UTFPR - DAELN - CT
Sistemas Embarcados - CSW42
Prof. Douglas Renaux
Raissa A N Higa

Lab 2 - Relatório

Dada as especificações do laboratório, sabe-se que o programa necessita de ao menos algum meio de contar o tempo, um Led e um botão.Sendo assim, ao estudar o manual da placa Tiva verificou-se a existências de botões e Leds com sua respectiva pinagem, além de alguns timers de propósito geral que podem ser usados como temporizadores, podendo ser configurados no modo periódico ou one-shot, como 16 ou 32 bits, e com ou sem pre-scale.

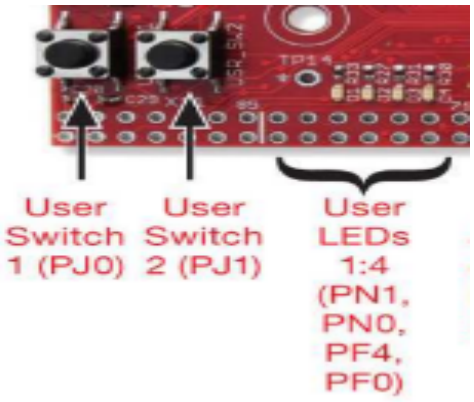


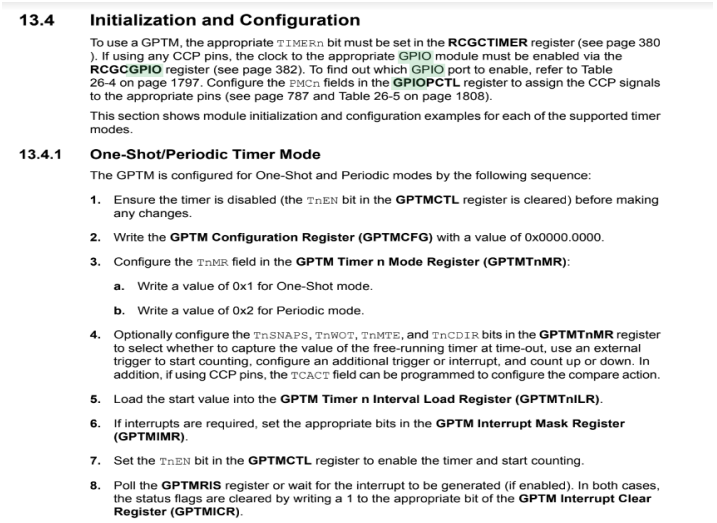
Table 13-3. General-Purpose Timer Capabilities

Mode	Timer Use	Count Direction	Counter Size	Prescaler Size[a]
One-shot	Individual	Up or Down	16-bit	8-bit
	Concatenated	Up or Down	32-bit	-
Periodic	Individual	Up or Down	16-bit	8-bit
	Concatenated	Up or Down	32-bit	-
RTC	Concatenated	Up	32-bit	-
Edge Count	Individual	Up or Down	16-bit	8-bit
Edge Time	Individual	Up or Down	16-bit	8-bit
PWM	Individual	Down	16-bit	8-bit

a. The prescaler is only available when the timers are used individually

O manual do processador mostra mais detalhes sobre a inicialização e configuração dos pinos dos Timers e GPIOs, além de mostrar seus endereços, registradores e meios de interrupções. Os GPIOs podem atuar como input ou output, dependendo das configurações setadas.

- GPIO Port J (AHB): 0x4006.0000
- GPIO Port K (AHB): 0x4006.1000
- GPIO Port L (AHB): 0x4006.2000
- GPIO Port M (AHB): 0x4006.3000
- GPIO Port N (AHB): 0x4006.4000



13.4 Initialization and Configuration

To use a GPTM, the appropriate TIMERn bit must be set in the **RCGCTIMER** register (see page 380). If using any CCP pins, the clock to the appropriate GPIO module must be enabled via the **RCGCGPIO** register (see page 382). To find out which GPIO port to enable, refer to Table 26-4 on page 1797. Configure the PMCn fields in the **GPIOPCTL** register to assign the CCP signals to the appropriate pins (see page 787 and Table 26-5 on page 1808).

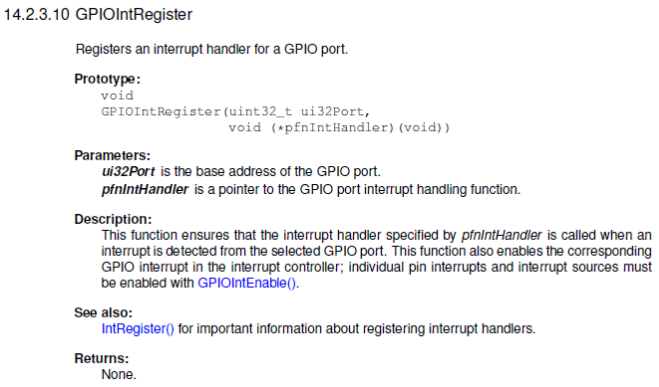
This section shows module initialization and configuration examples for each of the supported timer modes.

13.4.1 One-Shot/Periodic Timer Mode

The GPTM is configured for One-Shot and Periodic modes by the following sequence:

1. Ensure the timer is disabled (the TnEN bit in the **GPTMCTL** register is cleared) before making any changes.
2. Write the **GPTM Configuration Register (GPTMCFG)** with a value of 0x0000.0000.
3. Configure the TnMR field in the **GPTM Timer n Mode Register (GPTMTnMR)**:
   a. Write a value of 0x1 for One-Shot mode.
   b. Write a value of 0x2 for Periodic mode.
4. Optionally configure the TnSNAPS, TnWOT, TnMTE, and TnCDIR bits in the **GPTMTnMR** register to select whether to capture the value of the free-running timer at time-out, use an external trigger to start counting, configure an additional trigger or interrupt, and count up or down. In addition, if using CCP pins, the TCACT field can be programmed to configure the compare action.
5. Load the start value into the **GPTM Timer n Interval Load Register (GPTMTnILR)**.
6. If interrupts are required, set the appropriate bits in the **GPTM Interrupt Mask Register (GPTMIMR)**.
7. Set the TnEN bit in the **GPTMCTL** register to enable the timer and start counting.
8. Poll the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the appropriate bit of the **GPTM Interrupt Clear Register (GPTMICR)**.

No manual do TivaWare situa-se toda a descrição do software, descrevendo as funções implementadas e seus propósitos gerais, mostrando os parâmetros e saídas, além de exemplos de implementações. Interessando mais as funcionalidades que devem ser usadas dos GPIOs e timers, por exemplo as funções de enable , inicialização, configurações e interrupções.



14.2.3.10 GPIOIntRegister

Registers an interrupt handler for a GPIO port.

Prototype:

```
void
GPIOIntRegister(uint32_t ui32Port,
                void (*pfnIntHandler)(void))
```

Parameters:

ui32Port is the base address of the GPIO port.
pfnIntHandler is a pointer to the GPIO port interrupt handling function.

Description:

This function ensures that the interrupt handler specified by *pfnIntHandler* is called when an interrupt is detected from the selected GPIO port. This function also enables the corresponding GPIO interrupt in the interrupt controller; individual pin interrupts and interrupt sources must be enabled with GPIOIntEnable().

See also:

IntRegister() for important information about registering interrupt handlers.

Returns:

None.

Lendo os manuais e observando os exemplos disponíveis, nota-se um padrão de implementação dos periféricos a serem utilizados que em sua maioria necessitam primeiramente ter seu acesso ativado, e então setar seus configurações e modos, se forem usados com interrupções, deve-se habilitar a função de interrupção e dizer quais sua função tratadora, após isso tudo habilita-se o periférico.

Examinando a biblioteca driverlib, encontra-se diversos arquivos .c e .h os quais controlam e manipulam seus componentes.Nos .h, há as declarações das constantes dos pinos, valores de atribuídos a modos de funcionamento e das funções.Enquanto nos .c, as implementações das funções.

Em particular sobre o que é de relevância para esse projeto, estão as bibliotecas gpio, timer , sysctl e interrupt.Que detalham funções como: SysCtlPeripheralEnable, TimerDisable, TimerConfigure, TimerLoadSet, TimerIntEnable, IntEnable, TimerEnable, GPIOPinTypeGPIOInput, GPIOPinWrite , entre outras.

Na realização do laboratório, as maiores dificuldades encontradas foram as configurações iniciais da IDE IAR Workbench para a plataforma Tiva, que possuíam diversos detalhes de linkers e setups, que ocasionaram muitos erros ao compilar. Além disso, configurar e habilitar corretamente o clock do sistema para os contadores foi um grande, pois além dos diversos modos de funcionamento possíveis, haviam muitas chamadas de funções que não funcionam para um ou outro tipo de placa, o que gerou confusão.

Ao final do projeto, foi possível ativar interrupções, acionar o Led , fazer a contagem do tempo e detectar o acionamento do botão. Porém, não consegui implementar a função que printa graficamente no terminal i/o o tempo obtido, através das funções printf, cout , ou de uart.

Fontes: Imagens retiradas dos manuais TivaWare™ Peripheral Driver Library USER'S GUIDE, Getting Started with TivaWare™ for C Series e Tiva TM4C1294NCPDT Microcontroller DATA SHEET.