar los datos (fecha y hora) se deberá presionar 'A'. Para cancelar y no aceptar la modificación se deberá presionar 'D' en cuyo caso se volverá al estado por defecto.

Problema c: El proceso de modificación es secuencial, empezando por el año y siguiendo con el mes, el día, luego la hora, minutos y por último segundos. Para pasar de un campo al siguiente se deberá presionar 'A'. Si es el campo segundos al presionar 'A' se establecerán los nuevos datos al sistema volviendo al estado por defecto.

Problema d: Cada campo a modificar tendrá su rango de validez, por ejemplo la hora tiene un rango de 0 a 23, los minutos y segundos de 0 a 59, el año de 0 a 99, los meses de 1 a 12. Los días dependen de cada mes. Para incrementar el valor actual de un campo se deberá presionar 'B' y para decrementarlo se deberá presionar 'C'.

Para realizar estas tareas hizo falta la incorporación de un teclado matricial 4x4 por lo cual teníamos la posibilidad de configurar 16 teclas (en la figura 7 puede observarse como es internamente). Los pines correspondientes a las filas del mismo fueron conectados a puertos del MCU configurados como salida, por otro lado los correspondientes a columnas fueron conectados a puertos del MCU configurados como entrada con pull-up interno.



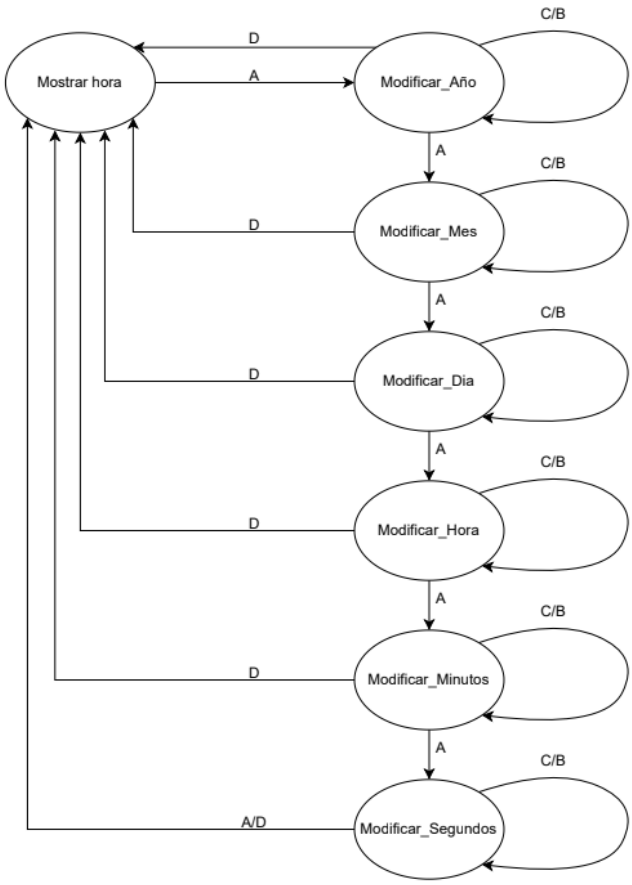
(Figura 7: Diagrama interno de un teclado matricial 4x4)

Para detectar la pulsación de una tecla actuaremos de forma parecida a la lectura de un pulsador, es decir, ponemos a tierra un extremo del pulsador, y el otro lo conectamos a una entrada digital con una resistencia de pull-up interna dada por el microcontrolador. Ahora para leer todas las teclas tendremos que hacer un barrido por filas, para ello ponemos todas las filas a 5V. Prosiguiendo ponemos una fila a 0V, y leemos las entradas de la columna. Una vez realizada la lectura de toda la fila volvemos a ponerla a 5V, pasamos a la siguiente, y volvemos a realizar el proceso hasta recorrer todas las filas. Todo esto fue realizado en la función KEYPAD_Scan(*tecla) que incorporamos de la librería KEYPAD.h (en la figura 8 se puede observar el pseudocodigo de dicha función). A partir del parámetro tecla se consigue saber la fila y columna del botón presionado utilizando la formula Tecla=fila*4+columna. Luego este valor se pasa como parámetro a la función Chequear_Car(*car) que indica el carácter correspondiente a Tecla.

```
Recorro Filas de a una(for);
    Pongo fila correspondiente en 0 logico y las restantes en 1;
    Recorro Columnas de a una;
        Pregunto si la entrada (correspondiente a fila y columna) esta en 0 logico
         en caso verdadero devuelvo 1 y asigno tecla=fila*4+columna;
devuelvo 0 si no se pulso ninguna tecla;
```

(Figura 8: Pseudocodigo para detectar pulsación de tecla)

Se explicaron las funcionalidad b, c y d en conjunto ya que entre estas forman una máquina de estados. Para nuestro caso optamos por utilizar una maquina del tipo Mealy ya que la funcionalidad (cambios de estado) depende del estado actual y la entrada. En la figura 9 se puede ver como fue planteada la MEF para resolver las funcionalidades b, c y d.



(Figura 9: MEF planteada como Mealy)

Para la funcionalidad B se definió el estado Mostrar_Hora y se definieron las entradas A y D donde A realiza el cambio de transición de estado para modificar los distintos campos mientras que D vuelve al estado inicial (Mostrar_Hora). Cabe aclara que mientras se está en algún estado de modificación de campo el reloj sigue contabilizando por mas que no se actualice en la pantalla de manera tal que si se cancelan las modificaciones (se pulsa D), la hora sigue de forma correcta.

Para la funcionalidad C se definieron los estados Modificar_Año, Modificar_Mes, Modificar_Dia, Modificar_Hora, Modificar_Minutos y Modificar_Segundos, donde cada uno de estos modifica su campo en el LCD.

Para la funcionalidad D se definen las entradas B y C donde si la MEF se encuentra en un estado de modificación B realiza un incremento de la variable mientras que C un decremento de la misma.

En la figura 10 se puede observar el pseudocodigo de la máquina de estados.

```
En caso de estar en estado Mostrar_Hora:
    Si se presiona A se pasa de estado a Modificar_Año;
En caso de estar en estado Modificar_Año:
    Si se presiona A se pasa de estado a Modificar_Mes;
    Sino si se presiona B se incrementa el año;
    Sino si se presiona C se decrementa el año;
    Sino si se pulsa D se vuelve a Mostrar_Hora no habiendo modificado nada;
En caso de estar en estado Modificar_Mes:
    Si se presiona A se pasa de estado a Modificar_Dia;
    Sino si se presiona B se incrementa el mes;
    Sino si se presiona C se decrementa el mes;
    Sino si se pulsa D se vuelve a Mostrar_Hora no habiendo modificado nada;
En caso de estar en estado Modificar_Dia:
    Si se presiona A se pasa de estado a Modificar_Hora;
    Sino si se presiona B se incrementa el dia;
    Sino si se presiona C se decrementa el dia;
    Sino si se pulsa D se vuelve a Mostrar_Hora no habiendo modificado nada;
En caso de estar en estado Modificar_Hora:
    Si se presiona A se pasa de estado a Modificar_Minutos;
    Sino si se presiona B se incrementa la hora;
    Sino si se presiona C se decrementa la hora;
    Sino si se pulsa D se vuelve a Mostrar_Hora no habiendo modificado nada;
En caso de estar en estado Modificar_Minutos:
    Si se presiona A se pasa de estado a Modificar_Segundos;
    Sino si se presiona B se incrementan los minutos;
    Sino si se presiona C se decrementan los minutos;
    Sino si se pulsa D se vuelve a Mostrar_Hora no habiendo modificado nada;
En caso de estar en estado Modificar_Segundos:
    Si se presiona A se pasa de estado a Mostrar_Hora con la hora actualizada;
    Sino si se presiona B se incrementan los segundos;
    Sino si se presiona C se decrementan los segundos;
    Sino si se pulsa D se vuelve a Mostrar_Hora no habiendo modificado nada;
```

(Figura 10: Pseudocodigo de la implementación de MEF por software)

Funcionalidad E:

Problema: Mientras transcurre la modificación de un campo, el valor del mismo deberá mostrarse parpadeando cada un segundo hasta que el usuario finalice presionando ‘A’ para pasar al campo siguiente o para finalizar, o presionando ‘D’ para cancelar.

Para realizar el parpadeo cada un segundo se utilizó de la interrupción generada por el timer0 (vista en Funcionalidad a) donde para cada segundo que transcurre, en caso de encontrarse en un estado de modificación, se invocara una función para hacer parpadear los dígitos del estado correspondiente y que en caso de no estar en uno de estos estados (encontrarse en Mostrar_Hora) ir actualizando el LCD mostrando la hora actual. Esta interrupción puede observarse bien en la figura 11.

```
Dentro de la interrupcion
    si paso un segundo
        Si me encuentro en un estado que no es Mostrar_Hora
            Invoco la funcion para parpadear
        Caso contrario
            Muestro hora y fecha actual en el LCD
```

(Figura 11: Pseudocodigo de interrupción)

La función parpadear consta de una variable interna para saber si los dígitos están encendidos o apagados de modo tal que si están apagados se enciendan y caso contrario se apaguen. Para ello utilizamos funciones como LCDstring(string, cantidad) y de la función LCDescribeDato(valor, tamaño). En la Figura 12 se puede observar un pseudocodigo de ello.

```
Dentro de la funcion parpadear
    Posiciono el cursor en los digitos correspondientes al estado actual
    si tengo el digito apagado
        escribo en el LCD el digito
    sino
        escribo en el LCD espacios en blanco
```

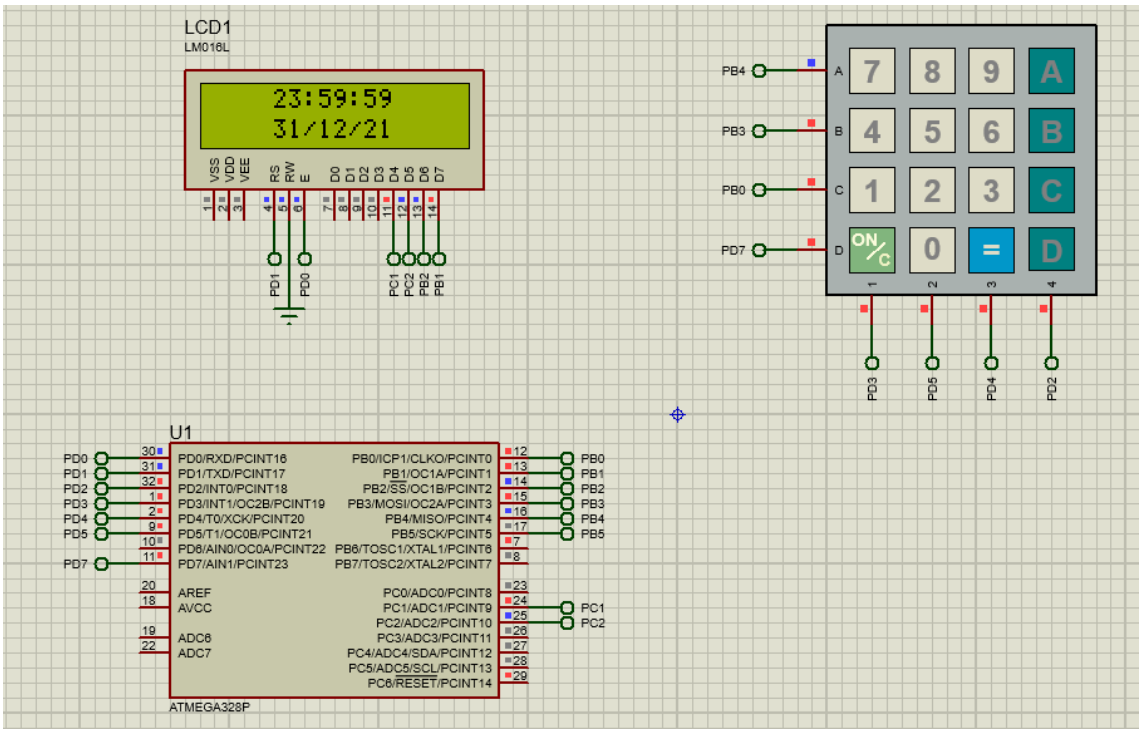
(Figura 12: Pseudocodigo de función parpadeo)

Modularizacion del programa:

Para adaptarlo a posibles cambios en el futuro el programa fue modularizado de la forma en la que se observa en la figura 13. En esta se puede observar que fue separado en 4 grandes bloques, por un lado el LCD que se encarga de todas las operaciones involucradas en la visualización de los datos como son el muestreo de la hora y la interacción con el usuario a la hora de modificarlos, por otro lado el Teclado que se encarga de captar lo que el usuario precise, la MEF encargada de realizar el funcionamiento general entre teclado y LCD como entrar al modo configuración apretando una tecla o cancelar dicho modo con otra y además incrementar y decrementar campos como hora y fecha, y por último, se encuentra el main (programa principal) en el que se planteó la interrupción cada un segundo, la inicialización de los periféricos y por ultimo un bucle en el que se llama a los demás bloques actualizando la MEF y detectando si se pulso una tecla.

(Figura 13: Forma en la que se modularizo el problema)

Validación:



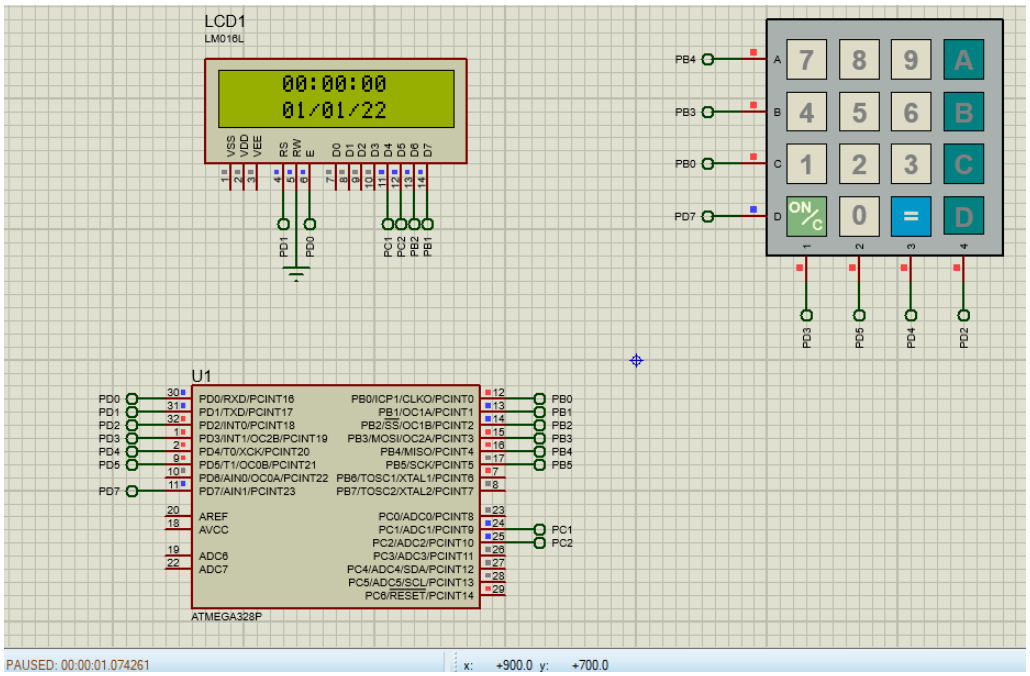
(Figura 14: imagen del circuito usado para resolver el problema)

En la figura 14 se puede observar una captura del circuito a resolver. Como se puede ver, el LCD está mostrando la hora dada como ejemplo en el enunciado del inciso b. Esta hora esta elegida a modo tal de que luego de un segundo se actualicen todos los campos.

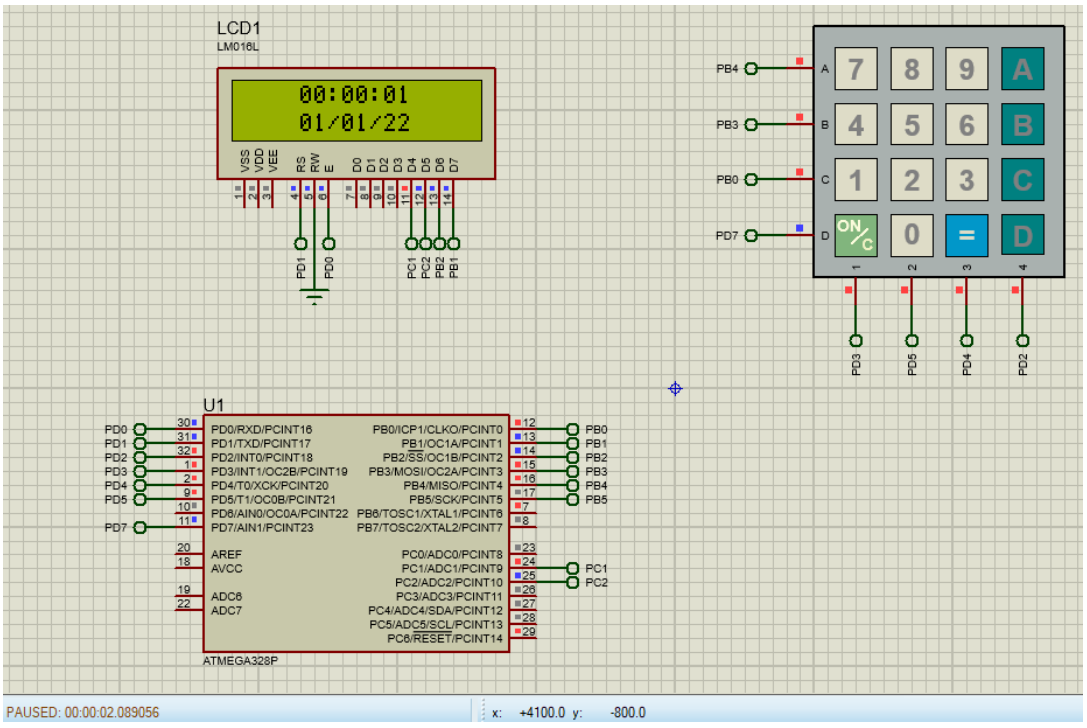
```
07E2      ISR (TIMER0_OVF_vect){
--------          static int pulsos=0;
0804              TCNT0 = 100;
0808              if(++pulsos == 100){
0820                      actualizar_datos();
0824                      MEF_inter(t);
0840                      pulsos=0;
--------          }
0848      }
```

(Figura 15: Breakpoint cada 1 segundo)

En la figura 15 se puede ver cómo fue definido un breakpoint luego de actualizar los datos para mostrar que en la figura 16 se encuentra atrasado unos milisegundos por las inicializaciones de los demás periféricos mientras que ya en la figura 17 puede verse un menor atraso debido a que solamente participa el error de conteo mostrado en la resolución de problema a. Dentro de estas figuras el tiempo de ejecución se encuentra debajo a la izquierda.



(Figura 16: circuito luego de pasar 1 segundo)



(Figura 17: circuito luego de pasar 1 segundo desde la figura 16)

Codigo:

LCD.c y LCD.h fueron proporcionados por la catedra por lo que no adjuntamos código.

Teclado.h:

```
#ifndef TECLADO_H_
#define TECLADO_H_
#include <stdint.h>
#include <avr/io.h>
```

```
#define F_CPU 16000000ul
#include <util/delay.h>
uint8_t Teclado_KEYPAD_Scan(uint8_t *); //Devuelve la posición en el puntero y
retorna si se pulso o no una tecla
void Teclado_Chequear_Car(uint8_t *); //Funcion que detecta que tecla se pulso a
partir del parametro de la anterior funcion


#endif
```

Teclado.c:

```
#include "Teclado.h"
uint8_t Teclado_KEYPAD_Scan(uint8_t *key){
	int c, f;

	for (f=0;f<4;f++){//se hace el barrido por filas
		PORTB|=(1<<PORTB0) | (1<<PORTB3) | (1<<PORTB4);
		PORTD|=(1<<PORTD7);
		PORTD |= (1<<PORTD3) | (1<<PORTD5) | (1<<PORTD4) | (1<<PORTD2);
		switch(f) //se pone en 0 la fila correspondiente
		{
			case 0:
			PORTB &= ~(1<<PORTB4);
			break;

			case 1:
			PORTB &= ~(1<<PORTB3);
			break;

			case 2:
			PORTB &= ~(1<<PORTB0);
			break;

			case 3:
			PORTD &= ~(1<<PORTD7);
			break;
		}

		_delay_ms(5);

		for(c=0;c<4;c++){//se hace el barrido por columnas dentro de cada
una de las filas
			switch(c) //chequeo si se pulso algún botón de la fila f
			{
				case 0:
				if (!(PIND & (1<<PIND3))){
					*key=(f*4+c);
					return 1;
				}
				break;

				case 1:
				if (!(PIND & (1<<PIND5))){
					*key=(f*4+c);
					return 1;
				}
				break;

				case 2:
				if (!(PIND & (1<<PIND4))){
					*key=(f*4+c);
					return 1;
```

```
                        }
                        break;

                        case 3:
                        if (!(PIND & (1<<PIND2))){
                                *key=(f*4+c);
                                return 1;
                        }
                        break;
                }
        }
    }
    return 0;
};

void Teclado_Chequear_Car(uint8_t *c){
    switch(*c) //asignación de las teclas del teclado matricial
    {
        case 0:
        *c=(int)'1';
        break;
        case 1:
        *c=(int)'2';
        break;
        case 2:
        *c=(int)'3';
        break;
        case 3:
        *c=(int)'A';
        break;
        case 4:
        *c=(int)'4';
        break;
        case 5:
        *c=(int)'5';
        break;
        case 6:
        *c=(int)'6';
        break;
        case 7:
        *c=(int)'B';
        break;
        case 8:
        *c=(int)'7';
        break;
        case 9:
        *c=(int)'8';
        break;
        case 10:
        *c=(int)'9';
        break;
        case 11:
        *c=(int)'C';
        break;
        case 12:
        *c=(int)'*';
        break;
        case 13:
        *c=(int)'0';
        break;
        case 14:
        *c=(int)'#';
```

```
                break;
                case 15:
                *c=(int)'D';
                break;
        }
};
```

MEF.h:

```
#ifndef MEF_H_
#define MEF_H_
typedef enum {Mostrar_Hora,
Modificar_Anio,Modificar_Mes,Modificar_Dia,Modificar_Hora,Modificar_Minutos,Modif
icar_Segundos} estados; //definición de los estados de la MEF
#include <stdint.h>
#include <avr/io.h>
#define F_CPU 16000000ul
typedef struct{
        unsigned char second;
        unsigned char minute;
        unsigned char hour;
        unsigned char date;
        unsigned char month;
        unsigned char year;
}time1; //Estructura de tipo fecha y hora definida como time1

void MEF_Inciar_modificar ( estados ); //Inicializacion del estado que tendra la
MEF
void MEF_Actualizar_Modificar(uint8_t ,volatile time1 *); //Actualizacion del
estado de la MEF a partir de la tecla pulsada incluyendo la posibilidad de
incrementar o decrementar un campo en un estado de modificacion
void MEF_Parpadear(void); //Funcion que cada un segundo realiza el parpadeo del
campo correspondiente en caso de estar en un estado de modificación
void MEF_inter(time1 ); //Funcion encargada de llamar al parpadeo o llamar a
mostrar los datos cada un segundo
void MEF_prender(void); //Funcion que enciende la variable de tiempo auxiliar
cuando nos encontramos cambiando de estados en configuración para que no queden
campos apagados
#endif
```

MEF.c:

```
#include "MEF.h"
#include "lcd.h"
static estados Modificar;
static uint8_t posx; // variable encargada de tener la posición del estado actual
en x
static uint8_t posy; // variable encargada de tener la posición del estado actual
en y
static time1 taux; //variable auxiliar encargada de modificar el tiempo actual
void MEF_Inciar_modificar ( estados m)
{
        Modificar = m; // poner un estado inicial
}

void MEF_Actualizar_Modificar(uint8_t tecla,volatile time1 *t)
{
        switch (Modificar)
        {
                case Mostrar_Hora:
```

```
		if(tecla == 'A') // con la tecla A se solicita pasar al siguiente
estado
		{
			taux=*t; // guardamos el tiempo actual para ir modificandolo
			Modificar = Modificar_Anio;
		}
		break;
		case Modificar_Anio:
		posx=10;
		posy=1;
		if(tecla == 'A')
		{
			Modificar = Modificar_Mes;
			MEF_prender();
		}
		else{
			if (tecla == 'B'){ // con la tecla B se incrementa el campo
				taux.year++;
				if (taux.year==100)
				{
					taux.year=00;
				}
			}
			else{
				if (tecla == 'C'){ // con la tecla C se decrementa el
campo
					taux.year--;
					if (taux.year==255)
					{
						taux.year=99;
					}
				}
				else{
					if (tecla == 'D'){ // con la tecla D se
cancelan las modificaciones
						Modificar = Mostrar_Hora;
					}
				}
			}
		}// los siguientes casos contemplan lo mismo
		break;
		case Modificar_Mes:
		posx=7;
		posy=1;
		if(tecla == 'A')
		{
			Modificar = Modificar_Dia;
			MEF_prender();
		}
		else{
			if (tecla == 'B'){
				taux.month++;
				if (taux.month==13)
				{
					taux.month=1;
				}
			}
			else{
				if (tecla == 'C'){
					taux.month--;
					if (taux.month==0)
					{
```

```
                                        taux.month=12;
                                }
                        }
                        else{
                                if (tecla == 'D'){
                                        Modificar = Mostrar_Hora;
                                }
                        }
                }
        }
        break;
        case Modificar_Dia:
        posx=4;
        posy=1;
        if(tecla == 'A')
        {
                Modificar = Modificar_Hora;
                MEF_prender();
        }
        else{
                if (tecla == 'B'){
                        taux.date++;
                        if (taux.month==1 || taux.month==3 || taux.month==5 ||
taux.month==7 || taux.month==8 || taux.month==10 || taux.month==12){
                                if (taux.date==32)
                                {
                                        taux.date=1;
                                }
                        }
                        else{
                                if (taux.month==2){
                                        if (taux.date==29)
                                        {
                                                taux.date=1;
                                        }
                                }
                                else{
                                        if (taux.date==31)
                                        {
                                                taux.date=1;
                                        }
                                }
                        }

                }
                else{
                        if (tecla == 'C'){
                                taux.date--;
                                if (taux.date==0)
                                {
                                        if (taux.month==1 || taux.month==3 ||
taux.month==5 || taux.month==7 || taux.month==8 || taux.month==10 ||
taux.month==12){
                                                taux.date=31;
                                        }
                                        else{
                                                if (taux.month==2){
                                                        taux.date=28;
                                                }
                                                else{
                                                        taux.date=30;
```

```
                                        }
                                }
                        }
                }
                else{
                        if (tecla == 'D'){
                                Modificar = Mostrar_Hora;
                        }
                }
        }
}
break;
case Modificar_Hora:
posx=4;
posy=0;
if(tecla == 'A')
{
        Modificar = Modificar_Minutos;
        MEF_prender();
}
else{
        if (tecla == 'B'){
                taux.hour++;
                if (taux.hour==24)
                {
                        taux.hour=0;
                }
        }
        else{
                if (tecla == 'C'){
                        taux.hour--;
                        if (taux.hour==255)
                        {
                                taux.hour=23;
                        }
                }
                else{
                        if (tecla == 'D'){
                                Modificar = Mostrar_Hora;
                        }
                }
        }
}
break;
case Modificar_Minutos:
posx=7;
posy=0;
if(tecla == 'A')
{
        Modificar = Modificar_Segundos;
        MEF_prender();
}
else{
        if (tecla == 'B'){
                taux.minute++;
                if (taux.minute==60)
                {
                        taux.minute=0;
                }
        }
        else{
                if (tecla == 'C'){
```

```
                                taux.minute--;
                                if (taux.minute==255)
                                {
                                        taux.minute=59;
                                }
                        }
                        else{
                                if (tecla == 'D'){
                                        Modificar = Mostrar_Hora;
                                }
                        }
                }
        }
        break;
        case Modificar_Segundos:
        posx=10;
        posy=0;
        if(tecla == 'A')
        {
                Modificar = Mostrar_Hora;
                MEF_prender();
                *t=taux; // se guarda en el tiempo actual la modificación
solicitada para comenzar a contabilizar a partir de ahi
        }
        else{
                if (tecla == 'B'){
                        taux.second++;
                        if (taux.second==60)
                        {
                                taux.second=0;
                        }

                }
                else{
                        if (tecla == 'C'){
                                taux.second--;
                                if (taux.second==255)
                                {
                                        taux.second=59;
                                }
                        }
                        else{
                                if (tecla == 'D'){
                                        Modificar = Mostrar_Hora;
                                }
                        }
                }
        }
        break;
    }
}

void MEF_Parpadear(void){
       static uint8_t encoap=0;
       LCDGotoXY(posx, posy);
       if (encoap==0){ // si encoap es 0 se apaga el campo de modificación actual
              encoap=1;
              LCDstring("  ",2);
       }
       else{  //si encoap es 1 se enciende el campo de modificación actual
              encoap=0;
              switch(Modificar){
```

```
                        case Modificar_Dia:
                        LCDescribeDato(taux.date,2);
                        break;
                        case Modificar_Mes:
                        LCDescribeDato(taux.month,2);
                        break;
                        case Modificar_Anio:
                        LCDescribeDato(taux.year,2);
                        break;
                        case Modificar_Hora:
                        LCDescribeDato(taux.hour,2);
                        break;
                        case Modificar_Minutos:
                        LCDescribeDato(taux.minute,2);
                        break;
                        case Modificar_Segundos:
                        LCDescribeDato(taux.second,2);
                        break;
                        default: break;
                }
        }
};

void MEF_inter(time1 t){
        if(Modificar!=Mostrar_Hora){ //si el estado es de modificación se llama a
que parpadee
                MEF_Parpadear();
        }
        else{ //si el estado es de mostrar_hora se imprime la hora
                LCDGotoXY(4,1);
                LCDescribeDato(t.date,2);
                LCDsendChar('/');
                LCDescribeDato(t.month,2);
                LCDsendChar('/');
                LCDescribeDato(t.year,2);
                LCDGotoXY(4,0);
                LCDescribeDato(t.hour,2);
                LCDsendChar(':');
                LCDescribeDato(t.minute,2);
                LCDsendChar(':');
                LCDescribeDato(t.second,2);
        }
}

void MEF_prender(){
        LCDGotoXY(4,1);
        LCDescribeDato(taux.date,2);
        LCDsendChar('/');
        LCDescribeDato(taux.month,2);
        LCDsendChar('/');
        LCDescribeDato(taux.year,2);
        LCDGotoXY(4,0);
        LCDescribeDato(taux.hour,2);
        LCDsendChar(':');
        LCDescribeDato(taux.minute,2);
        LCDsendChar(':');
        LCDescribeDato(taux.second,2);
}
```

Main.h:

```
#ifndef MAIN_H_
#define MAIN_H_

#define F_CPU 16000000ul
#include <util/delay.h>
#include <avr/io.h>
#include <stdint.h>
#include <avr\interrupt.h>
#include <time.h>
#include "lcd.h"
#include "MEF.h"
#include "Teclado.h"//se incluyen las librerías de los periféricos

static char not_leap(void); //función para saber si es año bisiesto
void Actualizar_datos(void); //actualiza todos los campos de la estructura time1
void Iniciar_puertos(void); //inicialización de interrupción por timer0 y de la
configuración establecida por los por los perifericos
void actualizar_lcd(void); //mostrar en el lcd la hora actual


#endif
```

Main.h:

```
#include "main.h"
uint8_t tecla; //variable donde se almacena la tecla pulsada
volatile time1 t={59,59,23,31,12,21}; //establecer tiempo de compilación por
defecto


int main(void)
{
	Iniciar_puertos();
	sei();//habilito interrupciones


	LCDinit();
	_delay_ms(5);
	LCDclr();
	_delay_ms(5);
	actualizar_lcd();
	MEF_Inciar_modificar(Mostrar_Hora); //establezco estado inicial en mostrar
hora
	while (1) {
		MEF_Actualizar_Modificar(tecla,&t); //actualizamos estado de la
maquina de acuerdo a la tecla pulsada
		tecla=-1; //limpio tecla si es que se pulso anteriormente
		if (Teclado_KEYPAD_Scan(&tecla)){
			Teclado_Chequear_Car(&tecla);
			_delay_ms(500);
		}
	}
}



void actualizar_lcd(void){
	LCDGotoXY(4,1);
	LCDescribeDato(t.date,2);
	LCDsendChar('/');
```

```
	LCDescribeDato(t.month,2);
	LCDsendChar('/');
	LCDescribeDato(t.year,2);
	LCDGotoXY(4,0);
	LCDescribeDato(t.hour,2);
	LCDsendChar(':');
	LCDescribeDato(t.minute,2);
	LCDsendChar(':');
	LCDescribeDato(t.second,2);
}


void actualizar_datos(){
	if (++t.second==60)
	{
		t.second=0;
		if (++t.minute==60)
		{
			t.minute=0;
			if (++t.hour==24)
			{
				t.hour=0;
				if (++t.date==32)
				{
					t.month++;
					t.date=1;
				}
				else if (t.date==31)
				{
					if ((t.month==4) || (t.month==6) ||
(t.month==9) || (t.month==11))
					{
						t.month++;
						t.date=1;
					}
				}
				else if (t.date==30)
				{
					if(t.month==2)
					{
						t.month++;
						t.date=1;
					}
				}
				else if (t.date==29)
				{
					if((t.month==2) && (not_leap()))
					{
						t.month++;
						t.date=1;
					}
				}
				if (t.month==13)
				{
					t.month=1;
					t.year++;
				}
			}
		}
	}
}
```

```
static char not_leap(void)      //check for leap year
{
       if (!(t.year%100))
       {
              return (char)(t.year%400);
       }
       else
       {
              return (char)(t.year%4);
       }
}

void Iniciar_puertos(void){
       DDRB|=(1<<PORTB0) | (1<<PORTB3) | (1<<PORTB4);
       DDRD|=(1<<PORTD7);
       DDRD &= ~(1<<PORTD3) | ~(1<<PORTD5) | ~(1<<PORTD4) | ~(1<<PORTD2);

       TCCR0A = 0x00; //modo normal del reloj
       TCCR0B = 0x05; //preescalador=1024
       TCNT0 = 100; //ponemos en 100 el limite inferior
       TIMSK0 |= (1<<TOIE0);
}

ISR (TIMER0_OVF_vect){
       static int pulsos=0; //variable estatica para almacenar cuantas veces se
interrumpio 10 ms
       TCNT0 = 100;
       if(++pulsos == 100){ //cada 1 segundo actualizo el tiempo actual y chequeo
si debo parpadear o mostrar la hora en el lcd
              actualizar_datos();
              MEF_inter(t);
              pulsos=0; //reseteo el contador para volver a contabilizar 1
segundo
       }
}
```