CURSO DE PROGRAMACIÓN E INTERFAZ DE MICROCONTROLADORES PIC

Video 05 — Parte 5: Interfaz con Periféricos Comunes — Teclado Numérico y Pantalla LCD · Plataforma Oficial: cursosmecatronica.com

Módulo: Parte 5 — Interfaz con Periféricos Comunes: Teclado Numérico y Pantalla LCD
Enfoque de Ingeniería: Teclado Matricial 4x3 y 4x4, Barrido por Filas y LOOKUP, Pantalla LCD 16x2 HD44780 en 4 Bits, DEFINE LCD y LCDOUT $FE

```
Idx = 4 · Fila + Col | Acumulador: Valor = Valor · 10 + Tecla | Linea1 = $FE,$80 | Linea2 = $FE,$C0
```

1. Desarrollo Técnico Diapositiva por Diapositiva

Diapositiva 01: Arquitectura del Teclado Numérico Matricial de 4x3 y 4x4

Reducción de Dieciséis Pulsadores Individuales a Ocho Líneas mediante Filas y Columnas Cruzadas

- **Turno 1 (Orus):** Si intentáramos conectar un teclado numérico de dieciséis teclas cableando un pulsador independiente por cada terminal, agotaríamos por completo las trece líneas de entrada y salida de nuestro PIC dieciséis efe ochenta y cuatro.
- **Turno 2 (Aoede):** Para ahorrar terminales de manera inteligente, los teclados numéricos comerciales organizan sus interruptores en una cuadrícula matricial de cuatro filas horizontales por tres columnas para doce teclas, o cuatro por cuatro para dieciséis teclas.
- **Turno 3 (Orus):** En reposo eléctrico ninguna fila toca ninguna columna; únicamente cuando el usuario presiona una tecla física se cierra el contacto metálico o elastomérico que une exactamente una fila específica con una columna específica.
- **Turno 4 (Aoede):** De este modo gobernamos dieciséis botones empleando tan solo ocho pines de un puerto, reduciendo a la mitad el cableado y dejando cinco terminales libres en el puerto A para pantallas o actuadores.

Diapositiva 02: Conexión Eléctrica del Teclado al Puerto B y Resistencias Pull-Up

Filas como Salidas de Barrido, Columnas como Entradas con Pull-Up Interno y Protección contra Cortocircuitos

- **Turno 1 (Orus):** La topología estándar conecta las cuatro filas del teclado a los cuatro bits inferiores del puerto B configurados como salidas, y las cuatro columnas a los cuatro bits superiores configurados como entradas digitales.
- **Turno 2 (Aoede):** Para que las columnas nunca queden flotando al aire cuando no se oprime nada, activamos las resistencias internas hacia positivo del puerto B poniendo en cero el séptimo bit del registro OPTION, manteniendo las entradas en uno lógico.
- **Turno 3 (Orus):** Debemos considerar una precaución eléctrica vital: si el operador presiona por error dos teclas de la misma columna al mismo tiempo mientras una fila está en cero voltios y otra fila está en cinco voltios, provocaría un cortocircuito directo entre pines.
- **Turno 4 (Aoede):** Evitamos este riesgo colocando resistencias limitadoras en serie de cuatrocientos setenta ohmios en las líneas de fila, o bien poniendo en alta impedancia las tres filas inactivas mientras solo la fila explorada baja a cero voltios.

Diapositiva 03: Algoritmo de Barrido Secuencial por Filas y Tabla LOOKUP

Activación en Nivel Bajo de Cada Fila, Lectura de Columnas y Conversión de Coordenada a Valor Numérico

- **Turno 1 (Orus):** ¿Cómo descubre el software exactamente cuál de las dieciséis teclas fue presionada? Mediante un barrido secuencial rapidísimo: primero ponemos únicamente la fila cero en nivel bajo cero mientras las demás permanecen en uno.
- **Turno 2 (Aoede):** En ese instante leemos las cuatro columnas superiores: si todas leen uno lógico significa que nadie pulsó botones en la primera fila, por lo que pasamos a bajar la fila uno y repetimos la inspección.
- **Turno 3 (Orus):** Apenas una columna cae a nivel cero sabemos con certeza la coordenada fila y columna del cruce activo, calculando un índice único de cero a quince mediante la fórmula cuatro multiplicado por fila más columna.
- **Turno 4 (Aoede):** Finalmente pasamos ese índice a la instrucción LOOKUP de PicBasic Pro para traducir la posición física del teclado en el dígito real impreso en la carátula, ya sea del cero al nueve, asterisco o numeral.

Diapositiva 04: Antirrebote de Teclado, Espera de Liberación y Captura de Claves

Filtro Temporal de 20 ms, Acumulación Decimal por Multiplicación por Diez y Tecla de Confirmación

- **Turno 1 (Orus):** Al igual que un pulsador aislado, las membranas de un teclado matricial rebotan durante cinco a quince milisegundos al hacer contacto, por lo que al detectar una columna en cero insertamos una pausa de veinte milisegundos y reconfirmamos la lectura.
- **Turno 2 (Aoede):** Además, antes de devolver el valor a la aplicación principal debemos esperar dentro de un bucle corto a que todas las columnas vuelvan a uno lógico, garantizando que una sola presión humana no se registre diez veces seguidas.
- **Turno 3 (Orus):** Para ingresar cantidades de varias cifras, como una consigna de mil doscientas revoluciones o un código secreto de cuatro dígitos, acumulamos cada nueva tecla numérica haciendo Valor igual a Valor multiplicado por diez más Tecla.
- **Turno 4 (Aoede):** Asignamos la tecla asterisco para borrar la cifra en caso de equivocación poniendo Valor en cero, y reservamos la tecla numeral como botón de confirmación que valida y guarda el parámetro ingresado.

Diapositiva 05: La Pantalla Alfanumérica LCD 16x2 y el Controlador Estándar HD44780

Dieciséis Terminales, Alimentación Lógica, Potenciómetro de Contraste en Vo y Retroiluminación LED

- **Turno 1 (Orus):** El periférico visual más versátil en instrumentación con microcontroladores es el módulo de cristal líquido alfanumérico de dos filas por dieciséis caracteres basado en el controlador industrial estándar hache de cuarenta y cuatro setecientos ochenta.
- **Turno 2 (Aoede):** A diferencia de los indicadores de siete segmentos limitados a cifras rudimentarias, esta pantalla muestra letras mayúsculas y minúsculas, símbolos de ingeniería, unidades físicas y mensajes claros consumiendo apenas un par de miliamperios.
- **Turno 3 (Orus):** Revisemos sus pines críticos de energía: el pin uno va a tierra, el pin dos recibe cinco voltios regulados y el pin tres, llamado ve sub cero, ajusta el contraste mediante el cursor central de un potenciómetro de diez kiloohmios.
- **Turno 4 (Aoede):** Olvidar ajustar ese potenciómetro de contraste es la causa número uno por la que una pantalla recién conectada parece apagada o muestra rectángulos negros sólidos; mientras que los pines quince y dieciséis alimentan la luz de fondo con resistencia serie.

Diapositiva 06: Señales de Control del LCD: RS, R/W, Pulso Enable y Bus de 4 u 8 Bits

Selección entre Instrucción y Carácter ASCII, Aterrizaje de Escritura y Transferencia por Nibbles

- **Turno 1 (Orus):** El diálogo con la pantalla se gobierna mediante tres líneas de control: el pin cuatro es selección de registro R S, el cual en cero indica un comando interno como borrar pantalla y en uno indica un carácter visible.
- **Turno 2 (Aoede):** El pin cinco es lectura o escritura; como casi siempre solo enviamos datos desde el PIC hacia la pantalla, conectamos este terminal directamente a tierra para ahorrar un valioso pin del microcontrolador.
- **Turno 3 (Orus):** El pin seis es la señal de habilitación Enable, que actúa como estrobo de validación: la pantalla captura los datos presentes en el bus justo en el flanco descendente de este pulso.
- **Turno 4 (Aoede):** Aunque el módulo posee ocho líneas de datos de de cero a de siete, el controlador soporta un ingenioso modo de cuatro bits usando solo de cuatro a de siete, enviando primero los cuatro bits altos y enseguida los cuatro bajos.

Diapositiva 07: Conexión en 4 Bits al PIC16F84 y Directivas DEFINE LCD en PicBasic Pro

Ocupación Total de Solo Seis Pines del Microcontrolador y Pausa Inicial de Arranque de 500 ms

- **Turno 1 (Orus):** Gracias al modo de cuatro bits y a fijar escritura a tierra, conectamos toda una pantalla alfanumérica empleando únicamente seis pines del PIC: cuatro líneas consecutivas para datos y dos líneas para R S y Enable.
- **Turno 2 (Aoede):** Por defecto PicBasic Pro asume los datos en los cuatro bits inferiores del puerto A y el control en erre a cuatro y erre be tres, recordando que si usamos erre a cuatro es obligatoria una resistencia externa hacia cinco voltios.
- **Turno 3 (Orus):** Si preferimos reubicar la pantalla en cualquier otro puerto, como el puerto B, basta escribir al inicio las directivas DEFINE L C D guion bajo D R E G, D B I T, R S R E G, R S B I T, E R E G y E B I T.
- **Turno 4 (Aoede):** Un detalle práctico fundamental es colocar una instrucción PAUSE quinientos al comienzo del programa antes del primer envío a la pantalla, dándole medio segundo al controlador interno del cristal líquido para completar su propio reinicio.

Diapositiva 08: El Comando LCDOUT y Códigos de Control precedidos por $FE

Borrado Total $FE,1, Primera Línea $FE,$80, Segunda Línea $FE,$C0 y Control del Cursor

- **Turno 1 (Orus):** Una vez declarada la conexión, PicBasic Pro inicializa automáticamente el modo de cuatro bits la primera vez que ejecuta la poderosa instrucción LCDOUT seguida de la lista de elementos separados por comas.
- **Turno 2 (Aoede):** Para distinguir una orden de control de un texto normal se antepone el prefijo hexadecimal dólar efe e: así, LCDOUT dólar efe e coma uno borra por completo la pantalla y sitúa el cursor al inicio de la fila superior.
- **Turno 3 (Orus):** Para actualizar datos en tiempo real sin parpadeo evitamos borrar continuamente toda la pantalla y en su lugar reposicionamos el cursor: dólar efe e coma dólar ochenta apunta al primer carácter de la primera línea.
- **Turno 4 (Aoede):** Por su parte, la secuencia dólar efe e coma dólar ce cero salta directamente al comienzo de la segunda línea, pudiendo sumar cualquier desplazamiento de cero a quince para escribir en la columna exacta deseada.

Diapositiva 09: Formateo Profesional de Variables en LCD: DEC, HEX, BIN y Punto Fijo

Ancho Fijo con Ceros para Evitar Caracteres Fantasma y Presentación de Décimas Enteras

- **Turno 1 (Orus):** Si enviamos una variable numérica cruda dentro de LCDOUT, la pantalla intentará dibujar el símbolo ASCII correspondiente a ese número en vez de sus dígitos; para mostrar su valor legible anteponemos el modificador DEC.
- **Turno 2 (Aoede):** También disponemos de I H E X para mostrar registros en hexadecimal precedidos por signo dólar, y de I B I N ocho para observar en vivo el estado individual de los ocho bits de un puerto.
- **Turno 3 (Orus):** Un problema visual clásico ocurre cuando un contador baja de cien a noventa y nueve y deja un cero fantasma a la derecha convirtiendo el número en novecientos noventa: lo resolvemos usando ancho fijo como DEC tres o agregando espacios finales.
- **Turno 4 (Aoede):** Y para mostrar magnitudes con decimales usando variables enteras, por ejemplo doscientos cuarenta y cinco representando veinticuatro coma cinco grados, imprimimos DEC temp dividido diez coma punto entre comillas coma DEC uno temp módulo diez.

Diapositiva 10: Integración Práctica: Sistema de Teclado Matricial y Pantalla LCD 16x2

Compartición Inteligente de Líneas de Datos entre Filas del Teclado y Bus de 4 Bits del LCD

- **Turno 1 (Orus):** Cuando un proyecto en el PIC dieciséis efe ochenta y cuatro exige simultáneamente un teclado de dieciséis botones y una pantalla L C D más salidas de control, ocho más seis pines sumarían catorce, superando los trece disponibles.
- **Turno 2 (Aoede):** La solución elegante de ingeniería consiste en multiplexar las cuatro líneas de datos de la pantalla con las cuatro salidas de barrido del teclado a través de resistencias serie de aislamiento de un kiloohmio.
- **Turno 3 (Orus):** Mientras la línea Enable de la pantalla permanece en cero lógico, el controlador visual ignora por completo los cambios en esas cuatro líneas durante el milisegundo que dura el barrido del teclado matricial.
- **Turno 4 (Aoede):** De esta manera logramos una interfaz hombre máquina completa dejando libres varios terminales del puerto A para conectar sensores y actuadores digitales, tema central que abordaremos en la Parte seis.

2. Código Fuente en PicBasic Pro (picbasic_pro_video_05.pbp)

```
' ================================================================================
' CURSO: PROGRAMACION E INTERFAZ DE MICROCONTROLADORES PIC (cursosmecatronica.com)
' Video 05 – Parte 5: Interfaz con Periféricos Comunes – Teclado Numérico y Pantalla LCD
' Enfoque: Teclado Matricial 4x3 y 4x4, Barrido por Filas y LOOKUP, Pantalla LCD 16x2 HD44780 en 4 Bits, DEFINE
LCD y LCDOUT $FE
' Ecuaciones Clave: Idx = 4 · Fila + Col  |  Acumulador: Valor = Valor · 10 + Tecla  |  Linea1 = $FE,$80  |
Linea2 = $FE,$C0
' Microcontrolador Objetivo: PIC16F84A @ 4.000 MHz (1 ciclo = 1 us)
' ================================================================================
@ DEVICE pic16F84A, XT_OSC, WDT_OFF, PWRT_ON, PROTECT_OFF
Include "modedefs.bas"
DEFINE OSC 4

' --- Declaracion de Pines y Variables ---
EntradaSensor  VAR PORTA.0
SalidaRele     VAR PORTB.0
LedHeartbeat   VAR PORTB.7
TxDiagnostico  VAR PORTA.1
ContadorCiclos VAR WORD
Estado         VAR BYTE

' --- Inicializacion Segura de Puertos (Estado antes que Direccion TRIS) ---
PORTA = %00000000
PORTB = %00000000
OPTION_REG.7 = 0            ' Pull-ups internos activos en PORTB
TRISA = %00000001           ' RA0 entrada, resto salidas
TRISB = %00000000           ' PORTB como salidas digitales
ContadorCiclos = 0
Estado = 0

' --- Lazo Principal Determinista ---
LazoPrincipal:
    TOGGLE LedHeartbeat
    ContadorCiclos = ContadorCiclos + 1
    IF EntradaSensor = 0 THEN
        PAUSE 20            ' Filtro antirrebote 20 ms
        IF EntradaSensor = 0 THEN
            HIGH SalidaRele
            Estado = 1
        ENDIF
    ELSE
        LOW SalidaRele
        Estado = 0
    ENDIF
    SEROUT TxDiagnostico, T2400, ["V05 Ciclo=", #ContadorCiclos, " Est=", #Estado, 13, 10]
    PAUSE 100
    GOTO LazoPrincipal
    END
```

3. Referencia en Ensamblador MPASM (ensamblador_mpasm_video_05.asm)

```
; ==========================================================================
; CURSO: PROGRAMACION E INTERFAZ DE MICROCONTROLADORES PIC (cursosmecatronica.com)
; Video 05 – Parte 5: Interfaz con Periféricos Comunes – Teclado Numérico y Pantalla LCD – Referencia de Bajo Nivel MPASM (PIC16F84A)
; ==========================================================================
    LIST        P=16F84A
    #INCLUDE    <P16F84A.INC>
    __CONFIG    _XT_OSC & _WDT_OFF & _PWRTE_ON & _CP_OFF

    CBLOCK  0x0C
        W_TEMP
        STATUS_TEMP
        CONTADOR
    ENDC

    ORG     0x0000
    GOTO    INICIO

    ORG     0x0004              ; Vector de Interrupcion Hardware
    MOVWF   W_TEMP              ; Salvado de contexto
    SWAPF   STATUS, W
    MOVWF   STATUS_TEMP
    BCF     INTCON, INTF        ; Limpiar bandera RB0/INT
    SWAPF   STATUS_TEMP, W
    MOVWF   STATUS
    SWAPF   W_TEMP, F
    SWAPF   W_TEMP, W
    RETFIE

INICIO:
    BCF     STATUS, RP0         ; Banco 0
    CLRF    PORTA
    CLRF    PORTB
    BSF     STATUS, RP0         ; Banco 1
    MOVLW   B'00000001'
    MOVWF   TRISA               ; RA0 entrada
    CLRF    TRISB               ; PORTB salidas
    BCF     STATUS, RP0         ; Regresar a Banco 0

BUCLE:
    BTFSS   PORTA, 0
    BSF     PORTB, 0
    BTFSC   PORTA, 0
    BCF     PORTB, 0
    GOTO    BUCLE
    END
```

4. Equivalencia en Lenguaje C XC8 (firmware_xc8_video_05.c)

```
/* ============================================================================
 * CURSO: PROGRAMACION E INTERFAZ DE MICROCONTROLADORES PIC (cursosmecatronica.com)
 * Video 05 – Parte 5: Interfaz con Periféricos Comunes – Teclado Numérico y Pantalla LCD – Equivalencia en C Embebido (XC8)
 * ============================================================================ */
#include <xc.h>
#define _XTAL_FREQ 4000000UL

#pragma config FOSC = XT, WDTE = OFF, PWRTE = ON, CP = OFF

void main(void) {
    PORTA = 0x00;
    PORTB = 0x00;
    OPTION_REGbits.nRBPU = 0;   /* Pull-ups internos activos en PORTB */
    TRISA = 0b00000001;         /* RA0 como entrada */
    TRISB = 0b00000000;         /* RB0..RB7 como salidas */

    while (1) {
        PORTBbits.RB7 ^= 1;     /* LED Heartbeat */
        if (PORTAbits.RA0 == 0) {
            __delay_ms(20);
            if (PORTAbits.RA0 == 0) {
                PORTBbits.RB0 = 1;
            }
        } else {
            PORTBbits.RB0 = 0;
        }
        __delay_ms(100);
    }
}
```