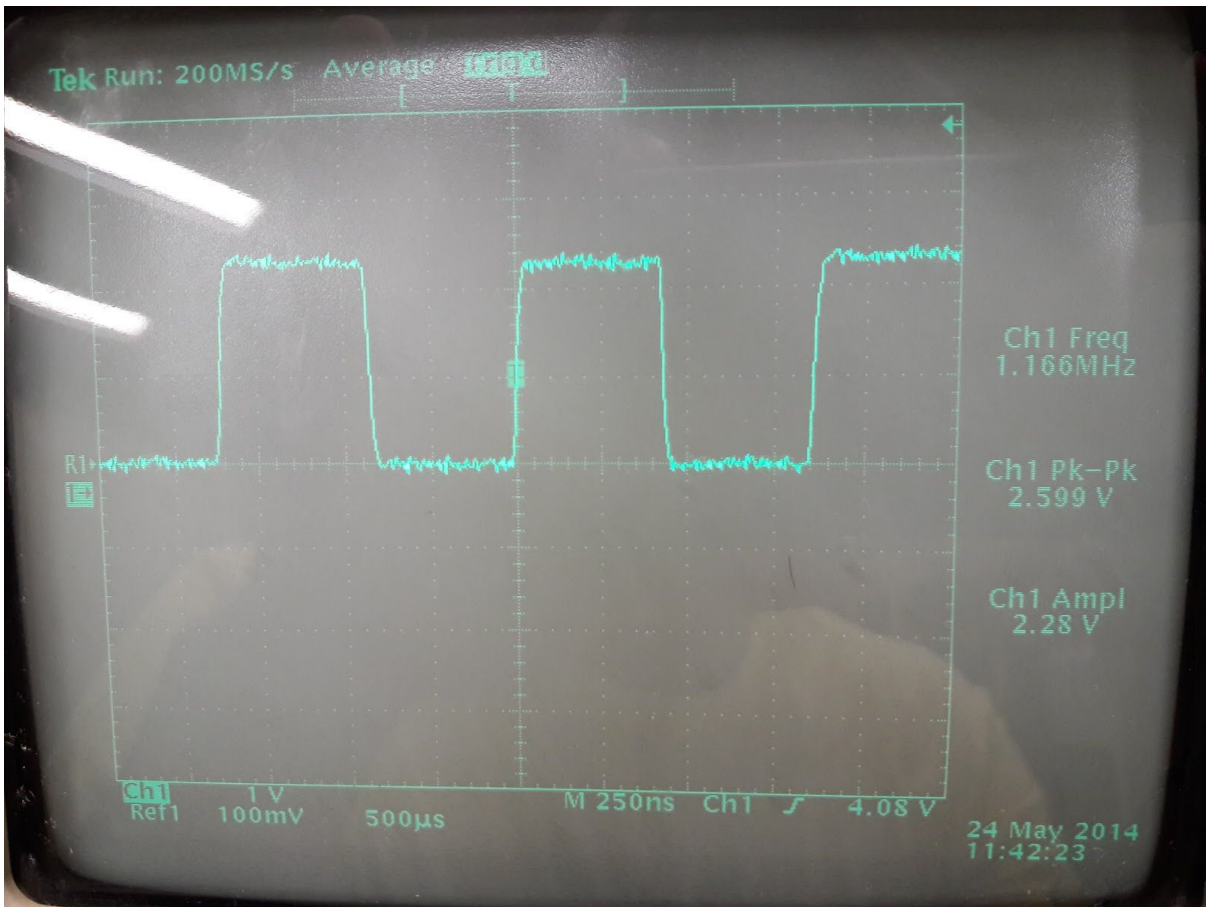
175480_05

EA871

Patrick de Carvalho Tavares Rezende Ferreira

Atividade 1

1.



Frequência: 1,17MHz

2.

ldr r3,[pc,#8] @ Salva o valor de PC + 8 bits em r3.

movs r2,#128 @ Move o valor 0x80 para r2.

lsls r2,r2,#16 @ Transforma o valor de r2 em 0x800000, que equivale a setar apenas o bit 23 desta palavra, para realizar somente o toggle do pino 23.

str r2,[r3,#12] @ O conteúdo de r3 + 12 bits equivale à posição da memória que guarda o endereço do GPIOE_PTOR. Assim, passando a palavra salva em r2, realizamos o toggle do terminal 23.

b main+0x8 @ A posição relativa main + 0x8 equivale à instrução **ldr r3,[pc,#8]**, ou seja, aqui recomeçamos o loop.

3.

Para realizar este cálculo, utilizou-se as seguintes instruções dentro da subrotina for:

```
for:
 mov r4, r5
 mov r4, r5
 str r2,[r3,#0] @ Faz o toggle no hardware.

 b for
```

Após isso, deixou-se ambas as instruções mov (instruções de duração de apenas um ciclo de relógio) comentadas, ou seja, não gerando códigos para a execução e mediu-se novamente o período da onda.

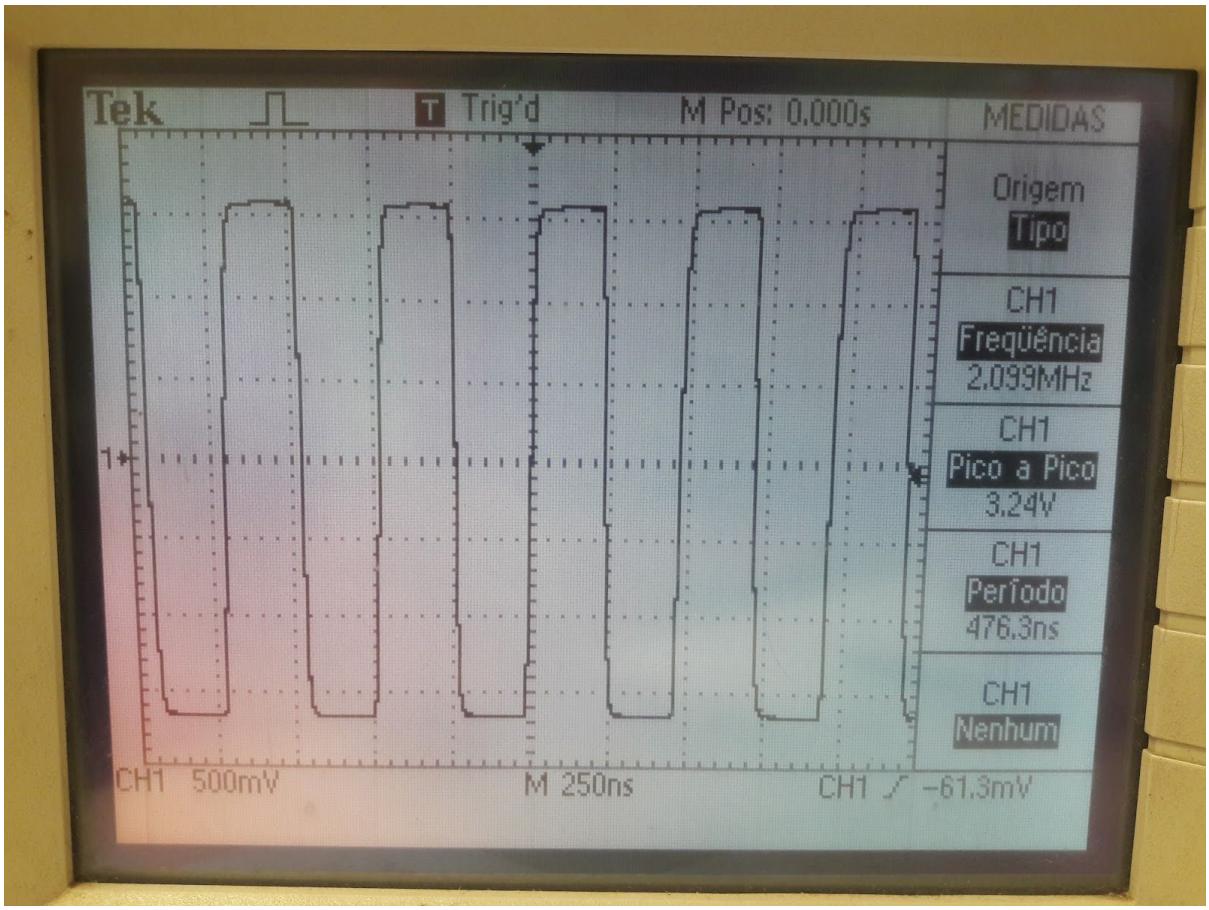
```
for:
 @mov r4, r5
 @mov r4, r5
 str r2,[r3,#0] @ Faz o toggle no hardware.

 b for
```

Período com os MOV: 668 ns.
Período sem os MOV: 476 ns.

Logo, o tempo gasto com cada MOV foi de (668ns - 476ns)/2 = 192ns/2 = 96ns.
Assim, cada ciclo de execução é estimado em **96ns**.

Atividade 2



Maior frequência obtida: 2,1MHz.

Código utilizado:

```
/* This assembly file uses GNU syntax */

 .text

 .global main

@b main @ Inicia o programa em main.

inicGPIO:

  @SIM_SCGC5 |= (1<<13);
  ldr r2, SIM_SCGC5 @ o sinal "=" indica que estamos obtendo o endereço da variavel
SIM_SCGC5, mas queremos seu conteúdo. entao nao o usaremos.
  ldr r3, [r2,#0]  @ Guarda em r3 o conteúdo do conteúdo de r2.
  ldr r1, =0x2000 @ Guarda o valor a se passar.
```

```
  orr r3, r3, r1 @ Faz o OU bit a bit de SIM_SCGC5 com 001000000000.
  str r3, [r2] @ Atualiza o valor do hardware de SIM_SCGC5.

  @PORTE_PCR23 = 0x00000100;
  ldr r1, =0x00000100 @ Guarda o valor a que desejamos inserir.
  ldr r2, PORTE_PCR23 @ Obtém o endereço do hardware de PORTE_PCR23.
  str r1, [r2] @ Guarda o novo valor no hardware de PORTE_PCR23.

  @GPIOE_PDDR |= (1<<23);
  ldr r2, GPIOE_PDDR @ O sinal "=" indica que estamos obtendo o endereço da variavel
SIM_SCGC5, mas queremos seu conteúdo. entao nao o usaremos.
  ldr r3, [r2,#0]  @ Guarda em r3 o conteúdo do conteúdo de r2.
  ldr r1, =0x800000 @ Guarda o valor a se passar.
  orr r3, r3, r1 @ Faz o OU bit a bit de SIM_SCGC5 com 001000000000.
  str r3, [r2] @ Atualiza o valor do hardware de SIM_SCGC5.

  b main_continua @ Continua a execução de main.

main:

 @ inicGPIO();
 b inicGPIO


main_continua:

 @ GPIOE_PTOR = 0x00800000;
 ldr r3,GPIOE_PTOR @ Salva em r3 o endereço físico de GPIOE_PTOR.
 ldr r2, =0x800000 @ Guarda o valor que estaremos constantemente utilizando para fazer o
toggle.

for:

 str r2,[r3,#0] @ Faz o toggle no hardware.

 b for


 .align 2

SIM_SCGC5: .word 0x40048038
GPIOE_PDDR: .word 0x400FF114
GPIOE_PTOR: .word 0x400FF10C
PORTE_PCR23: .word 0x4004D05C
```