



Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica
e de Computação



EA871 - Laboratório de Programação Básica de Sistemas Digitais

Roteiro 9 - Relógio em Tempo Real

Autores: Fernando Teodoro de Cillo e Rafael Silva Cirino

RA: 197029 223730

Campinas
Junho de 2022

Introdução:

O intuito deste experimento é compreender a configuração e programação do MKL25Z128 para processamento de eventos temporais, a fim de demonstrar o princípio de funcionamento de temporizadores e uma aplicação em relógios digitais.

Experimento:

1

1.a

Nesta primeira questão temos que medir o tempo para cada relógio: Em systick temos o seguinte intervalo de contagem setado:

$$\frac{5242880}{20971520} = 0.25s$$

Assim foi obtido o seguinte tempo conforme a imagem abaixo:

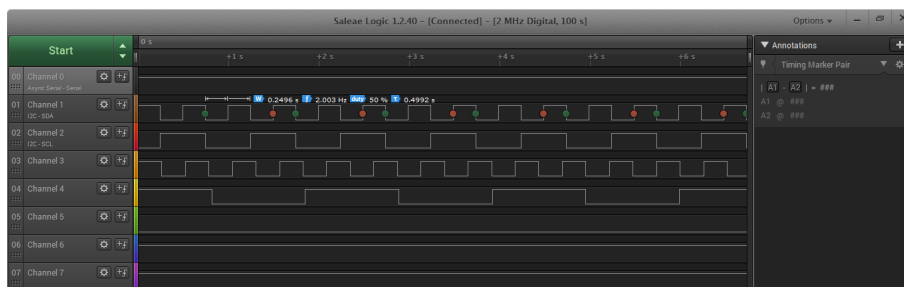


Figura 1: Caption

Em PIT temos o seguinte intervalo de contagem setado:

$$\frac{5242880}{20971520} = 0.25s$$

Entretanto temos que SIM_setOUTDIV4 está com o clock multiplicado por 2:

```
/*!  
 * Seta o divisor de frequencia para o sinal de barramento  
 */  
SIM_setOUTDIV4 (0b001);
```

Figura 2: Caption

Portanto o tempo será:

$$2 \cdot \frac{5242880}{20971520} = 0.5s$$

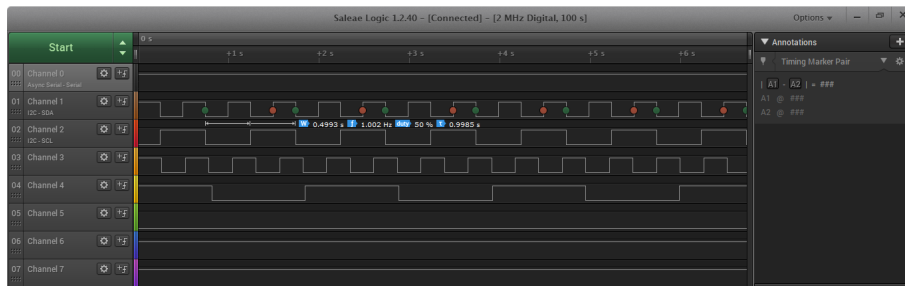


Figura 3: Caption

Para LPTMR0 com frequência de 1Hz, temos:

$$2 \cdot \frac{125}{1000} = 0.25s$$

Conforme esperado:

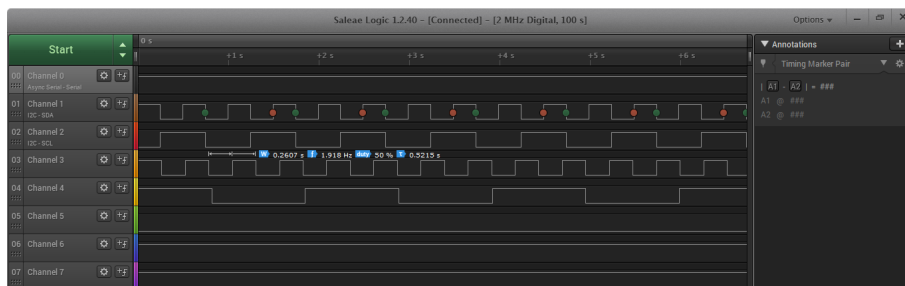


Figura 4: Caption

Para RTC está setado que será ativado a cada 1 segundo, conforme esperado temos:

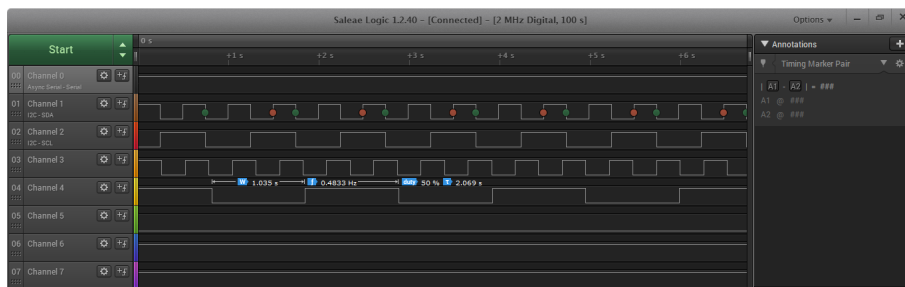


Figura 5: Caption

1.b

Para SysTick, PIT e LPTMR0 temos o mesmo intervalo de tempo obtido anteriormente:

Já para o RTC, neste ensaio ele está funcionando via interrupção, que acontece a cada aproximadamente 33 segundos.

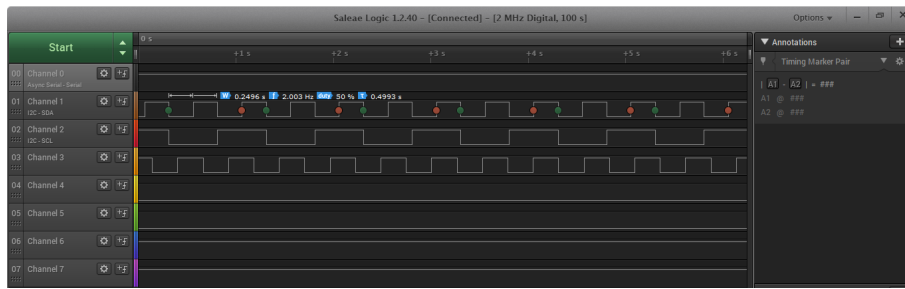


Figura 6: Caption

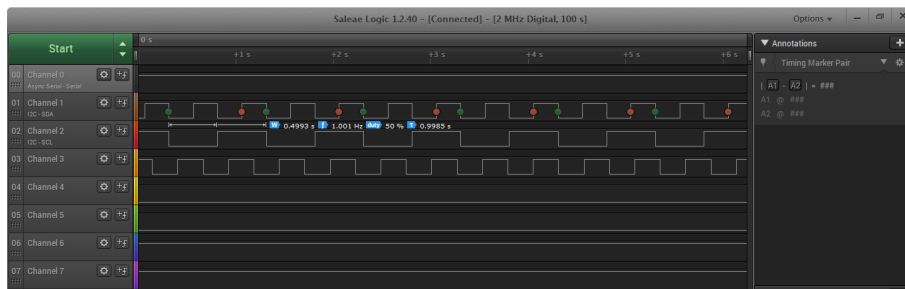


Figura 7: Caption

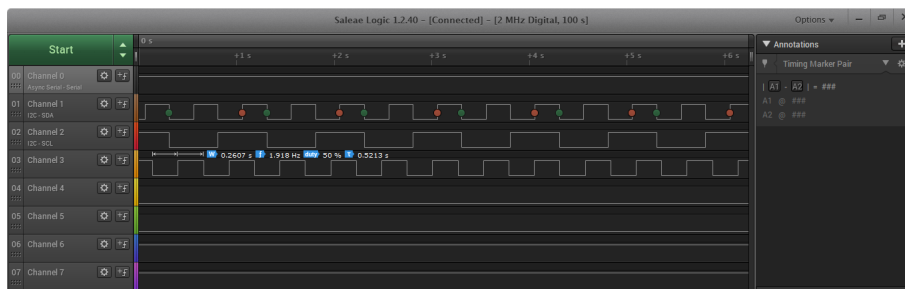


Figura 8: Caption

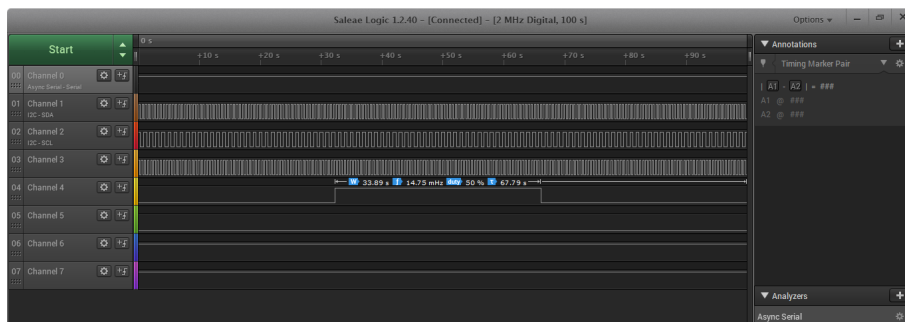


Figura 9: Caption

2

2.a

- Configurações para void GPIO_ativaConLCD(void):

– PORTC.PCR (0 ao 7) - Dados: 0x4004B000u = (0xFFFFF8FF) — 0x00000100
 // Zera os bits 10, 9 e 8 e depois habilita somente o bit 8

- PORTC_PCR (8 ao 10) - RS, E, LE: 0x4004B000u = (0xFFFFF8FF) — 0x00000100
// Zera os bits 10, 9 e 8 e depois habilita somente o bit 8
- GPIOC_PDDR: 0x400FF094u = 0x7FF // Ativar bits do 0 ao 10
- Configurações para void GPIO_initLCD (void):
 - GPIOC_PDOR: 400F_F080 = 0x38(Function Set) - delay 39us
 - GPIOC_PDOR: 400F_F080 = 0x0C(Display ON/OFF Control) - delay 39us
 - GPIOC_PDOR: 400F_F080 = 0x01(Display Clear) - 1530us
 - GPIOC_PDOR: 400F_F080 = 0x06(Entry mode set) - delay 39us

2.b

- Configurações para void GPIO_initSwitches (void):
 - NMI(PTA4)
 - * PORTA_PCR4: 0x40049010u = (&0x7) | 0x1 Limpa os bits, em seguida, aciona o primeiro
 - * GPIOA_PDDR: 0x400FF014u = (1<<4) Aciona o bit 4 da porta
 - * GPIOA_PDIR: 0x400FF010u = (1<<4) Caso no bit 4 esteja o valor 0, botão acionado, caso contrário não acionado
 - IRQA5(PTA5)
 - * PORTA_PCR5: 0x40049014u = (0x7) | 0x1 Limpa os bits, em seguida, aciona o primeiro
 - * GPIOA_PDDR: 0x400FF014u = (1<<5) Aciona o bit 5 da porta
 - * GPIOA_PDIR: 0x400FF010u = &(1<<5) Caso no bit 5 esteja o valor 0, botão acionado, caso contrário não acionado
 - IRQA12(PTA12)
 - * PORTA_PCR12: 0x4004902Cu = (0x7) | 0x1 Limpa os bits, em seguida, aciona o primeiro
 - * GPIOA_PDDR: 0x400FF014u = (1<<12) Aciona o bit 12 da porta
 - * GPIOA_PDIR: 0x400FF010u = &(1<<12) Caso no bit 12 esteja o valor 0, botão acionado, caso contrário não acionado

2.c

- Configurações para void RTC_init (void):
 - RTC_CR: 4003_D010 &= not(0x1) - Reseta SWR

- RTC_TCR: 4003_D00C = RTC_TCR_CIR(0x00) | RTC_TCR_TCR(0x00) - ajusta o erro de frequência em cada segundo
- RTC_TPR: 4003_D004 = 0 - Reseta prescaler
- RTC_TSR: 4003_D000 = 0 - Reseta o contador
- RTC_SR: 4003_D014 | = RTC_SR_TCE_MASK - Habilita o relógio

2.d

- Configurações para SysTick_init(unsigned int periodo):
 - SYST_RVR = SysTick_RVR_RELOAD(5242880) - Seta o período de 0,5s para ativar a interrupção
 - SYST_CVR = SysTick_CVR_CURRENT(0) - Reseta flag
 - SYST_CSR | = (SysTick_CSR_CLKSOURCE_MASK | SysTick_CSR_ENABLE_MASK) - Ativa o SysTick

2.e

- Configurações SIM para ativar o relógio:
 - SIM_SCGC6: 4004_803C |= SIM_SCGC6_RTC_MASK -
 - SIM_SOPT1: 4004_7000 | = SIM_SOPT1_OSC32KSEL(0b11)

2.f

- Para habilitar interrupção nas 3 botoeiras:
 - NVIC_IUSER | = GPIO_PIN(46-16)
 - NVIC_ICPR | = GPIO_PIN(46-16)
 - NVIC_IPR7 | = NVIC_IP_PRL30(priority << 6)

4 Quando o programa foi executado obteve-se a figura 10a e, 29 segundos depois, a figura 10b, que exemplifica o funcionamento correto do contador, pois houve uma variação em *hora[2]* de 45 para 14 (15 segundos para completar 60 e mais 14) e, como consequência, também houve um incremento em *hor[1]*.

5 Como pode-se observar nas figura 11a e 11b, os segundos são atualizados de um instante para o outro, comprovando que a função funciona e que na placa é atualizado o horário.

Name	Value	Location
segundos	12345	0x20002ff4
hor	0x20002fe4	0x20002fe4
[0]	3	0x20002fe4
[1]	25	0x20002fe8
[2]	45	0x20002fec
[3]	0	0x20002ff0

(a) Programa executado

Name	Value	Location
segundos	12374	0x20002ff4
hor	0x20002fe4	0x20002fe4
[0]	3	0x20002fe4
[1]	26	0x20002fe8
[2]	14	0x20002fec
[3]	0	0x20002ff0

(b) 29 segundos depois

Figura 10: Alteração no vetor `hor[]` que indica que a contagem está ocorrendo corretamente

Name	Value	Location
segundos	12379	0x20002ff4
end	0x20002fe4	0x20002fe4
[0]	0x00	0x20002fe4
[1]	0x07	0x20002fe5
[2]	0x0a	0x20002fe6
str_hhddss	0x20002fdc	0x20002fdc
[0]	'0'	0x20002fdc
[1]	'3'	0x20002fdd
[2]	':'	0x20002fde
[3]	'2'	0x20002fdf
[4]	'6'	0x20002fe0
[5]	':'	0x20002fe1
[6]	'1'	0x20002fe2
[7]	'9'	0x20002fe3
str	'0'	0x20002fe7

(a)

Name	Value	Location
segundos	12404	0x20002ff4
end	0x20002fe4	0x20002fe4
[0]	0x00	0x20002fe4
[1]	0x07	0x20002fe5
[2]	0x0a	0x20002fe6
str_hhddss	0x20002fdc	0x20002fdc
[0]	'0'	0x20002fdc
[1]	'3'	0x20002fdd
[2]	':'	0x20002fde
[3]	'2'	0x20002fdf
[4]	'6'	0x20002fe0
[5]	':'	0x20002fe1
[6]	'4'	0x20002fe2
[7]	'4'	0x20002fe3
str	'0'	0x20002fe7

(b)

Figura 11: Aba Variables com os valores sendo atualizados

6 Podemos ver na figura 12a que o valor 0x0F é setado no registrador PDOR para ativar o cursor piscante, e depois é registrado o valor 0x0C para desativá-lo, como mostra a figura 12b.



Figura 15: Display LCD após ajuste fino nos valores das variáveis

11

INÍCIO

main:

inicialização

loop:

atualiza o estado

atualiza o horário

Se o estado não for NORMAL:

cursor começa a piscar

para de atualizar o horário

Senão:

atualiza o horário

atualiza o horário exibido no display LCD

SysTick_Handler:

conta 3 segundos ($6 * 0.5s$)

Se o estado não for NORMAL:

desativa o cursor piscante

termina a interrupção

altera o estado para NORMAL

PORTA_IRQHandler

caso NORMAL

troca de estado para NORMAL_HORA ou mantém em NORMAL

caso SEGUNDO_HORA

caso NORMAL_HORA

caso HORA

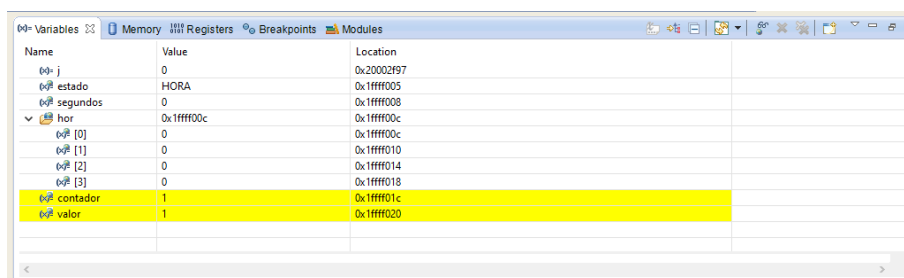
troca de estado para HORA_MINUTO, DECREMENTA_HORA ou INCREMENTA_HORA

```

caso HORA_MINUTO
caso MINUTO
    troca de estado para MINUTO_SEGUNDO, DECREMENTA_MINUTO ou INCRE-
    MENTA_MINUTO
caso MINUTO_SEGUNDO
caso SEGUNDO
    troca de estado para SEGUNDO_HORA, DECREMENTA_SEGUNDO ou INCREMEN-
    TA_SEGUNDO
caso DECREMENTA_HORA
caso INCREMENTA_HORA
    Se hor[0] = 23:
        hor[0] volta para 0
    Senão:
        incrementa hor[0]
caso DECREMENTA_MINUTO
caso INCREMENTA_MINUTO
    Se hor[1] = 0 e decrementa minuto:
        hor[1] vai para 59
    Se hor[1] = 59 e incrementa minuto:
        hor[1] volta para 0
caso DECREMENTA_SEGUNDO
caso INCREMENTA_SEGUNDO
    Se hor[2] = 0 e decrementa segundo:
        hor[2] vai para 59
    Se hor[2] = 59 e incrementa segundo:
        hor[2] volta para 0
fimescolha
FIM

```

13 Ao inicializar o programa e pressionar a botoeira NMI pela primeira vez, obtém-se *estado* = *normal* e o vetor *hor[]* inicializa zerado, como na figura 16.



Name	Value	Location
0x+ j	0	0x20002f97
0x@ estado	HORA	0x1ffff005
0x@ segundos	0	0x1ffff008
0x@ hor	0x1ffff00c	0x1ffff00c
0x@ [0]	0	0x1ffff00c
0x@ [1]	0	0x1ffff010
0x@ [2]	0	0x1ffff014
0x@ [3]	0	0x1ffff018
0x@ contador	1	0x1ffff01c
0x@ valor	1	0x1ffff020

Figura 16: Variáveis após o primeiro acionamento da botoeira NMI

Depois disso, ao acionar a botoeira IRQA5, o estado é alterado para *decrementa_hora* e é

subtraído o valor 1 da hora atual, como visto na figura 17, em que a hora (hor[0]) retrocede de 0 a 23.

Name	Value	Location
j	71	0x20002f5f
estado	DECREMENTA_HORA	0x1ffff005
segundos	82800	0x1ffff008
hor	0x1ffff00c	0x1ffff00c
hor[0]	23	0x1ffff010
hor[1]	0	0x1ffff014
hor[2]	0	0x1ffff018
hor[3]	0	0x1ffff01c
contador	3	0x1ffff020
valor	2	0x1ffff020

Figura 17: Variáveis após o acionamento da botoeira IRQA5

Acionando NMI pela segunda vez, alteramos a variável *estado* para *minuto*, que possibilita incrementar ou decrementar os minutos do nosso relógio, como visto na figura 18.

Name	Value	Location
j	71	0x20002f5f
estado	MINUTO	0x1ffff005
segundos	79200	0x1ffff008
hor	0x1ffff00c	0x1ffff00c
hor[0]	23	0x1ffff010
hor[1]	0	0x1ffff014
hor[2]	0	0x1ffff018
hor[3]	0	0x1ffff01c
contador	1	0x1ffff020
valor	1	0x1ffff020

Figura 18: Variáveis após o segundo acionamento da botoeira NMI

15 Duas sequências de teste do sistema podem ser vistas nas tabelas 1 e 2.

Tabela 1: Primeira sequência

Estado	Botão pressionado	Horário esperado
Normal	-	
Hora	NMI	
Incrementa hora	IRQA12	
Incrementa hora	IRQA12	
Minuto	NMI	
Incrementa minuto	IRQA12	
Incrementa minuto	IRQA12	
Segundo	NMI	
Normal	Espera 3s	02:02:00

Tabela 2: Segunda sequência

Estado	Botão pressionado	Horário esperado
Normal	-	
Hora	NMI	
Minuto	NMI	
Decrementa minuto	IRQA5	
Decrementa minuto	IRQA5	
Segundo	NMI	
Incrementa segundo	IRQA12	
Incrementa segundo	IRQA12	
Incrementa segundo	IRQA12	
Normal	Espera 3s	00:58:03

16 Na figura 19 temos o print size obtido no experimento 9, comparando com a figura 20, referente ao experimento 8, podemos perceber que há um aumento somente no arquivo `.txt`, que corresponde ao tamanho das instruções e dados constantes armazenados na memória flash, o que é um aumento esperado, tendo em vista que foram adicionadas novas funções. Nas variáveis sem inicialização armazenadas na memória RAM, representado por `.bss`, também ocorreu um aumento. Como `.data` se refere a valores com inicialização armazenada na memória RAM, comparando com o último experimento é esperado que não haja aumento.

```

CDT Build Console [religio_digital]
'Executing target #13 religio_digital.siz'
'Invoking: ARM Ltd Windows GNU Print Size'
"C:/Freescale/CW MCU v10.6/Cross_Tools/arm-none-eabi-gcc-4_7_3/bin/arm-none-eabi-size" --format=berkeley religio_digital.elf
text  data  bss   dec   hex filename
4136   24   2108  6268  187c religio_digital.elf
'Finished building: religio_digital.siz'

```

Figura 19: *Print size* do projeto `religio_digital`

```

CDT Build Console [escolha_cor_bitmap]
"C:/Freescale/CW MCU v10.6/gnu\\bin\\mingw32-make" -j8 all
'Executing target #12 escolha_cor_bitmap.siz'
'Invoking: ARM Ltd Windows GNU Print Size'
"C:/Freescale/CW MCU v10.6/Cross_Tools/arm-none-eabi-gcc-4_7_3/bin/arm-none-eabi-size" --format=berkeley escolha_cor_bitmap.elf
text  data  bss   dec   hex filename
3628   24   2080  5732  1664 escolha_cor_bitmap.elf
'Finished building: escolha_cor_bitmap.siz'

```

Figura 20: *Print size* do projeto `escolha_cor_bitmap`