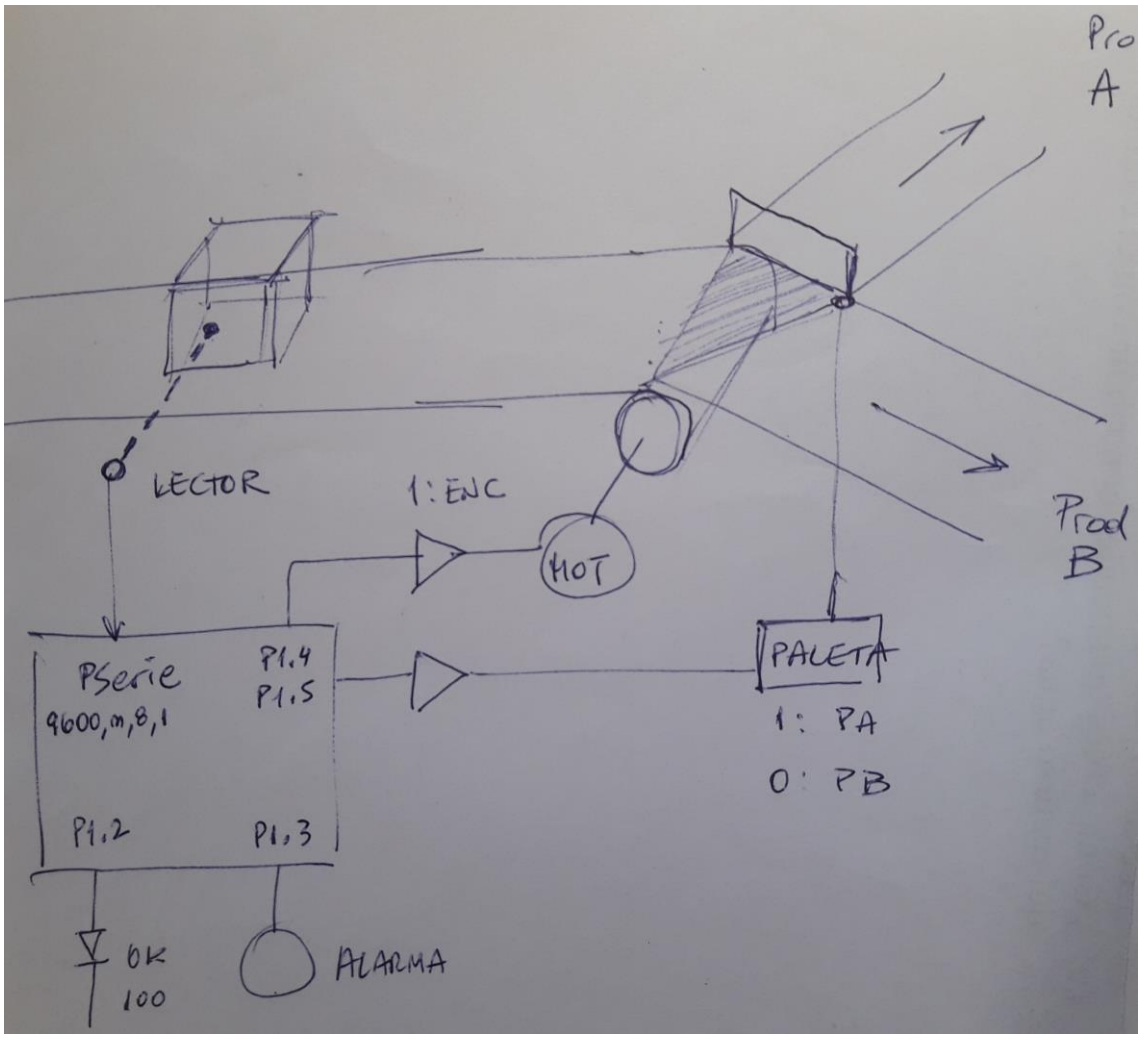
FIE

Examen parcial Técnicas Digitales II

Martes, 23 junio 2020
Profesor: Ing. Daniel Steiner
Alumno: Seery Brian Alfredo



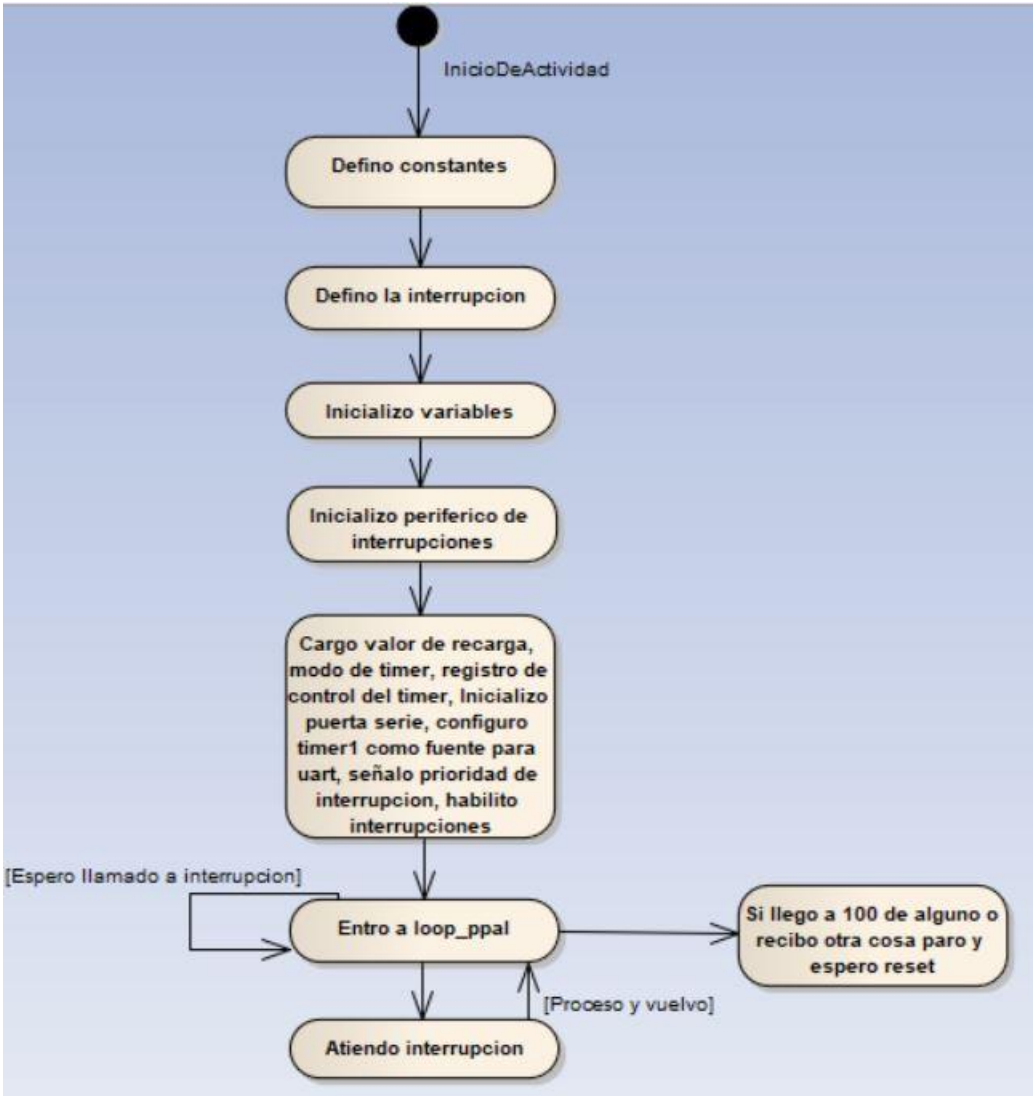
- El microcontrolador controla una cinta transportadora mediante un motor en P1.4, luego de RESET debe encender el motor
- Se dispone un sensor que emite por comunicación serie RS232 el código del producto que pasa: (0AH para producto A) (0BH para el producto B) y otros códigos
- Se dispone una paleta en P1.5 para desviar los productos hacia cajas de A o B (con 1 van hacia la caja de A, y con 0 van hacia la caja de B)
- **Realizar la codificación en Assembler para un 8052 con la siguiente función:**

Si llegan 100 productos de A, o 100 productos de B, detener la cinta y avisar con LED en P1.2

- Si llega un producto distinto al A o B, detener la cinta inmediatamente y dar alarma con P1.3 = 1

Restriccion: Usar interrupción serie para leer código desde el lector

Resolución



En base al diagrama que se plasma en el documento es como se planea dar solución al problema con respecto al código
Luego en base al témplate, se procede a realizar los cambios necesarios en las constantes para configurar las interrupciones, timers, puerta serie de acuerdo a las necesidades del problema planteado. Luego se define la interrupción de la puerta serie, se inicializan las variables que necesitare para prender la alarma, led cinta y paleta; y a continuación inicializo los periféricos de interrupción y cargo en los registros las variables declaras al inicio.
Una vez realizado estos pasos entro al loop_ppal que va a estar entrando y saliendo de la subrutina procesar datos hasta que se reciba algo por la puerta serie, una vez atendida esta interrupción en la subrutina se encuentra la lógica para operar sobre el dato.
El programa va a estar en un bucle hasta que pase una de las siguientes cosas:

- Se llegó a 100 productos de A o B → LED y paro cinta
- Se recibió algo que no sea 0AH o 0BH →ALARMA y paro cinta

En ese momento la cinta quedara detenida y esperando que se resetee.

A continuación el código, comentado para hacer más fácil de seguir la lógica del programa

```
;-------------------------------------------------------------------------------
; Fecha: 23/06/2020
;
; Programador: Seery Brian Alfredo
;
; Descripción: Recibo por puerto serie hasta que se cumplan 100 A y hago prender un
;LED. Tengo una paleta que distribuye para un lado u otro los productos A y B. Si llega
;algo que no es A ni B, tengo que parar la cinta y prender alarma
;-------------------------------------------------------------------------------

$bitdef.h       ;manejo de a bits
$ioAT89C52.h    ;incluye los registros del microcontrolador

        NAME    main
        PUBLIC  main

        ;--- ISRs ---
        PUBLIC  Isr_Serie
        ;--- Declaro públicas las subrutinas ---
        PUBLIC  InitPerif_Interrupciones        ;En esta subrutina voy a configurar las
interrupciones

;---CONSTANTES----------------------------------------------------------------

  ;Registros generales
CTE_SUMA              EQU     0x3E       ;Número cte para operar
LONG_BUFFER           EQU     10         ;Longitud del buffer
ADDR_INICIAL          EQU     0x40       ;Dirección de memoria inicial

  ;Constantes para los segmentos
SEG_CODIGO            EQU     0030h    ; inicio seg codigo
SEG_DATOS1            EQU     0050h    ;seg datos
SEG_CONST             EQU     0090h    ;Constantes en rom
SEG_STACK             EQU     00C0h    ;Segmento de stack

LONG_STACK            EQU     10     ;Longitud en bytes del stack

  ;Para interrupciones
INT_PRIORIDAD   EQU     0x00          ;prioridad baja para todas las interrupciones
INT_HABILITADAS EQU     b'10010000     ;Registro de interrupciones. habilitación
de llave general y puerta serie

  ;Para TIMERs
TMOD_INICIAL    EQU     b'00100000    ;Registro de modo de timers
                                      ;timer 0 sin uso
                                      ;modo 2 timer 1, con uso de Fosc

TCON_INICIAL    EQU     b'01000000    ;Registro de control de timers, solo pongo en
                                      ;1 el TR1
```

```
 ;TIMER 1
RECARGA_TMR1    EQU     0xFD   ;Valor de Recarga del Timer para poder operar a
                                  9600 baudios

 ;UART
SCON_INICIAL    EQU     b'01010000      ;Modo 1 (SM0=0;SM1=1) y Recepción
habilitada

;Puerto I/O
led             EQU     P1.2        ;Prende cuando llegan 100 A
alarma          EQU     P1.3        ;Alarma prende cuando no llega ni a, ni B
cinta           EQU     P1.4        ;Cinta transportadora
paleta          EQU     P1.5        ;paleta -> Si es A, paleta =1.. Si es B paleta =0
RX_SERIE_I      EQU     P3.0    ;entrada de recepción serie
;-------------------------------------------------------------------------
;---SEGMENTO DE CODIGO-------------------------------------------------------
        ; Direct the reset vector to the program entry label.
        ;asegn   INTVEC:CODE, 0x0000
         ASEG
         org 0000h
         ljmp    main

        ASEG
        org  0023H              ;(1°) Defino el vector de Interrupción
        ljmp Isr_Serie          ;(2°) Defino el salto a la Isr.

;---SEGMENTO DE STACK--------------------------------------------------------
         ;RSEG ISTACK:IDATA:NOROOT(0)
         aseg
         org 00b8h
stack_init  DS          LONG_STACK          ;Define el segmento de stack
;---SEGMENTO DE DATOS--------------------------------------------------------
     RSEG    IDATA_Z:DATA:NOROOT

dato_rx         ds    1       ;dato recibido del puerto serie, se carga en ISR
hay_dato_rx     ds    1   ;flag de que hay un dato recibido canal serie. En '1', hay dato, en
                            '0', no hay dato .
;-------------------------------------------------------------------------
;Inicia el código

      ;RSEG    NEAR_CODE:CODE:ROOT          ;defino un segmento de código
      RSEG    RCODE
      ORG     SEG_CODIGO                  ;define el punto de inicio del Programa
(code)
main

;**Inicializaciones
      ;Inicializo el STACK
      mov SP,#(stack_init - 1)              ;inicializo el stack
```

```
        ;Inicializo variables
        mov     hay_dato_rx,#0x00       ;Canal libre en 0, canal ocupado en 1
        setb    RX_SERIE_I              ;Inicializa puerto de recepción serie como entrada
        setb    cinta                   ;Inicializo el motor prendido
        clr     alarma                  ;Inicializo la alarma apagada
        clr     led                     ;Inicializo el led apagado
        mov     R2,#00H                 ;Cuento la cantidad de producto A
        mov     R3,#00H                 ;Cuento la cantidad de producto B
        ;Inicializo Interrupciones
        call InitPerif_Interrupciones

;**Inicio loop principal
Loop_ppal:
          ;Voy a ir y volver de la subrutina hasta que se dispare la interrupcion
          ;de puerta serie
          call ProcesarDatoRecibido

salir:  jmp     Loop_ppal
;fin del loop principal

;-------Subrutinas----------------------------------------------------------------
        RSEG NEAR_CODE:CODE:NOROOT(0)

;===========SUBRUTINAS=======================================
==================
;-------------------------------------------------------------------------------
; Nombre: ProcesarDatoRecibido
;-------------------------------------------------------------------------------
ProcesarDatoRecibido:
    push        A                               ;Guardo A
    ;analiza el estado de hay_dato_rx
    mov         A,hay_dato_rx                   ;Muevo el flag a A
    mov         hay_dato_rx,#0x00               ;limpio el flag
    cjne        A,#1,Salir_ProcesarDatoRecibido    ;Si es 0 salto, sino es que tenia un 1 y
ejecuto

    ;----hago algo con el dato recibido
    mov         A,dato_rx                   ;Guardo lo recibido por serie en A
    cjne        A,#0AH, prod_B                  ;Si no es A salto a prod_B
    setb        paleta                      ;paleta en 1
    INC         R2                          ;Cuento un producto A
    cjne        R2,#100D,Salir_ProcesarDatoRecibido     ;Si es distinto  a 100 salto, sino
ejecuto
    clr         cinta                       ;Paro la cinta
    setB        led                         ;Enciendo led
    jmp         Salir_ProcesarDatoRecibido
```

```
prod_B:
      cjne   A,#0BH,distinto                  ;Si es distinto salto, sino ejecuto
      clr    paleta
      INC    R3
      cjne   R3,#100D,Salir_ProcesarDatoRecibido    ;Si es distinto a 100 salto, sino
ejecuto
      clr    cinta
      setB   led
      jmp    Salir_ProcesarDatoRecibido

distinto:
     clr     cinta          ;Si llego aca es porque no es A ni B, paro cinta
     setB    alarma         ;Y prendo la alarma voy a volver al loop_ppal pero al no
                            ;mover la cinta no recibo nada mas por puerta serie.
Salir_ProcesarDatoRecibido:
   pop       A              ;Recupero A
 ret                        ;Salgo de la subrutina

;-------------------------------------------------------------------------------
; Nombre: InitPerif_Interrupciones
; Descripción:
;      Inicializa las interrupciones y algunos periféricos como el timer
;-------------------------------------------------------------------------------
InitPerif_Interrupciones:
      ;Para el timer 1
      mov    TL1, #RECARGA_TMR1       ; Recargo la parte baja del Timer 1
        mov    TH1, #RECARGA_TMR1     ; Recargo la parte alta del Timer 1


      mov    TMOD, #TMOD_INICIAL      ; Inicializo Timer/Counter Mode Register
        mov    TCON, #TCON_INICIAL    ; Inicializo Timer/Counter Control Register

      mov    SCON,#SCON_INICIAL       ;Inicializo puerto serie

      ;Configuro Timer 1 como fuente de reloj para uart en tx y rx
      clr    T2CON_RCLK
      clr    T2CON_TCLK

         ;(3°) Seleccionar la prioridad de las interrupciones
      mov    IP,#INT_PRIORIDAD        ;Prioridad baja para todos

      ;(4°) Habilitar interrupción, es lo último a hacer antes de entrar al lazo ppal.
      mov    IE,#INT_HABILITADAS      ;Habilito las interrupciones según la cte

    ret
```

```
;-------------------------------------------------------------------------------
; Nombre: Isr_Serie
; Descripción:
;       Rutina de servicio de interrupción del puerto serie
;-------------------------------------------------------------------------------
Isr_Serie:
     push         PSW                          ; Resguardo registros modificados por esta
subrutina
     push      A

Recepcion:                                     ; Proceso la Recepción
       clr          SCON_RI                          ; Reseteo Flag de Recepción
          mov     A, SBUF                         ; Recupero Dato Recibido
          mov     dato_rx,A                 ;lo guardo en memoria
       ;mov     dato_rx,#0AH              ;SOLO PARA PROBAR LA RECEPCION DE A
       mov     hay_dato_rx,#0x01             ;levanto el flag

       pop       A
       pop         PSW

     reti

;================================================================
=================

  END     main
```