Universidad Autónoma de Baja California

Facultad de ciencias químicas e ingeniería

Materia.

Microprocesadores y Microcontroladores.

Maestro.

Garcia Lopez Jesus Adan.

Alumno

Gonzalez Cardiel Luis Enrique

Matricula:

1217258

Grupo:

561

Trabajo:

Practica No. 5b

05/10/2018

Práctica No. 5b

• Sección de Entrada/Salida

Objetivo: Acceso a los puertos genéricos del dispositivo programable PPI-8255 que se encuentra emulado en la tarjeta T-Juino.

Material: - Programa ensamblador y encadenador (TASM/TLINK)
- Tarjeta T-Juino.

Equipo: - Computadora Personal

Teoría: * * * INVESTIGACION: INTERFAZ DE PUERTOS PARALELO 8255 * * *

Desarrollo:

Parte A: Programación y uso de puertos del PPI8255.

1) Crear archivo Prac5b.c que contenga a partir del listado 1

Listado 1

```
#define BYTE unsigned char
#define WORD unsigned int

#define PA     0x40
#define PB     0x41
#define PC     0x42
#define RCtr   0x43
#define PTOs_all_out 0x80

char dato;

void main( void ){
    puts("Practica 5b\n\r");          /* imprimir mensaje          */
    outportb(RCtr, PTOs_all_out);     /* inicializar 8255          */
    outportb(PA,0x55);                /* presentar 55h en el Pto A */
    while(1){
        dato = getch();               /* leer tecla                */
        outportb(PB,dato);            /* presentar tecla en PB     */
        printBin(dato);
        puts("\n\r");
    }
}

/* función para lectura de puertos usando ensamblador in-line     */
void outportb( WORD port, BYTE dato){
   asm  mov dx, port
   asm  mov al, dato
   asm  out dx, al
}

/* función simple para desplegar un byte en formato binario       */
void printBin( BYTE dato ){
    BYTE msk=0x80;
    do{
        putchar( (dato & msk) ? '1' : '0' );
        msk>>=1;
    }while( msk );
}
```

2) Realizar proceso para ejecutar el programa en T-Juino.

3) La siguiente función llamada SetBitPort manipula la información de un puerto dado para activar un determinado bit. Es decir mediante ella se puede activar (hacer uno) un bit del puerto. El número del bit está en el rango de 0 a 7 siendo el bit 7 es más significativo.

Listado 2

```
void SetBitPort(WORD Puerto, BYTE num_bit)
{

   BYTE mask=0x01;                    /* mascara inicial                 */
   BYTE temp;                         /* dato auxiliar                   */

    temp = inportb( Puerto );         /* leer dato del puerto            */
    mask = mask << num_bit;           /* ajustar mascara según num_bit   */
    temp = temp | mask;               /* aplicar mascara con operador OR */
    outportb( Puerto , temp );        /* presentar resultado en el puerto */

}
```

Listado 2 – Simplificado

```
void SetBitPort(WORD Puerto, BYTE num_bit)
{

    outportb( Puerto , inportb( Puerto )|( 0x01 << num_bit ) );

}
```

4) Diseñe las siguientes funciones para manipulación de bit de puertos .

Nota: *Es necesario implementar las funciones básica de E/S inportb( ) y outportb( ) en lenguaje ensamblador en archivo ASM independiente -- no usar funciones in-line.*

a) Función ClrBitPort la cual borrar un bit; es decir hace cero el bit de la posición num_bit del puerto dado por el parámetro Puerto.

void **ClrBitPort**(WORD Puerto, BYTE num_bit)

Función NotBitPort la cual invierte el bit de la posición num_bit del puerto dado por el parámetro Puerto.

void **NotBitPort**(WORD Puerto, BYTE num_bit)

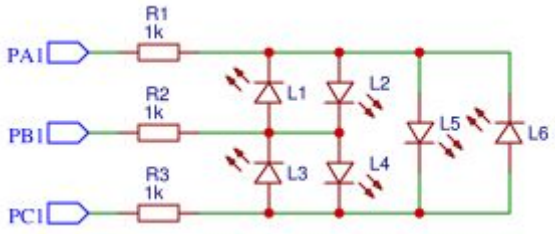
b) Función TstBitPort la cual retorna el estado del bit de la posición num_bit del puerto dado por el parámetro Puerto. Si el bit del puerto está en 0 lógico entonces la función retorna valor cero (0) de otra forma retorna valor uno (1).

BYTE **TstBitPort** (WORD Puerto, BYTE num_bit)

5) Verifique el funcionamientos de las funciones del punto anterior realizando los siguientes pasos:

a. Capturar 8 bits mediante TstBitPort en PC4, dando tiempo para que el usuario pueda alterar el valor de PC4 (por ejemplo, detener el programa mediante un getchar()), y mostrar los bits capturados en la terminal.

b. Mostrar los 6-bits menos significativos sobre el siguiente arreglo de LEDs:



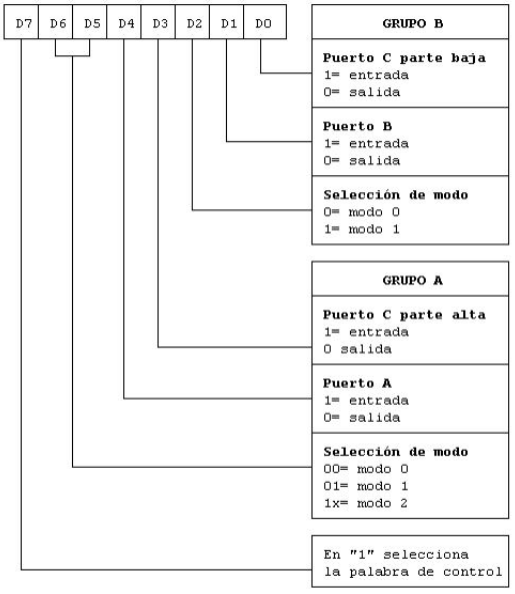
Respetando L1 como el LSB, y L6 como el MSB.

* * * INVESTIGACION: INTERFAZ DE PUERTOS PARALELO 8255 * * *

El PPI 8255 es un dispositivo de E/S general, programable, capaz de controlar 24 líneas con diferentes configuraciones (entrada/salida) y en hasta 3 modos de operación.

El 8255 soporta 3 modos de operación: el modo 0 (entrada y salida básica), el modo 1 (entrada y salida con señales de control) y el modo 2 (bus bidireccional de comunicaciones). Tras un Reset, los 3 puertos quedan configurados en modo entrada, con las 24 líneas puestas a "1" gracias a la circuitería interna. Esta configuración por defecto puede no obstante ser alterada con facilidad. El modo para el puerto A y B se puede seleccionar por separado; el puerto C está dividido en dos mitades relacionadas con el puerto A y el B. Todos los registros de salida son reseteados ante un cambio de modo, incluyendo los biestables de estado. Las configuraciones de modos son muy flexibles y se acomodan a casi todas las necesidades posibles. Los tres puertos pueden ser accedidos en cualquier momento a través de la dirección E/S que les corresponde, como se vio en el apartado anterior. La palabra de control a enviar a la 4ª dirección es:

Aviso: los PC tienen un byte de identificación 0FFh; los XT 0FEh (este byte está en la posición de memoria 0FFFF:0Eh); por otro lado, parte de esta información es accesible también por medio de la variable BIOS ubicada en 40h:10h, método mucho más recomendable.



Conclusiones y Comentarios.

Los puertos son una parte esencial de las computadoras ya que sin ellos no se daría el intercambio de datos o información entre una computadora y sus periféricos o entre una computadora y otra.

Esta práctica resultó un poco complicada a la hora de hacer encender 0Fh ya que encendían un poco los 2 bits más significativos pero a la hora de mostrar AAh todo salió correctamente.

Bibliografía

http://icaro.eii.us.es/descargas/Transparencias_i8255.pdf