SENAI

Processadores Embarcados

Atividade em Dupla

Prof° Fernando Simplicio

Processadores Embarcados

- **Configure os pinos a seguir como:**

//CONFIGURADO COMO SAÍDA E DRIVER HABILITADO – NÍVEL 1 INICIAL
PTA19/XTAL0/UART1_TX/FTM_CLKIN1/LPTMR0_ALT1

//CONFIGURADO COMO ENTRADA, PULL-DOWN HABILITADO.
PTB0/LLWU_P5/ADC0_SE8/TSI0_CH0/I2C0_SCL/FTM1_CH0

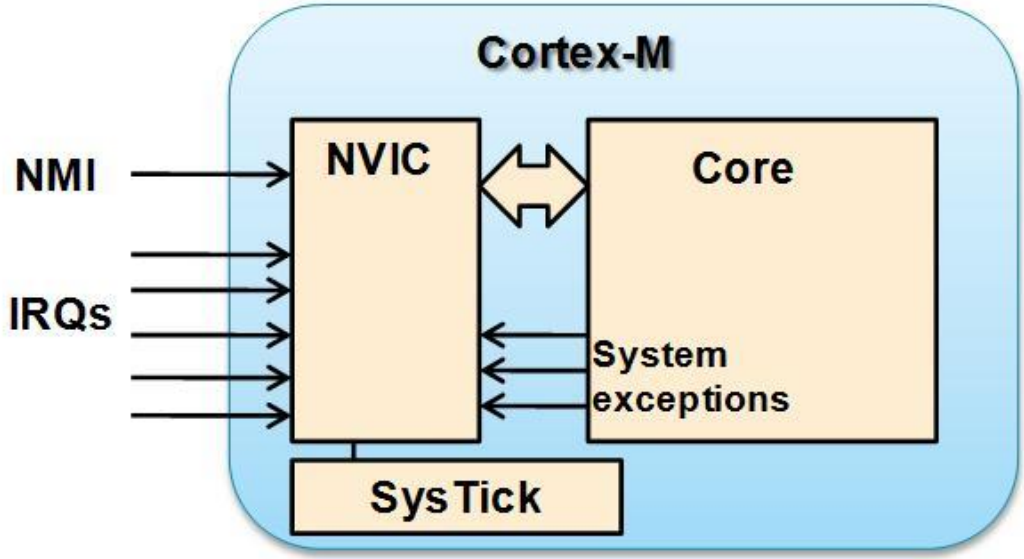
//CONFIGURADO COMO ANALÓGICO
PTC0/ADC0_SE14/TSI0_CH13/EXTRG_IN/CMP0_OUT

//CONFIGURADO COMO SAÍDA E DRIVER DESABILIADO – NÍVEL 0 INICIAL
PTC13/FTM_CLKIN17

Obs:Não se esqueça de ligar o CLOCK no PORT!!!

Processadores Embarcados

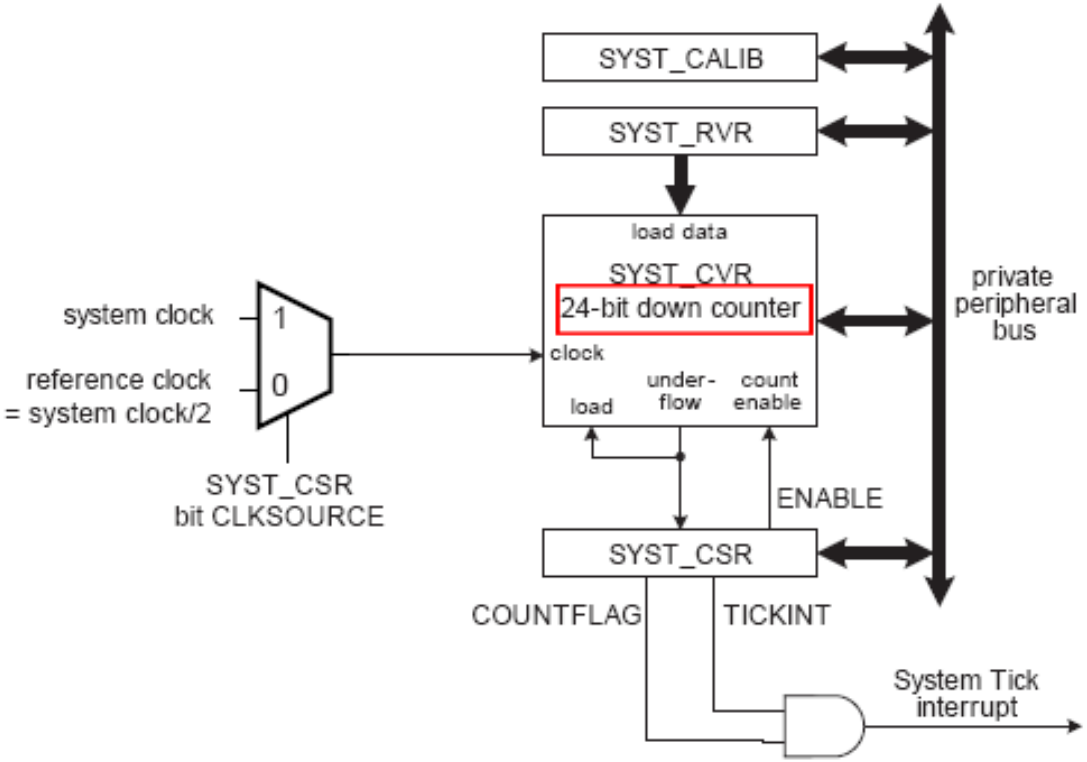
- **System Timer (SysTick)**
- **Os microcontroladores ARM Cortex M possuem um timer de sistema (System Timer – SysTick).**
- **Este timer pertence ao núcleo ARM e não é um periférico.**
- **Muito utilizado em Sistemas Operacionais e aplicações onde existe a necessidade de periodicidade na execução das tarefas.**

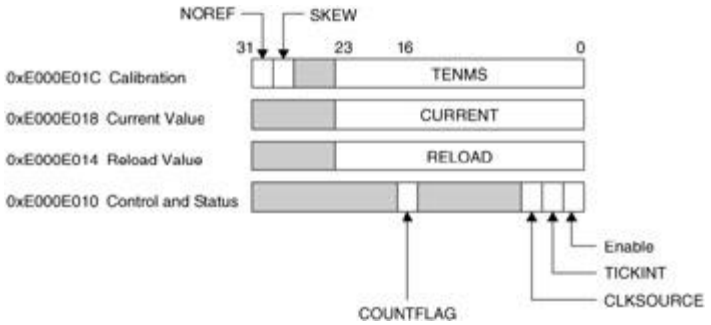


Processadores Embarcados

- **System Timer (SysTick)**
- **Possui 24 bits e é um contador decrescente com intervalor normalmente máximo de 10ms.**
- **O underflow (passagem de 1 para 0) provoca a sinalização de um flag chamado COUNTFLAG e uma nova recarga é feita, iniciando novamente a contagem.**
- **O valor inicial de carga deverá estar entre 0x000000 a 0xFFFFFF;**
- **O contador decrescente é chamado de STCURRENT e é configurado através SysTick->Val no CMSIS ARM.**
- **O clock pode vir de System Clock ou através de um pino externo PIOSC; A escolha da fonte do clock é configurada através do registrador CLK_SRC (SysTick->CTRL).**
- **O registrador de carga é chamado de STRELOAD.**

- **System Timer (SysTick)**





Processadores Embarcados

- **System Timer (SysTick)**
- **Para contar 1000 clocks, o valor de carga em STRELOAD (SysTick Reload Value) deverá ser 999, pois o evento ocorre na passage de 1 para 0 (n-1).**
- **Inicialmente SysTick é configurado para operar em 41.94MHz.**
- **Para se calcular o tempo usa-se a seguinte formula:**

$$(N+1)\times(1/sysclk)=(N+1)/sysclk$$

Onde N representa o valor de carga inicial e sysclk é a frequencia do clock de entrada.

Exemplo: SysTick de 1ms

(N+1)=delay×sysclk = 0.001sec × 41.94MHz = 41.940

N=41.940–1=41939.

Processadores Embarcados

- **System Timer (SysTick)**

```
void SysTick_Init(void) {
   SysTick->CTRL = 0;
   SysTick->VAL = 41939;
   SysTick->CTRL = ( /*SysTick_CTRL_TICKINT_Msk   |*/  /* Enable SysTick exception */
                SysTick_CTRL_ENABLE_Msk) |    /* Enable SysTick system timer */
                SysTick_CTRL_CLKSOURCE_Msk;   /* Use processor clock source */
}
void delay1ms(void)
{
     while((SysTick->CTRL & 0x10000) ==0);//aguarda o flag sinalizar.
}
```

Processadores Embarcados

- **System Timer (SysTick)**

```
void SysTick_Init(void) {
   SysTick->CTRL = 0;
   SysTick->VAL = 41939;
   SysTick->CTRL = ( /*SysTick_CTRL_TICKINT_Msk   |*/  /* Enable SysTick exception */
                SysTick_CTRL_ENABLE_Msk) |    /* Enable SysTick system timer */
                SysTick_CTRL_CLKSOURCE_Msk;   /* Use processor clock source */
}
void delay1ms(void)
{
     while((SysTick->CTRL & 0x10000) ==0);//aguarda o flag sinalizar.
}
```

Processadores Embarcados

- System Timer (SysTick) (MKL25Z4.h)

- /** Interrupt Number Definitions */
- #define NUMBER_OF_INT_VECTORS 48 /**< Number of interrupts in the Vector table */

- typedef enum IRQn {
- /* Core interrupts */
- *NonMaskableInt_IRQn = -14, /**< Non Maskable Interrupt */*
- *HardFault_IRQn = -13, /**< Cortex-M0 SV Hard Fault Interrupt */*
- *SVCall_IRQn = -5, /**< Cortex-M0 SV Call Interrupt */*
- *PendSV_IRQn = -2, /**< Cortex-M0 Pend SV Interrupt */*
- *SysTick_IRQn = -1, /**< Cortex-M0 System Tick Interrupt */*

- /* Device specific interrupts */
- *DMA0_IRQn = 0, /**< DMA channel 0 transfer complete */*
- *DMA1_IRQn = 1, /**< DMA channel 1 transfer complete */*
- *DMA2_IRQn = 2, /**< DMA channel 2 transfer complete */*
- *DMA3_IRQn = 3, /**< DMA channel 3 transfer complete */*
- *Reserved20_IRQn = 4, /**< Reserved interrupt */*

Processadores Embarcados

```
FTFA_IRQn          = 5,    /**< Command complete and read collision */
LVD_LVW_IRQn       = 6,    /**< Low-voltage detect, low-voltage warning */
LLWU_IRQn          = 7,    /**< Low leakage wakeup Unit */
I2C0_IRQn          = 8,    /**< I2C0 interrupt */
I2C1_IRQn          = 9,    /**< I2C1 interrupt */
SPI0_IRQn          = 10,   /**< SPI0 single interrupt vector for all sources */
SPI1_IRQn          = 11,   /**< SPI1 single interrupt vector for all sources */
UART0_IRQn         = 12,   /**< UART0 status and error */
UART1_IRQn         = 13,   /**< UART1 status and error */
UART2_IRQn         = 14,   /**< UART2 status and error */
ADC0_IRQn          = 15,   /**< ADC0 interrupt */
CMP0_IRQn          = 16,   /**< CMP0 interrupt */
TPM0_IRQn          = 17,   /**< TPM0 single interrupt vector for all sources */
TPM1_IRQn          = 18,   /**< TPM1 single interrupt vector for all sources */
TPM2_IRQn          = 19,   /**< TPM2 single interrupt vector for all sources */
RTC_IRQn           = 20,   /**< RTC alarm */
RTC_Seconds_IRQn   = 21,   /**< RTC seconds */
PIT_IRQn           = 22,   /**< PIT interrupt */
Reserved39_IRQn    = 23,   /**< Reserved interrupt */
USB0_IRQn          = 24,   /**< USB0 interrupt */
DAC0_IRQn          = 25,   /**< DAC0 interrupt */
TSI0_IRQn          = 26,   /**< TSI0 interrupt */
MCG_IRQn           = 27,   /**< MCG interrupt */
LPTMR0_IRQn        = 28,   /**< LPTMR0 interrupt */
Reserved45_IRQn    = 29,   /**< Reserved interrupt */
PORTA_IRQn         = 30,   /**< PORTA Pin detect */
PORTD_IRQn         = 31    /**< PORTD Pin detect */
} IRQn_Type;
```

Processadores Embarcados

```
.syntax unified
   .arch armv6-m

   .section .isr_vector, "a"
   .align 2
   .globl __isr_vector
__isr_vector:
   .long   __StackTop                    /* Top of Stack */
   .long   Reset_Handler                  /* Reset Handler */
   .long   NMI_Handler                   /* NMI Handler*/
   .long   HardFault_Handler                 /* Hard Fault Handler*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   SVC_Handler                       /* SVCall Handler*/
   .long   0                          /* Reserved*/
   .long   0                          /* Reserved*/
   .long   PendSV_Handler                    /* PendSV Handler*/
   .long   SysTick_Handler                  /* SysTick Handler*/

                                 /* External Interrupts*/
   .long   DMA0_IRQHandler                   /* DMA channel 0 transfer complete*/
   .long   DMA1_IRQHandler                   /* DMA channel 1 transfer complete*/
   .long   DMA2_IRQHandler                   /* DMA channel 2 transfer complete*/
   .long   DMA3_IRQHandler                   /* DMA channel 3 transfer complete*/
   .long   Reserved20_IRQHandler                 /* Reserved interrupt*/
```

Processadores Embarcados

```
volatile uint32_t ticks;

void SysTick_Handler (void) {
  ticks++;
}


void _delay_ms(unsigned int TicksIn_mS)
{
      uint32_t DelayTimerTick = ticks;

      while((ticks - DelayTimerTick) < TicksIn_mS);
}

int main(void)
{
   SysTick_Config(SystemCoreClock/1000);   /* Generate interrupt each 1 ms  */
```

SENAI

Processadores Embarcados

Atividade em Dupla. Programar o SysTick

Prof° Fernando Simplicio

SENAI

Processadores Embarcados

Atividade em Dupla. Programar LCD usando GPIO e SysTick.

Prof° Fernando Simplicio