Basic Express BX-24
Nota de aplicación

Programación del Timer1 para modulación de pulsos de anchura duales

Temporizador Timer1 y la modulación de pulsos duales

El procesador BX-24 incluye un temporizador integrado denominado Timer1. El temporizador puede utilizarse para múltiples funciones, una de las cuales es para generar salidas de hardware duales PWM sin consumir recursos de la CPU. Los pines OC1A y OC1B se utilizan para estas salidas, que corresponden con los pines 26 y 27 respectivamente. Puede elegir entre las salidas PWM de 8, 9 ó 10 bits, así como entre diferetes frecuencias.

El temporizador es capaz de operar con 5 frecuencias de ticks, desde aproximadamente 7,20 a 7.37 MHz. Después de establecer la frecuencia, la frecuencia de ticks se convierte a tasas de pulsos PWM desde 3,52 Hz (para salidas de 10 bits) a 14,4 kHz (para salidas de 8 bits).

A continuación se incluye una lista con los principales registros necesarios para acceder al temporizador Timer1:

Tipo	Nombre	Descripción
Byte	TCNT1L	Timer/Counter1 Low Byte
Byte	TCNT1H	Timer/Counter1 High Byte
Byte	TCCR1B	Timer/Counter1 Control Register B
Byte	TCCR1A	Timer/Counter1 Control Register A

Consulte también las páginas 32 a 38 del fichero AT90_8535.pdf si desea obtener más detalles del temporizador Timer1, que en realidad recibe el nombre Timer/Counter1. Este fichero viene incluido en la instalación de BasicX y los documentos del chip Atmel 8535, que se utiliza como procesador del BX-24.

Advertencia -- Timer1 debería sólo utilizarse siempre que no afecte al funcionamiento de otros recursos del sistema, tales como InputCapture y OutputCapture, los cuales dependen de Timer1.

Programación del temporizador Timer1 para Modulación de Pulsos (PWM) de anchura dual.

Iniciación

Este ejemplo ilustra cómo utilizar el temporizador Timer1 para generar salidas PWM de 8 bits en los pines OC1A y OC1B, que son los pines 26 y 27 respectivamente del sistema BX-24. Tenga en cuenta que el pin 27 es compartido por el indicador LED verde integrado, que confirma de manera visual el funcionamiento de la modulación de ancho de pulso en dicho pin.

El primer paso del proceso de la iniciación es detener el temporizador (Timer1):

```
Register.TCCR1B = 0
```

El siguiente paso es configurar el Timer1 con el modo PWM de 8 bits (los modos de 9 y 10 bits son similares):

```
Const PWMmode8bit  As Byte = bx0000_0001
Const PWMmode9bit  As Byte = bx0000_0010
Const PWMmode10bit As Byte = bx0000_0011
Const PWMmodeOff   As Byte = bx0000_0000

Register.TCCR1A = PWMmode8bit
```

A continuación, debemos definir los ciclos de activación para cada pin, que están cargados en los registros 0CR1A y OCR1B. Los ciclos de activación pueden ser un número de 8, 9 ó 10 bits. En este ejemplo, utilizaremos valores de 8 bits para los ciclos de activación de 25 % y 75 % (64 y 191). Tenga en cuenta que es necesario escribir bytes superiores en primer lugar para estos registros:

```
Register.OCR1AH = 0
Register.OCR1AL = 64    ' = 25 %

Register.OCR1BH = 0
Register.OCR1BL = 191   ' = 75 %
```

Inicio del temporizador Timer1

Ahora ya podemos iniciar el temporizador escribiendo uno de los 5 valores enumerados para el registro de control del temporizador (TCCR1B). Los valores permitidos son los siguientes:

TCCR1B Value	Tick Frequency (Hz)	8-bit PWM Pulse Rate (Hz)
1	7 372 800	14 456
2	921 600	1 807
3	115 200	225.9
4	28 800	56.47
5	7 200	14.12

En este ejemplo, utilizaremos la frecuencia más baja de 7.2 kHz:

```
Register.TCCR1B = 5
```

Aquí la tasa de pulsos es igual a la frecuencia del temporizador Timer1 dividido entre 510, que en este caso es aproximadamente 14.1 Hz. Para salidas de 9 bits, el divisor es 1022, y para salidas de 10 bits el divisor es 2046.

El último paso es conectar el temporizador a los pines OC1A y OC1B:

```
Const MaskOC1A As Byte = bx1000_0000
Const MaskOC1B As Byte = bx0010_0000

' Inicializar ambos pines a niveles lógicos bajos.
Call PutPin(PinOC1A, bxOutputLow)
Call PutPin(PinOC1B, bxOutputLow)

' Activar PWM para ambos pines.
Register.TCCR1A = Register.TCCR1A Or MaskOC1A
Register.TCCR1A = Register.TCCR1A Or MaskOC1B
```

Después de este punto puede modificar continuamente los ciclos de activación PWM escribiendo nuevos valores en los registros OCR1A y OCR1B. Las salidas PWM se generan en un segundo plano.

Código de ejemplo

El código fuente para este programa se suministra en un fichero independiente. El nombre del fichero es PWMexample.bas.

2.00.A

Traducción Española: Alicia Bernal, revision: Pablo Pompa www.superrobotica.com