Práctico 8

Tema: Entrada/Salida programada e interrupciones.

Objetivo: Manejo de Entrada Salida (Polling e Interrupciones) a nivel conceptual. Los conceptos se manejan independientemente de la arquitectura, por lo tanto no se tienen en cuenta los detalles de implementación.

Notas:

- Para manejar E/S en alto nivel deben utilizar las pseudo-instrucciones del tipo **dato=in(DIR E/S)** y **out (DIR E/S, dato)**.
- Para habilitar y deshabilitar interrupciones en alto nivel se pueden invocar las rutinas Habilitar_interrupciones() y Deshabilitar_interrupciones() respectivamente.
- Los ejercicios marcados con (C), son complementarios.
- Los ejercicios a ser hechos en clase son: 2, 8.
- Las soluciones a ser publicadas serán: 5, 11.

Ejercicio 1
Se considera el teclado de una terminal de caracteres controlado por un microprocesador. Este dispositivo es accesible mediante 2 registros de 1 byte:

ESTADO_TECLADO 22h
DATO_TECLADO 23h

donde 22h, 23h son las direcciones de E/S donde están accesibles los registros, de SOLO LECTURA. El bit más significativo de ESTADO_TECLADO es seteado en 1 por el hardware cuando hay un carácter disponible. Cuando se lee el carácter disponible en DATO_TECLADO, el bit más significativo de ESTADO_TECLADO es puesto a 0 por el hardware.

Se pide: Escribir un procedimiento para aceptar caracteres de entrada, organizarlos en palabras que contienen dos caracteres y almacenarlos en posiciones consecutivas en un buffer circular de memoria. El tamaño del buffer es de 512 palabras.

Ejercicio 2
Se desea controlar una bomba de agua mediante un microprocesador.
La bomba puede activarse poniendo en 1 el bit 2 del puerto de E/S BOMBA, se puede apagar poniendo en cero dicho bit; existe un sensor externo que pone en 1 el bit 0 del puerto BOMBA cuando se activa efectivamente la bomba, y en 0 cuando está desactivada. Cuando la bomba se activa se debe activar un LED conectado al bit 4 del puerto BOMBA, en otro caso se debe desactivar el LED.

Se pide: Escribir los procedimientos activarBomba() y desactivarBomba() sabiendo que el puerto de E/S BOMBA es de lectura y escritura.

Ejercicio 3
Se dispone de tres rutinas utilitarias llamadas RUTINA1, RUTINA2 y RUTINA3. El programa en ejecución es interrumpido en cualquier momento para ejecutar estas rutinas por acción del teclado. Esta ejecución comienza con la secuencia de teclas <ESC> <1>, <ESC> <2> y <ESC> <3> respectivamente. <ESC> es un carácter especial, conocido. La única manera de retornar de las rutinas es por la tecla <ESC>.

Se pide: Escribir un procedimiento, en un lenguaje de alto nivel, que atienda la interrupción del teclado, lea el carácter contenido en TECLA y en caso que la sucesión de caracteres recibida sea válida, pase a ejecutar la rutina correspondiente. La interrupción debe retornar al programa adecuado luego de procesar una tecla.

Notas:

- Para obtener la dirección de una rutina se dispone de la función radr(nombre rutina).
- Para pasar el control a una dirección determinada se usa la función jmp(dirección).
- Para obtener la dirección del programa interrumpido se usa la función padr().

Ejercicio 4

Un procesador posee tres dispositivos conectados a una misma línea de interrupción. El procesador atiende por nivel y los dispositivos interrumpen también por nivel. Existen tres direcciones de Entrada/Salida Estado_1, Estado_2 y Estado_3 asociadas a ellos, cuyo bit menos significativo es colocado en uno por el hardware cuando el dispositivo correspondiente solicita una interrupción.

Se pide: Escribir una rutina de interrupción en alto nivel capaz de atender a los tres dispositivos. Para las siguientes situaciones:

(a) Un esquema de prioridades fijas (pri(disp. 1) > pri(disp 2) > pri(disp. 3).
(b) Un esquema de prioridades variables. Existe la función pri(dispositivo).

Ejercicio 5

Se desea implementar un concentrador de comunicaciones cuya misión es recibir datos a través de 4 líneas de entrada y enviarlos por 1 única línea de salida. Cada línea de entrada tiene un controlador asociado que se maneja por medio de 2 registros, uno de control (EST_CONT_n) y otro de datos (DATO_n), de 1 byte cada uno, ubicados en las direcciones de memoria:

- controlador 0: EST_CONT_0 10H, DATO_0 11H
- controlador 1: EST_CONT_1 12H, DATO_1 13H
- controlador 2: EST_CONT_2 14H, DATO_2 15H
- controlador 3: EST_CONT_3 16H, DATO_3 17H

Interpretación de EST_CONT_n:

- bit 0: en 1 indica un dato disponible en DATO_n. El bit 0 es el menos significativo y solo puede ser leído. Se apaga si es leído el registro DATO_n.
- bit 1 al bit 7 no se usan.

DATO_n: contiene el dato recibido por la línea n.
Cuando algún controlador de entrada tiene un dato disponible genera, en caso de estar habilitado, una interrupción que activa la rutina INT_ENTRADA.

La línea de salida tiene su propio controlador con un registro de datos DATO_S, de un byte, ubicado en la dirección de E/S 20H. Toda vez que se escriba un byte en DATO_S el controlador trasmite el byte por la línea de salida, y al terminar la transmisión genera una interrupción que activa la rutina INT_SALIDA.

Se deben cumplir las siguientes reglas:

- Al ingresar un dato (1 byte) por la línea n (0..3) se trasmite a la salida el numero n (1 byte) y luego el dato ingresado (1 byte).

- Si no hay datos a trasmitir se enviará el valor 0FFH como salida (para que la salida esté siempre trasmitiendo algo).

Se pide: Escribir en alto nivel las rutinas necesarias para implementar el concentrador.

Nota:
- El procesador está **dedicado** exclusivamente a la ejecución de las rutinas que implementan el concentrador.

Ejercicio 6
Sea un sistema que controla las luces de iluminación de los pasillos de un edificio. Este cuenta con un botón de encendido en cada piso y se desea que una vez transcurrido 45 segundos desde que fue presionado el botón de algún piso, estas se apaguen.
Para esto se sabe que:
(a) Cuando se aprieta un botón en cualquier piso, se invoca una rutina de interrupción llamada BOTÓN.
(b) El encendido de todas las luces se hace poniendo en 1 el bit menos significativo del byte de la dirección 20H y se apagan colocando este mismo en 0.
(c) Además se cuenta con un timer que genera una interrupción cada un segundo la cual invoca a la rutina de interrupción TIEMPO.

Escribir en lenguaje de alto nivel las rutinas necesarias para implementar el controlador de luces.

Ejercicio 7 (C)
Se desea controlar el escape de gas en una fábrica de recarga de garrafas. Por este motivo se instala un sistema de seguridad controlado por un procesador utilizado en forma dedicada. El sistema está formado por un sensor de gas, un extractor y una válvula de corte.
- El sensor controla la concentración de gas en el ambiente. Es posible conocer si la concentración de gas es peligrosa o no consultando el registro de E/S del sensor.
- El extractor y la válvula pueden controlarse manipulando el registro de lectura/escritura asociado a ellos. La válvula una vez cerrada sólo podrá abrirse manualmente.

El sistema debe encender el extractor toda vez que este detecte escape de gas (la concentración de gas es peligrosa) y apagarlo cuando dicho escape culmina. Si el escape persiste por más de 30 segundos se debe cerrar la válvula y apagar el extractor cuando el escape haya finalizado.

Descripción de los registro de E/S:
1. Para el sensor
   - El bit 0: vale 1 si la concentración de gas es peligrosa y 0 cuando no lo es.
   - Del 1 al 7: reservados.
   - Dirección de E/S SENSOR.

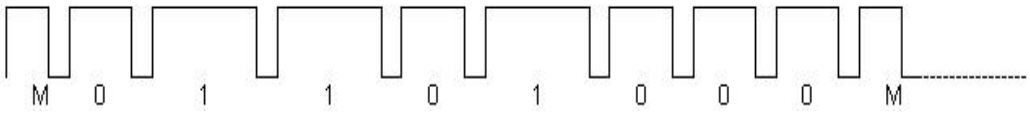
2. Para el extractor y la válvula
   - El bit 0: puede ser seteado a 1 para encender el extractor o a 0 para apagarlo.
   - El bit 1: puede ser seteado a 0 para cerrar la válvula.

- Del 2 al 7: reservados.
- Dirección de E/S ACCESORIO.

Se pide: Escribir todas las rutinas necesarias para implementar el sistema de seguridad en un lenguaje da alto nivel.

Nota: Se dispone de un reloj externo que genera una interrupción cada 10 Hz la cual es atendida por la rutina TIEMPO.

Ejercicio 8
Se desea construir un dispositivo que decodifique una transmisión digital binaria codificada en ancho de pulso PWM (Pulse Width Modulation).
La técnica PWM consiste en modificar la duración de un pulso (tiempo en que la señal está en "1"). En este caso concreto un 0 se codifica como un pulso que dura 30 ms (en estado 1), mientras que un 1 se codifica con un ancho de pulso de 50 ms (en estado 1). Un pulso de ancho menor que 30 ms se considera una "marca" que indica que finaliza un byte y comienza otro (la marca no es un bit y eventualmente puede aparecer antes de completarse 8 bits en cuyo caso se rellena con 0 a la izquierda o luego de más de 8 bits en cuyo caso se toman los 8 últimos recibidos).



El dispositivo recibe el tren de pulsos a través de una entrada binaria que puede ser leída en el bit menos significativo del byte de E/S en la dirección PULSO. En paralelo los pulsos son presentados a una entrada de pedido de interrupción con detección por flanco ascendente que invoca a la rutina de atención flanco().
Se dispone además de un timer de frecuencia 1 KHz que genera una interrupción que invoca a la rutina de atención timer().

El sistema deberá decodificar los bits recibidos en forma serial codificados en el ancho de los pulsos, rearmar y escribir cada byte recibido en el puerto de E/S (solo escritura) en la dirección SALIDA.

Se pide implementar en un lenguaje de alto nivel todas las rutinas necesarias para que funcione el decodificador según los requerimientos, teniendo en cuenta que la maquina es dedicada.

Ejercicio 9 (C)
Se desea diseñar un dispositivo para controlar una caja de seguridad con combinación para ser instalada en una cadena de hoteles. La caja de seguridad cuenta con un teclado numérico con 10 teclas (0 - 9) y la tecla ENTER y un display de 8 dígitos de 7 segmentos.
El funcionamiento será el siguiente:

- Cuando la caja está abierta el huésped ingresará una clave de entre 4 y 8 dígitos en el teclado y cerrará la puerta de la caja.

- Cuando presiona la tecla ENTER la caja se cerrará mediante una serie de barras de metal que se extienden desde la puerta. Si el código ingresado tiene menos de 4 dígitos no se hará nada.

- Luego de cerrada si se ingresa la misma secuencia usada para cerrarla y luego se presiona la tecla ENTER la caja se abrirá retrayendo las barras.

- Si la clave es ingresada en forma equivocada, al presionar la tecla ENTER se mostrarán una serie de 8 guiones en el display de 7 segmentos.

- Si la clave se ingresa mal 3 veces seguidas la caja quedará bloqueada por 2 horas como medida de seguridad. Esto quiere decir que cualquier combinación de teclas que se intente siempre dará error durante ese lapso de tiempo.

- El display y la memoria de los dígitos presionados se vaciaran automáticamente si no se presiona ninguna tecla por 5 segundos. Si se presionan más de 8 dígitos solo se considerarán los últimos 8 ingresados antes de presionar ENTER.

Para manejar el teclado se dispone de una interrupción llamada tecla() que se llamará cada vez que se presiona una tecla. La tecla presionada puede leerse en el registro de E/S de solo lectura TECLADO de un byte. Las teclas de dígitos se leerán codificadas en binario en los 4 bits más significativos y la tecla ENTER tendrá el código 0xD0.

Para activar el motor que extiende o retrae las barras se dispone de un registro de E/S de solo escritura de un byte llamado BARRAS. Escribiendo un 1 en el bit más significativo extiende las barras mientras que escribiendo un 1 en el menos significativo las retrae.

Para manejar el display se dispone de un registro de E/S de solo escritura de 4 bytes llamado DISPLAY donde se debe escribir el número que se desee mostrar en BCD (los dígitos más significativos del número a mostrar coinciden con los más significativos del registro). El guión a mostrar en caso de error se codifica como 0xFF mientras que para no mostrar nada se debe escribir 0xEE.

Se dispone además de un timer que interrumpe con una frecuencia de 2 Hz invocando a la rutina tiempo().

Se pide:
Implementar en un lenguaje de alto nivel el programa principal y todas las rutinas necesarias para que funcione el sistema de control para el dispositivo descripto.

Ejercicio 10(C)
Se desea construir un sistema controlador de un semáforo para un cruce peatonal en una avenida muy ancha.
El semáforo contará, como es habitual, con luces de color verde, amarillo y rojo. Adicionalmente dispondrá de un panel construido con dos “displays” numéricos los cuales deberán indicar el tiempo que le resta para permanecer encendida la luz verde. Cuando no esté encendida la luz verde el panel deberá permanecer apagado.

El controlador del semáforo deberá encender las luces en el orden: verde, amarillo y rojo. El valor por defecto de la luz verde es de 50 segundos. El tiempo de permanencia en la luz roja es de 2 minutos y el de la luz amarilla 5 segundos. También se contará con un botón para uso de los peatones el cual al presionarse hará que se incremente en 10 segundos el tiempo por defecto de la luz verde. El botón puede presionarse en cualquier momento. No es posible acumular tiempo mediante sucesivas activaciones del botón, que será tomado en cuenta una sola vez en un ciclo amarillo – rojo – verde (es decir si el botón se pulsa estando el semáforo en amarillo o rojo, incrementará el tiempo de la siguiente verde, si se pulsa estando

en verde alargará el tiempo de la actual).

El estado del botón para los peatones se puede consultar en el byte de E/S de sólo lectura en la dirección BOTON, donde el bit menos significativo está en “1” mientras se encuentre presionado el botón.
Los “displays” para mostrar el tiempo restante se manejan en el byte de E/S de solo escritura en la dirección DISPLAYS, el cual recibe codificado en BCD en sus 4 bits más significativos el valor a mostrar en el panel izquierdo y en sus 4 bits menos significativos el valor del panel derecho. Para que el panel aparezca apagado debe escribirse el valor 0xFF en el mencionado puerto.
Las luces del semáforo se controlan en el byte de E/S de solo escritura en la dirección LUCES: el bit 0 (menos significativo) es la luz verde, el bit 1 es la luz amarilla y el bit 2 la roja. El bit correspondiente en “0” apaga la luz y en “1” la enciende.
Se dispone de un timer de frecuencia 5 Hz que interrumpe invocando a la rutina tiempo().

Se pide:
Escribir en un lenguaje de alto nivel todas las rutinas necesarias para implementar el sistema de semáforo, suponiendo que el procesador está dedicado a la tarea.

Ejercicio 11
Un fabricante de juguetes desea fabricar un juguete que dispone de un teclado similar a un piano con 7 notas diferentes.
Cuando el niño presiona una tecla se debe reproducir por el parlante del dispositivo la nota asociada a la tecla presionada. Mientras la tecla se mantenga presionada la nota seguirá sonando y si se presiona otra tecla la misma será ignorada.
Para reproducir un sonido se debe generar con el parlante una onda con una frecuencia lo más cercana posible a la correspondiente a la nota.

- Do: 262 Hz
- Re: 294 Hz
- Mi: 330 Hz
- Fa: 349 Hz
- Sol: 392 Hz
- La: 440 Hz
- Si: 494 Hz

Para implementar el juguete se dispone de un procesador dedicado y de los siguientes elementos:

- Un timer de frecuencia 1MHz que genera una interrupción que invoca a la rutina tiempo()
- Cuando se presiona una tecla del piano se produce una interrupción que invoca a la rutina piano() y la tecla presionada se escribe en el puerto de 8 bits de sólo lectura TECLA con la siguiente codificación: se pone en 1 el bit i cuando se presiona la tecla i del piano, i=1..7. Si el bit 0 (menos significativo) está prendido la tecla se está presionando, mientras que si está apagado la tecla se está soltando.
- El parlante se controla con el bit 0 el registro de E/S (solo escritura) PARLANTE. Para que el parlante reproduzca el sonido se debe generar una onda cuadrada con la frecuencia de la nota que se desea emitir.

Se pide implementar en un lenguaje de alto nivel todas las rutinas necesarias para que funcione el juguete según los requerimientos.