CURSO DE PROGRAMACIÓN E INTERFAZ DE MICROCONTROLADORES PIC

Video 04 — Parte 4: Ejemplos Prácticos en PicBasic Pro y Uso de Interrupciones · Plataforma Oficial: cursosmecatronica.com

Módulo: Parte 4 — Ejemplos Prácticos en PicBasic Pro y Uso de Interrupciones
Enfoque de Ingeniería: Secuenciador LED, BUTTON, SOUND, PWM + Filtro RC, PULSOUT/PULSIN (10 us @ 4 MHz), SEROUT/SERIN, INTCON y ON INTERRUPT GOTO

```
t_servo = N_pulsout · 10 us | d_cm = (t_eco_us · 0.0343) / 2 | V_avg = (Duty / 255) · 5.0 V | INTCON = %10010000
```

1. Desarrollo Técnico Diapositiva por Diapositiva

Diapositiva 01: Ejemplo Práctico I: Secuenciador Bidireccional de Ocho LEDs en Puerto B

Desplazamiento Binario Izquierda y Derecha sobre PORTB con Resistencias Limitadoras de 330 Ohmios

- **Turno 1 (Orus):** Nuestro primer ejemplo práctico convierte los ocho terminales del puerto B en un secuenciador luminoso bidireccional, conectando cada pin a un diodo emisor de luz en serie con una resistencia limitadora de trescientos treinta ohmios hacia tierra.
- **Turno 2 (Aoede):** Tras declarar DEFINE O S C cuatro, limpiamos P O R T B igual a cero y configuramos T R I S B igual a cero para que las ocho patitas operen como salidas digitales capaces de suministrar unos diez miliamperios por canal.
- **Turno 3 (Orus):** En lugar de escribir ocho líneas repetitivas encendiendo y apagando cada pin, cargamos una variable patrón tipo BYTE con el valor inicial uno y dentro de un ciclo FOR de siete pasos asignamos P O R T B igual a patrón y desplazamos un bit a la izquierda.
- **Turno 4 (Aoede):** Al llegar al extremo en el bit siete, un segundo bucle desplaza el bit encendido un lugar hacia la derecha cada cien milisegundos, demostrando cómo los operadores de desplazamiento producen código mínimo y elegante.

Diapositiva 02: Ejemplo Práctico II: Lectura Confiable de Pulsadores y Comando BUTTON

Eliminación de Rebote Mecánico, Polaridad Activa en Bajo y Parámetros de Autorrepetición

- **Turno 1 (Orus):** En el segundo ejemplo leemos pulsadores mecánicos para incrementar o decrementar un contador: si solo evaluamos un condicional simple dentro de un lazo rápido, una sola pulsación del dedo se contará decenas de veces debido al rebote metálico.
- **Turno 2 (Aoede):** La solución manual consiste en detectar el nivel activo, ejecutar una pausa de veinte milisegundos para dejar extinguir las oscilaciones mecánicas del resorte, confirmar que sigue presionado y esperar o registrar el flanco de liberación.
- **Turno 3 (Orus):** PicBasic Pro incluye además el potente comando dedicado BUTTON, el cual gestiona en una sola línea el pin de entrada, el estado activo cero o uno, el retardo de antirrebote, la velocidad de autorrepetición y una variable byte de trabajo.
- **Turno 4 (Aoede):** Así podemos crear botones de ajuste industrial que al tocarse brevemente avanzan una sola unidad precisa, pero al mantenerse oprimidos durante un segundo comienzan a avanzar de forma continua y controlada.

Diapositiva 03: Ejemplo Práctico III: Tonos Acústicos con SOUND y Regulación con PWM

Excitación de Transductor Piezoeléctrico y Generación de Tensión Analógica Media mediante Filtro RC

- **Turno 1 (Orus):** El tercer ejemplo explota dos comandos temporales para producir sonido y controlar potencia: la instrucción SOUND pin coma corchete nota coma duración genera ondas cuadradas audibles para excitar un zumbador piezoeléctrico pasivo a través de un capacitor en serie.
- **Turno 2 (Aoede):** Las notas de uno a ciento veintisiete producen frecuencias musicales ascendentes, mientras que los valores de ciento veintiocho a doscientos cincuenta y cinco generan ruido blanco útil para alarmas sonoras de maquinaria.
- **Turno 3 (Orus):** Por otro lado, aunque el PIC dieciséis efe ochenta y cuatro no posee módulo hardware de modulación por ancho de pulso ni conversor digital a analógico, el comando P W M pin coma ciclo coma ráfagas sintetiza trenes modulados de cero a doscientos cincuenta y cinco.
- **Turno 4 (Aoede):** Si enviamos esa salida digital hacia un filtro pasa bajas compuesto por una resistencia y un capacitor seguido de un seguidor operacional, obtenemos una tensión analógica proporcional entre cero y cinco voltios.

Diapositiva 04: Ejemplo Práctico IV: Servomotores con PULSOUT y Medición con PULSIN

Resolución de 10 Microsegundos a 4 MHz para Trenes de 1.0 a 2.0 ms y Cronometraje de Ecos Ultrasónicos

- **Turno 1 (Orus):** En robótica y mecatrónica necesitamos posicionar servomotores estándar enviándoles cada veinte milisegundos un pulso positivo cuyo ancho exacto entre mil y dos mil microsegundos codifica el ángulo del eje de cero a ciento ochenta grados.
- **Turno 2 (Aoede):** Con un cristal de cuatro megahercios, cada unidad del comando PULSOUT equivale exactamente a diez microsegundos; por tanto, PULSOUT Servo coma ciento cincuenta genera un pulso impecable de mil quinientos microsegundos que centra el servomotor a noventa grados.
- **Turno 3 (Orus):** Su comando complementario es PULSIN pin coma estado coma variable Word, el cual mide cuánto dura un pulso entrante en unidades de diez microsegundos e incluye un límite automático de seiscientos cincuenta y cinco milisegundos para no colgarse si falta la señal.
- **Turno 4 (Aoede):** Combinando un pulso corto de disparo y una lectura con PULSIN sobre un sensor ultrasónico calculamos la distancia en centímetros multiplicando el tiempo medido por la velocidad del sonido y dividiendo entre dos.

Diapositiva 05: Ejemplo Práctico V: Comunicación Serie Asíncrona con SEROUT y SERIN

Trama UART 8N1 a 2400 o 9600 Baudios, Modos Verdadero con MAX232 e Invertido Directo con Resistencia

- **Turno 1 (Orus):** El quinto ejemplo dota al PIC dieciséis efe ochenta y cuatro de comunicación serial asíncrona hacia una computadora o hacia otro microcontrolador, aun careciendo de puerto serial físico interno, gracias a las rutinas por software SEROUT y SERIN.
- **Turno 2 (Aoede):** Tras incluir el archivo de definiciones modedefs punto bas disponemos de constantes para seleccionar velocidades estándar como dos mil cuatrocientos o nueve mil seiscientos baudios tanto en polaridad verdadera como en polaridad invertida.
- **Turno 3 (Orus):** Si usamos un circuito conversor de niveles como el clásico M A X doscientos treinta y dos elegimos modo verdadero; mientras que en modo invertido PicBasic Pro permite enlazar directamente la recepción a través de una resistencia limitadora de veintidós kiloohmios.
- **Turno 4 (Aoede):** Además, el comando avanzado SERIN dos incorpora temporizador de espera máxima y filtro calificador de encabezado, asegurando que el microcontrolador solo procese paquetes válidos sin bloquear el lazo de control.

Diapositiva 06: Teoría de Interrupciones frente a Encuesta Continua Polling

Atención Asíncrona Inmediata a Eventos Críticos, Salto al Vector 0x0004 y Retorno Transparente

- **Turno 1 (Orus):** Hasta ahora nuestros programas revisaban sensores de forma secuencial mediante encuesta continua o polling; pero ¿qué sucede si llega un pulso fugaz de emergencia justo cuando el procesador está ejecutando un PAUSE de quinientos milisegundos?
- **Turno 2 (Aoede):** En un esquema de encuesta ese pulso se pierde irremediablemente. Para resolverlo existen las interrupciones por hardware: señales asíncronas capaces de suspender el programa principal para atender de inmediato un evento prioritario.
- **Turno 3 (Orus):** Cuando ocurre una interrupción habilitada en el PIC, el hardware termina la instrucción en curso, apaga el permiso global para evitar reentradas, guarda el contador de programa en la pila y salta automáticamente a la dirección cuatro hexadecimal.
- **Turno 4 (Aoede):** Allí se ejecuta la rutina de servicio de interrupción, limpia la bandera que originó el aviso y al finalizar restaura la dirección desde la pila y reactiva las interrupciones globales exactamente donde se había quedado.

Diapositiva 07: Las Cuatro Fuentes de Interrupción del PIC16F84 y el Registro INTCON

Permiso Global GIE, Flanco Externo RB0/INT, Cambio en RB4..RB7, Desbordamiento TMR0 y Escritura EEPROM

- **Turno 1 (Orus):** El PIC dieciséis efe ochenta y cuatro posee cuatro fuentes de interrupción gobernadas por el registro de ocho bits I N T C O N: primero, una entrada externa por flanco ascendente o descendente en el pin dedicado erre be cero.
- **Turno 2 (Aoede):** La segunda fuente detecta cualquier cambio de estado en cualquiera de los cuatro pines superiores del puerto B, desde erre be cuatro hasta erre be siete, ideal para despertar al sistema al pulsar un teclado matricial.
- **Turno 3 (Orus):** La tercera fuente se dispara cuando el contador interno temporizador cero pasa de doscientos cincuenta y cinco a cero; y la cuarta avisa cuando concluye una grabación lenta en la memoria EEPROM de datos.
- **Turno 4 (Aoede):** Cada fuente posee en el registro I N T C O N dos bits gemelos: un bit de habilitación individual y un bit de bandera de ocurrencia que nuestro código debe poner obligatoriamente en cero antes de salir para no reentrar sin fin.

Diapositiva 08: Programación de Interrupciones en PicBasic Pro: ON INTERRUPT GOTO

Directivas DISABLE y ENABLE, Retorno con RESUME y Limpieza Explícita de la Bandera en INTCON

- **Turno 1 (Orus):** PicBasic Pro simplifica enormemente el manejo de interrupciones mediante la declaración ON INTERRUPT GOTO seguida del nombre de la etiqueta manejadora, ubicada al inicio del código antes de habilitar el registro I N T C O N.
- **Turno 2 (Aoede):** Al escribir la rutina manejadora al final del archivo debemos encerrarla rigurosamente entre la directiva DISABLE justo arriba de su etiqueta y la directiva ENABLE inmediatamente después de la instrucción de cierre RESUME.
- **Turno 3 (Orus):** ¿Por qué es indispensable colocar DISABLE antes de la rutina de servicio? Porque impide que el compilador siga insertando verificaciones de interrupción dentro del propio manejador, lo que provocaría un bucle recursivo fatal.
- **Turno 4 (Aoede):** Dentro del bloque manejador ejecutamos la acción rápida requerida, limpiamos manualmente la bandera correspondiente, como I N T C O N punto uno igual a cero para el pin erre be cero, y terminamos con RESUME.

Diapositiva 09: Ejemplo Completo: Parada de Emergencia en RB0 y Reloj Base con TMR0

Selección de Flanco con OPTION_REG.6, Habilitación INTCON = %10010000 y Preescalador Temporizado

- **Turno 1 (Orus):** Implementemos un caso industrial real: un actuador opera en el lazo principal mientras un sensor óptico de seguridad conectado al pin erre be cero debe cortar la potencia al instante ante un flanco de bajada.
- **Turno 2 (Aoede):** Primero seleccionamos el flanco de bajada poniendo en cero el sexto bit del registro OPTION; después declaramos ON INTERRUPT GOTO ParadaSegura y cargamos I N T C O N con binario uno cero cero uno cero cero cero cero.
- **Turno 3 (Orus):** Ese byte activa únicamente el interruptor maestro global en el bit siete y el permiso específico del pin erre be cero en el bit cuatro, dejando limpias todas las banderas inferiores.
- **Turno 4 (Aoede):** De manera análoga, si habilitamos el quinto bit de I N T C O N con el preescalador asignado al temporizador cero, cada desbordamiento incrementa un contador en segundo plano para llevar el tiempo exacto sin detener el proceso.

Diapositiva 10: Latencia de ON INTERRUPT frente a Interrupciones en Ensamblador

Espera al Fin del Comando Básico en Curso frente a Respuesta de 3 Microsegundos con Salvado de W y STATUS

- **Turno 1 (Orus):** Todo ingeniero debe conocer la diferencia de latencia en PicBasic Pro: ON INTERRUPT no interrumpe a mitad de una instrucción compuesta para no corromper variables internas del compilador, sino que revisa la bandera al terminar cada comando básico.
- **Turno 2 (Aoede):** Esto significa que si nuestro lazo principal contiene un comando largo como PAUSE mil, ON INTERRUPT esperará hasta terminar ese segundo; la solución práctica es sustituir pausas largas por un bucle corto de mil iteraciones de PAUSE uno.
- **Turno 3 (Orus):** Y cuando un sistema mecatrónico exige latencia dura garantizada de apenas tres o cuatro microsegundos, PicBasic Pro permite declarar DEFINE INTHAND y escribir una breve rutina en ensamblador puro entre bloques A S M y ENDASM.
- **Turno 4 (Aoede):** Con el dominio completo del lenguaje y de las interrupciones, pasemos a la Parte cinco para interconectar los dos periféricos de usuario más universales: el teclado numérico matricial y la pantalla L C D.

2. Código Fuente en PicBasic Pro (picbasic_pro_video_04.pbp)

```
' ================================================================================
' CURSO: PROGRAMACION E INTERFAZ DE MICROCONTROLADORES PIC (cursosmecatronica.com)
' Video 04 – Parte 4: Ejemplos Prácticos en PicBasic Pro y Uso de Interrupciones
' Enfoque: Secuenciador LED, BUTTON, SOUND, PWM + Filtro RC, PULSOUT/PULSIN (10 us @ 4 MHz), SEROUT/SERIN,
INTCON y ON INTERRUPT GOTO
' Ecuaciones Clave: t_servo = N_pulsout · 10 us  |  d_cm = (t_eco_us · 0.0343) / 2  |  V_avg = (Duty / 255) ·
5.0 V  |  INTCON = %10010000
' 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, ["V04 Ciclo=", #ContadorCiclos, " Est=", #Estado, 13, 10]
    PAUSE 100
    GOTO LazoPrincipal
    END
```

3. Referencia en Ensamblador MPASM (ensamblador_mpasm_video_04.asm)

```
; ==========================================================================
; CURSO: PROGRAMACION E INTERFAZ DE MICROCONTROLADORES PIC (cursosmecatronica.com)
; Video 04 – Parte 4: Ejemplos Prácticos en PicBasic Pro y Uso de Interrupciones – 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_04.c)

```
/* ================================================================================
 * CURSO: PROGRAMACION E INTERFAZ DE MICROCONTROLADORES PIC (cursosmecatronica.com)
 * Video 04 – Parte 4: Ejemplos Prácticos en PicBasic Pro y Uso de Interrupciones – 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);
    }
}
```