

# PSI3441 - Arquitetura de Sistemas Embarcados

Relatório Exercício Prático 01

Nome: Kevin Kirsten Lucas nºUSP: 10853306

Utilizei algumas funções auxiliares para melhorar a leitura da saída serial no aplicativo de monitoramento Tera Term, como seguem abaixo:

```
float calculate_timer_diff(float timer_start, float timer_end) {  
    float diff = timer_end - timer_start;  
    return diff;  
}  
  
void printf_line(int lines_amount) {  
    for (int i = 0; i < lines_amount; i++) {  
        printf("\n ");  
    }  
}
```

Ao final, fica o link do código completo.

1) Meça o tempo para realizar uma aquisição analógica. Qual é a maior taxa de aquisição (Amostras por segundo) que se pode conseguir com as configurações padrão do microcontrolador?

Com o código abaixo, obtive 47ms de tempo de leitura para uma aquisição analógica.

```
// Tarefa 1 -----  
timer_start = t.read_us();  
float temp = ain;  
timer_end = t.read_us();  
timer_diff = calculate_timer_diff(timer_start, timer_end);  
printf("\n1) Tempo para realizar uma aquisicao analogica:\n");  
printf("        > %f ms\n", timer_diff);
```

*Código da Tarefa 01*

```
1) Tempo para realizar uma aquisicao analogica:  
  > 47.000000 ms
```

*Print do Tera Term*

Segundo o manual da KL25Z, utilizando-se algum dos modos  $\leq$  que 13bits, é possível obter uma taxa de 818.330 Ksps (Kilosamples por segundo), no modo de 16bits, a taxa é um pouco menor, sendo de 461.467 Ksps.

**Table 25. 16-bit ADC operating conditions (continued)**

Symbol	Description	Conditions	Min.	Typ. <sup>1</sup>	Max.	Unit	Notes
V <sub>ADIN</sub>	Input voltage	<ul style="list-style-type: none"> <li>16-bit differential mode</li> <li>All other modes</li> </ul>	VREFL	—	31/32 * VREFH	V	—
C <sub>ADIN</sub>	Input capacitance	<ul style="list-style-type: none"> <li>16-bit mode</li> <li>8-bit / 10-bit / 12-bit modes</li> </ul>	—	8	10	pF	—
R <sub>ADIN</sub>	Input series resistance		—	2	5	kΩ	—
R <sub>AS</sub>	Analog source resistance (external)	13-bit / 12-bit modes f <sub>ADCK</sub> < 4 MHz	—	—	5	kΩ	4
f <sub>ADCK</sub>	ADC conversion clock frequency	≤ 13-bit mode	1.0	—	18.0	MHz	5
f <sub>ADCK</sub>	ADC conversion clock frequency	16-bit mode	2.0	—	12.0	MHz	5
C <sub>rate</sub>	ADC conversion rate	≤ 13-bit modes No ADC hardware averaging Continuous conversions enabled, subsequent conversion time	20.000	—	818.330	Ksps	6
C <sub>rate</sub>	ADC conversion rate	16-bit mode No ADC hardware averaging Continuous conversions enabled, subsequent conversion time	37.037	—	461.467	Ksps	6

Manual: <https://www.nxp.com/docs/en/data-sheet/KL25P80M48SF0.pdf>

2) Determine o tempo para realizar uma vez o cálculo matemático abaixo:  $x=x*3$ ;  $x=x*4$ ;  $y=x+max$ ;  $x=y-i$ ;  $x=x<<4$ ; Como se pode melhorar a precisão da medida? Qual é a resolução e a acurácia da medida?

Obtive os tempos abaixo para os cálculos pedidos. Para melhorar a precisão da medida, utilizei um loop com 10.000 iterações e realizei a média no final.

```

2) Tempo para realizar o calculo matematico <sem loop>:
  > 11.000000 ms

Tempo para realizar o calculo matematico <media com loop 10000x>:
  > 10.960700 ms
  
```

Print do Tera Term

Foi utilizado o código abaixo para realizar as operações:

```
// Tarefa 2 -----
int x = 1;
int y;
int max = 0;
int i = 0;

// sem loop
timer_start = t.read_us();
x = x * 3;
x = x * 4;
y = x + max;
x = y - i;
x = x << 4;
timer_end = t.read_us();
timer_diff = calculate_timer_diff(timer_start, timer_end);
printf("\n2) Tempo para realizar o calculo matematico (sem loop):\n");
printf("      > %f ms\n", timer_diff);

// com loop
x = 1;
max = 0;
float diff_sum = 0;
int threshold = 10000;
for(int i = 0; i < threshold; i++)
{
    timer_start = t.read_us();
    x = x * 3;
    x = x * 4;
    y = x + max;
    x = y - i;
    x = x << 4;
    timer_end = t.read_us();
    timer_diff = calculate_timer_diff(timer_start, timer_end);
    diff_sum += timer_diff;
}
timer_diff = diff_sum/threshold;
printf("\n   Tempo para realizar o calculo matematico (media com loop %dx):\n", threshold);
printf("      > %f ms\n", timer_diff);
```

*Código da Tarefa 02*

**3) Quanto tempo leva para fazer uma: a. Adição; b. Subtração; c. Multiplicação; d. Divisão; utilizando números inteiros e com ponto flutuante?**

Utilizei o método de calcular as médias para estas operações e obtive os resultados abaixo:

```
3) Tempo para realizar operacoes matematicas:
a) Adicao de int:
  > 10.849600 ms
  Adicao de float:
  > 10.958000 ms

b) Subtracao de int:
  > 10.849600 ms
  Subtracao de float:
  > 10.958200 ms

c) Multiplicao de int:
  > 10.811800 ms

  Multiplicao de float:
  > 10.959400 ms

d) Divisao de int:
  > 10.843000 ms
  Divisao de float:
  > 10.956400 ms
```

*Print do Tera Term*

Abaixo segue o código utilizado para a adição com inteiros e float. De maneira similar procedi com as outras operações.

```
// Tarefa 3 -----  
printf("\n3) Tempo para realizar operacoes matematicas:\n");  
  
// a. Adição Int  
int a_int = 1;  
int b_int = 2;  
int r_int;  
diff_sum = 0.0;  
for(int i = 0; i < threshold; i++)  
{  
    timer_start = t.read_us();  
    r_int = a_int * b_int;  
    timer_end = t.read_us();  
    timer_diff = calculate_timer_diff(timer_start, timer_end);  
    diff_sum += timer_diff;  
}  
timer_diff = diff_sum/threshold;  
printf("    a) Adicao de int:\n");  
printf("        > %f ms\n", timer_diff);  
  
// a. Adição Float  
float a_float = 1.0;  
float b_float = 2.0;  
float r_float;  
diff_sum = 0.0;  
for(int i = 0; i < threshold; i++)  
{  
    timer_start = t.read_us();  
    r_float = a_float * b_float;  
    timer_end = t.read_us();  
    timer_diff = calculate_timer_diff(timer_start, timer_end);  
    diff_sum += timer_diff;  
}  
timer_diff = diff_sum/threshold;  
printf("        Adicao de float:\n");  
printf("        > %f ms\n", timer_diff);
```

*Código da Tarefa 03*

#### 4) Qual o valor da soma de $2,147,483,647 + 1$ ?

O resultado está mostrado abaixo. Como podemos ver, houve um overflow nesta operação, uma vez que o número 2.147.483.647 é o maior inteiro que pode ser representado com 32 bits.

2.147.483.647	
HEX	7FFF FFFF
DEC	2.147.483.647
OCT	17 777 777 777
BIN	0111 1111 1111 1111 1111 1111 1111 1111

*Representação binária*

```
4> Qual o valor da soma de 2.147.483.647 + 1?  
> Soma: -2147483648
```

*Print do Tera Term*

```
// Tarefa 4 -----  
printf("\n4) Qual o valor da soma de 2.147.483.647 + 1?\n");  
a_int = 2147483647;  
b_int = 1;  
r_int = a_int + b_int;  
printf("      > Soma: %d\n", r_int);  
printf_line(10);
```

*Código da Tarefa 04*

#### 5) Como poderia ser medido o tempo de uma determinada operação sem utilizar timers?

Uma alternativa, seria ter o auxílio de um osciloscópio. Desta forma, poderia ser realizada a medição de dois pinos GPIO, um seria ativado antes do início de uma operação e o outro seria ativado logo após o fim da operação. Em seguida, basta utilizar os cursores do osciloscópio para medir a janela de tempo entre as medidas.

Segue o link com o código completo:

[https://github.com/kevinkirsten/psi3441-arquitetura-de-sistemas-embarcados/blob/main/exercicio\\_pratico\\_01/main.cpp](https://github.com/kevinkirsten/psi3441-arquitetura-de-sistemas-embarcados/blob/main/exercicio_pratico_01/main.cpp)