Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Electrónica

Microcontroladores

Materia

Práctica No. 5

Camarillo Bautista Alfredo, González Escalona Miguel Ángel, Lázaro Bonilla Ramiro, López Arce Roberto

Integrantes del equipo

Ingeniería en mecatrónica

Carrera

Ricardo Álvarez González

Docente

04 de mayo de 2021, Primavera 2021

Fecha de entrega

Contenido

Objetivo

Marco teórico

El pic 18f4550 es un microcontrolador de 8 bits de la empresa Microchip. Este microcontrolador cuenta con una gran cantidad de memoria RAM, diferentes módulos de comunicación, una gran cantidad de pines de entrada y salida y algunas otras grandes cualidades. Algunas de sus características son:

- 40 pines tipo DIP
- Interfaz USB 2.0 de alta velocidad, EEPROM 256 bytes
- Memoria RAM 2048 bytes, EEPROM 256 bytes
- Memoria de programa (memoria flash) 32 kb
- Voltaje de operación 2 a 5.5 V
- Frecuencia máxima 48 MHz
- 35 pines de entrada / salida

Las instrucciones utilizadas en la práctica se enlistarán y explicarán a continuación.

Directivas utilizadas

<label> EQU <value> (equate): La directiva de ensamblador ***EQU*** simplemente equipara un nombre simbólico a un valor numérico.

ORG <value> (origin): La directiva ***ORIGIN*** le dice al ensamblador donde cargar instrucciones y datos dentro de la memoria.

CBLOCK [expr]

Label [:increment] [,label [:increment]]

Endc: Define una lista de símbolos secuenciales nombrados. La lista de nombres termina cuando la directiva *endc* es encontrada.

Para dispositivos **PIC18**, sólo números pares en *expr* son permitidos.

Display de 7 segmentos

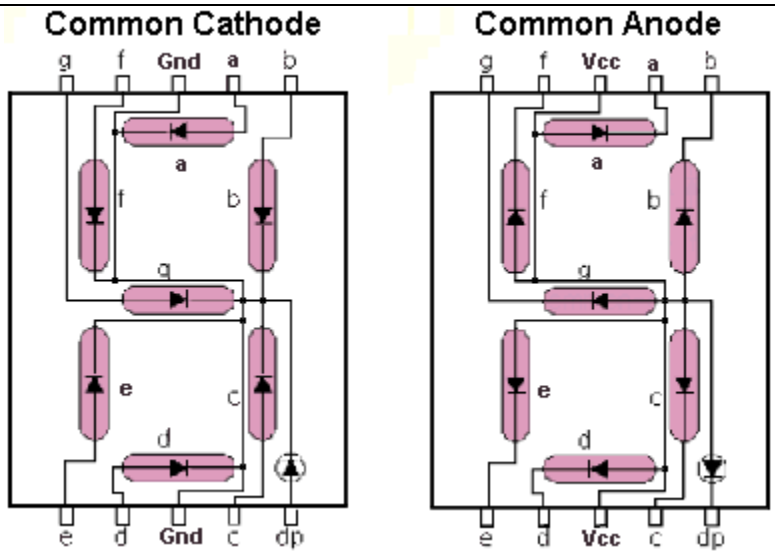


Imagen 1. Display de 7 segmentos de cátodo común (izquierda) y ánodo común (derecha).

El display de 7 segmentos es un dispositivo opto – electrónico que permite visualizar números del 0 al 9. Existen dos tipos de display, de cátodo común y de ánodo común. Especificaciones:

- Voltaje: 3 VCD
- Amperaje: 10 mA
- Número de segmentos: 7
- Cátodo común
- Color del LED: Rojo
- Posiciones de los pines con respecto al punto: Vertical
- Dimensiones: 1.8 cm x 0.9 cm x 0.4 cm

Un display de este tipo está compuesto por siete u ocho leds de diferentes formas especiales y dispuestos sobre una base de manera que puedan representarse todos los símbolos numéricos y algunas letras. Los primeros siete segmentos son los encargados de formar el símbolo y con el octavo podemos encender y apagar el punto decimal. Cada uno de los segmentos que forman la pantalla están marcados con siete primeras letras del alfabeto (‘a’ – ‘g’).

En los tipos de ánodo común, todos los ánodos de los segmentos están unidos internamente a una patilla común que debe ser conectada a potencial positivo (nivel ‘1’). El encendido de cada segmento individual se realiza aplicando potencial negativo (nivel ‘0’) por la patilla correspondiente a través de una resistencia que límite el paso de la corriente.

En los de tipo de cátodo común, todos los cátodos de los segmentos están unidos internamente a una patilla común que debe ser conectada a potencial negativo (nivel ‘0’). El encendido de cada segmento individual se realiza aplicando potencial positivo (nivel ‘1’) por la patilla correspondiente a través de una resistencia que limite el paso de la corriente.

Interrupciones externas

Las interrupciones externas en los pins RB0/AN12/INT0/FLT0/SDI/SDA, RB1/AN10/INT1/SCK/SCL y RB2/AN8/INT2/VMO son activados por flanco. Si el bit correspondiente INTEDGx en el registro INTCON2 es puesto (=1), la interrupción es activa en un flanco; si el bit está limpio, entonces la interrupción se activa en un flanco de bajada.

Cuando un flanco válido aparece en el pin RBx/INTx, el bit de bandera correspondiente, INTxIF, es puesto. Esta interrupción puede deshabilitar limpiando el bit de activación correspondiente, INTxIE. El bit de bandera, INTxIF, debe ser limpiado en software en la rutina de servicio de interrupción antes de habilitar de nuevo la interrupción.

Todas las interrupciones externas (INT0, INT1 e INT2) pueden "despertar" al procesador del modo de manejo de energía si el bit INTxIE fue establecido para configurar el manejo de energía. Si el bit de habilitación de interrupción global, GIE, este puesto, el procesador se ramificará al vector de interrupción después de "despertarse".

La prioridad de interrupción de INT1 e INT2 está determinada por el valor contenido en los bits de interrupción de prioridad INT1IP (INTCON3<6>) e INT2IP (INTCON3<7>). No hay algún bit de prioridad asociado a INT0, siempre es una fuente de interrupción de alta prioridad.

Bits de control para la configuración del puerto A/D

PCFG3:PCFG0: A/D Port Configuration Control bits:

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7(2)	AN6(2)	AN5(2)	AN4	AN3	AN2	AN1	AN0
0000(1)	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111(1)	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input D = Digital I/O

Imagen 2. Bits de control para la configuración del puerto A/D.

Configuración del puerto D como salida

	7	6	5	4	3	2	1	0
TRISD	0	0	0	0	0	0	0	0
PORTD	Salida	Salida	Salida	Salida	Salida	Salida	Salida	Salida

Timer 0

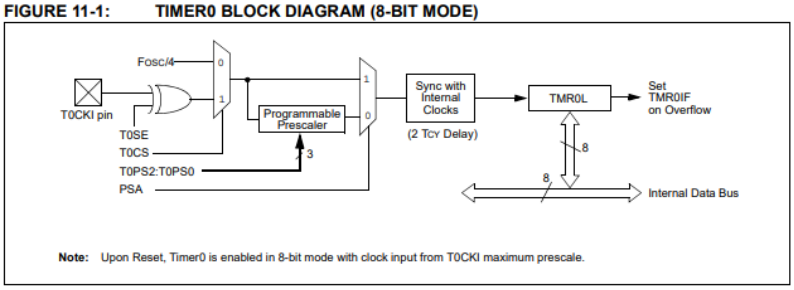
El módulo del Timer0 incorpora las siguientes características:

- Operación seleccionable por software como temporizador o contador, ambos en modo de 8 bits o 16 bits.
- Registros para lectura y escritura.
- Prescaler programable por software de 8 bits.
- Fuente seleccionable de reloj (interna o externa).
- Selección de borde para reloj externo.
- Interrupción en desbordamiento.

El registro T0CON controla todos los aspectos de las operaciones del módulo, incluyendo la selección de prescala. En ambos sirve de lectura y escritura.

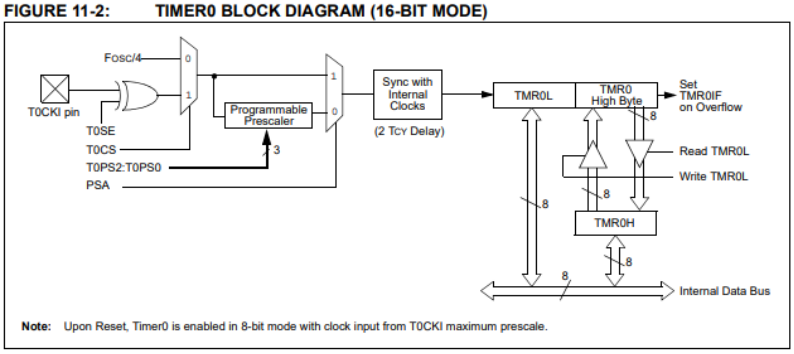
Un diagrama de bloques simplificado del módulo del timer0 en su modalidad de 8 bits y 16 bits se muestra en las imágenes siguientes.

FIGURE 11-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI maximum prescale.

FIGURE 11-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI maximum prescale.

Imagen 3. Timer0 en su modalidad de 8 bits (Figure 11-1) y en su modalidad de 16 bits (Figure 11-2).

T0: Control de registro del Timer0

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Imagen 4. Funcionalidad de los bits del registro T0CON

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-Bit/16-Bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0**: Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

Imagen 5. Función de acuerdo con los estados de los bits del registro T0CON

Timer 2

El módulo del Timer2 incorpora las siguientes características:

- Temporizador de 8 bits y registros de periodo (TMR2 y PR2, respectivamente).
- Registros para lectura y escritura
- Prescaler programable por software (1:1, 1:4 y 1:16)
- Postscaler programable por software (1:1 hasta 1:16)
- Interrupción en TMR2 hasta PR2
- Uso opcional como reloj de cambio para el módulo MSSP

El módulo es controlado a través del registro T2CON el cuál habilita y deshabilita el temporizador y configura el prescaler y postscaler. El timer2 puede ser apagado limpiando el bit de control, TMR2ON (T2CON<2>), para minimizar el consumo de poder.

Un diagrama de bloques simplificado del módulo del timer2 se muestra en la figura 13-1.

FIGURE 13-1: TIMER2 BLOCK DIAGRAM

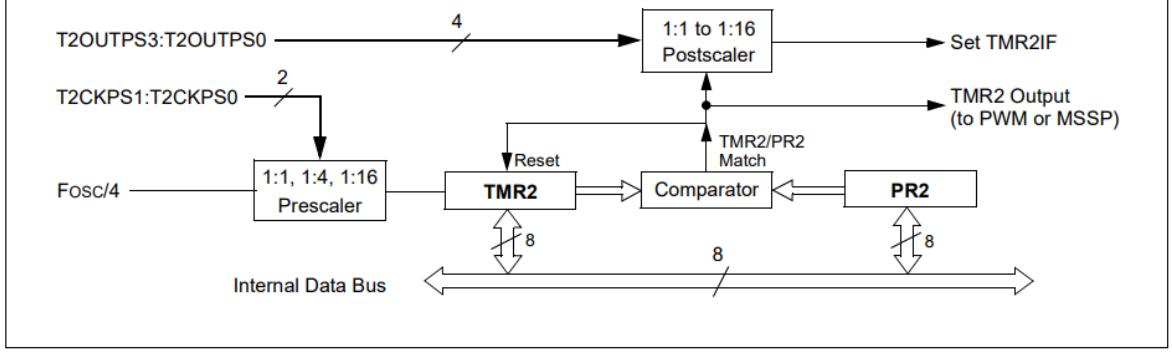


Imagen 6. Diagrama de bloques del Timer2

REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-3 **T2OUTPS3:T2OUTPS0**: Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON**: Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS1:T2CKPS0**: Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

Imagen 7. Registros de control del timer2

TABLE POINTER REGISTER (TBLPTR) y TABLE LATCH REGISTER (TABLAT)

El Table Latch (TABLAT) es un registro de 8 bits mapeado en el espacio de los SFR (Special Function Registers). El registro TABLAT es usado para guardar datos de 8 bits durante transferencia de datos entre la memoria de programa y los datos de la RAM.

El registro Table Pointer (TBLPTR) direcciona un byte dentro de la memoria de programa. El TBLPTR está compuesto de tres SFR: Table Pointer Upper Byte, Table Pointer High Byte y Table Pointer Low Byte (TBLPTRU:TBLPTRH:TBLPTRL). Estos tres registros se unen para formar un apuntador de 22 bits. El orden bajo de 21 bits permite al dispositivo direccionar hasta 2 MB de espacio de memoria de programa. El bit 22 permite el acceso al ID del dispositivo, del usuario y la configuración de bits. El puntero de tabla, TBLPTR, es usado por las instrucciones TBLRD y TBLWT. Estas instrucciones pueden actualizar el TBLPTR en una de cuatro maneras de acuerdo con la tabla de operaciones. Estas operaciones se muestran en la tabla 6 – 1. Estas operaciones en el registro TBLPTR solamente afectan el orden bajo de 21 bits.

TABLE 6-1: TABLE POINTER OPERATIONS WITH TBLRD AND TBLWT INSTRUCTIONS

Example	Operation on Table Pointer
TBLRD* TBLWT*	TBLPTR is not modified
TBLRD*+ TBLWT*+	TBLPTR is incremented after the read/write
TBLRD*- TBLWT*-	TBLPTR is decremented after the read/write
TBLRD+* TBLWT+*	TBLPTR is incremented before the read/write

FIGURE 6-3: TABLE POINTER BOUNDARIES BASED ON OPERATION

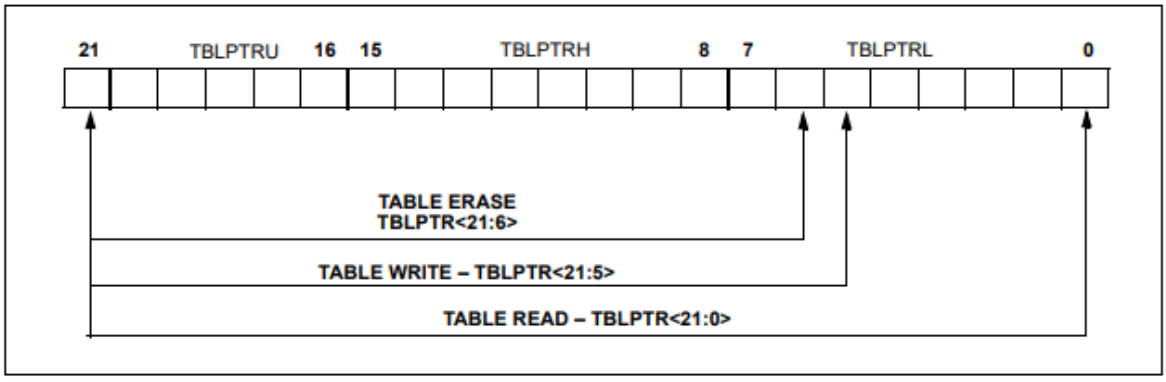


Imagen 8. Operaciones de apuntador de tabla (tabla 6 – 1) y límites de puntero de tabla basados en la operación (figura 6 – 3).

El registro TBLPTR es usado en lectura, escritura y borrado de la memoria flash del programa.

Cuando un TBLRD es ejecutado, todos los 22 bits del TBLPTR determinan cual byte es leído de la memoria del programa dentro de TABLAT.

Desarrollo práctico

Para la práctica realizada se utilizaron los siguientes materiales a enlistar.

Cantidad	Concepto
1	Tarjeta de desarrollo Miuva
1	Laptop
4	Display de 7 segmentos ánodo común
1	Software MPLAB v8.92
1	Datasheet del PIC 18F4550
7	Resistencia de 220
1	Resistencia de 330
3	Resistencia de 10k
4	Resistencia de 1k
4	Transistor BC547
2	Protoboard
1	Diodo LED
3	Pushbutton
20	Jumpers
1	Mt de alambre calibre 22

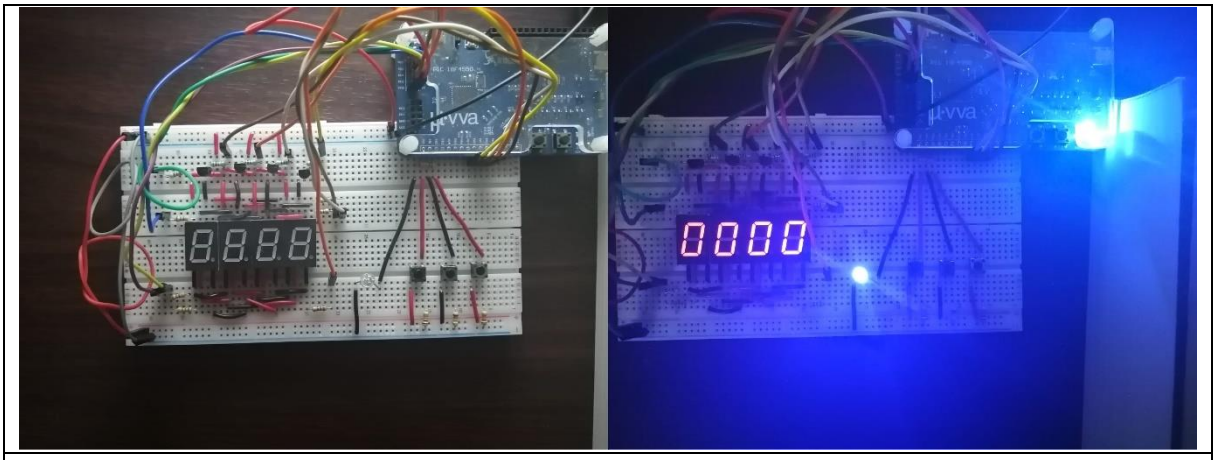


Imagen 9. Implementación del circuito y funcionamiento de este en la tarjeta de desarrollo Miuva

Explicación del código

```
        LIST P=18F4550      ;directiva para definir el procesador
        #include <P18F4550.INC> ;definiciones de variables especificas del procesador

;*****************************************************************************
;Bits de configuración
            CONFIG FOSC = INTOSC_XT ;Oscilador interno para el uC, XT para USB
            CONFIG BOR = OFF        ;Brownout reset deshabilitado
            CONFIG PWRT = ON        ;Pwr up timer habilitado
            CONFIG WDT = OFF        ;Temporizador vigia apagado
            CONFIG MCLRE = OFF      ;Reset apagado
            CONFIG PBADEN = OFF
            CONFIG LVP = OFF
;*****************************************************************************
;Definiciones de variables
        cblock      0x0 ;ejemplo de definición de variables en RAM de acceso
        flags           ;banderas
        iumin           ;índice de unidades de minuto
        cumin           ;código de 7 segmentos de unidades de minuto
        idmin           ;índice de decenas de minuto
        cdmin           ;código de 7 segmentos de decenas de minuto
        iuhr            ;índice de unidades de hora
        cuhr            ;código de 7 segmentos de unidades de hora
        idhr            ;índice de decenas de hora
        cdhr            ;código de 7 segmentos de decenas de hora
        scnds
        cont
        endc            ;fin de bloque de constantes
;Todo lo anterior son palabras de configuración
;*****************************************************************************asdasd
;Reset vector
        ORG 0x0000
        bra     inicio
        org     0x08
        bra     RSI
```

Imagen 10. Código Pt 1

En la primera parte del código se realizan las declaraciones necesarias para que el PIC funcione, además de que se declaran variables y la ubicación de las subrutinas.

```
;Inicio del programa principal
inicio  bsf     OSCCON, IRCF2, 0
        bsf     OSCCON, IRCF1, 0
        bcf     OSCCON, IRCF0, 0        ; 110, Oscilador interno a 4 MHz
        movlw   0x0F
        movwf   ADCON1, 0               ; Puertos digitales
        clrf    PORTD, 0
        clrf    TRISD, 0                ; Puerto D configurado como salida
        clrf    TRISA, 0                ; Puerto A configurado como salida
        bcf     TRISB, 3                ; RB3 salida
        bcf     TRISB, 4                ; RB4 salida
        movlw   0xF0                    ; 1111 0000
        movwf   INTCON, 0               ; Interrupciones externa INT0, TMR0 y PEIE

; Habilitando INT1 e INT2
        BCF     INTCON3, INT1IF         ; INT1IF: 0
        BSF     INTCON3, INT1IE         ; ENABLE INT1
        BCF     INTCON3, INT2IF         ; INT1IF: 0
        BSF     INTCON3, INT2IE         ; ENABLE INT1
        BSF     INTCON, GIE             ; ENABLE GIE

        movlw   0x95
        movwf   T0CON                   ; Timer 16 bits, preescaler *64
        movlw   0xE1
        movwf   TMR0H, 0
        movlw   0x7C
        movwf   TMR0L, 0                ; Valor de precarga para 1000ms a 4 MHz, preescaler 64 del timer0

        clrf    TBLPTRL, 0
        movlw   0x03
        movwf   TBLPTRH, 0              ; El apuntador TBLPTRH se pone en dirección 0x03
        clrf    TBLPTRU, 0              ; tblptr = 0x000300
        movlw   0x05
        movwf   PR2                     ; Timer2 period register
        movlw   0x07
        movwf   T2CON                   ;
        bsf     PIE1, TMR2IE            ; Interrupción timer2 activa

        clrf    iumin, 0
        clrf    idmin, 0                ; Iniciamos en cero
        clrf    iuhr, 0
        clrf    idhr, 0

        movlw   d'60'                   ; Se mueve el valor decimal "60" al Working Register
        movwf   scnds                   ; Se mueve el valor previamente cargado al WR hacia el registro "scnds"
```

Imagen 11. Código Pt 2

En esta parte del código, se realizan las configuraciones necesarias en los puertos para poder utilizar las interrupciones externas, además de establecer el apuntador en cierta dirección para poder empezar a contar desde ahí. Además, se inician desde cero los valores de los displays.

```
lee     movff   iumin, TBLPTRL          ; ajusta apuntador (unidades)
        tblrd   *                       ; lee la tabla sin modificar apuntador
        movff   TABLAT, cumin           ; Cuni tiene código 7 segmentos

        movff   idmin, TBLPTRL          ; ajusta apuntador (decimales)
        tblrd   *                       ; Lee la tabla sin modificar apuntador
        movff   TABLAT, cdmin           ; cdec tiene código 7 segmentos

        movff   iuhr, TBLPTRL           ; ajusta apuntador (centenas)
        tblrd   *                       ; lee la tabla sin modificar apuntador
        movff   TABLAT, cuhr            ; Ccen tiene código 7 segmentos

        movff   idhr, TBLPTRL
        tblrd   *
        movff   TABLAT, cdhr
; Inicio del programa principal
loop    movlw   0x01
        movwf   PORTA,0                 ; encendemos display unidades
        movff   cumin, PORTD              ; Mueve el valor del código de las unidades al puerto D (Codigo UNIdades)
        call    delay                   ; Timer2 establecerá el tiempo que esté encendido el display

        movlw   0x02
        movwf   PORTA, 0                ; encendemos display decenas
        movff   cdmin, PORTD              ; Mueve el valor del código de las decenas (Codigo DECenas)
        call    delay                   ; Timer2 establecerá el tiempo que esté encendido el display

        movlw   0x04
        movwf   PORTA, 0                ; encendemos display centenas
        movff   cuhr, PORTD             ;Mueve el valor del código de las centenas al puerto D (Código CENtenas)
        call    delay

        movlw   0X08                    ; encendemos display unidades de milar
        movwf   PORTA, 0
        movff   cdhr, PORTD
        call    delay
        btfss   flags, 0, 0
        bra     loop
```

Imagen 12. Código Pt 3

En la parte 3 del código se ubica la subrutina *lee*, que se encargará de ubicar los punteros de los índices de los minutos y las horas y de pasar estos valores a las variables de los códigos, mientras que la subrutina *loop* es la encargada de se encarga de encender los displays y de mover el valor de las variables de código al puerto D, además de seleccionar qué display se encenderá.

```
        bcf     flags, 0, 0             ; ¿Ya transcurrieron 500 ms?

        incf    iumin, F, 0             ; se incrementa el valor de la
        movf    iumin, W, 0             ; se mueve el valor de iumin a
        xorlw   0x0a                    ; se compara con el valor d'10
        btfss   STATUS, Z, 0            ; verifica límite de tabla (si
        bra     lee                 ; en caso de que los minutos llega
        clrf    iumin, 0

        incf    idmin, F, 0
        movf    idmin, W, 0
        xorlw   0x06
        btfss   STATUS, Z, 0
        bra     lee
        clrf    idmin, 0
```

Imagen 13. Código pt 3

En esta parte del código se realizan los incrementos de los minutos. El hecho de que sea corto ese segmento de código es que los límites de los minutos solamente se rigen por un parámetro (llegar hasta 59), las condiciones estarán en el incremento de las horas.

```
;Subrutina donde se incrementan las unidades de hora desde 00:59-19:59
            incf    iuhr, F, 0
            movf    iuhr, W, 0
            bra     check2                  ; hará un salgo para evaluar
bfrtwnty2   movf    iuhr, W, 0
            xorlw   0x0a
            btfss   STATUS, Z, 0
            bra     lee
            clrf    iuhr, 0

            incf    idhr, F, 0
            movf    idhr, W, 0
            xorlw   0x02
            btfss   STATUS, Z, 0
            bra     lee
            bra     ftrtwnty2
;Subrutina donde se incrementan las unidades de hora desde 20:00-23:59

check2  movf    idhr, W, 0              ; en esta subrutina se evalua en
        xorlw   0x02
        btfss   STATUS, Z, 0
        bra     bfrtwnty2
        movf    iuhr, W, 0
        xorlw   0x04
        btfss   STATUS, Z, 0
        bra     lee


ftrtwnty2   movf    iuhr, W, 0
            xorlw   0x04
            btfss   STATUS, Z, 0
            bra     lee
            clrf    iuhr
            clrf    idhr
            goto    lee
```

Imagen 14. Código Pt 4

En este segmento de código la condición radica en que primero se evaluarán en que decena de hora se encuentra el reloj por medio de la subrutina *Check2*. Si el reloj se encuentra en la segunda decena de las horas, entonces se hará una segunda evaluación para ver si las unidades de hora son 4 (es decir, si son 24 horas). Si ya se llegó a las 24 horas, entonces se seguirá a la subrutina *ftrtwnty2*, de lo contrario, se regresará a la subrutina *lee*. Si son antes de las 20 horas, entonces se saltará a la subrutina *bfrtwnty2*, donde las unidades de hora podrán llegar a 9. Estos mismos parámetros se aplican para la interrupción 0, que es la que modifica el valor de las unidades de minuto, por lo que se omitirá la explicación de esta subrutina.

Para las otras dos interrupciones, es un código más corto puesto que solo se modificarán las horas.

```
;xxxxxxxxxxXXXXXXXXXXXXXXXXXXXXxxxxxxxxx INTERRUPCION HORAS xxxxxxxxxXXXXXXXXXXXXXXXXxxxxxxxxxxxxxxXXXXXXXXXXXXXXX
SINT1   btfss   INTCON3, INT1IF
        bra     SINT2
        bcf     INTCON3, INT1IF         ; apaga bit de bandera

        movf    idhr, W, 0
        xorlw   0x02
        btfss   STATUS, Z, 0
        bra     ztonntn
        bra     ttotwnyfr

ztonntn     incf    iuhr, F, 0
            movf    iuhr, W, 0
            xorlw   0x0a
            btfss   STATUS, Z, 0
            bra     lee3
            clrf    iuhr, 0

            incf    idhr, F, 0
            movf    idhr, W, 0
            bra     lee3

ttotwnyfr   incf    iuhr, F, 0
            movf    iuhr, W, 0
            xorlw   0x04
            btfss   STATUS, Z, 0
            bra     lee3
            clrf    iuhr, 0
            clrf    idhr, 0


lee3    movff   iuhr, TBLPTRL           ; ajusta apuntador
        tblrd   *
        movff   TABLAT, cuhr

        movff   idhr, TBLPTRL           ; ajusta apuntador
        tblrd   *
        movff   TABLAT, cdhr
        retfie
```

Imagen 15. Código pt 5

En la parte 5 del código se muestra la interrupción externa 1, que es la encargada de modificar el valor de las unidades de hora. Primero se evalúa en qué decena de hora se encuentra y con base en eso se decide si avanzar hasta 9 en las unidades de hora o hasta 3.

```
SINT2   btfss   INTCON3, INT2IF
        bra     ST2
        bcf     INTCON3, INT2IF         ; apaga bit de bandera

        clrf    iumin, 0
        clrf    idmin, 0
        clrf    iuhr, 0
        clrf    idhr, 0

lee4    movff   iuhr, TBLPTRL           ; ajusta apuntador
        tblrd   *
        movff   TABLAT, cuhr

        movff   idhr, TBLPTRL           ; ajusta apuntador
        tblrd   *
        movff   TABLAT, cdhr

        movff   iumin, TBLPTRL          ; ajusta apuntador
        tblrd   *
        movff   TABLAT, cumin

        movff   idmin, TBLPTRL          ; ajusta apuntador
        tblrd   *
        movff   TABLAT, cdmin
        retfie

ST2     bcf     PIR1, TMR2IF            ;Peripheral Interrupts Request 1 & Timer2 Interrupt flag bit
        bsf     flags, 2                ; Bandera monitor del timer2
        retfie
;*******************************************************************************
delay   btfss   flags, 2
        bra     delay
        bcf     flags, 2
        return

;*******************************************************************************
        org     0x300                   ; DB Directiva que Define Byte
        DB      0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xB8, 0x80, 0x98, 0x88, 0x83, 0xc6, 0xa1, 0x86, 0x8e
        END
```

Imagen 16. Código pt 6

En esta última parte del código, se encuentra la interrupción de INT2, la cual reinicia los displays a cero, además de que se encuentra la interrupción *ST2* y la interrupción *delay*. Más abajo se encuentra la directiva que define los bytes (que son los códigos de los números).

Código completo escrito en ASM

```
        LIST P=18F4550          ;directiva para definir el procesador
                #include <P18F4550.INC>        ;definiciones de variables especificas del
procesador

;*******************************************************************************
;Bits de configuración
                        CONFIG FOSC = INTOSC_XT ;Oscilador interno para el uC, XT para USB
                        CONFIG BOR = OFF                ;Brownout reset deshabilitado
                        CONFIG PWRT = ON                ;Pwr up timer habilitado
                        CONFIG WDT = OFF                ;Temporizador vigia apagado
                        CONFIG MCLRE = OFF              ;Reset apagado
                        CONFIG PBADEN = OFF
                        CONFIG LVP = OFF
```

```
;*********************************************************************
;Definiciones de variables
                cblock          0x0     ;ejemplo de definición de variables en RAM de acceso
                flags                   ;banderas
                iumin                   ;índice de unidades      de minuto
                cumin                   ;código de 7 segmentos de unidades de minuto
                idmin                   ;índice de decenas de minuto
                cdmin                   ;código de 7 segmentos de decenas de minuto
                iuhr                    ;índice de unidades de hora
                cuhr                    ;código de 7 segmentos de unidades de hora
                idhr                    ;índice de decenas de hora
                cdhr                    ;código de 7 segmentos de decenas de hora
                scnds
                cont
                endc                    ;fin de bloque de constantes
;Todo lo anterior son palabras de configuración
;*********************************************************************asda
sd
;Reset vector
                ORG 0x0000
                bra     inicio
                org             0x08
                bra             RSI

;Inicio del programa principal
inicio  bsf             OSCCON, IRCF2, 0
                bsf     OSCCON, IRCF1, 0
                bcf             OSCCON, IRCF0, 0                ; 110, Oscilador interno a 4 MHz
                movlw   0x0F
                movwf   ADCON1, 0                               ; Puertos digitales
                clrf    PORTD, 0
                clrf    TRISD, 0                                ; Puerto D configurado como salida
                clrf    TRISA, 0                                ; Puerto A configurado como salida
                bcf     TRISB, 3                                ; RB3 salida
                bcf             TRISB, 4                                ; RB4 salida
                movlw   0xF0                                    ; 1111 0000
                movwf   INTCON, 0                               ; Interrupciones externa INT0,
TMR0 y PEIE

; Habilitando INT1 e INT2
                BCF             INTCON3, INT1IF                 ; INT1IF: 0
                BSF             INTCON3, INT1IE                 ; ENABLE INT1
                BCF             INTCON3, INT2IF                 ; INT1IF: 0
                BSF             INTCON3, INT2IE                 ; ENABLE INT1
                BSF             INTCON, GIE                     ; ENABLE GIE


                movlw   0x95
```

```
                movwf  T0CON                            ; Timer 16 bits, preescaler *64
                movlw  0xE1
                movwf  TMR0H, 0
                movlw  0x7C
                movwf  TMR0L, 0                         ; Valor de precarga para 1000ms a
4 MHz, preescaler 64 del timer0

                clrf   TBLPTRL, 0
                movlw  0x03
                movwf  TBLPTRH, 0                       ; El apuntador TBLPTRH se pone en
dirección 0x03
                clrf   TBLPTRU, 0                       ; tblptr = 0x000300
                movlw  0x05
                movwf  PR2                                      ; Timer2 period register
                movlw  0x07
                movwf  T2CON                            ;
                bsf             PIE1, TMR2IE            ; Interrupción timer2 activa

                clrf   iumin, 0
                clrf   idmin, 0                         ; Iniciamos en cero
                clrf   iuhr, 0
                clrf   idhr, 0

                movlw  d'60'                            ; Se mueve el valor decimal "60" al
Working Register
                movwf  scnds                            ; Se mueve el valor previamente
cargado al WR hacia el registro "scnds"
; Subrutina Lee donde se pasa el valor de "iun" al apuntador "TBLPTRL" y se lee su valor
; Se mueve el valor de donde el apuntador apuntó (donde ahora está TABLAT) al código de las
unidades y se repite el proceso
; con los decimales (Lo que hace la sr lee es que va a conseguir los códigos para mostrar los
números en los displays)

lee             movff  iumin, TBLPTRL           ; ajusta apuntador (unidades)
                tblrd  *                                ; lee la tabla sin modificar
apuntador
                movff  TABLAT, cumin            ; Cuni tiene código 7 segmentos

                movff  idmin, TBLPTRL           ; ajusta apuntador (decimales)
                tblrd  *                                ; Lee la tabla sin modificar
apuntador
                movff  TABLAT, cdmin            ; cdec tiene código 7 segmentos

                movff  iuhr, TBLPTRL            ; ajusta apuntador (centenas)
                tblrd  *                                ; lee la tabla sin modificar
apuntador
                movff  TABLAT, cuhr             ; Ccen tiene código 7 segmentos
```

```
                movff   idhr, TBLPTRL
                tblrd   *
                movff   TABLAT, cdhr
; Inicio del programa principal
loop    movlw  0x01
                movwf  PORTA,0                                  ; encendemos display
unidades
                movff   cumin, PORTD                    ; Mueve el valor del código de las
unidades al puerto D (Codigo UNIdades)
                call    delay                           ; Timer2 establecerá el tiempo que
esté encendido el display

                movlw  0x02
                movwf  PORTA, 0                         ; encendemos display decenas
                movff   cdmin, PORTD                    ; Mueve el valor del código de las
decenas (Codigo DECenas)
                call    delay                           ; Timer2 establecerá el tiempo que
esté encendido el display

                movlw  0x04
                movwf  PORTA, 0                         ; encendemos display centenas
                movff   cuhr, PORTD                     ;Mueve el valor del código de las
centenas al puerto D (Código CENtenas)
                call    delay

                movlw  0X08                             ; encendemos display unidades de
milar
                movwf  PORTA, 0
                movff   cdhr, PORTD
                call    delay
                btfss   flags, 0, 0
                bra             loop
; Programa principal de loop hasta acá jejeps
; INCREMENTOS XXX------XXX INCREMENTOS XXX------XXX INCREMENTOS XXX------XXX
INCREMENTOS XXXXXXXXXXXXXXXXXXXXXXXXXXXX
                bcf             flags, 0, 0                     ; ¿Ya transcurrieron 500
ms?

                incf    iumin, F, 0                     ; se incrementa el valor de las
unidades de minuto
                movf    iumin, W, 0                     ; se mueve el valor de iumin a W
                xorlw   0x0a                            ; se compara con el valor d'10'
                btfss   STATUS, Z, 0                ; verifica límite de tabla (si se hace 0 con la
operación quiere decir que llegó al límite que pusimos)
                bra             lee                             ; en caso de que los
minutos llegasen a 10, limpia el índice de unidades de minutos con la siguiente instrucción, de lo
contrario regresa a la subrutina lee2 para seguir aumentando minutos
                clrf    iumin, 0
```

```
                incf    idmin, F, 0
                movf    idmin, W, 0
                xorlw   0x06
                btfss   STATUS, Z, 0
                bra             lee
                clrf    idmin, 0
;Subrutina donde se incrementan las unidades de hora desde 00:59-19:59
                        incf    iuhr, F, 0
                        movf    iuhr, W, 0
                        bra             check2                          ; hará un salgo para
evaluar. Si está en 20 horas el límite será 4, de lo contrario será 9
bfrtwnty2       movf    iuhr, W, 0
                        xorlw   0x0a
                        btfss   STATUS, Z, 0
                        bra             lee
                        clrf    iuhr, 0

                        incf    idhr, F, 0
                        movf    idhr, W, 0
                        xorlw   0x02
                        btfss   STATUS, Z, 0
                        bra             lee
                        bra             ftrtwnty2
;Subrutina donde se incrementan las unidades de hora desde 20:00-23:59

check2  movf    idhr, W, 0                              ; en esta subrutina se evalua en que
decena de hrs se encuentra el reloj, si se encuentra en 20 hará salto a una subrutina especial
donde las unidades lleguen a 4, de lo contrario irá a dónde las unidades lleguen a 9
                xorlw   0x02
                btfss   STATUS, Z, 0
                bra             bfrtwnty2
                movf    iuhr, W, 0
                xorlw   0x04
                btfss   STATUS, Z, 0
                bra             lee


ftrtwnty2       movf    iuhr, W, 0
                        xorlw   0x04
                        btfss   STATUS, Z, 0
                        bra             lee
                        clrf    iuhr
                        clrf    idhr
                        goto    lee

;*****RUTINA DE SERVICIO DE
INTERRUPCIÓN**********************************************
```

```
RSI             btfss   INTCON, TMR0IF
                bra             SINT0
                bcf             INTCON, TMR0IF
                movlw   0xC2
                movwf   TMR0H, 0
                movlw   0xF7
                movwf   TMR0L, 0                        ; valor de precarga para 1000 ms a
4 MHz preescaler 64
                btg             PORTB, 3                        ; Monitor de segundos
(cambia cada que transcurre 1 segundo)
                decfsz  scnds, 1                        ; le resta 1 a "scnds", si llega a cero
saltará la siguiente instrucción
                retfie
                movlw   d'60'
                movwf   scnds
                bsf             flags, 0                ; monitor interrupción del timer 0
                retfie

SINT0   btfss   INTCON, INT0IF
                bra             SINT1
                bcf             INTCON, INT0IF          ; apaga bit de bandera

                incf    iumin, F, 0                     ; se incrementa el valor de las
unidades de minuto
                movf    iumin, W, 0                     ; se mueve el valor de iumin a W
                xorlw   0x0a                            ; se compara con el valor d'10'
                btfss   STATUS, Z, 0            ; verifica límite de tabla (si se hace 0 con la
operación quiere decir que llegó al límite que pusimos)
                bra             lee2                            ; en caso de que los
minutos llegasen a 10, limpia el índice de unidades de minutos con la siguiente instrucción, de lo
contrario regresa a la subrutina lee2 para seguir aumentando minutos
                clrf    iumin, 0

                incf    idmin, F, 0
                movf    idmin, W, 0
                xorlw   0x06
                btfss   STATUS, Z, 0
                bra             lee2
                clrf    idmin, 0
;Subrutina donde se incrementan las unidades de hora desde 00:59-19:59
                        incf    iuhr, F, 0
                        movf    iuhr, W, 0
                        bra             check                           ; hará un salgo para
evaluar. Si está en 20 horas el límite será 4, de lo contrario será 9
bfrtwnty        movf    iuhr, W, 0
                        xorlw   0x0a
                        btfss   STATUS, Z, 0
                        bra             lee2
```

```
                clrf    iuhr, 0

                incf    idhr, F, 0
                movf    idhr, W, 0
                xorlw   0x02
                btfss   STATUS, Z, 0
                bra             lee2
                bra             ftrtwnty
;Subrutina donde se incrementan las unidades de hora desde 20:00-23:59

check   movf    idhr, W, 0                      ; en esta subrutina se evalua en que
decena de hrs se encuentra el reloj, si se encuentra en 20 hará salto a una subrutina especial
donde las unidades lleguen a 4, de lo contrario irá a dónde las unidades lleguen a 9
                xorlw   0x02
                btfss   STATUS, Z, 0
                bra             bfrtwnty
                ;incf   iuhr, F, 0
                movf    iuhr, W, 0
                xorlw   0x04
                btfss   STATUS, Z, 0
                bra             lee2


ftrtwnty        movf    iuhr, W, 0
                xorlw   0x04
                btfss   STATUS, Z, 0
                bra             lee2
                clrf    iuhr
                clrf    idhr



lee2    movff   iumin, TBLPTRL          ; ajusta apuntador
                tblrd   *                                       ; lee la tabla sin modificar
apuntador
                movff   TABLAT, cumin

                movff   idmin, TBLPTRL          ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cdmin

                movff   iuhr, TBLPTRL           ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cuhr

                movff   idhr, TBLPTRL           ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cdhr
```

```
                retfie
;xxxxxxxxxxXXXXXXXXXXXXXXXXXXxxxxxxxxx INTERRUPCION HORAS
xxxxxxxxxXXXXXXXXXXXXXXXxxxxxxxxxxxxxXXXXXXXXXXXXXX
SINT1   btfss   INTCON3, INT1IF
                bra             SINT2
                bcf             INTCON3, INT1IF                 ; apaga bit de bandera

                movf    idhr, W, 0
                xorlw   0x02
                btfss   STATUS, Z, 0
                bra             ztonntn
                bra             ttotwnyfr

ztonntn         incf    iuhr, F, 0
                        movf    iuhr, W, 0
                        xorlw   0x0a
                        btfss   STATUS, Z, 0
                        bra             lee3
                        clrf    iuhr, 0

                        incf    idhr, F, 0
                        movf    idhr, W, 0
                        bra             lee3

ttotwnyfr       incf    iuhr, F, 0
                        movf    iuhr, W, 0
                        xorlw   0x04
                        btfss   STATUS, Z, 0
                        bra             lee3
                        clrf    iuhr, 0
                        clrf    idhr, 0



lee3    movff   iuhr, TBLPTRL                   ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cuhr

                movff   idhr, TBLPTRL                   ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cdhr
                retfie

SINT2   btfss   INTCON3, INT2IF
                bra             ST2
                bcf             INTCON3, INT2IF                 ; apaga bit de bandera

                clrf    iumin, 0
```

```
                clrf    idmin, 0
                clrf    iuhr, 0
                clrf    idhr, 0

lee4    movff   iuhr, TBLPTRL                  ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cuhr

                movff   idhr, TBLPTRL                  ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cdhr

                movff   iumin, TBLPTRL                 ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cumin

                movff   idmin, TBLPTRL                 ; ajusta apuntador
                tblrd   *
                movff   TABLAT, cdmin
                retfie

ST2             bcf             PIR1, TMR2IF                    ;Peripheral Interrupts Request 1 &
Timer2 Interrupt flag bit
                bsf             flags, 2                        ; Bandera monitor del timer2
                retfie
;*********************************************************************************
*****
delay   btfss   flags, 2
                bra             delay
                bcf             flags, 2
                return

;*********************************************************************************
*****
                org             0x300                                   ; DB Directiva que Define
Byte
                DB              0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xB8, 0x80, 0x98, 0x88,
0x83, 0xc6, 0xa1, 0x86, 0x8e
                END
```

Conclusiones

Con la realización de esta práctica podemos tener un panorama más amplio de las aplicaciones que se le pueden dar al uso de interrupciones y de apuntadores, además de que se pudo hacer uso de todas las interrupciones.

Link del vídeo: https://youtu.be/TFscBrqM5Ms

Bibliografía

Microchip PIC18F Instruction Set. (2021). Retrieved 9 February 2021, from http://technology.niagarac.on.ca/staff/mboldin/18F_Instruction_Set/

(2021). Retrieved 23 February 2021, from http://ww1.microchip.com/downloads/en/devicedoc/33014j.pdf

Rafiquzzaman, M., 2007. *Microprocessors theory and applications*. 2nd ed. New Delhi, II: Prentice-Hall of India.