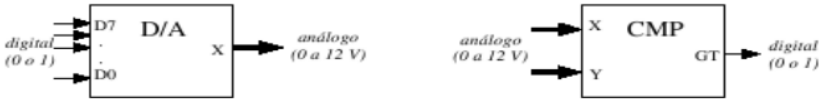
Auxiliar 8
Rodrigo Cánovas
18 de Junio del 2010

1. Problema 1

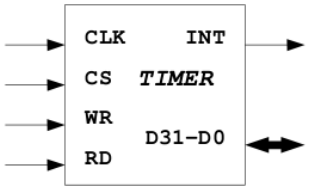


La figura muestra un conversor digital/análogo (D/A) y un comparador (CMP). El conversor digital/análogo recibe como entrada un número entero binario en D7-D0 (entre 0 y 255) y entrega en X una señal análogica consistente en un voltaje entre 0 y 12V proporcional al valor de la entrada. El comparador recibe dos valores análogicos X e Y y los compara entregando un valor binario 1 que indica que X es mayor que Y o 0 en caso contrario.

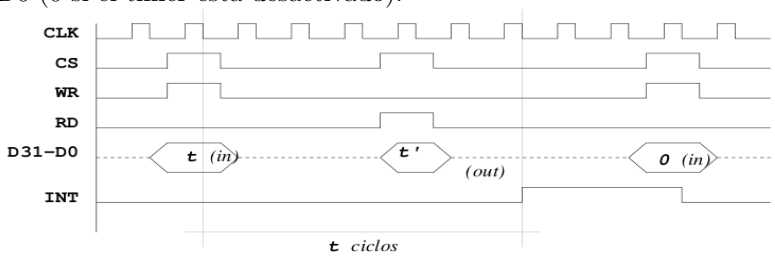
(a) Diseñe e implemente una interfaz de entrada/salida para ambas componentes. Considere un microprocesador con un bus de datos de 8 bits y un bus de direcciones de 16 bits. Haga que al escribir en la dirección 0xffff se establezca la entrada del conversor D/A (preocúpese que esta entrada se mantenga constante hasta que se reescriba un nuevo valor) y al leer en la misma dirección se lea la salida GT de CMP. Las señales X e Y salen del sistema para ser conectadas por el usuario.

(b) Explique cómo un usuario puede usar estas componentes para convertir una señal analógica entre 0 y 12V en un valor binario entre 0 y 255. Escriba en C la rutina **convertirAD()** que hace esta conversión (busque la simplicidad en esta rutina y no la eficiencia).

2. Problema 2



El *timer* de la figura es un dispositivo que se programa para que produzca una interrupción al cabo de una cierta cantidad de ciclos del reloj. Para programarlo se coloca simultáneamente un 1 en la entrada CS, un 1 en WR y la cantidad de ciclos t en $D31 - D0$. Como se muestra en el diagrama de tiempo de la figura, el timer activará la línea INT después de t cicls del reloj, la que permanecerá en 1 hasta que se desactive el timer. El timer se desactiva colocando un 1 en CS, un 1 en WR y un 0 en $D31 - D0$. También se puede recuperar la cantidad de ciclos que restan para que se active la interrupción, lo que se logra colocando un 1 en CS y un 1 en RD. La cantidad de ciclos restantes t(prima) aparece por D31-D0 (0 si el timer está desactivado).



i. Implemente una interfaz que conecte el timer de la figura con un procesador M32. Haga que cada vez que se escribe un dato t en la dirección 0xff ff 00 00 se programa el timer para que interrumpa en t ciclos del reloj, si es que t es mayor que 0, o se desactiva el timer, si t es cero. Además haga que cada vez que se lea esa misma dirección se obtiene la cantidad de ciclos que restan para la interrupción.

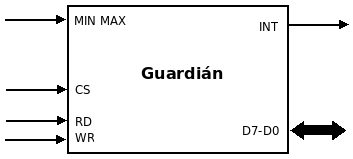
ii. Programe en C los siguientes procedimientos:

progTimer(int t, void (*f)()): programa el timer para que produzca una interrupción en t ciclos del reloj. Además registra el procedimiento f para que se invoque cuando ocurra la interrupción. Si t es 0, se desactiva el timer.

handleTimer(): rutina de atención de la interrupción que desactiva el timer e invoca el procedimiento f.

3. Problema 3

La siguiente figura muestra un circuito guardián que se encarga de vigilar que la temperatura ambiente no salga de un rango dado. El guardián tiene dos puertas de E/S que permiten especificar la mínima y la máxima temperatura aceptable. Cuando la temperatura se sale de estos parámetros, el sensor activa la línea INT.



El siguiente procedimiento sirve para configurar el guardián:

```
void setTempRange(char min, char max){
  /*min y max son enteros de un byte*/
  char *port_min = (char *)0xff00;
  char *port_max = (char *)0xff01;
  *port_min = min;
  *port_max = max;
}
```

Los valores 0 y 255 se usan para indicar al guardián que no se debe producir ninguna interrupción.

a) Implemente la interfaz que se necesita para conectar el guardián con el bus del procesador, de modo que el procedimiento anterior funcione correctamente. Considere un microcontrolador con un bus de datos de 8 bits y un bus de dirección de 16 bits.

b) Implemente en pseudo-C la rutina de atención alertTemp que debe procesar la interrupción causada por el guardián. Esta rutina debe desplegar un mensaje en pantalla (usando printf) y evitar que se siga produciendo la interrupción.