UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Medición de tiempos con la familia de microcontroladores STM32

Apellidos, nombre	Yuste Pérez, Pedro (pyuste@disca.upv.es)
Departamento	Departamento de Informática de Sistemas y Computadores
Centro	Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

1 Resumen de las ideas clave

En este artículo se introduce el modo de medición de tiempos en los timers de la familia STM32 de microcontroladores ARM Cortex. Se explica cómo se configura y qué funciones aportan las librerías CMSIS para su utilización. Se suponen conocimientos básicos del sistema de relojes de esta familia de microcontroladores y de la utilización de las librerías CMSIS.

2 Introducción

Una característica fundamental de los sistemas empotrados basados en microcontrolador es la capacidad para trabajar con tiempos con una gran precisión. El periférico que realiza estas funciones son los timers. Cada fabricante tiene un diseño propio para los timers, aunque los fundamentos son los mismos. Los timers suelen tener dos modos. Uno habitualmente llamado de captura y el otro de comparación. El primer modo se puede utilizar para medir el tiempo entre dos eventos. Este artículo se basa en el conocimiento básico de los timers de la familia STM32 para introducir la utilización del modo captura y su configuración mediante librerías CMSIS. Se asume que el lector ya tiene un conocimiento de esta familia de microcontroladores y de las librerías CMSIS.

3 Objetivos

Una vez que el alumno se lea con detenimiento este documento, será capaz de:

- Comprender el funcionamiento del modo captura de los timers de los microncontroladores STM32
- Escribir programas que utilizan los timers para medir tiempos.

4 Desarrollo

Cada uno de los timers de propósito general de la familia de microcontroladores STM32 tiene 4 canales independientes que se pueden utilizar para:

- Captura de señal (de entrada)
- Comparación de cuenta (para salida)
- Generación automática de señales PWM
- Salida en modo "un pulso"

En primer lugar nos vamos a centrar en cómo utilizar el modo captura para medir tiempos, luego veremos cómo se configura y qué funciones hay disponibles en las librerías CMSIS para facilitar su uso.

4.1 Modo captura

En modo captura, los registros de los canales (TIMx_CCRy) se usan para almacenar el valor del contador cada vez que hay una transición en la entrada

correspondiente. (ICy). De esta manera podemos medir el instante (respecto al reloj interno) en el que ha sucedido esa transición. En la figura 1 podemos apreciar un ejemplo: La señal de la entrada tiene un flanco positivo en el instante 3 y de bajada en el instante 10. La duración es 10-3=7 ticks de reloj.

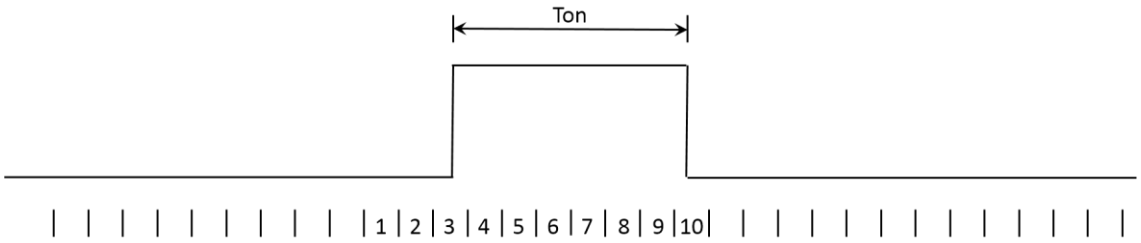

Figura 1. Diagrama de bloques del circuito de base de tiempos

El timer indica que se ha producido un evento de este tipo activando el bit CCyIF del registro de estado del timer (TIMx_SR) y generando una interrupción o petición de DMA si se han habilitado previamente. Hay cuatro de estos bits, uno por canal: CC1IF, CC2IF, CC3IF y CC4IF. Este bit se borra al hacer una lectura en el registro asociado al canal (TIMx_CCRy). Además, si llega un nuevo flanco sin que se haya leído el registro se activa el bit de "overflow" correspondiente al canal (CCyOF). El significado del contenido del bit CCyIF es

0: No ha sucedido ninguna captura.

1: El valor del contador del timer se ha copiado en el registro TIMx_CCRy, lo que quiere decir que se ha detectado un flanco en la entrada ICy de acuerdo con la polaridad configurada.

4.2 ¿Cómo se configura?

El primer paso es decirle al timer de dónde queremos tomar la señal a medir. Esto se hace en el campo CCyS (Capture/Compare "y" Selection) del registro TIMx_CCMRy (Timer "x" Capture/Compare mode register "y"). Son válidos los siguientes valores:

00: canal CCy configurado como salida.

01: canal CCy configurado como entrada, tomándola del pin TI1.

10: canal CCy configurado como entrada, tomándola del pin TI2.

11: canal CCy configurado como entrada, tomándola de la señal TRC.

El procedimiento es el siguiente:

1. Programar la duración del filtro de entrada. Se puede configurar para eliminar los pulsos que sean más cortos de una duración concreta. Se hace con los bits ICxF del registro TIMx_CCMRy.
2. Elegir el flanco que queremos que dispare la captura, escribiendo en los bits CCyP y CCyNP (polaridad positiva y negativa respectivamente) del registro TIMx_CCER .

3. Programar el prescaler de entrada: cuántos de esos flancos queremos que pasen antes de cada captura. Si queremos contar todos debemos escribir 00 en el registro TIMx_CCMRy.
4. Habilitar la captura poniendo a uno el bit CCyE en el registro TIMx_CCER.
5. Habilitar las interrupciones y/o DMA si es necesario.

4.3 Implementación, librerías.

Para simplificar la configuración de este modo podemos utilizar las librerías de periféricos que nos proporciona el fabricante del microcontrolador. En concreto la librería stm32f4xx_tim incluye las siguientes funciones específicas para el modo de captura:

```
/* Input Capture management ******************************************************/

void TIM_ICInit(TIM_TypeDef* TIMx, TIM_ICInitTypeDef* TIM_ICInitStruct);

void TIM_ICStructInit(TIM_ICInitTypeDef* TIM_ICInitStruct);

void TIM_PWMIConfig(TIM_TypeDef* TIMx, TIM_ICInitTypeDef* TIM_ICInitStruct);

uint32_t TIM_GetCapture1(TIM_TypeDef* TIMx);

uint32_t TIM_GetCapture2(TIM_TypeDef* TIMx);

uint32_t TIM_GetCapture3(TIM_TypeDef* TIMx);

uint32_t TIM_GetCapture4(TIM_TypeDef* TIMx);

void TIM_SetIC1Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);

void TIM_SetIC2Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);

void TIM_SetIC3Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);

void TIM_SetIC4Prescaler(TIM_TypeDef* TIMx, uint16_t TIM_ICPSC);
```

Se recomienda al lector la búsqueda de las definiciones del tipo TIM_ICInitTypeDef en el archivo de librería stm32f4xx_tim.h y de las funciones void TIM_ICInit() y TIM_SetIC1Prescaler() en el archivo de librería stm32f4xx_tim.c.

Merecen especial atención dos funciones, la primera es GetCapturex, que devuelve el valor del registro del canal, con la captura del instante en que se ha recibido el flanco.

```
uint32_t TIM_GetCapture1(TIM_TypeDef* TIMx)
```

La segunda función importante es GetFlagStatus, que nos devuelve el estado del canal, indicando si ha habido una nueva captura. Un ejemplo de utilización conjunta es el siguiente:

```
while (TIM_GetFlagStatus(TIM4, TIM_FLAG_CC1)==RESET);
Instante_del_flanco= TIM_GetCapture1(TIM4);
```

En este ejemplo el bucle inicial espera a que haya disponible una medida (evento de captura) en el canal 1 del timer 4. Cuando esto sucede se lee el contenido del canal.

5 Cierre

A lo largo de este objeto de aprendizaje hemos tratado los fundamentos del modo captura de los timers de propósito general de la familia STM32 y se ha visto el código necesario para configurarlos. Para comprobar que se ha entendido se recomienda hacer el siguiente ejercicio:

- Escribe un programa que calcule de manera continua el tiempo a nivel alto de una señal que entra por un pin.
- La señal podrá tener tiempos a nivel alto entre 1 y 2 ms.
- El programa deberá devolver el tiempo en microsegundos.

6 Bibliografía

[1] STMicroelectronics: "RM0090 Reference manual. STM32F40xxx, STM32F41xxx, STM32F42xxx, STM32F43xxx advanced ARM-based 32-bit MCUs", Doc ID 018909 Rev 4. 2013. URL: http://www/st.com