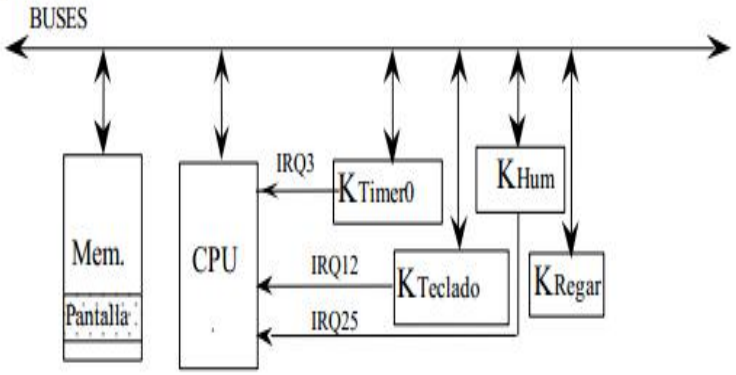
3.- Se desea diseñar un **sistema automático de la temperatura y humedad de un invernadero**. El sistema está basado en la arquitectura de la siguiente figura, similar a la vista en clase, a la que se han añadido unos sensores de temperatura y humedad.



KHum :

Dispositivo que interrumpe cuando la humedad relativa sea inferior a un valor mínimo (HUMEDAD_ESCASA).

KRegar: Válvulas de regadío. Disponen de un registro de control (RCON_REGAR) en que se debe indicar que se quiere hacer: 1 para abrir las válvulas y 0 para cerrarlas. Al igual que en el caso anterior, para simplificar, suponed que no pasa nada si se intenta cerrar las válvulas estando ya cerradas o si se intenta abrir estando ya abiertas.

El **funcionamiento** del sistema es el siguiente. Si se detecta que la humedad relativa es inferior al mínimo, se deben abrir las válvulas de regado durante 5 minutos.

En cualquier momento el operario puede finalizar el sistema pulsando la tecla 'Select'.

Los controladores de reloj y el teclado y el gestor de interrupciones son los mismos que los vistos en clase. En este sistema en particular, la sincronización del **teclado** se debe realizar por **encuesta**. Los registros de los dispositivos nuevos están en el espacio de E/S, no están mapeados en memoria. Disponemos de las siguientes instrucciones para acceder a ellos:

unsigned char ***InPort*** *(registro);*
void ***OutPort*** *(registroa, valor);*

Se pide lo siguiente:

- Escribe en lenguaje algorítmico todas las rutinas de atención que creas necesarias y el programa principal. Comenta cualquier supuesto que consideres para la resolución del problema.
- Escribe en lenguaje de programación C las instrucciones de configuración del temporizador y el teclado.
- Escribe en lenguaje de programación C la rutina que permita las interrupciones de KHum.
- Explica que cambiaría en el ejercicio si la sincrinización con el controlador de humedad se tuviera que realizar por encuesta. Supón para ello, que el controlador KHum dispone de un registro de control, RCON_KHUM, cuyo valor será 1 siempre que la humedad descienda del mínimo establecido.

Para solucionar este problema realizaremos ciertas suposiciones:

- Supondremos que tenemos definida la mascara de la tabla de interrupciones que se encarga del sensor de humedad, la llamaremos IRQ_KHum y utilizará el bit 14 que está libre.

- Usaremos el archivo "defines.h" que tenemos en la plantilla del proyecto, pero con unas pequeñas diferencias:
  -Añadiremos la variable global segundos, que cuenta la cantidad de segundos que llevan abiertas las válvulas.
  -Sustituiremos SELECT por un entero, que en binario será todo ceros y un 1 en el bit 2 (el de la tecla select).

Suponiendo esto el problema se vuelve bastante sencillo. El programa principal lo dividiremos en dos partes. Una primera de inicialización donde prepararemos todo para el bucle principal y el bucle principal. En este sólo comprobaremos si hay que cerrar el riego, ya que únicamente abriremos las válvulas a través de la interrupción del sensor de humedad.

Así nos quedaría el main:

```
void main(){
    extern int segundos;
    segundos = 0;

    //Determinamos la rutina de interrupción del temporizador
    //intTemporizador suma uno a la variable global segundos (lo hará
    //una vez por segundo)
    irqSet(IRQ_TIMER0,intTemporizador);

    //Determinamos la rutina de interrupción del sensor de humedad.
    //intHum inicia la cuenta del temporizador, desactiva las
    //interrupciones del sensor de humedad y abre las válvulas
    irqSet(IRQ_KHum,intHum);

    //Habilitamos las interrupciones del temporizador
    HabilitarIntTemp();

    //Habilitamos las interrupciones del sensor de humedad
    HabilitarIntHum();

    //Hacemos que el temporizador interrumpa una vez por segundo
    prepararTemporizador();

    while(1){
        if ((segundos == 300) || (TECLAS_CNT & SELECT)){
            /* Si select se pulsa o llegamos a los 5 minutos se
            interrumpe el riego */
            pararTemporizador();//Paramos el temporizador
            segundos = 0;       //Ponemos a cero la cuenta de segundos
            OutPort(RCON_REGAR,0); //Cerramos las válvulas
```

```
            HabilitarIntHum(); /*Volvemos a atender las interrupciones
            del sensor de humedad */
        }
    }
}
```

Como se ve, en el programa debemos también definir las funciones encargadas del temporizador y del sensor de humedad.

Las del temporizador:

```
extern int segundos;

//Rutina de interrupción
void intTemporizador(){
    segundos ++;       //Añade un segundo a la cuenta
}

// Fija la frecuencia de interrupciones a una por segundo
void prepararTemporizador(){
    latch = 32764;
    divisor = 3;
    //Valores para una interrupción por segundo
    TIMER0_DAT = latch;
    TIMER0_CNT = divisor;
}

// Inicia el temporizador 0
void iniciarTemporizador()
{
    TIMER0_CNT = TIMER0_CNT | 1 << 7;     // Enciende el bit 7
    DeshabilitarIntHum();                 //Deshabilita las
    //interrupciones por humedad, ya que ya se está regando
}

// Detiene el temporizador 0
void pararTemporizador()
{
    TIMER0_CNT = TIMER0_CNT & ~(1 << 7); // apaga el bit 7
}

//Habilita las interrupciones del temporizador
void HabilitarIntTemp()
{
    //Para ello primero se deshabilitan todas las interrupciones
    DisableInts();
  //Escribir un 1 en el bit correspondiente al temporizador del REG_IE
    IE = IE | 1 << 3;

    //Para acabar, se habilitan todas las interrupciones
    EnableInts();
}
```

Las del sensor de humedad:

```
//Habilita las interrupciones del sensor de humedad
void HabilitarIntHum ()
{
    //Para ello primero se deshabilitan todas las interrupciones
    DisableInts();
    //Escribir un 1 en el bit correspondiente al sensor de humedad
    //del REG_IE
    IE = IE | 1 << 14;

    //Para acabar, se habilitan todas las interrupciones
    EnableInts();
}


//Deshabilita las interrupciones del sensor de humedad
void DeshabilitarIntHum()
{
    //Para ello primero se deshabilitan todas las interrupciones
    DisableInts();

    //Escribir un 0 en el bit correspondiente al sensor del REG_IE
    IE = IE & ~(1 << 14);

    //Para acabar, se habilitan todas las interrupciones
    EnableInts();
}


//Rutina de interrupción
void intHum(){
    OutPort(RCON_REGAR,1);     //Abre las válvulas
    DeshabilitarIntHum();//Deshabilita las interrupciones del sensor
    iniciarTemporizador();//Pone a contar el temporizador
}
```

Si el controlador de humedad lo tratamos por encuesta en vez de por interrupción debemos realizar algunos cambios en nuestro main:

```
void main(){
    extern int segundos;
    segundos = 0;

    //Determinamos la rutina de interrupción del temporizador
    //intTemporizador suma uno a la variable global segundos (lo hará
    //una vez por segundo)
    irqSet(IRQ_TIMER0,intTemporizador);

    //Habilitamos las interrupciones del temporizador
    HabilitarIntTemp();

    //Hacemos que el temporizador interrumpa una vez por segundo
    prepararTemporizador();

    while(1){
        if ((segundos == 300) || (TECLAS_CNT & SELECT)){
            /* Si select se pulsa o llegamos a los 5 minutos se
            interrumpe el riego */
            pararTemporizador();//Paramos el temporizador
            segundos = 0;       //Ponemos a cero la cuenta de segundos
            OutPort(RCON_REGAR,0); //Cerramos las válvulas
            HabilitarIntHum(); /*Volvemos a atender las interrupciones
            del sensor de humedad */
        }
        //Si la humedad ha bajado y no se está regando
        if (InPort(RCON_KHUM) && segundos == 0){
            OutPort(RCON_REGAR,1); //Abre las válvulas
            iniciarTemporizador();//Pone a contar el temporizador
        }
    }
}
```

También deberemos eliminar la llamada a DeshabilitarIntHum() desde iniciarTemporizador(), ya que no hace falta.