Sistemas Electrónicos Para Automatización

PRÁCTICA 1: *Manejo básico del Connected Launchpad de Texas Instruments.*

13/10/2021

Francisco Javier Román Cortés 4º GIERM

Introducción

En esta primera práctica se busca lograr una toma de contacto con el microcontrolador propuesto para esta asignatura, que como se comentó, a diferencia del MSP del año pasado en "Sistemas Electrónicos", es un microcontrolador mucho más potente y con muchas más posibilidades a nivel industrial.
Concretamente se trabaja con la Connected Launchpad de Texas Instruments en conjunción con el microcontrolador TIVA TMC1294NCPDT. Como toma de contacto de la que se trata, se busca conocer el sistema de desarrollo de este microcontrolador (se programa con Code Composer Studio (CCS)), realizar algunos ejemplos sencillos de funcionamiento del mismo, así como trabajar e investigar las posibilidades que ofrece la librería DriverLib.h

Fundamentos Teóricos

Para esta práctica se necesita conocer el funcionamiento de alguno de los periféricos explicados (botones, leds…). También se usarán las funciones de configuración y manejo de las interrupciones de los pines de entrada/salida. Para realizar esto, se necesita trabajar con algunas funciones sobre las que se puede buscar información de su funcionamiento:

- Funciones de gpio.c/gpio.h

-*GPIOIntTypeSet*

-GPIOIntEnable

-GPIOIntStatus

-GPIOIntClear

-GPIOIntRegister

- Funciones de interrupt.c/interrupt.h (funciones de manejo de interrupciones)

-*IntMasterEnable*

-*IntMasterDisable*

-*IntEnable*

-*IntDisable*

Realización de la práctica

1. Primer ejemplo básico

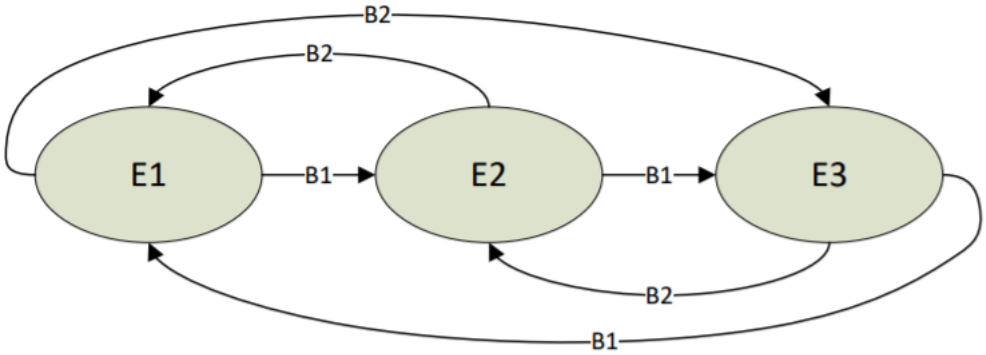
Se busca realizar un programa que gestione una especie de máquina de estados, la cual tiene varios modos de funcionamiento:

-En el modo 1, deberán parpadear los 4 leds con período de 1s y duty cycle de 10%

-En el modo 2, hará una secuencia en la que, empezando por todos los leds apagados, los enciende de uno en uno esperando 1s, y cuando estén todos encendidos esperará 3s antes de repetir la secuencia.

-En el modo 3, se busca una secuencia 1010-0101 con 500ms entre ambos.

El esquema de transiciones como se ve en la memoria corresponde al siguiente:



Una vez recogido en la memoria el enunciado de este primer ejercicio se procede a comentar cómo se ha resuelto. Es importante **destacar** que en este primer ejercicio no hacemos uso de las interrupciones de los pines. Por tanto como se espera, hay altas posibilidades de que aunque se pulse el botón para cambiar de estado, no se capte la pulsación-no pulsación debido a que se encontraba en una orden SysCtlDelay(), que impide que el microcontrolador atienda a las condiciones utilizadas para este primer ejercicio para transicionar entre estados (tipo if).

Comentado esto, se va a comentar brevemente cómo se ha planteado el código para este primer ejercicio, ya que como se adjuntan también los .c comentados no es necesario extenderse tanto.

Al comenzar, se incluyen las librerías necesarias, así como se definen una serie de “macros” para comprobar el estado de los botones y evitar repetir el uso de la orden GPIOPinRead cada vez que se quiere comprobar dicho estado. También se define el valor de MSEC = 40000 para que corresponda a 1ms ya que se ha configurado un reloj a 120MHz. También se define el número de estados de la matriz LED, aunque realmente luego no lo llego a usar en el resto del código.

Se declaran tres variables tipo uint32_t: Puerto, Pin y reloj, de manera que con las dos primeras podremos referirnos a los puertos y a los pines recorriendo su vector en lugar de estar repitiendo la orden cada vez. Se inicializa reloj a 0.

Ahora en la función “main” se declaran una serie de variables: estado, e1, e2, e3, j.

Se configura el reloj a 120MHz, se habilitan los periféricos: GPIOF, J y N.

Se definen los pines tal que:

-F0 y F4 son salidas.

-N0 y N1 son salidas.

-J0 y J1 son entradas.

-Además se configura pull-up en J0 y J1 (los botones).

Se inicializa estado a 0, siendo estado la variable a usar para conmutar en la estructura switch-case.

Dentro del while(1) se introduce una estructura switch-case como se ha comentado para abarcar los 3 modos que tiene el sistema.

En el case 0 (Modo 1 del esquema):

Se inicializa e1 a 0, y usando la línea:

For(j=0;j<4;j++) GPIOPinWrite(Puerto[j],Pin[j],Pin[j]*LED[e1][j]);

Se recorre la “primera fila de la matriz” y por tanto se tienen todos los LEDS apagados.

Con esta lógica, tras la espera de 900ms mediante SysCtlDelay(), se incrementa e1 y se repite la orden anterior, donde ahora se recorrería la “segunda fila de la matriz” y por tanto se tienen todos los LEDS encendidos.

Tras abarcar el funcionamiento de este modo, se introduce unas condiciones en las que si el botón se pulsa y deja de pulsarse (1 pulsación), se pasa a estado = 1 (modo 2 del esquema).

En el case 1 (Modo 2 del esquema):

Siguiendo con la misma lógica se inicializa e2 = 2 que recorre la tercera fila (todos los leds apagados) y se va incrementando e2 de manera que se recorren los diferentes estados de la secuencia: 0000-0001-0011-0111-1111 y se repite. Con SysCtlDelay() se incluyen las esperas de 1s entre estados intermedios y los 3s para pasar de todos encendidos a repetir la secuencia.

En el case 2 (Modo 3 del esquema):

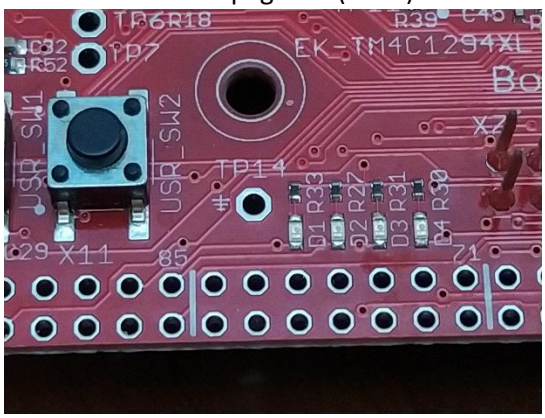
También se sigue con la misma forma de implementación, por tanto, se inicializa e3 = 7 que recorre la octava fila que en este caso corresponde a los LEDS en esta configuración 1010 y tras 500ms se incrementa e3 para lograr la configuración 0101 y tras 500ms se repite esto hasta que entren en juego los botones.

Con esto queda comentado, aunque merece la pena mencionar que en cada case, hay una serie de condiciones según qué botón se pulse para realizar las transiciones que especificaba el esquema.

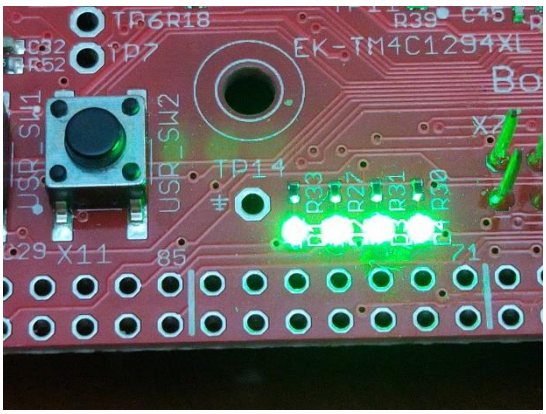
Antes de continuar explicando el Segundo Ejercicio merece la pena mostrar los modos de funcionamiento con imágenes para comprenderlo mejor visualmente:

MODO 1:

-Leds apagados (0000):

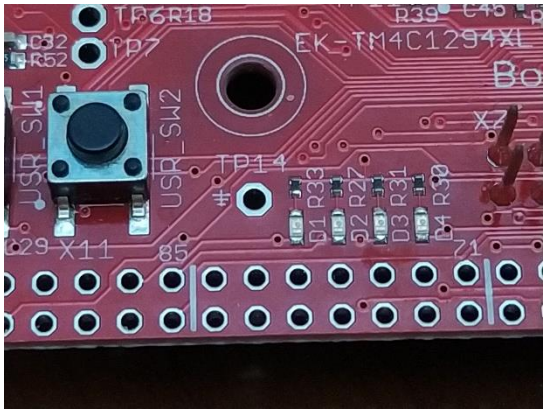


-Leds encendidos (1111):

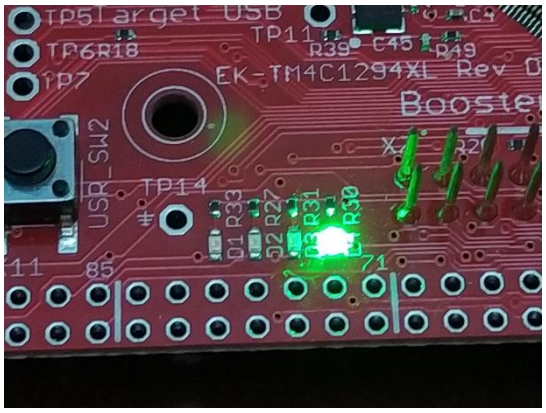


MODO 2:

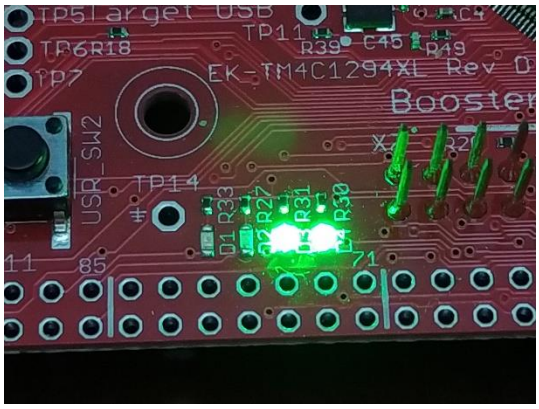
-Leds apagados (0000):



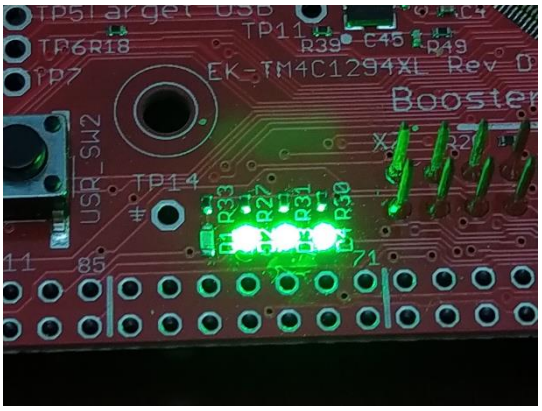
-Led derecha encendido (0001):



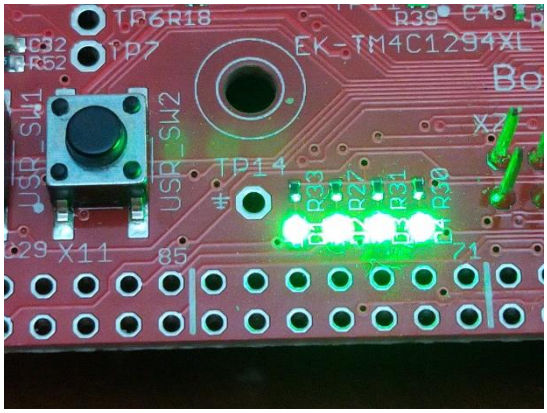
- 2 leds derecha encendidos (0011):



-3 leds encendidos (0111):

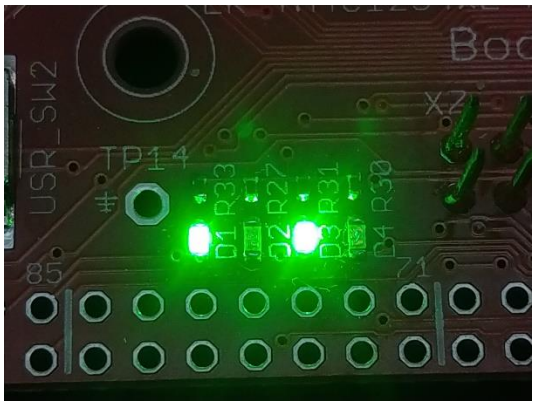


-4 leds encendidos (1111):

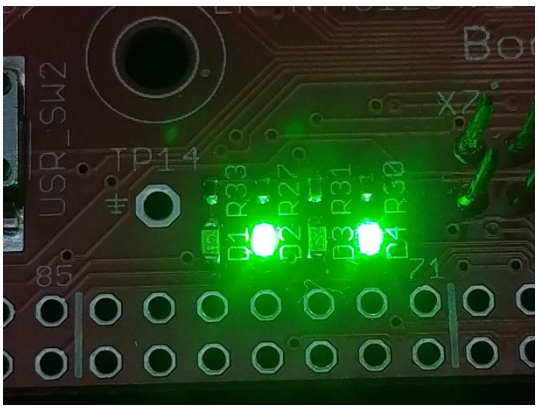


MODO 3:

-Combinación 1010:



-Combinación 0101



2. Segundo ejemplo: con interrupciones

Para este segundo ejemplo, se busca solucionar el problema de la pérdida de pulsaciones haciendo uso de las interrupciones de los pines de entrada/salida, de manera que se atienda a las pulsaciones de los botones de manera inmediata. Para este ejercicio el funcionamiento es exactamente igual al anterior, luego el código es básicamente el mismo, pero añadiendo el tema de las interrupciones, por tanto, solo comentaré los nuevos aspectos de este programa para no repetir lo mismo que el ejercicio 1 en el resto de cosas.

Ahora se declara estado como variable global, ya que se trabaja con ella tanto en la función main como en la función de interrupción.

Ahora hay que confeccionar una rutina de interrupción a la que se accederá cada vez que se pulse uno de los 2 botones (por la configuración realizada de las interrupciones). En esta función se realizan las transiciones entre modos en función del botón pulsado y el modo en que estuviese al pulsar:

- Si se pulsa B1 y estaba en Modo 1 --> Pasar a modo 2
- Si se pulsa B1 y estaba en Modo 2 --> Pasar a modo 3
- Si se pulsa B1 y estaba en Modo 3 --> Pasar a modo 1
- Si se pulsa B2 y estaba en Modo 1 --> Pasar a modo 3
- Si se pulsa B2 y estaba en Modo 2 --> Pasar a modo 1
- Si se pulsa B2 y estaba en Modo 3 --> Pasar a modo 2

Posteriormente se crea una función “enciende_leds” para encender los leds según la matriz sin tener que recurrir a los bucles *for* como se hizo en el ejercicio 1.

Finalmente, en el while(1) se tiene la estructura switch-case igual que antes de manera que se van incrementando e1, e2 y e3 como se comentó y las esperas se siguen haciendo con SysCtlDelay(). No tiene interés repetirlo.

Como conclusión, con esto quedan explicado ambos ejercicios de manera que en la entrega adjuntaré además los códigos de ambos programas, así como un video demostrativo del funcionamiento de ambos ejercicios.