*Nota: Si hay algun error sepan disculpar, estuve trabajando en base a la documentación de VonSim, videos de Genaro Camele y conocimiento propio, cualquier apoyo a mejorar este documento bienvenido es, exitos en el parcial del día 29/09. Gracias por leer <3.*

**Manual para Arquitectura de Computadoras 2023**

Hecho por: San

\* Memoria

|

\* Guardar datos en memoria: ORG 1000H

|

\* Programa principal: ORG 2000H

|

\* Subrutinas: ORG 3000H

|

\* Pila: ORG 8000H

\* Interrupciones por Software:

|

\* INT 0 (Termina la ejecución del programa, equivalente a HLT)

|

\* INT 3 (Inicia el modo depuración) \*NO SE USA\*

|

\* INT 6 (Lectura de caracteres por teclado. Se requiere que BX tenga una dirección donde se guarden caracteres)

|

\* INT 7 (Escribe en pantalla una cadena de caracteres mediante un registro BX que contiene la dirección de la cadena a imprimir y un registro AL que contiene la cantidad de caracteres a imprimir)

\* Registros de Entrada/Salida:

[TIMER]

\* CONT EQU 10H

\* COMP EQU 11H

[PIC]

\* PIC EQU 20H

\* EOI EQU 20H

\* IMR EQU 21H

\* IRR EQU 22H

\* ISR EQU 23H

[INTERRUPCIONES PIC]

\* INT 0 EQU 24H (TECLA F10)

\* INT 1 EQU 25H (TIMER)

\* INT 2 EQU 26H (HANDSHAKE)

...

[PIO]

\* PA EQU 30H

\* PB EQU 31H

\* CA EQU 32H

\* CB EQU 33H

[HANDSHAKE]

\* DATA EQU 40H

\* ESTADO EQU 41H

\*CONFIGURAR IMR PARA INTERRUPCIONES:

El IMR es el que se utiliza para enmascarar (o “inhabilitar”) líneas de interrupción. Si el bit correspondiente a una línea es 1, esta línea está enmascarada y no se emitirá la interrupción a la CPU. Si el bit es 0, la línea está habilitada y se emitirá la interrupción a la CPU. El registro IMR posee 1 byte (8 bits 1111 1111 por defecto) y los 3 primeros bits de derecha a izquierda significan las interrupciones de: F10, TIMER y HANDSHAKE, en ese orden.

\* 1111 1110 = Habilitamos la interrupción tecla F10

\* 1111 1101 = Habilitamos la interrupción TIMER

\* 1111 1011 = Habilitamos la interrupción HANDSHAKE

Configurar las interrupciones para que se realice una acción (Llamar una subrutina de interrupción)

Se debe utilizar una variable que contenga la dirección de la subrutina a llamar en caso de que la interrupción del PIC suceda:

La fórmula para asignar direcciones es 4xID (ID es un número que no sea 0, 3, 4, 6, 7 ya que estos ya están ocupados por interrupciones)

Si queremos que la dirección 40 (ORG 40) \*NO ESCRIBIR ORG 40H\* sea nuestro lugar a donde vamos a asignar la dirección de la subrutina lo hacemos de la siguiente forma:

ID\_SUBR EQU 10

Y cuando aplicamos la fórmula: 4x10 = 40, escribimos lo siguiente:

ORG 40

dir\_subr DW NOMBRE\_SUBRUTINA\_A\_LLAMAR

Este "NOMBRE\_SUBRUTINA\_A\_LLAMAR" contendra el nombre de donde nosotros hagamos esta subrutina, puede ser 3000h o cualquiera otra dirección perteneciente a las subrutinas.

ORG 40

dir\_subr DW 3000h

También es válido, aunque tendríamos que crear nuestra subrutina de interrupción en la ORG 3000H

Configurar el PIC para que una interrupción haga un llamado a subrutina de interrupción:

Recordar que cuando se hace un llamado a una subrutina de interrupción se utiliza el IRET y no el RET.

Una vez asignada nuestra dirección en ORG y la dirección de la subrutina, debemos asignarle a la interrupción que cada vez que suceda, se llame a la interrupción

\* IN = Entrada

\* OUT = Salida

1.

| MOV AL, ID\_SUBR (Donde ID\_SUBR tiene la dirección de la subrutina)

| OUT PIC+X, AL (Donde X es un número que me posiciona en la interrupción)

\* 20H + 4 = INT 0 (F10)

\* 20H + 5 = INT 1 (TIMER)

\* 20H + 6 = INT 2 (HANDSHAKE)

2. (Utilizando constantes)

| MOV AL, ID\_SUBR

| OUT INT0, AL (INT0 previamente tendria que estar declarado como constante, INT0 EQU 24H)

Asi que cada vez que suceda esta interrupción, se llamara a la subrutina que este en DIR\_SUBR.

Configuración del TIMER:

El TIMER posee 2 registros, el CONT y el COMP

\* CONT: Se incrementa en 1 cada que pase 1 segundo

\* COMP: Cuanto el registro CONT coincide con COMP, se produce una interrupción

CONT = 0

COMP = X (Donde X es la cantidad de segundos que tienen que pasar para que se produzca una interrupción)

Configuración del TIMER (Código):

\* TIMER EQU 10H

\* CONT EQU 10H (Opcional)

\* COMP EQU 11H (Opcional)

1.

| MOV AL, X (X es la cantidad de segundos)

| OUT TIMER+1, AL; TIMER: registro COMP / Seteo en X el Comparador

| MOV AL, 0; Setea en 0 el Contador

| OUT TIMER, AL; TIMER: registro CONT

2. (Utilizando constantes)

| MOV AL, X

| OUT COMP, AL

| MOV AL, 0

| OUT CONT, AL

Configuración del PIO

El PIO posee 4 registros: PA, PB, CA, CB

\* 0 = Salida

\* 1 = Entrada

Puede estar conectado a llaves y luces o a una impresora

El valor del Puerto A se encuentra en el registro PA y su configuración en el registro CA. El registro CA también es de 1 byte (8 bits) y le indica al PIO el modo cada bit: un 0 si es de salida y un 1 si es de entrada. Por ejemplo, para CA = 00001111b, los cuatro bits más significativos son de salida y los cuatro menos significativos son de entrada. El puerto B funciona idénticamente.

Configuración de la impresora por PIO (Código)

1. CA Configura PA | Strobe 0 (Salida) Busy 1 (Entrada)

a.

| MOV AL, 11111101B

| OUT CA, AL

\* Configuro Strobe en 0 y Busy en 1

2. CB Configura PB | Configuro CB todo en 1

(Recordar que previamente declare CB como constante [33H])

a.

| MOV AL, 0

| OUT CB, AL

\* Configuro CB como todo salida (todo 0)

\*TODO LO DE ARRIBA SE HACE 1 VEZ\* (y se puede hacer en una subrutina en el programa principal)

CALL CONFIG\_PIO (¿Trucazo no?)

Para lograr imprimir consulto:

Si busy = 0

Imprimo

Sino

No imprimo

Luego de consultar si está desocupada

Seteo el STROBE en 1 y luego en 0

\*TODO LO DE ARRIBA SE HACE POR CADA CARACTER\*

Hacer consulta de estado para impresora PIO (polling method)

(Las partes de código son como si se escribieran y lo que sigue abajo es el código, los separo para poder explicar que hace cada parte pero en realidad se copiaria como si fuese todo un código entero)

Ejemplo de lo que digo en el mensaje de arriba:

POLL: ; Consulta de estado

IN AL, PA; Mando el binario de PA al registro AL

AND AL, 1; Hago XXXXXXSB AND 00000001 y ahí me entero si está desocupada la impresora o no

JNZ POLL; Si la flag zero = 1 (ocupada), flag zero = 0 (desocupada)

MOV AL, [BX]; Muevo el CARACTER que apunta la cadena en BX al registro AL

OUT PB, AL; Mando el carácter que hay en el registro AL al puerto PB que es donde yo mando el dato a imprimir

; Strobe en 1

IN AL, PA; Tomamos el estado de PA

OR AL, 2; Forzamos en 1 el bit de Strobe

OUT PA, AL; Mando la cadena binaria XXXXXX1B al puerto PA

\* Básicamente el strobe en 1 significa que hay un dato y lo va a imprimir

; Strobe en 0

IN AL, PA; Tomamos el estado de PA

AND AL, 11111101B; Forzamos en 0 el bit de Strobe

OUT PA, AL; Mando la cadena binaria XXXXXX0B al puerto PA

INC BX; Incremento la posición para pasar al siguiente carácter

CMP BX, OFFSET FIN; En este caso es para preguntar si llegue al final, si no llegue al final sigo preguntando

JNZ POLL; Si no llegue al final entonces vuelvo a la consulta de estado, y si llego al final entonces debería terminar de imprimir

- Explicación por partes del código -

POLL:

IN AL, PA; Tomo lo que haya en PA y se lo mando al registro AL (XXXXXXSB) | S = Strobe | B = Busy

AND AL, 1; XXXXXXSB AND 00000001 = 0000000B (No confundir la letra de binario con Busy, en este caso lo que se hace es "**Z**")

JNZ POLL; Si la flag me da 0 entonces vuelvo a consultar, si la flag me da 1 entonces salgo del bucle

**Z**: [B AND 1 = B] (Si busy = 1 entonces 1 y 1 = 1 entonces está ocupada, y si busy = 0 entonces 0 y 1 = 0 entonces está desocupada)

En caso de estar desocupada, mando el caracter del registro al que apunta BX ([BX]) al registro AL (En [BX] voy a tener mi cadena de caracteres)

MOV AL, [BX]

OUT PB, AL; Mando el caracter al puerto PB (Siempre utilizando AL ya que es el único registro que me lo permite)

\* Strobe 1

| IN AL, PA; (El estado de PA en ese momento va a ser XXXXXXSB) recuerden que S = Strobe y B = Busy

| OR AL, 2; XXXXXXSB OR 00000010 = XXXXXX1B (Forzamos el Strobe en 1)

| OUT PA, AL; Mando ahora si la cadena modificada al puerto PA

\* Strobe 0

| IN AL, PA; (El estado de PA en ese momento va a ser XXXXXX1B, ya que quedamos con lo que haya en el anterior PA)

| AND AL, 11111101B

; XXXXXX1B AND 11111101 = XXXXXX0B (Forzamos el Strobe en 0)

| OUT PA, AL; Volvemos a mandar la cadena ahora con el strobe en 0 al puerto PA

Configuración del Handshake (Explicación)

HANDSHAKE posee dos registros:

\* DATO en la posición 40H

\* ESTADO en la posición 41H

| cada uno de 1 byte (XXXXXXXX)

Configuración del ESTADO (Explicación)

\* ESTADO = IXXXXXSB (I = 1 por interrupción, I = 0 por Polling, además, el ESTADO cuenta con los bits de Strobe y Busy en caso de ser por polling aka consulta de estado)

| Si el bit más significativo es un 1 es porque el Handshake se va a manejar como interrupción, caso contrario (0), es por consulta de estado

Configuración del HANDSHAKE (Sin interrupciones)

\* ORG 2000H

| IN AL, ESTADO; Tomo el estado actual del ESTADO del Handshake y lo mando al registro AL

| AND AL, 01111111B; IXXXXXXX AND 01111111 = 0XXXXXXX (Le digo al estado del Handshake que voy a utilizar interrupciones)

| OUT ESTADO, AL; Mando la configuración a ESTADO (Handshake por interrupciones)

Enviar caracteres para imprimirlos (Código)

\* MOV BX, OFFSET MENSAJE; Muevo la cadena entera a BX

| POLL: IN AL, ESTADO; Muevo el estado para consulta de estado (polling)

| AND AL, 1; Pregunto por bit de busy para ver si está desocupada

| JNZ POLL; FLAG ZERO = 0 Ocupada, FLAG ZERO = 1 DESOCUPADA

| MOV AL, [BX]; Muevo el carácter a imprimir al registro AL

| OUT DATO, AL; Mando el dato a imprimir al registro DATO del HANDSHAKE

| INC BX; Paso al siguiente carácter

| CMP BX, OFFSET FIN; Comparo si llegue al final de mi cadena

| JNZ POLL; Si FLAG ZERO = 0 entonces no llegue al final y sigo consultando, FLAG ZERO = 1 entonces llegue al final y sigo abajo

- En caso de terminar de imprimir simplemente no salta a POLL y sigue debajo de la línea "JNZ POLL" (arriba de este mensaje) y continua con lo que tenga que hacer

Configuración del HANDSHAKE (Interrupciones) (Código)

; Declaro constantes

EOI EQU 20H

IMR EQU 21H

INT2 EQU 26H

ESTADO EQU 41H

DATO EQU 40H

ID\_RUTH EQU 10

ORG 40

DIR\_RHI DW IMPRIMIR; Dirección\_Ruta\_Handshake\_Interrupción = 3000H (IMPRIMIR está en la 3000H)

ORG 1000H

MENSAJE DB "Arquitectura de Computadoras 2023"

FIN DB?

ORG 3000H

IMPRIMIR: ; IMPRIMIR está en la dirección 3000H

MOV AL, [BX]; Muevo el carácter a imprimir al registro AL

OUT DATO, AL; Muevo a DATO el carácter a imprimir

INC BX; Paso a la siguiente posición

MOV AL, 20H; Le mando al registro AL el 20H

OUT EOI, AL; Se lo mando a EOI para avisarle que termine de interrumpir

IRET; Vuelvo de la subrutina de interrupción

ORG 2000H

CLI; Desactivo temporalmente las interrupciones

MOV AL, 11111011B; Habilito la interrupción HANDSHAKE en el IMR

OUT IMR, AL; Configuro el IMR para interrupción HANDSHAKE

MOV AL, ID\_RUTH; Muevo al registro AL 10, que va a ser la ID que se va a multiplicar en la interrupción (IDx4 = ORG IDx4 (10x4 = ORG 40))

OUT INT2, AL; Le paso el 10 del registro AL a la interrupción para que sepa a donde tiene que ir cuando mande un pedido de interrupción

STI; Habilito las interrupciones

MOV BX, OFFSET MENSAJE; Mando a BX la cadena de caracteres

IN AL, ESTADO; Tomo el estado

OR AL, 10000000B; Deshabilito el Handshake por interrupciones

OUT ESTADO, AL; Mando la configuración al ESTADO desde el registro AL

POLL: CMP BX, OFFSET FIN; Consulto si llegue al final

JNZ POLL; Si no llegue al final, vuelvo a preguntar si llegue al final

IN AL, ESTADO; Cuando llegue al final, tomo el ESTADO y lo mando al registro AL

AND AL, 01111111B; Deshabilito el estado Handshake por interrupciones

OUT ESTADO, AL; Mando la configuración al estado

INT 0; Finalizar

END