UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE CIENCIAS EXACTAS, FÍSICAS Y NATURALES

ELECTRÓNICA DIGITAL II

TRABAJO FINAL:

“TURNERO COVID”

GRUPO 6

Integrantes:

- AGUIRRE, Matias - 40576886
- ERLICHER, Ezequiel - 42051917
- VIETTI, Emilia - 43882437

Docente:

- Ing. DEL BARCO, Martín

Introducción

En este informe vamos a abordar el armado y funcionamiento de un turnero programado en assembler, utilizando el microcontrolador PIC16F887. El turnero ha sido diseñado para su implementación en un centro médico. Este sistema ofrece diversas funcionalidades, como la configuración de la fecha, la visualización del consultorio y número de paciente a llamar, la exhibición de la hora actual, la medición de temperatura de los pacientes y el envío de información mediante transmisión serie. El informe detallará los componentes utilizados, así como la lógica de programación implementada.

Desarrollo

El diagrama de estados del sistema es el siguiente, cada uno de estos estados es visualizado en 6 display 7 segmentos (cátodo común):

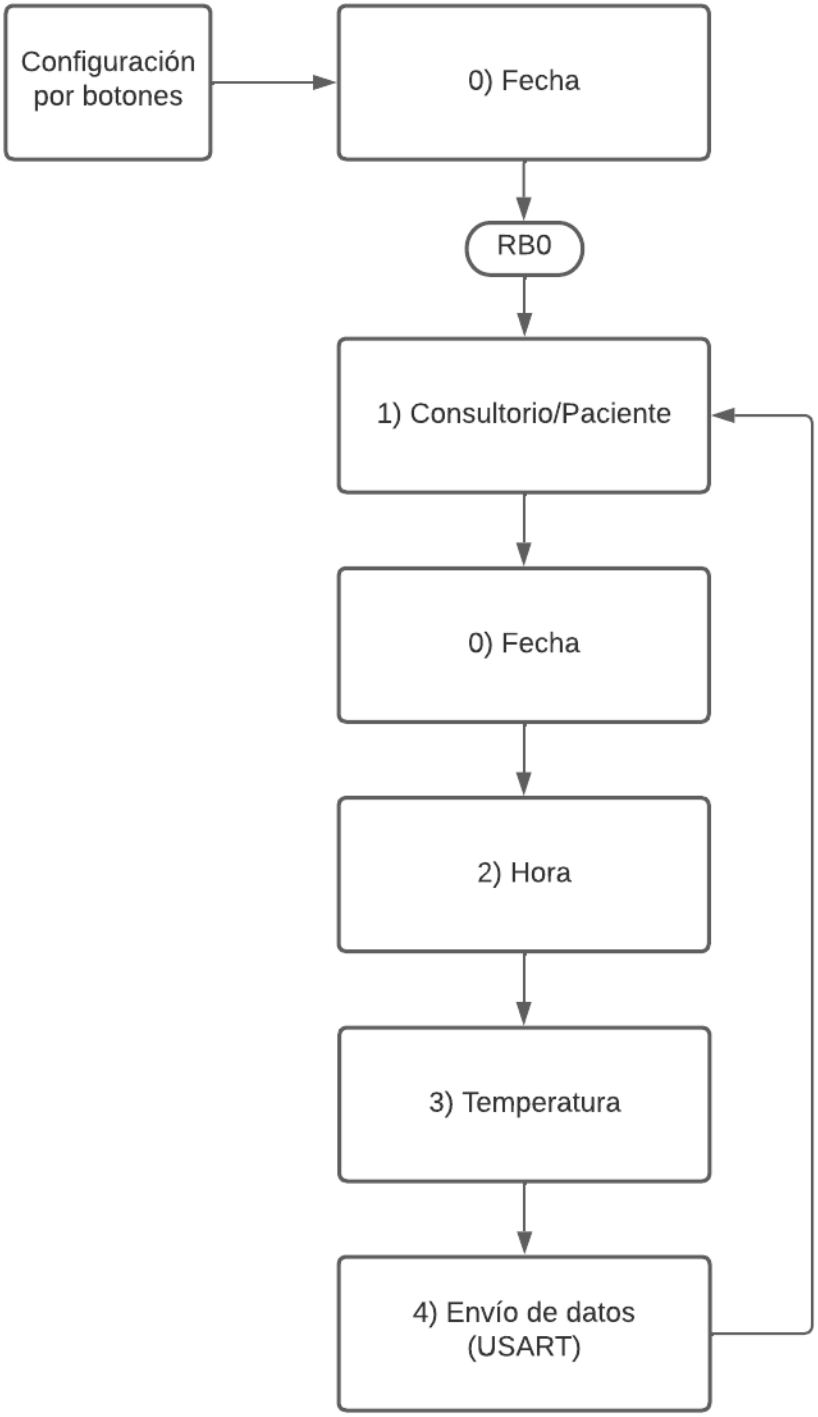

Figura 1: Diagrama de estados del Turnero

Especificación de funcionamiento del turnero

El estado que debe mostrarse en un momento particular es definido por una variable llamada STATE, la cual indica que sección de código debe ejecutarse y mostrarse en los display. El sistema cuando arranca en el estado 0, permite configurar la fecha por un miembro del personal, una vez terminada esta tarea, mediante una interrupción por RB0, comienza la secuencia 1–0–2–3–4–1… como se muestra en la imagen 1 de manera indefinida mientras se mantenga encendido el turnero. Las transiciones entre cada estado, una vez iniciada la secuencia, se manejan con la ayuda de una del Timer1. Dicho timer es configurado de forma tal que cada 5 segundos modifique la variable STATE, modificando por lo tanto la información que es mostrada en los displays.

Fecha: Se indica el día, el mes y el año, cada uno posee un botón dedicado para modificar su valor. Dichos botones son RB1 para el día, RB2 para el mes y RB3 para el año. El rango de valores para el día es de entre 1 y 31, el del mes entre 1 y 12 y el del año entre 0 y 99.

Consultorio/Paciente: Se indica el número de consultorio al que debe ingresar el paciente en los primeros 3 dígitos de los display y el número de paciente en los 3 restantes. El consultorio tiene un rango de 0-999 y es generado de manera aleatoria en rutina de interrupción del TMR1 tomando los últimos 3 dígitos del reloj (correspondiente a los segundos y el dígito más pequeño de los minutos). El número de paciente por otra parte, se incrementa en 1 en cada iteración de la secuencia mencionada.

Hora: Se muestra Horas-Minutos-segundos en formato 24hs. El reloj se modifica cada 1 segundo de manera permanente mediante TMR0 una vez que ocurre la interrupción por RB0.

Temperatura: Se mide la temperatura del paciente mediante un sensor y el ADC del PIC. En caso de que dicha temperatura sea mayor o igual a 38°C, se muestra “0000” indicando que dicho paciente posee fiebre y por lo tanto, no puede ingresar al consultorio, en caso contrario se muestra “PASE”

Transmisión de Datos USART: Se muestra una “E” indicando que se están enviando datos por el transmisor serie. Se envía el número de paciente y si posee fiebre o no.

Configuración de puertos e interrupciones

Configuración de puertos:

En la configuración de puertos, se asignaron funciones específicas a cada uno de ellos. A continuación se detalla la asignación de puertos utilizada en el sistema:

Puerto A:

- El pin A0 fue designado para conectar el sensor de temperatura. Para lograr esto, se configuró como una entrada analógica a través del registro ANSEL.

Puerto B:

- El pin RB0 se utilizó para la interrupción por RB0 .
- Los pines RB1 a RB3 se emplean para conectar los botones destinados a configurar la fecha. Estos pines se configuran como entradas digitales a través de los registros ANSELH y TRISB.

Puertos C y D:

- Los puertos C y D se utilizan para conectar los displays, en donde se muestran los distintos estados del turnero.
- Para lograr esto, se configuran los puertos C y D como salidas digitales mediante los registros TRISC y TRISD.

Configuración de interrupciones:

Para habilitar correctamente las interrupciones que se utilizan en el turnero, es fundamental en primera instancia configurar los siguientes bits del registro INTCON:

```
MOVLW   B'11011000'    ;(GIE=1,PEIE=1,T0IE=0,INTE=1,RBIE=1,T0IF=0,INTF=0,RBIF=0)
MOVWF   INTCON
```

Figura 2: Configuración registro INTCON.

GIE (Global Interrupt Enable): Este bit debe establecerse en uno para permitir cualquier interrupción en el sistema. Si se encuentra deshabilitado (en cero), no se podrán procesar las interrupciones.

PEIE(Perpherical interrupt enable): Debe setearse en 1 para habilitar las interrupciones por TMR1,ADC y transmisor USART.

TOIE: Este bit habilita las interrupciones por TMR0, en un principio este está en 0 (se pone en 1 en la subrutina de TMR0 como se explica más adelante)

INTE: Permite interrupción por INT/RB0

RBIE: Habilita interrupciones por puerto B

T0IF,INTF,RBIF: Son Flags que se ponen en 1 cuando ocurre una interrupción por TMR0,INT/RB0 y Puerto B respectivamente.

➢ Configuración del temporizador TMR0:

En el caso específico del temporizador TMR0, se requiere realizar las siguientes configuraciones:

Registro OPTION_REG:

- Se establece el bit TOCS (TMR0 Clock Source) en cero para seleccionar el clock interno como fuente y poder utilizar el Timer 0 como temporizador.
- Se limpia el bit PSA (Prescaler Assignment). De esta manera, se habilita el uso del prescaler para el TMR0 .

Configuración del prescaler del TMR0:

- Se configuran los bits PS2, PS1 y PS0 en uno para usar un prescaler de 256. Esta configuración permite alcanzar el tiempo máximo para el temporizador TMR0.

```
MOVLW   B'00000111'
MOVWF   OPTION_REG
```

Figura 3: Configuración registro OPTION_REG

El bit TOIE del registro INTCON (necesario para habilitar las interrupciones por TMR0) es seteado en 1 en la subrutina de interrupción de RB0

➢ Configuración del TMR1 (Temporizador 1):

Para habilitar las interrupciones por TMR1, primero se setea en 1 el bit TMR1IE (TMR1 Interrupt Enable) en el registro PIE1 y luego se pone en 0 el bit TMR1IF en el registro PIR1 (TMR1 Interrupt Flag). Cabe aclarar que previamente

PEIE(INTCON) fue configurado en 1. Luego, se utiliza el registro T1CON de la siguiente manera:

Registro T1CON: Se carga el valor B '00110000' en este registro para generar la siguiente configuración de bits (Se describe partiendo del menos significativo):

- TMR1ON = 0: El TMR1 se encuentra apagado (se activa en la interrupción por RB0) .
- TMR1CS = 0: Fuente de reloj interna seleccionada para el TMR1.
- T1SYNC = 0: dado que no se utiliza una fuente de clock externa, da igual el valor de este bit
- T1OSCEN = 0: Oscilador LP del TMR1 deshabilitado.
- T1CKPS1 y T1CKPS0 = 1: Preescaler establecido en 8.
- TMR1GE=0 El TMR1 cuenta siempre (cuando TMR10N=1)
- T1GINV=0

➢ Configuración del ADC (Convertidor Analógico-Digital):

Para habilitar las interrupciones por ADC, se setea en 1 el bit ADIE del registro PIE1 y se limpia el flag ADIF del registro PIR1 (cabe recordar que previamente se puso en 1 el bit PEIE del registro INTCON). Luego, se procedió a realizar las siguientes configuraciones:

Registro ADCON0: El registro ADCON0 se carga con el valor B '01000000'. Lo cual genera la siguiente configuración (se describe partiendo del bit menos significativo):

- ADON=0 (bit 0): en un comienzo el ADC se encuentra apagado (se habilita en la interrupción por TMR1)
- GO/DONE=0 (bit 1): no se inicia una conversión
- CHS0,CHS1,CHS2,CHS3=0 (bits 2,3,4 y 5): se selecciona el pin AN0 como puerto de entrada analógico
- ADCS0=1,ADCS1=0 (bits 6 y 7): se configura la frecuencia del ADC en Fosc/8, es decir, en 500Khz (F0sc=4MHz)

- En esta configuración, el bit ADFM (A/D Result Format Select) se ha establecido en 0, lo que indica que los resultados de conversión se almacenan en los bits de menor peso del registro ADRESH y los bits de mayor peso en el registro ADRESL.
- Se ha seleccionado el canal de entrada analógica utilizando los bits CHS (Analog Channel Select). El valor específico utilizado en los bits CHS dependerá del canal de entrada que se haya designado para conectar el sensor de temperatura en el pin A0 del puerto A.

Registro ADCON1: El registro ADCON1 se configura con el valor 10000000,seteando los bits de este registro de la siguiente manera (nuevamente, se comienza describiendo desde el bit menos significativo):

- VCFG0=0,VCFG0=1 (bits 4 y 5 respectivamente,Voltage Reference Configuration): configuran los valores VDD(+) y VSS(-) como los voltajes de referencia para el funcionamiento del ADC,
- ADFM=1(bit 7): indica que los 10 bits que arroja el ADC en cada conversión son almacenados en los registros ADRESL y en los 2 bits menos significativos de ADRESH (justificación a la derecha).
- El resto de los bits no tienen uso alguno por lo tanto da igual el valor que se cargue en estos.

➢ <u>Configuración de la interrupción por puerto B:</u>

En el contexto de la interrupción por puerto B, se realiza las siguientes configuraciones en el microcontrolador:

Registro INTCON:

- Se selecciona el bit correspondiente a la habilitación de la interrupción por Puerto B (RBIE) y establecerlo en estado activo (1) tal como se mostraba en la página 5.
- Se limpia la bandera de interrupción (RBIF) asociada a la interrupción por puerto B se encuentra en el mismo registro INTCON.

Registros OPTION_REG y WPUB:

- Se deshabilita el bit RBPU (OPTION_REG) para permitir el uso de las resistencias de pull-up en el puerto B.

- Se activan las resistencias de pull-up individuales para los pines que se utilizan como entradas en el puerto B (RB1-RB3). Esta activación se realiza mediante el registro WPUB.

```
MOVLW   B'00000111'          MOVLW   B'00001111'
MOVWF   OPTION_REG           MOVWF   WPUB
```

Figura 4: configuración del bit RBPU y del registro WPUB

Registro IOCB:

- Mediante este registro se habilitan de manera individual las interrupciones por cambio de nivel en los pines del puerto B que fueron previamente configurados como entradas en la página 4. Es decir, se setean en 1 los bits 1, 2 y 3 del registro IOCB para habilitar la interrupción en los pines RB1,RB2 Y RB3 (el resto se pone a 0).

➢ Configuración de la interrupción por RB0:

Para habilitar la interrupción por RB0, se lleva a cabo las siguientes configuraciones utilizando el registro INTCON como mostraba en las página 4 y 5:

Registro INTCON:

- Se establece el bit INTE (External Interrupt Enable) en uno para habilitar la interrupción por RB0.
- Se limpia el flag de interrupción la INTF (External Interrupt Flag), también ubicada en el registro INTCON.

➢ Configuración transmisor USART:

A continuación se detalla cómo se configuran los registros asociados al transmisor USART

Registro SPBRG: Este registro permite establecer a qué velocidad (medida en baud, símbolos por segundo) se transmite información. Se desea transmitir a 9600 bauds con un clock interno de 4Mhz, por lo tanto se carga el valor 25 decimal (este valor es el que produce el menor error entre la tasa de bauds calculada y la deseada. El mismo se extrae de las tablas 12-5 del datasheet del PIC16F887).

TABLE 12-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 0											
	Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	—	—	—	—	—	—	300	0.16	207
1200	1202	0.16	207	1200	0.00	191	1202	0.16	103	1202	0.16	51
2400	2404	0.16	103	2400	0.00	95	2404	0.16	51	2404	0.16	25
9600	9615	0.16	25	9600	0.00	23	9615	0.16	12	—	—	—
10417	10417	0.00	23	10473	0.53	21	10417	0.00	11	10417	0.00	5
19.2k	19.23k	0.16	12	19.2k	0.00	11	—	—	—	—	—	—
57.6k	—	—	—	57.60k	0.00	3	—	—	—	—	—	—
115.2k	—	—	—	115.2k	0.00	1	—	—	—	—	—	—

Figura 5: Tabla de valores para Baud rates para modos síncronos

Registro TXSTA: se carga el valor B'00100100' produciendo la siguiente configuración de bits (se comienza describiendo a partir del bit menos significativo):

- TX9D=0 (bit 0): En este caso, no se envía un bit adicional (tra
- TRMT(bit 1): Este bit es solo de lectura, asi que da igual el valor en que se intente setear.
- BRGH=1(bit 2): Se selecciona el modo de High baud rate
- SENDB(bit 3)=0, no se envía un carácter de break.
- SYNC=0(bit 4), se transmite en modo asincrónico
- TXEN=1(bit 5), el transmisor queda habilitado para transmitir
- TX9=0, Se selecciona transmisión de 8 bits
- CSRC=0, como la transmisión es asíncrona, no importa el valor en que se setea este bit.

REGISTRO RCSTA: se pone en 1 el bit SPEN, haciendo que el pin TX/CK se configure automáticamente como salida.

Interrupciones (Rutinas):

En la sección de código titulada “RUTINA DE SERVICIO A LAS INTERRUPCIONES” se identifica la fuente de interrupción, llamando a la correspondiente subrutina. El orden de prioridad de las interrupciones es el siguiente: 1)TMR1,2)TMR0,3)PORTB,4)INT/RB0,5)ADC,6)Transmisor USART.

```
;------------------RUTINA DE SERVICIO A LAS INTERRUPCIONES-------------------
     ORG 0x0220
INTERRUPT

; Guardado del contexto
    MOVWF   W_TEMP
    SWAPF   STATUS,W
    MOVWF   STATUS_TEMP

    BANKSEL PIR1
    BTFSC   PIR1,TMR1IF
    GOTO    TMR1_ISR
    BANKSEL INTCON
    BTFSC   INTCON,T0IF
    GOTO    TMR0_ISR
    BTFSC   INTCON,RBIF
    GOTO    PortB_ISR
    BTFSC   INTCON,INTF
    GOTO    INT_ISR
    BTFSC   PIR1,ADIF
    GOTO    ADC_ISR
    BTFSC   PIR1,TXIF
    GOTO    TX_ISR

END_INTERRUPT
;Recuperación del contexto y retorno a MAIN
    SWAPF STATUS_TEMP,W
    MOVWF STATUS
    SWAPF W_TEMP,F
    SWAPF W_TEMP,W
    RETFIE
```

Figura 6: rutina de servicio a las interrupciones

➢ **Subrutina de interrupción TMR1:**

Es la subrutina más importante del programa, ya que es la encargada de generar un retardo de tiempo de 5 segundos aproximadamente y modificar el estado del turnero (cada vez que transcurre dicho período de tiempo) haciendo uso de la variable STATE. El tiempo máximo que se puede generar con el Timer1 con un preescaler de 8 y un oscilador interno de 4Mhz es de 0,52 segundos, por lo tanto, se

utiliza un contador (COUNTER_TMR1) para recargar el TMR1 10 veces para generar el retardo de 5 segundos entre cada estado. Una vez que COUNTER_TMR1 llega a 0(es decir, una vez transcurridos los 5 segundos) la subrutina ejecuta la siguiente secuencia :

- Si STATE es igual a 0 (Fecha) se pone a 2 (Clock)
- Si STATE es igual a 1 (consultorio/paciente) se pone a 0 (fecha)
- Si STATE es igual a 2 (Reloj) se pone a 3 y se habilita el ADC para que comienze a muestrear temperatura a través del sensor.
- Si STATE es igual a 3(ADC-Sensor) se pone a 4 , se deshabilita el ADC y se pone a funcionar el transmisor USART (habilitando su interrupción y mandando el primer carácter “-”).
- Si STATE es igual a 4 se pone en 1 (consultorio/paciente), se incrementa el numero de paciente y se generan números aleatorios utilizando los últimos 3 dígitos del reloj.
- La rutina finaliza limpiando la bandera de interrupción TMR1IF y se realiza un salto a la etiqueta END_INTERRUPT.

➢ **Subrutina de interrupción TMR0:**

El TMR0 se utiliza para mostrar un reloj digital en el que se muestra hora, minutos y segundos (formato de 24Hs). Este reloj comienza a funcionar de manera constante una vez que se ejecuta la subrutina de interrupción asociada a RB0. Para que el reloj se actualize cada 1 segundo (teniendo un oscilador interno de 4MHz y un prescaler seteado en 256), se carga el TMR0 con el valor decimal 61 para que interrumpa cada 50ms ,dicha carga se realiza 20 veces (20*50ms=1s) con la ayuda de un contador (COUNTER_TMR0). Lo que se realiza una vez transcurrido 1 segundo (es decir, COUNTER_TMR0=0) es lo siguiente:

- Se incrementa”Second2”, si dicho valor se pasa de 9, se limpia y se incrementa “Second1”
- Si “Second1” se pasa de 5, se limpia y se incrementa “Minute2”
- Si “Minute2” se pasa de 9, se limpia y se incrementa “Minute1”
- Si “Minute1” se pasa de 5, se pone a 0 y luego se incrementan “Hour2” y “AmountOfHours”

- Si AmountOfHours alcanza el valor de 24, se resetean las horas (se limpian “Hour1” y “Hour2”) , caso contrario, se verifica si “Hour2” supera el valor de 9, en cuyo caso se incrementa “Hour1”, si “Hour1” se pasa de 2, se limpia.
- Se borra la bandera de interrupción T0IF y se realiza un salto a la etiqueta END_INTERRUPT.

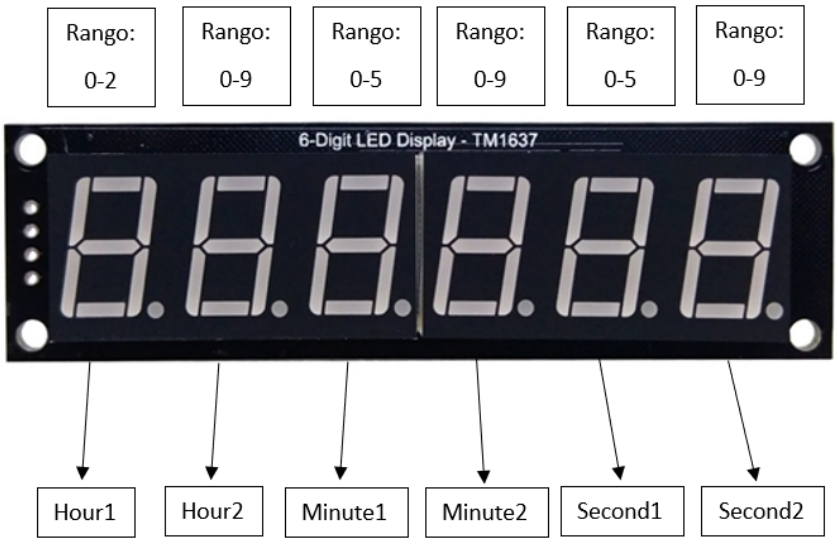

Figura 7: Representación de las variables del clock y el rango de valores de cada dígito (Se utiliza un display de 6 segmentos con fines ilustrativos, el modelo de display que en verdad se utiliza se especifíca en el apartado de hardware).

➢ **Subrutina de interrupción PORTB:**

En esta interrupción se modifican las variables que muestran la fecha, los pasos realizados son los siguientes :

- Se verifica el estado de los pines del puerto B para identificar en qué puerto ocurrió un flanco descendente.
- **RB1_ISR:** Se incrementa “Day2” y el contador “Days”, si “Days” es mayor a 31, se resetean todas las variables para el manejo de los días (“Day2”=1,”Day1”=0,”Days”=1). En caso de que “Days” sea menor o igual a 31, se incrementa “Day1”.
- **RB2_ISR:** Se incrementa “Month2” y el contador “Months”, si “Months” es mayor a 12, se resetean todas las variables para el manejo del mes

(“Month2”=1,”Month1”=0,”Days”=1). En caso de que “Months” sea menor o igual a 12, se incrementa “Month1”.

- **RB3_ISR**: Se incrementa “Year2”, si es mayor a 9, se pone a 0 y se incrementa “Year1”. Si “Year1” es mayor a 9, se pone ”Year1”=0.
- Se borra la bandera de interrupción RBIF y se realiza un salto a la etiqueta END_INTERRUPT.

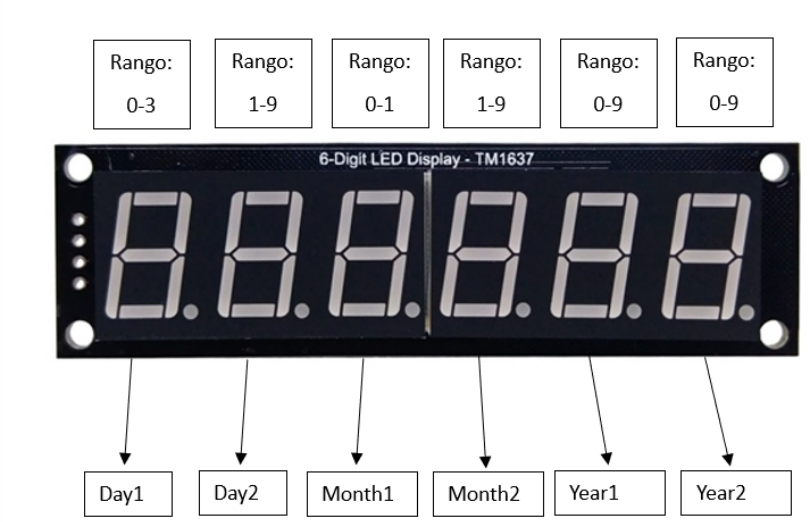

Figura 8: Variable y rango correspondiente a cada dígito de la fecha

➢ **Subrutina de interrupción INT/RB0:**

En esta subrutina se habilita, por un lado, el TMR1 para que comience a modificar el estado del turnero cada 5 segundos en forma secuencial tal como se ve en la figura X. Por último, se habilita el TMR0 para que entre en funcionamiento el reloj. La secuencia de instrucciones en concreto es la siguiente:

- Se habilitan los temporizadores TMR0 y TMR1.
- Se establecen los bits correspondientes en los registros PIE1, T1CON e INTCON para habilitar las interrupciones del TMR1 y del TMR0.
- Se borran las banderas de interrupción INTF y RBIF.
- Se realiza un salto a la etiqueta END_INTERRUPT.

➢ **Subrutina de interrupción ADC:**

Esta subrutina es utilizada para leer el valor que fue obtenido por el sensor de temperatura, el cual está seteado para indicar “PASE” si la temperatura es menor a 38° y “000000” si la temperatura obtenida es mayor o igual a 38°, lo que indica que el paciente tiene fiebre y no puede pasar. El valor que el ADC inserta en el registro ADRESL (en decimal) queda determinado por la siguiente fórmula:

$$ADRESL(decimal) \approx Vout * \frac{1024}{5V}$$

Donde Vout es el voltaje de salida del sensor LM35 para una determinada temperatura. Dado que la función de transferencia del sensor es $0,01\frac{V}{C°}$, implica que si la temperatura que se lee es de, por ejemplo, 34°, Vout es igual a 0,34. Luego, el coeficiente 1024/5 es el inverso del menor voltaje que el ADC es capaz de convertir. Como el ADC tiene una resolución de 10 bits (2^10=1024) y la fuente de alimentación es 5V, dicho voltaje es aproximadamente igual 0,005V.

Por lo tanto, si se quiere obtener el valor real de temperatura (en grados) que lee el sensor hay que (aproximadamente) dividir el valor de ADRESL en 2, cabe aclarar que no es un cálculo exacto como resultado de que el coeficiente 5/1024 es redondeado, esto se debe a las limitaciones del lenguaje assembly para operar con números decimales. A continuación se muestra un ejemplo:

$$Para\ Tsensor = 36°: \quad ADRESL(decimal) = 0,36 * \frac{1024}{5V} \approx 74\ d$$

$$Por\ lo\ tanto: ADRESL(decimal)/2 = 36°$$

Por lo tanto, esta subrutina ejecuta las siguientes acciones:

- Se lee el valor del registro ADRESL y se divide dicho valor por 2 debido a los motivos explicitados en el anterior párrafo.
- Luego, se determina si el valor de temperatura leído es mayor/igual o menor a 38°. Dependiendo del resultado obtenido, se realizan ajustes en los dígitos de visualización (indicando “PASE” o “0000”).

- Se borra la bandera de interrupción ADIF, se realiza una pausa de muestreo y se inicia una nueva conversión del ADC.
- Se realiza un salto a la etiqueta END_INTERRUPT.

➢ **Subrutina TX USART**

Los datos que se envían y su formato son los siguientes:

"-PACIENTE: {numero de paciente},FIEBRE:{SI/NO}"

Para indicar a la subrutina el orden en que se deben enviar los caracteres, se usa por un lado la variable "USART_COUNTER" la cual permite recorrer en cada interrupción las tablas que contienen cada uno de los strings que se deben enviar. Luego mediante la variable "String_Flag", se divide la información a enviar en 2 "Substrings" de la siguiente manera:

- String_Flag=0: En cada interrupción se envía un carácter correspondiente a "*PACIENTE: {número de paciente}"* (Ver tabla: "String1"). Si USART_COUNTER vale 10,11 o 12 se envían los dígitos correspondientes al número de paciente, siendo 10 el de mayor, para mayor claridad, se usa de ejemplo el número de paciente "123":

Dígito Paciente	1	2	3
USART_COUNTER	10	11	12

Figura 9: Dígito del número de paciente que se muestra según el valor de USART_COUNTER

- String_Flag=1: En cada interrupción se envía un carácter correspondiente a "*FIEBRE:"* (Ver tabla: "String2") haciendo uso nuevamente de la variable USART_COUNTER. Finalmente, mediante la variable "YesOrNo" se envían los caracteres de "SI" en caso de que el paciente tenga fiebre o de "NO" en caso contrario (Ver tablas: "Fever" y "Not_fever").

- String_Flag=2: Se introduce un salto de línea y se inhabilitan las interrupciones del transmisor (se apaga el flag TXIE).

Hardware

El proyecto fue en un principio simulado en Proteus y luego fue construído físicamente. **Debido a un inconveniente con la simulación de los transistores que se conectan a la entrada de los pines de selección del display, se excluyeron de la simulación, sin embargo, estos fueron agregados en el circuito físico y su diagrama de conexión se muestra en la figura 11.Esto también llevó a que se realizarán 2 versiones del software que controla el turnero, uno que funciona en la simulación y otro para el funcionamiento del circuito real. La única diferencia entre uno y otro es que en la versión para la simulación los bits de selección de los display se activan por bajo, mientras que en el del circuito real se activan por alto.**

El listado de componentes utilizados son los siguientes :

- PIC16F887
- 6 displays 7 segmentos cátodo común
- Un sensor de temperatura LM35
- 4 Botones, 3 para seleccionar la fecha y 1 para interrupción por RB0
- Transistores
- Cables
- Resistencias: 680Ω (base de los transistores), 220Ω (segmentos de los display) 10KΩ(resistencias de pull up externas),
- Capacitores

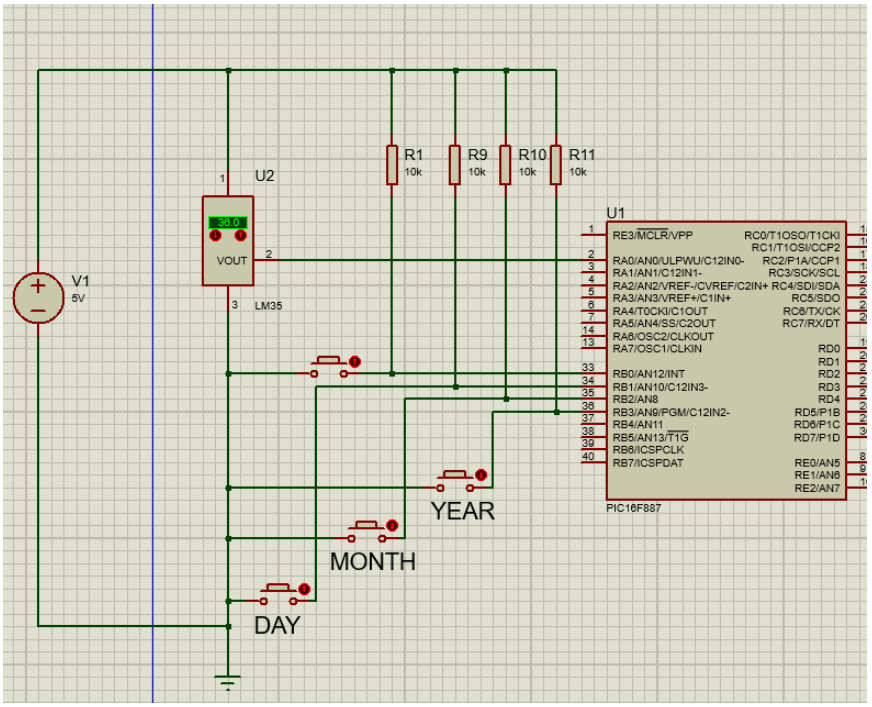

Figura 10 : Diagrama del circuito en Proteus parte 1: Fuente,sensor LM35, botones y resistencias de Pull-Up en RB0-RB3

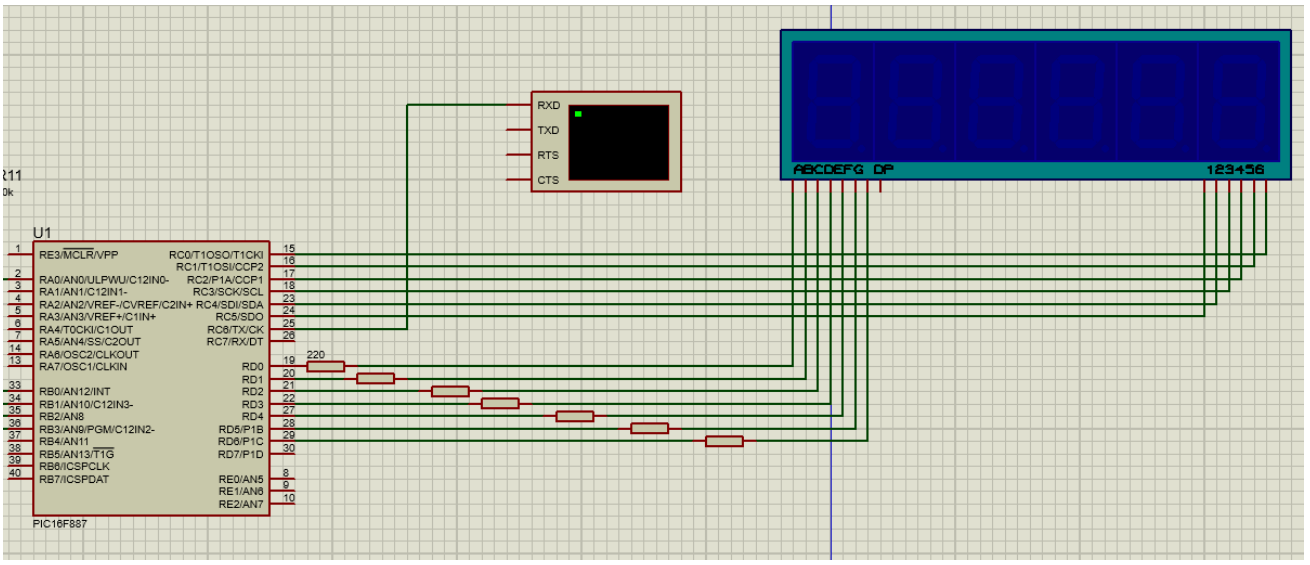

Figura 11 : Diagrama del circuito en Proteus parte 2: Conexiones del display y terminal virtual para simular la transmisión de datos por el puerto USART

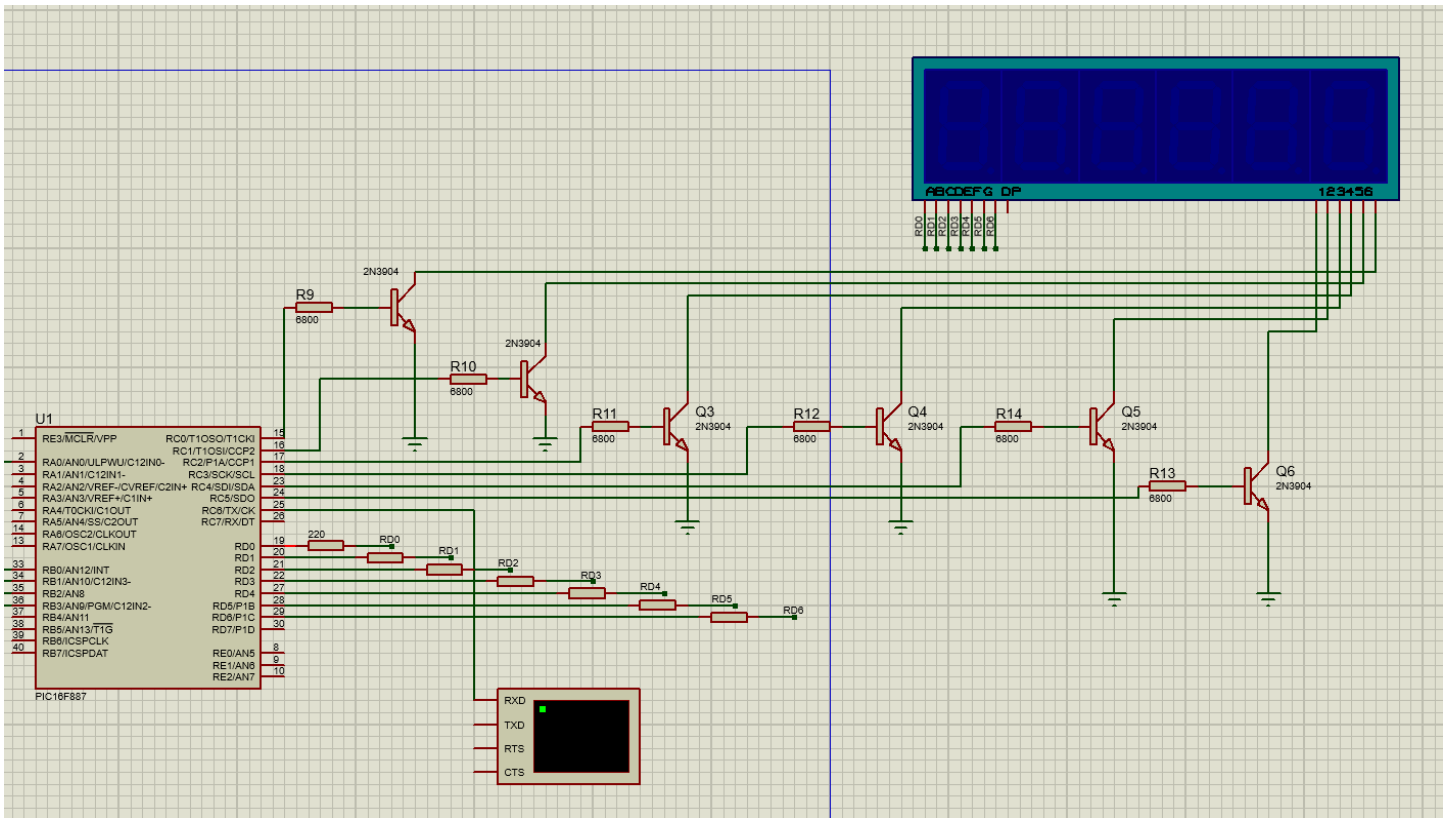

Figura 12: Diagrama de conexiones del display en Proteus (con Transistores)

Calculo de resistencias para los segmentos del display:

$$R_{LED} = \frac{V - Vled}{I}$$

$$V_{LED} = 2V$$

$$I_{LED} = 20mA \text{ (corriente deseada)}$$

$$\Rightarrow R = \frac{5V - 2V}{20mA}$$

$$R = \frac{5V - 2V}{0.02A}$$

$$R = \frac{3V}{0.02A}$$

$$R = 150\Omega$$

*En el circuito se utilizaron resistencias de 220Ω

Cálculo de la resistencia de base de los transistores (2N3904):

$$R_b = \frac{V - Vbe}{\frac{Ic}{hFE}}$$

se elige un hFE = 30 (ganancia de corriente)

Se calcula una resistencia para una I_C = 140mA (7 segmentos encendidos al mismo tiempo)

$$R_b = \frac{5V - 0.7V}{\frac{140mA}{30}}$$

$$R_b = \frac{4.3V}{\frac{0.14A}{30}}$$

$$R_b = \frac{4.3V}{0.0047}$$

$$R_b = 914\Omega$$

Se calcula una resistencia para una corriente de saturación de 200mA:

$$Rb_{Sat} = \frac{5V - 0.7V}{\frac{200mA}{30}}$$

$$Rb_{Sat} = \frac{4.3V}{\frac{0.2A}{30}}$$

$$Rb_{Sat} = \frac{4.3V}{0.0066}$$

$$Rb_{Sat} = \frac{4.3V}{0.007}$$

$$Rb_{Sat} = 614\Omega$$

⇒ Rb toma valores de entre 614Ω para una saturación de 200mA y 914Ω para una IC= 140mA

⇒ Se eligen resistencias comerciales de 680Ω