UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA

MICROCONTROLADORES

REPORTE
PRÁCTICA 5
PWM

SONIA PAOLA AGUILLÓN OLAMENDI
261520

PRÁCTICA PWM

1. OBJETIVO

El alumno conocerá y aprenderá a implementar el código de programación para la configuración y uso del PWM utilizando el microcontrolador Tiva TM4C123GH6PM.

2. MARCO TEÓRICO

PWM

La modulación de ancho de pulso está formada por una señal de onda cuadrada que no siempre tiene la misma relación entre el tiempo que está en alto y el tiempo que está en bajo.

Duty Cycle

La **variación de ancho de pulso** consiste en variar los tiempos de encendido y apagado, es decir Ton y Toff. Al cambiar el valor de un PWM, en realidad se están modificando estos tiempos.
Uno de las características más importantes de una señal PWM es su ciclo de trabajo o Duty Cycle, en inglés, ya que este es el que varía en un PWM.

El ciclo de trabajo no es otra cosa que la relación entre el tiempo de encendido y el periodo o tiempo total del PWM.

Cuanto mayor sea el duty cycle, mayor tiempo estará la señal de tensión en alto, sin variar el periodo. Por consecuencia, como el periodo no vario y la suma de Ton y Toff, si el tiempo de encendido aumenta, el tiempo de apagado disminuye.

Este microcontrolador tiene dos bloques PWM como PWM0 y PWM1. Además, cada bloque PWM contiene cuatro generadores PWM y bloques de control. Además, cada generador PWM proporciona dos salidas PWM como pwmA y pwmB. Pero estas dos señales PWM comparten la frecuencia o los bloques de contador. Por lo tanto, dos salidas PWM del mismo generador tendrán la misma frecuencia, pero pueden tener un ciclo de trabajo diferente o pueden usarse como una salida complementaria para controlar medio puente y puente completo en aplicaciones de control de motores. Estas salidas complementarias también proporcionan retardos de banda muerta programables.
Los bloques generadores de PWM se utilizan para generar señales de PWM. Por otro lado, los bloques de control se utilizan para enviar señales de salida PWM a los pines GPIO del microcontrolador TM4C123.

Por lo tanto, cada generador proporciona 8 canales PWM. Por lo tanto, el microcontrolador TM4C123 admite un total de 16 salidas PWM, a saber, M0PWM_n y M1PWM_n, donde n = 0 – 7 y M0 – M1 son para el módulo 0 y el módulo 1.

Configuración PWM

1. Habilitar el reloj del sistema para el módulo PWM usando SYSCTL->RCGCPWM configurando los bits.
2. Habilitar el reloj en el puerto GPIO a través del cual aparece la salida PWM en el pin GPIO relacionado (SYSCTL->RCGCGPIO).
3. Seleccionar la función de divisor de reloj del sistema antes de alimentarlo al módulo PWM. En caso de no desear reducir la frecuencia del módulo desactivarlo.
4. Configurar el pin GPIO de salida PWM como un pin de salida digital, seleccione su función alternativa para el módulo PWM y asignar el pin PWM al pin GPIO aplicable usando los registros GPIOF->AFSEL, GPIOF->PCTL y GPIOF->DEN, respectivamente.
5. Deshabilitar el contador del generador PWM antes de configurar el canal PWM.
6. Seleccionar el modo de contador, ya sea de cuenta ascendente o descendente, configurando o borrando el bit 1 del registro PWMn->_x_CTL.
7. Seleccionar el canal PWM y seleccionar las acciones de la señal de salida cuando el contador se recarga o el contador coincide con el registro PWMxCMPx.
8. Calcular el valor de carga según la frecuencia requerida.
9. Calcular el valor del registro de comparación para el ciclo de trabajo requerido y coloque el valor en PWMx->_x_CMPx
10. Al final, habilitar el contador del generador PWM configurando el bit D0 del registro PWMx->_x_CTL y también habilitar la salida PWM en el pin GPIO con el registro PWMx->ENABLE.

3. METODOLOGÍA

Experimento 1

Usando el módulo 0 de PWM con una frecuencia de reloj del sistema de 50,000,000 Hz, junto con el generador 1 habilitar alguno de los pwm's asociados y obtener un PWM cuya frecuencia sea de 10KHz.

Para comenzar a llevar a cabo la práctica, se inició configurando el PWM.

Inicialmente se habilito el reloj del módulo deseado, ya que en mi caso toco Módulo 0

Módulo 0	Poner 1

Al solicitar módulo 0, generador 1, fue necesario ir a la tabla de la página 1232

|PAG. 1233

M0PWM2	58	PB4 (4)	O	TTL	Motion Control Module 0 PWM 2. This signal is controlled by Module 0 PWM Generator 1.

Por medio de esa tabla podemos observar cual Pin Mux se va a utilizar, en este caso utilizaremos el M0PWM2.

Posteriormente se continua con la tabla de la pag 341, la cual nos dice que el pin 1 habilita el Puerto B del GPIO ya que es el que ocupamos PB**4.**

```
//EXPERIMENTO 1 ------------------------------------------------------------------------
//*/

    SYSCTL->RCGCPWM |= (1<<0); //Enable reloj de modulo PWM0 pag 354 ------------    Se habilita reloj del modulo 0
    SYSCTL->RCGCGPIO |= (1<<1); //Habilitar reloj de GPIO Puerto B  pin 4 (Solo se esta poniendo 1 del puerto b)   - Pag 1233
   // GPIOF->AFSEL |= (1<<3)|(1<<2)|(1<<1); //Control de registros ya sea por GPIO o Otros Pag 672    //ESTO SI SE COMENTA
```

|PAG. 254

Para llevar a cabo el registro RCC, primero se pusieron en cero todos los bits para limpiarlo, posteriormente, habilitamos para que se divida el reloj, por ello se le pone un 1 en el bit 20, es decir, activamos USEPWMDIV para dividir la señal del reloj . Una vez activado, se indica por cuanto se va a dividir la señal del reloj, en este caso indicamos que se dividirá entre 2.

Configure the **Run-Mode Clock Configuration (RCC)** register in the System Control module to use the PWM divide (USEPWMDIV) and set the divider (PWMDIV) to divide by 2 (000).

Run-Mode Clock Configuration (RCC)
Base 0x400F.E000
Offset 0x060
Type RW, reset 0x078E.3AD1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved				ACG	SYSDIV				USESYSDIV	reserved	USEPWMDIV	PWMDIV			reserved
Type	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0

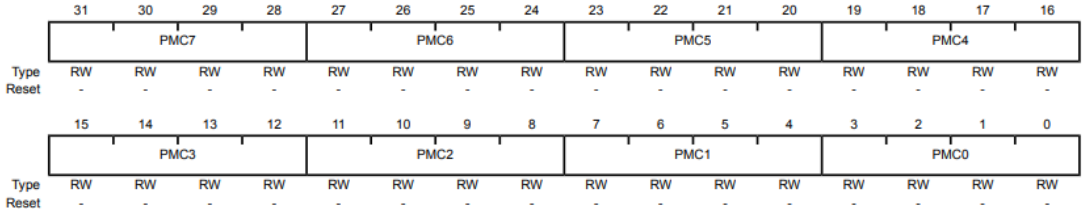
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		PWRDN	reserved	BYPASS	XTAL					OSCSRC		reserved			MOSCDIS
Type	RO	RO	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO	RO	RW
Reset	0	0	1	1	1	0	1	0	1	1	0	1	0	0	0	1

Posteriormente en el registro AFSEl indicamos que queremos que en el pin 4 del puerto B tenga una función alternativa. Después en el registro PCTL indicaremos cual es la función que hará, ya que queremos habilitar el pin PB4 la función de M0PWM2, queremos que la función sea la del bit 4 como se muestra.

PB4	58	AIN10	-	SSI2Clk	-	M0PWM2	-	-	T1CCP0	CAN0Rx	-	-	-

M0PWM2	Bit 4

|Pagina 689



Bit/Field	Name	Type	Reset	Description
31:28	PMC7	RW	-	Port Mux Control 7 This field controls the configuration for GPIO pin 7.
27:24	PMC6	RW	-	Port Mux Control 6 This field controls the configuration for GPIO pin 6.
23:20	PMC5	RW	-	Port Mux Control 5 This field controls the configuration for GPIO pin 5.
19:16	PMC4	RW	-	Port Mux Control 4 This field controls the configuration for GPIO pin 4.
15:12	PMC3	RW	-	Port Mux Control 3 This field controls the configuration for GPIO pin 3.
11:8	PMC2	RW	-	Port Mux Control 2 This field controls the configuration for GPIO pin 2.
7:4	PMC1	RW	-	Port Mux Control 1 This field controls the configuration for GPIO pin 1.
3:0	PMC0	RW	-	Port Mux Control 0 This field controls the configuration for GPIO pin 0.

Por medio del registro DEN, se indica que el pin 4 será digital. A continuación, se deshabilita el generador 1 del PWM0 para poder configurarlo.

En mi caso, elegí el generador A, ya que no se especifico cual utilizar, se le dio un valor hexadecimal de 008C para indicar que cuando coincidan los flancos, el otro baje, es decir, se va a tener un contador descendente cuando el contador llegue al valor de la carga y va a bajar cuando sea igual al valor del comparador va a bajar, generando el pulso.

```c
PWM.c > Configura_Reg_PWM1(uint16_t)

#include "lib/include.h"

extern void Configura_Reg_PWM1(uint16_t freq)
{

//EXPERIMENTO 1 ------------------------------------------------------------------------
//*/

    SYSCTL->RCGCPWM |= (1<<0); //Enable reloj de modulo PWM0 pag 354 ------------    Se habilita reloj del modulo 0
    SYSCTL->RCGCGPIO |= (1<<1); //Habilitar reloj de GPIO Puerto B  pin 4 (Solo se esta poniendo 1 del puerto b)   - Pag 1233
   // GPIOF->AFSEL |= (1<<3)|(1<<2)|(1<<1); //Control de registros ya sea por GPIO o Otros Pag 672    //ESTO SI SE COMENTA

    SYSCTL->RCC &= 0xFFF0FFFF; //Indicamos en los pines 17-19 que se divida entre 2
    SYSCTL->RCC |=(1<<20);  //Activamos USEPWMDIV para dividir la señal del reloj
    SYSCTL->RCC |= 0xFFF0FFFF; //Indicamos en los pines 17-19 que se divida entre 2

    GPIOB->AFSEL |= (1<<4); //Indicamos que queremos una funcion alterna en el pin b4
    GPIOB->PCTL |= 0x00040000; //Combinado con la tabla Pag 1351 y el registro PCTL le digo que es pwm Pag 689
    //Aqui habilitamos la funcion 4 del pin PB4 - en el port mux 4
    //Se usa el enmascaramiento porque no tiene valores por default

    GPIOB->DEN |= (1<<4); // para decirle al pin 4 que es digital Pag 682
    // PWM0->CC &= ~(1<<8);  //Enable o Disable Divisor  Pag 1747 ------TIVA GRANDE ESTO SI SE COMENTA

    //El número indica el generador que se va a usar
    PWM0->_1_CTL &= ~(1<<0);  //Se deshabilita el generador 1 del PWM0 para poder configurarlo POR??
    PWM0->_1_GENA = 0x008C; //Se utilizara el generador A del generador 1 del
    // El 8C es para decir que cuando coincidan, el otro baje para que se genere el PWM
    //Se va a tener un contador descendente cuando el contador llegue al valor de la carga y va a bajar cuando sea igual
    //al valor del comparador va a bajar
```

El siguiente paso es llevar a cabo los cálculos por medio de la siguiente formula:

$$Cuentas = \frac{fclk}{fpwm}$$

Para 10khz:

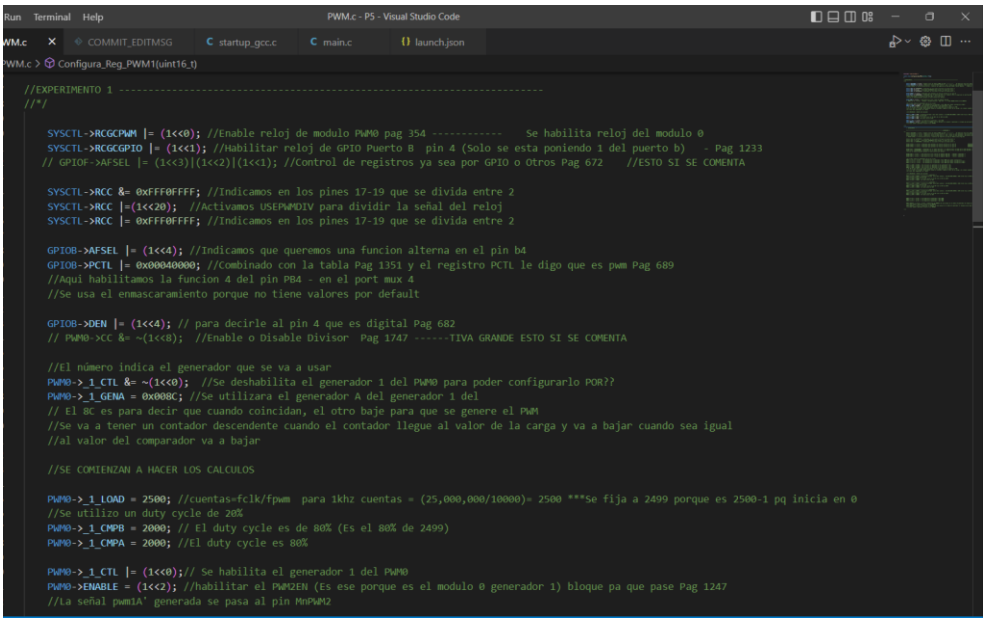
$$cuentas = (\frac{25{,}000{,}000}{10000}) = 2500$$

Se busca tener un duty cycle del 80%, por ello, el valor dado al comparador, es el 20% de 2500, lo cual da un total de 2000.

Por ultimo, se vuelve a habilitar el generador 1 del PWM0 y el PWM2 para que la señal pwm1A generada se pasa al pin MnPWM2.

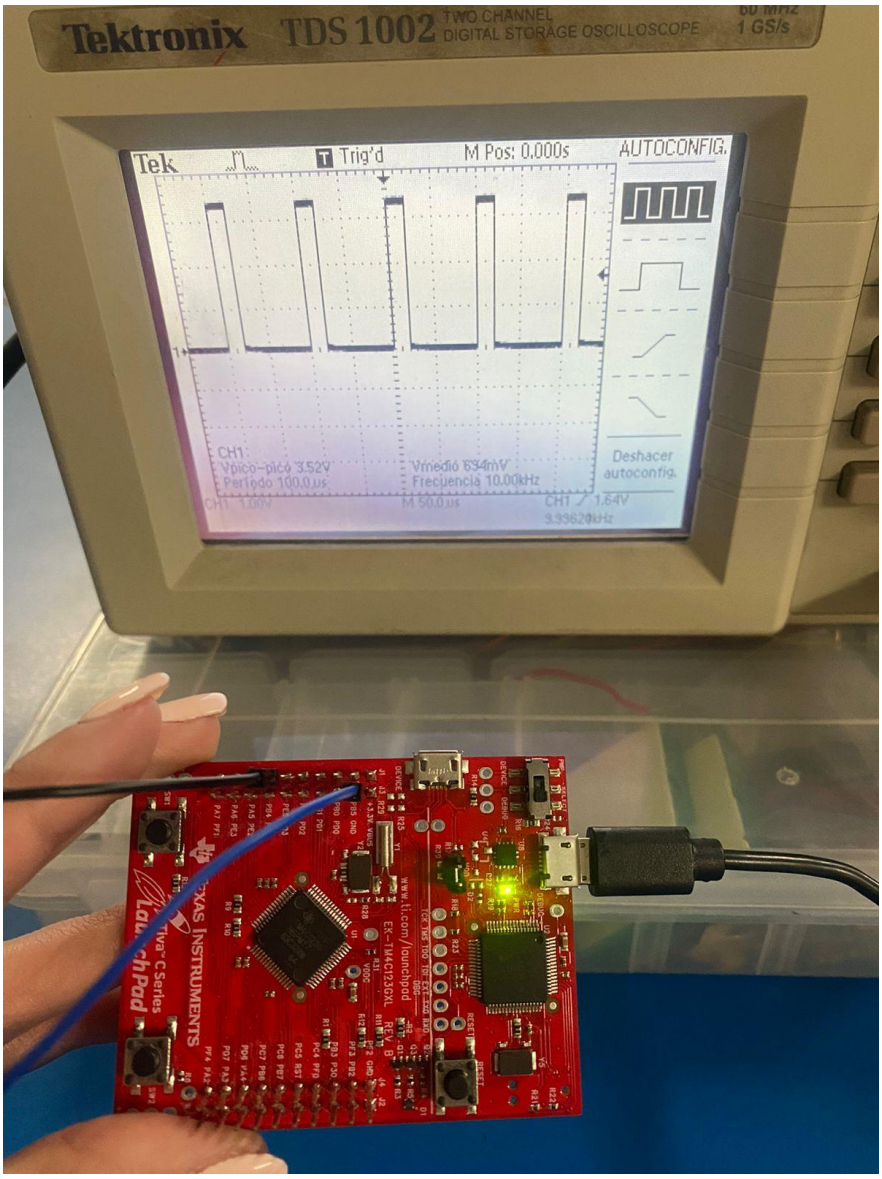
```c
    //SE COMIENZAN A HACER LOS CALCULOS

    PWM0->_1_LOAD = 2500; //cuentas=fclk/fpwm  para 1khz cuentas = (25,000,000/10000)= 2500 ***Se fija a 2499 porque es 2500-1 pq inicia en 0
    //Se utilizo un duty cycle de 20%
    PWM0->_1_CMPB = 2000; // El duty cycle es de 80% (Es el 80% de 2499)
    PWM0->_1_CMPA = 2000; //El duty cycle es 80%

    PWM0->_1_CTL |= (1<<0);// Se habilita el generador 1 del PWM0
    PWM0->ENABLE = (1<<2); //habilitar el PWM2EN (Es ese porque es el modulo 0 generador 1) bloque pa que pase Pag 1247
    //La señal pwm1A' generada se pasa al pin MnPWM2

```

Una vez terminado el código se probó y después de varios arreglos de obtuvieron los siguientes resultados.



Además, en el MAIN.c se configuro la velocidad del reloj a 50MHz y el PWM a 10KHz

```
Configurar_PLL(_50MHZ);  //Confiuracion de velocidad de reloj - Experimento 1
//Configurar_PLL(_20MHZ);  //Confiuracion de velocidad de reloj - Experimento 2
//Configura_Reg_ADC0();
//Configurar_UART7();//Yo FCLK 50MHZ Baudrate 57600
//Configurar_GPIO(); //Se configura GPIO
//Configura_Reg_ADC0();
Configura_Reg_PWM1(10000);//Configuro a 10khz el pwm
```



Se adjuntan videos en el git como evidencia

EXPERIMENTO 3

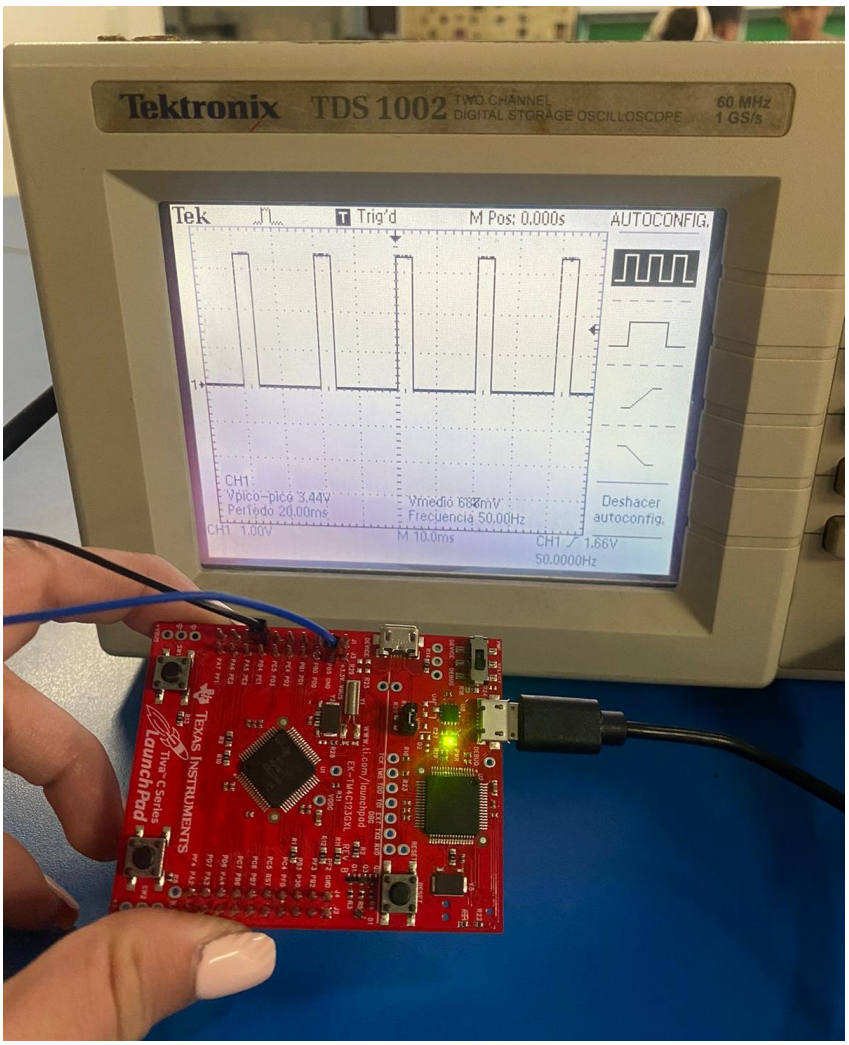
Para llevar a cabo este experimento, se utilizo la configuración anterior como base, se fueron siguiendo los mismos pasos. Se habilito el reloj del modulo 0, posteriormente se habilito el reloj del puerto B y E, debido a que los pines por donde nuestra señal generada saldrá son los pines E4, B4 y B6. Otra cosa que cambio fue que, en este caso, la señal del reloj se dividirá en 8, debido a que es la única forma en la que todos los bits quepan.

Todas las configuraciones se hicieron iguales para cada generador.

El arreglo que se utilizo fue:

Generador 0	Pin B6
Generador 1	Pin B4
Generador 2	Pin E4

De esta manera, haciendo los cálculos necesarios, se obtuvieron los siguientes resultados, logrando leer la señal en todos los pines.



Se adjuntan videos en el git como evidencia