Fundamentos de Microprocesadores y Microcontroladores

Mtro. Mario Alberto Peredo Durán

Práctica 4: Teclado matricial, pantalla (LCD), dispositivo de comunicación serial.

727272 - Cordero Hernández Marco Ricardo

727576 - Guzmán Claustro Edgar

727366 - Rodríguez Castro Carlos Eduardo

Jal., 14 de julio de 2022

Descripción y desarrollo

Para la realización de esta práctica es necesario contar con los siguientes materiales: 1 microcontrolador AT89S52, 1 cristal de 11.0592MHz, 2 capacitores de 33 pf, 1 capacitor de un uf, capacitor de 100 nf, 1 circuito HC-05, 1 pantalla LCD 16x2, 1 encoder 74C922, 6 resistencias de 10k ohm, 2 resistencias de 100 ohms, 1 resistencia de 200 ohms, 3 push buttons, 1 transistor 222A, 1 resistencia variable y 1 teclado matricial.

Como parte previa, se debe tomar a consideración obtener las hojas de especificaciones de la pantalla LCD y el encoder. Estos documentos son fáciles de obtener, basta con realizar una simple búsqueda en internet para conseguirlo (revisar referencias). Esto para evitar tener problemas al momento de alambrar y no realizar el trabajo dos veces. Posteriormente, es recomendable realizar un diagrama esquemático antes del trabajo de alambrado, de esta manera se sabe con exactitud el tipo de conexiones son necesarias para cada componente.

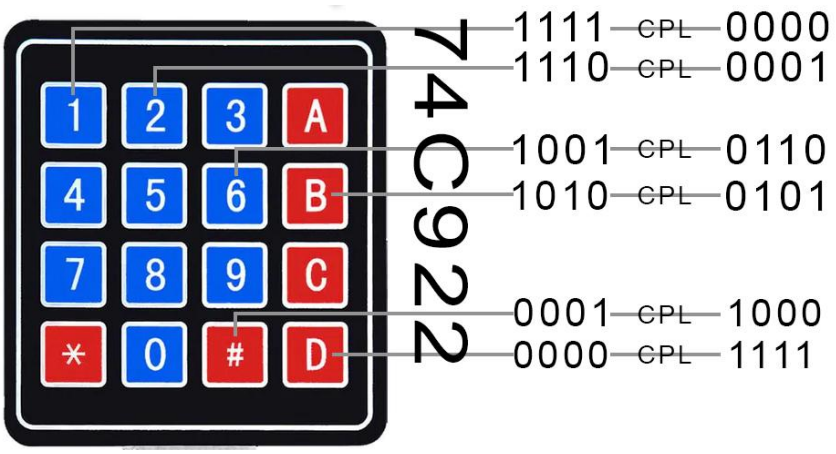
En cuanto a la planeación de la solución, se comienza inicializando la pantalla LCD, haciendo que esta muestre los dos renglones y se posicione en el primer lugar del renglón superior, haciendo parpadear el puntero. De esta manera siempre que se haga reset al sistema la pantalla será la primera en configurarse.

Prosiguiendo, iniciamos planteando las dos interrupciones que son obligatorias, la transmisión serial de los datos y la presión de teclas en el teclado matricial. Para esto indicamos cada una de ellas mediante el puerto 3.2 y 3.3. De esta manera cubrimos los dos requerimientos más importantes de la práctica. Finalizando con el gran problema al quedarnos sin interrupciones externas para el modo ALT. La solución más factible fue utilizar un temporizador que estuviera revisando si el botón fue presionado.

En la parte del teclado matricial, se realizó una decodificación por medio de software para que las teclas coincidieran con el valor mostrado en la pantalla, además de crear un retraso artesanal para que no escribiera varias veces el mismo caracter.

Especificaciones del teclado matricial

Contrario a lo que se esperaba, no existió la necesidad de realizar ningún tipo de recubrimiento externo para "enmascarar" los valores del teclado matricial, ya que la decodificación fue realizada directamente en software por medio de tablas de valores en código. Parte del proceso de resolución del problema fue tan sencilla como analizar los datos arrojados por el codificador y aplicar la lógica de las secuencias resultantes. Lo obtenido fue conciso: el orden era inverso. La solución también lo fue: hacer un complemento (negación lógica) al dato recibido. El siguiente diagrama trata de ilustrar lo expresado.



De forma que se tiene la siguiente tabla de valores asociados:

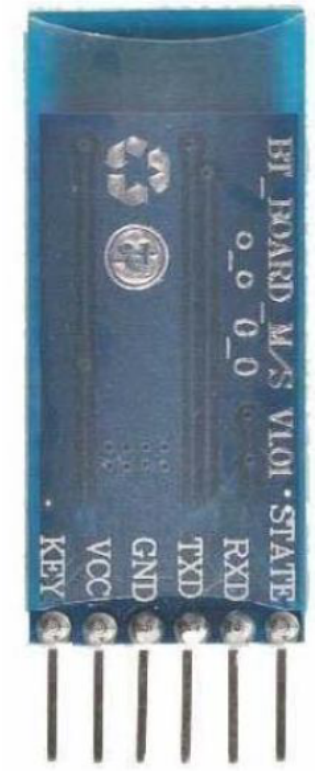
Tecla	Code	Valor	Tecla	Code	Valor	Tecla	Code	Valor	Tecla	Code	Valor
1	1111	'1'	2	1110	'2'	3	1101	'3'	A	1100	'A'
4	1011	'4'	5	1010	'5'	6	1001	'6'	B	1000	'B'
7	0111	'7'	8	0110	'8'	9	0101	'9'	C	0100	'C'
*	0011	'E'	0	0010	'0'	#	0001	'F'	D	0000	'D'

Una vez que se tiene el valor codificado recibido por medio del puerto 1 (P1.0 - P1.3), se obtiene el dato deseado mediante la instrucción CPL, con lo cual finalmente se podrá buscar el valor perteneciente a la tecla presionada, tanto para ASCII como para el valor directo de cada botón.

```
; Tabla de valores preedefinidos

            DB  '1', '2', '3', 'A'
            DB  '4', '5', '6', 'B'
            DB  '7', '8', '9', 'C'
            DB  'E', '0', 'F', 'D'
```

Especificaciones del módulo bluetooth (HC-05)



El módulo presentado sirve como el periférico crítico para demostrar el funcionamiento y entendimiento del puerto serial encontrado en el 89S52. El primer par pines para el puerto 3 son conectados a sus respectivas contrapartes de la siguiente forma:

- 89S52 : P3.0 = RXD ⇔ TXD : HC-05
- 89S52 : P3.1 = TXD ⇔ RXD : HC-05

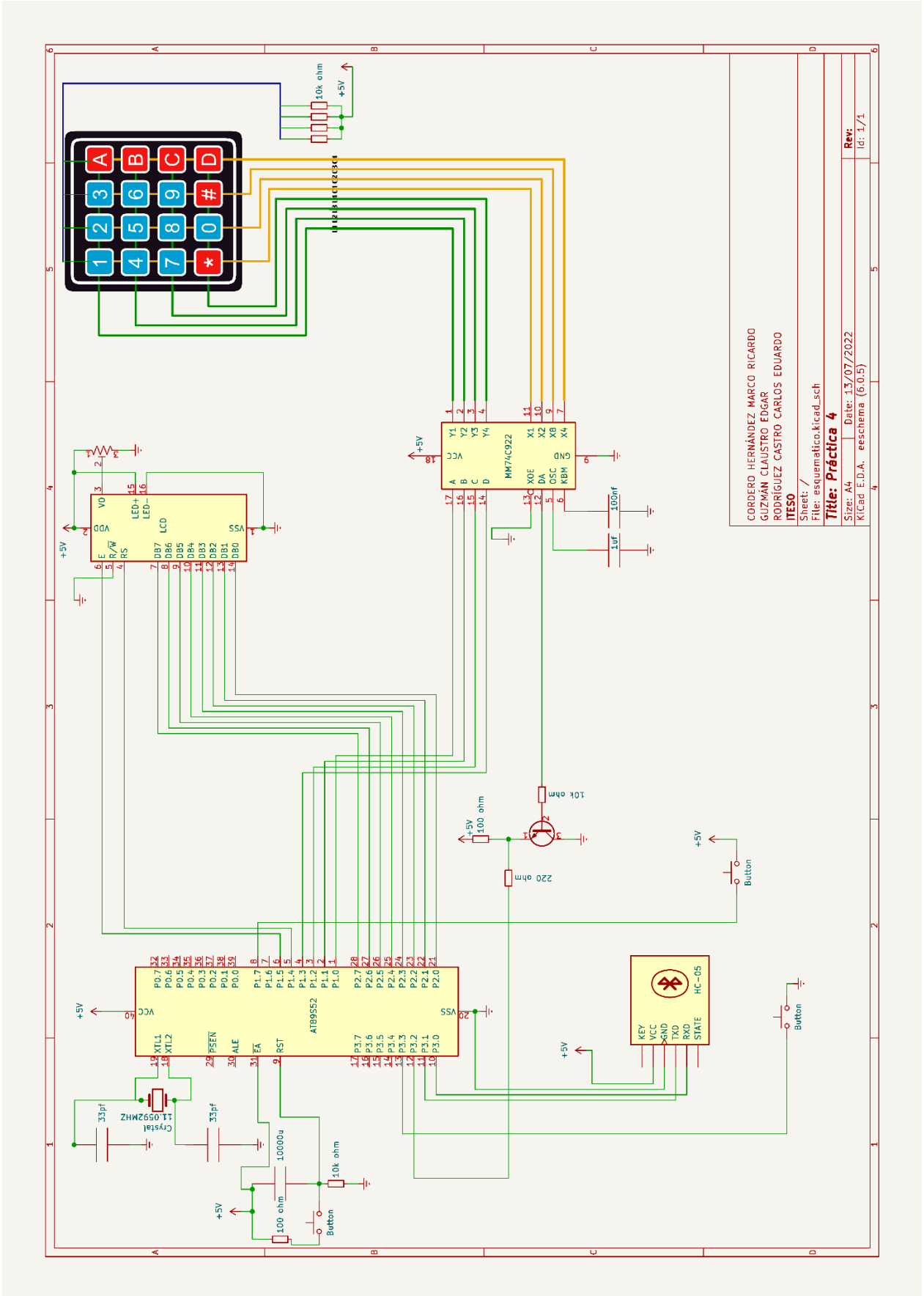
A pesar de que se conecta la recepción serial (P3.0), esta no se utiliza, ya que para la práctica únicamente se transmiten datos. Respecto a lo mencionado, el baud rate utilizado es un estándar de 9600Bd calculado a través de una herramienta en línea (Arm Keil, s.f.), la cual otorga el valor exacto que se le debe de asignar al Timer 1, ya que el Timer 0 no es compatible con lo que se desea obtener respecto al puerto serial.

El módulo bluetooth con el que se trabajó cuenta con 6 pines, en donde dos de ellos (KEY; STATE) no se utilizan por cuestiones del modelo utilizado.

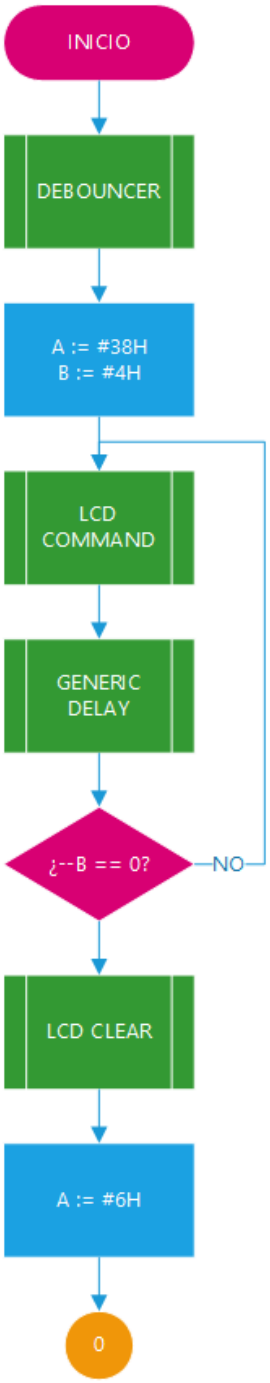
El módulo trabaja bajo la banda de 2.4 GHz, bajo el estándar de Bluetooth 2.0, siendo compatible con otros dispositivos que cuenten con alguna hyper terminal o emulador similar

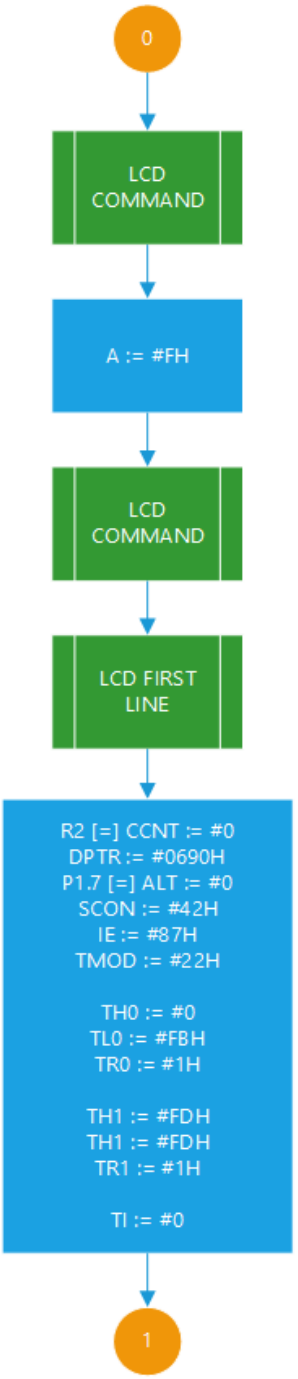
Algunas especificaciones adicionales incluyen la reconexión automática al último dispositivo vinculado, seguridad de vinculación mediante PIN y reconexión automática después de un tiempo de inactividad o lejanía con dispositivos receptores.

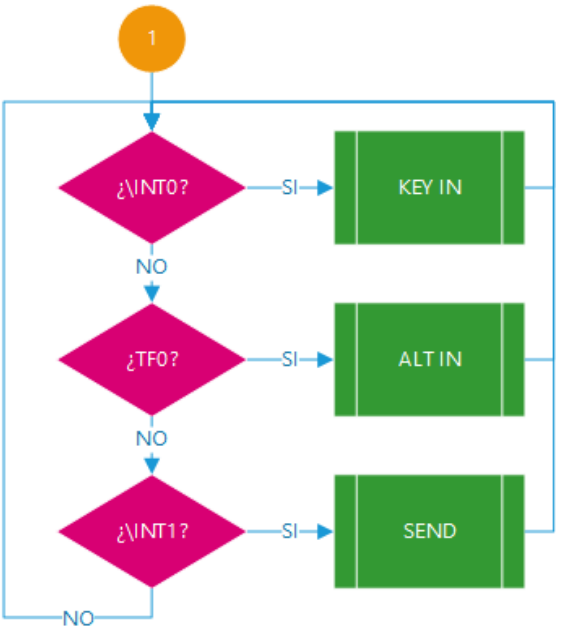
Esquemático

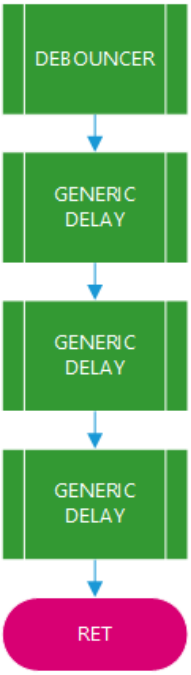


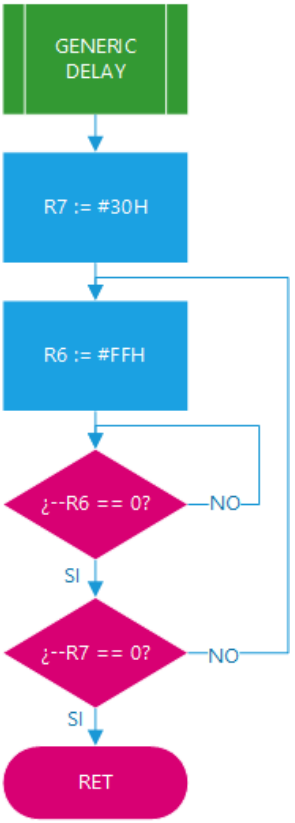
Diagrama(s) de flujo

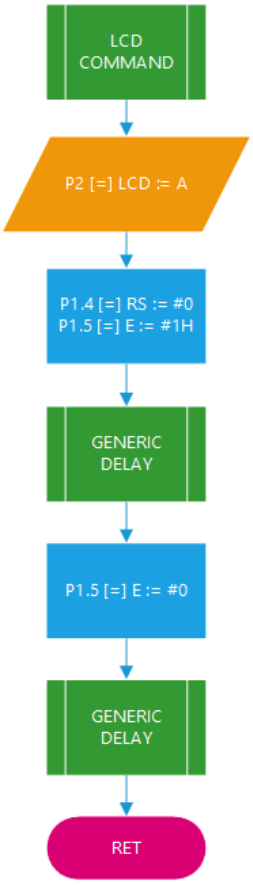


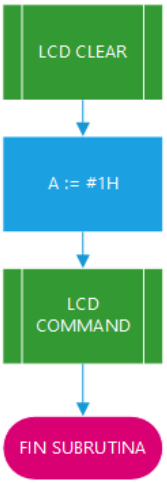


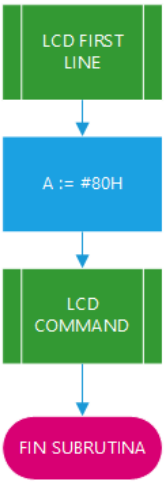


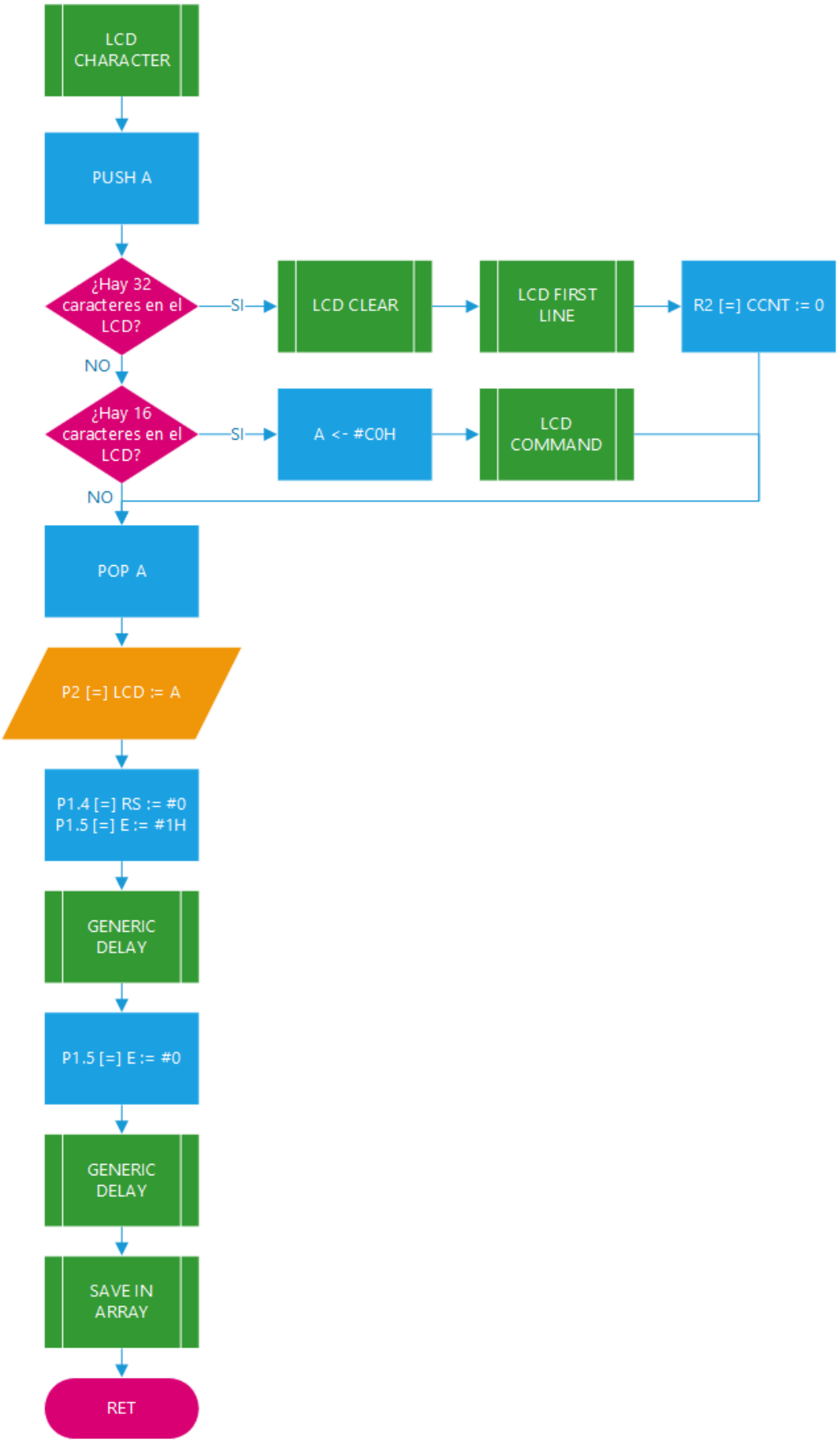

DEBOUNCER
GENERIC DELAY
GENERIC DELAY
GENERIC DELAY
RET

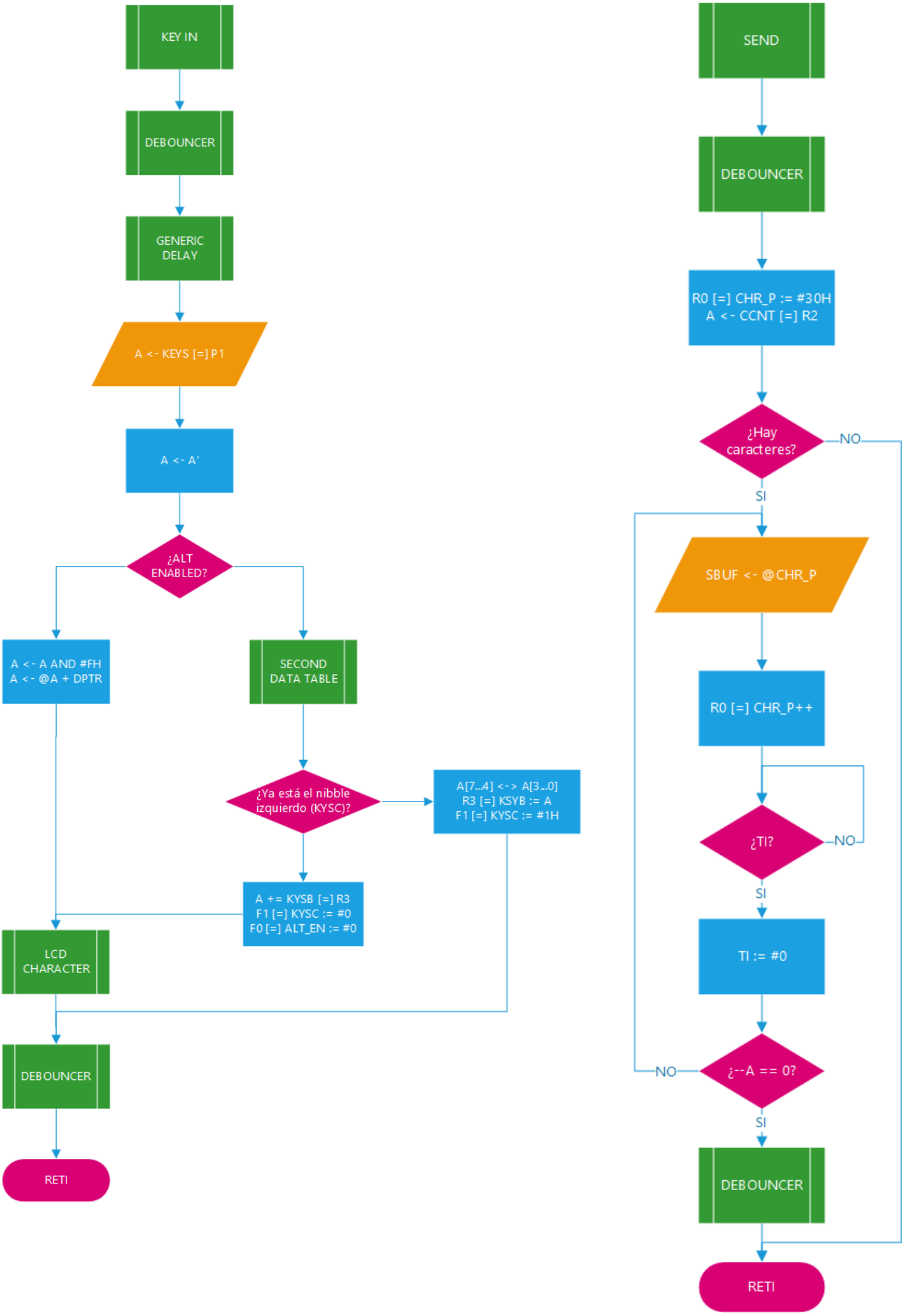

GENERIC DELAY
R7 := #30H
R6 := #FFH
¿--R6 == 0?
NO
SI
¿--R7 == 0?
NO
SI
RET

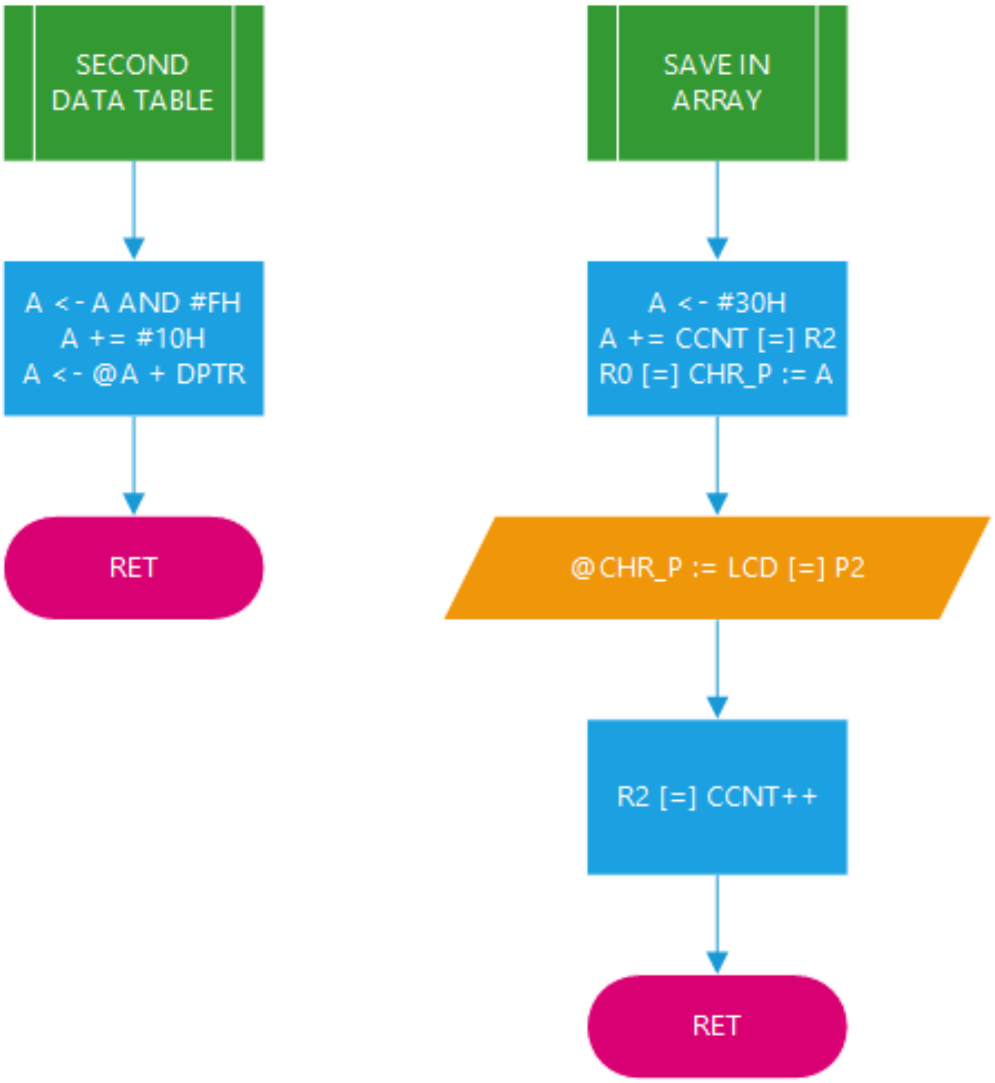

LCD COMMAND
P2 [=] LCD := A
P1.4 [=] RS := #0
P1.5 [=] E := #1H
GENERIC DELAY
P1.5 [=] E := #0
GENERIC DELAY
RET


LCD CLEAR
A := #1H
LCD COMMAND
FIN SUBRUTINA


LCD FIRST LINE
A := #80H
LCD COMMAND
FIN SUBRUTINA


LCD CHARACTER
PUSH A
¿Hay 32 caracteres en el LCD?
SI
LCD CLEAR
LCD FIRST LINE
R2 [=] CCNT := 0
NO
¿Hay 16 caracteres en el LCD?
SI
A <- #C0H
LCD COMMAND
NO
POP A
P2 [=] LCD := A
P1.4 [=] RS := #0
P1.5 [=] E := #1H
GENERIC DELAY
P1.5 [=] E := #0
GENERIC DELAY
SAVE IN ARRAY
RET


KEY IN
DEBOUNCER
GENERIC DELAY
A <- KEYS [=] P1
A <- A'
¿ALT ENABLED?
A <- A AND #FH
A <- @A + DPTR
SECOND DATA TABLE
¿Ya está el nibble izquierdo (KYSC)?
A[7..4] <-> A[3..0]
R3 [=] KSYB := A
F1 [=] KYSC := #1H
A += KYSB [=] R3
F1 [=] KYSC := #0
F0 [=] ALT_EN := #0
LCD CHARACTER
DEBOUNCER
RETI
SEND
DEBOUNCER
R0 [=] CHR_P := #30H
A <- CCNT [=] R2
¿Hay caracteres?
NO
SI
SBUF <- @CHR_P
R0 [=] CHR_P++
¿TI?
NO
SI
TI := #0
¿--A == 0?
NO
SI
DEBOUNCER
RETI


SECOND
DATA TABLE
A <- A AND #FH
A += #10H
A <- @A + DPTR
RET
SAVE IN
ARRAY
A <- #30H
A += CCNT [=] R2
R0 [=] CHR_P := A
@CHR_P := LCD [=] P2
R2 [=] CCNT++
RET

Listado del programa con comentarios

```
            RS      EQU P1.4            ; Resister Select de LCD
            E       EQU P1.5            ; Enable de LCD
            LCD     EQU P2              ; Bus para LCD
            KEYS    EQU P1              ; Tecla codificada
            ALT     EQU P1.7            ; Botón ALT
            ALT_EN  EQU 0D5H            ; ALT activado
            KYSC    EQU 0D1H            ; Bandera de nibble superior de ASCII (BBBB 0000)
            CHARR   EQU 030H            ; Inicio de arreglo para caracteres a enviar

            CHR_P   EQU R0              ; Apuntador a posición de arreglo
            CCNT    EQU R2              ; Contador de caracteres para display (max. 32 | 20H)
            KYSB    EQU R3              ; Contenedor para ASCII


            ORG 0000H
            JMP LCD_INIT

; ------------------ INTERRUPCIÓN TECLADO ------------------
            ORG 0003H
            JMP KEY_IN
; ----------------------------------------------------------


; ------------------- INTERRUPCIÓN ALT ---------------------
            ORG 000BH
            JMP ALT_IN
; ----------------------------------------------------------


; ----------------- INTERRUPCIÓN BLUETOOTH -----------------
            ORG 0013H
            JMP SEND_ND
; ----------------------------------------------------------


; --------------------- FLUJO PRINCIPAL --------------------
            ORG 0040H
/* ---------------------- SUBRUTINAS ---------------------- */
; Para interrupción externa 0 | Detección de tecla o ASCII
KEY_IN:     ACALL DEBOUNCER
            ACALL GEN_DEL               ; Retraso personalizado +1 retraso
            MOV A, KEYS
            CPL A

            JNB ALT_EN, KEY_SGL         ; Verificación para ASCII
            SJMP KEY_ALT

KEY_SGL:    ANL A, #0FH
            MOVC A, @A + DPTR            ; Decodificar caracter
KEY_SHW:    ACALL LCD_CHR               ; Enviar caracter a LCD
KEY_END:    ACALL DEBOUNCER
            RETI

KEY_ALT:    ACALL SND_DT                ; Decodificar tecla
            JB KYSC, ALT_1              ; Primer caracter

ALT_0:      SWAP A                      ; Posicionar primera mitad del ASCII
            MOV KYSB, A
            SETB KYSC                   ; Se indica la presencia del nibble inferior
            SJMP KEY_END

ALT_1:      ADD A, KYSB                 ; Posicionar segunda mitad del ASCII
            CLR KYSC
            CLR ALT_EN                  ; Resetear banderas para siguiente ASCII
            SJMP KEY_SHW                ; Enviar ASCII

SND_DT:     ANL A, #0FH
            ADD A, #10H                  ; Desfase para segunda tabla
            MOVC A, @A + DPTR
            RET


; Para interrupción externa 1 | Envío a bluetooth
SEND_ND:    ACALL DEBOUNCER

            MOV CHR_P, #CHARR           ; Inicio de arreglo
            MOV A, CCNT                 ; Cantidad de caracteres disponibles
            JZ ALT_END                  ; Cancelar si no hay caracteres
SN_L0:      MOV SBUF, @CHR_P
            INC CHR_P                   ; Siguiente caracter
            JNB TI, $
            CLR TI
            DJNZ ACC, SN_L0             ; Repetir hasta que se termine arreglo
            ACALL DEBOUNCER
```

```
            RETI

; Para detección de ALT
ALT_IN:     JNB ALT, ALT_END
            SETB ALT_EN                 ; Entrar en modo ALT
ALT_END:    RETI


; Atraso genérico para dejar libres temporizadores
GEN_DEL:    MOV R7, #30H
G_0:        MOV R6, #0FFH
G_1:        DJNZ R6, G_1
            DJNZ R7, G_0
            RET


; Múltiples retrasos para botones
DEBOUNCER:  ACALL GEN_DEL
            ACALL GEN_DEL
            ACALL GEN_DEL
            RET


; Limpiar LCD
LCD_CLR:    MOV A, #1H                  ; 20H a cada posición | Limpiar pantalla
            ACALL LCD_CMD


; Posicionar en primera línea, primera posición de LCD
LCD_FL:     MOV A, #80H                 ; Cursor en primera línea, primera posición
            ACALL LCD_CMD


; Enviar comandos a LCD
LCD_CMD:    MOV LCD, A                  ; Posicionar comando
            CLR RS                      ; Modo comando
            SETB E
            ACALL GEN_DEL
            CLR E                       ; Enviar comando
            ACALL GEN_DEL
            RET


; Enviar datos/caracteres a LCD
LCD_CHR:    PUSH ACC                    ; Guardar dato a desplegar

LCD_ESL:    CJNE CCNT, #20H, LCD_EFL    ; Fin de segunda línea
            ACALL LCD_CLR               ; Limpiar pantalla
            ACALL LCD_FL                ; y resetear a primera posición
            MOV CCNT, #0                ; Resetear conteo de caracteres

LCD_EFL:    CJNE CCNT, #10H, LCD_SND    ; Fin de primera línea
            MOV A, #0C0H                ; Cursor en segunda línea, primera posición
            ACALL LCD_CMD

LCD_SND:    POP ACC                     ; Recuperar dato a desplegar
            MOV LCD, A                  ; Posicionar datos
            SETB RS                     ; Modo datos
            SETB E
            ACALL GEN_DEL
            CLR E                       ; Enviar datos
            ACALL GEN_DEL
            ACALL ARR_SAVE              ; Guardar caracter introducido en arreglo
            RET


; Almacenar en arreglo de caracteres
; Inicio de arreglo: 30H
ARR_SAVE:   MOV A, #CHARR               ; Inicio de arreglo
            ADD A, CCNT                 ; Desplazamiento según caracteres actuales
            MOV CHR_P, A                ; Posición actual de arreglo
            MOV @CHR_P, LCD             ; Guardar elemento en posición actual
            INC CCNT                    ; Aumentar cantidad de elementos almacenados actuales
            RET

/* ------------------------------------------------------- */

LCD_INIT:   ACALL DEBOUNCER             ; Asegurar espera inicial

            MOV B, #4H
            MOV A, #38H                 ; Modo de 8 bits, 2 líneas (4 veces)
LI_0:       ACALL LCD_CMD
            ACALL GEN_DEL
            DJNZ B, LI_0

            ACALL LCD_CLR               ; Limpiar LCD

            ; Se incluye el siguiente comando por recomendación pero es omisible
```

```
            ; Incrementar posición, desplazamiento desactivado
            MOV A, #6H
            ACALL LCD_CMD

            MOV A, #0FH                 ; Display, cursor y parpadeo encendidos
            ACALL LCD_CMD

            ACALL LCD_FL                ; Cursor inicial

            MOV CCNT, #0                ; Resetear contador de caracteres en matriz

MAIN:       MOV DPTR, #0A8H * 0AH       ; Posicionar apuntador en tabla de valores
            CLR ALT                     ; Limpiar para detección de ALT

            MOV SCON, #42H              ; Habilitación de comunicación serial

            MOV IE, #87H                ; Interrupciones externas y T0 habilitadas
            MOV TMOD, #22H              ; T0 y T1 en modo autorrecarga

            MOV TH0, #0
            MOV TL0, #0FBH              ; Contar 500 nanosegundos
            SETB TR0

            MOV TH1, #0FDH
            MOV TL1, #0FDH              ; Estándar de 9600 baudios
            SETB TR1

            CLR TI

            SJMP $                      ; Loop principal

; ------------------------------------------------------------



; -------------- TABLA PARA DECODIFICAR TECLAS -------------

            ORG 0690H

            DB  '1', '2', '3', 'A'
            DB  '4', '5', '6', 'B'
            DB  '7', '8', '9', 'C'
            DB  'E', '0', 'F', 'D'

            DB  01H, 02H, 03H, 0AH
            DB  04H, 05H, 06H, 0BH
            DB  07H, 08H, 09H, 0CH
            DB  0EH, 00H, 0FH, 0DH

; ------------------------------------------------------------

            END
```

Problemas encontrados

Dentro de todos los problemas encontrados al realizar esta práctica, el que más presencia tuvo fue al transmitir datos por medio de bluetooth. El problema consistía en la mala transmisión del arreglo, ya que omitía la última letra o la primera, además de mandarlo con un gran conjunto de datos basura. Incluso si no se mandaba ningún mensaje, al realizar la transmisión los datos basura estaban presentes.

El segundo problema fue el cambio de renglón al llegar a los 16 caracteres y la limpieza de la pantalla al llegar a los 32 caracteres escritos, esto causado por error al momento de hacer el conteo de caracteres escritos en la pantalla. El tercer error fue parte del hardware, esto por una conexión realizada al teclado matricial sin antes haber consultado la documentación de este, pues al realizar una medición de voltaje esté siempre era cero.

El último problema fue la mala administración del tiempo. No tener una buena planeación generó que la realización del reporte se hiciera a último momento. Aunque esto va más allá que hardware y software, son simples problemas de comunicación.

Conclusiones

- Guzmán Claustro:

  La realización de esta práctica fue realmente un reto, aunque la buena investigación hizo que se lograra hacer de manera exitosa. La parte más complicada recayó en el software, ya que se tuvo que idear desde cero cómo manejar las interrupciones requeridas (las del teclado, ALT y serial). Sin duda me llevo una buena experiencia de esto, ya que la combinación de elementos con los que convivimos día a día y de los cuales somos conscientes de su funcionamiento como es el caso de la conexión Bluetooth hizo que pudiera ver el alcance que pueden llegar a tener las aplicaciones con el 8051 y su programación con tan bajo nivel. Me llevo una buena experiencia, pero sinceramente solo me hizo formalizar mis bases al estar seguro que la seguridad informática es lo mío y no la electrónica a un nivel tan profundo como este.

- Cordero Hernández:

  Existen misterios del mundo moderno de los cuales estoy seguro que la vida no me alcanzará para descifrarlos, como aquel en donde se relata acerca de la existencia de personas que opinan que esta práctica es la más sencilla de todas. Si bien no la consideré como una nueva instancia de repentina, considero a esta práctica como una digna candidata de proyecto final, ya que combina conceptos físicos en el hardware como lo fue la realización de una compuerta NOT de manera artesanal, junto con muchos otros conceptos de software como limitaciones de código, interrupciones, subrutinas, puerto serial y demás. Esto último mencionado se sabe que se encuentra dentro del infame microcontrolador AT89S52, sin embargo, su manipulación por medio de ensamblador nuevamente incitó al equipo a pensar como no se había hecho antes en cuanto a trabajo de bajo nivel se refiere; eso, en conjunto de algunos otros pensamiento intrusivos relacionados con la insoportable levedad del ser personas horribles, finalmente nos hizo desarrollar la que en mi opinión ha sido la mejor práctica del curso en cuanto a nuestro triada se refiere. Realmente puedo comprender el gusto por el resultado final y visualización del desarrollo, ya que fue sumamente satisfactorio poder observar el producto ejecutándose en conjunto de todos los elementos, especialmente el funcionamiento del módulo bluetooth de la mano del puerto serial.

  Finalmente y a decir verdad, al analizar todo el desarrollo encontrado en este documento no parece tan complicado, no obstante, no dejamos de ser principiantes en este ámbito. No queda ninguna duda de la diferencia en el nivel de conocimiento con el que entramos con respecto al actual, con el que afortunadamente saldremos la materia, un nivel que puede demostrar por lo menos la ejecución y funcionamiento de código casi completamente hexadecimal junto con un microprocesador e incluso periféricos externos.

- Rodríguez Castro:

  Esta práctica fue la que más problemas ocasionó ya que tuvimos problemas muy específicos que requirieron una búsqueda exhaustiva y minuciosa para encontrar las causas raíces y poder idear una solución correcta al problema. El funcionamiento armónico de los diferentes componentes fue un verdadero reto porque en ocasiones nos encontramos problemas en un componente que afectaba el funcionamiento de otro. Nos tomó más tiempo de lo esperado realizar el programa ya que ocurrían fallos inesperados y nos tomaba tiempo encontrar y solucionar pequeños fallos. Definitivamente fue la práctica que más contenido abordó, la más extensa y la que abarca casi todo el contenido del curso.

Referencias bibliográficas

- Fairchild Semiconductor. (1987). *MM74C922 MM74C923 16-Key Encoder 20-Key Encoder*. Recuperado de https://www.soemtron.org/downloads/disposals/74c922.pdf.
- Arm Keil. (s.f.). 8051 Baud Rate Calculator. Recuperado de https://www.keil.com/c51/baudrate.asp.
- ITeadStudio. (2010). *HC-05 Bluetooth module* (N.º 01). Recuperado de https://datasheetspdf.com/pdf-file/1418730/ITead/HC-05/1.
- *LCD MOUDULE SPECIFICATION FOR APPROVAL JHD162A* (1.0). (2004). http://www.datasheet.es/PDF/512991/JHD162A-pdf.html
- Mazidi, M. A., Mazidi, J. G., McKinlay, R. D. (2014). The 8051 Microcontroller and Embedded Systems. Harlow: Pearson.
- MacKenzie, I. S., Phan, R. C.-W. (2007). The 8051 Microcontroller. New Jersey: Pearson.