PSI2662 – Projeto em Sistemas Eletrônicos Embarcados: Sensores e Atuadores

LED RGB, PWM, Comunicação Serial, Acelerômetro

Escola Politécnica da Universidade de São Paulo

Prof. Gustavo Rehder – grehder@lme.usp.br

Segundo Semestre de 2015

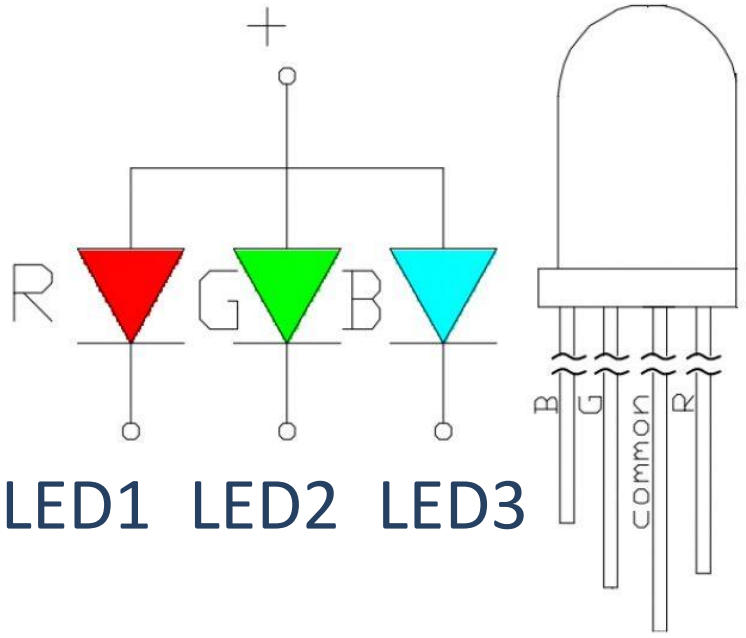
Sumário

1. **LED RGB.**
2. Módulo PWM.
3. Comunicação serial.
4. Acelerômetro.

LED RGB

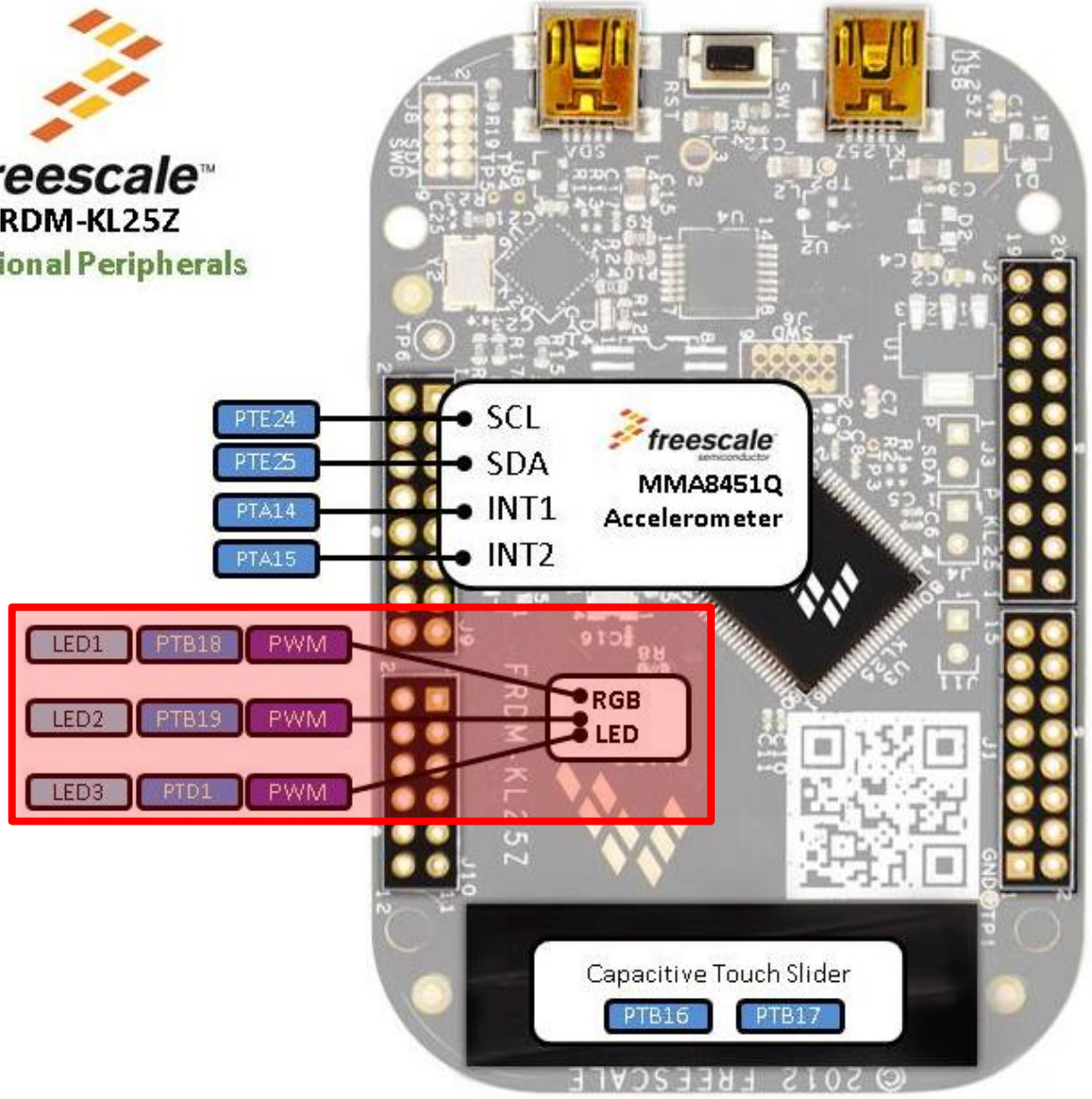
- O LED RGB na placa (componente D3), é constituído internamente pela ligação de 3 LEDs, um vermelho, um verde e um azul.

  - Ligação ANODO COMUM com os pinos do µcontrolador.
  - Ligados a entradas digitais com suporte a sinal PWM (para regular brilho).
  - Podem ser ligados em conjunto para formar outras cores.



LED RGB

freescale™
FRDM-KL25Z
Additional Peripherals



mbed Enabled

LED RGB

- “Hello World!”. Simples programa para a plataforma FRDM KL25Z – Piscar LED:

```
#include "mbed.h"
DigitalOut myled(LED1);
int main() {
    while(1) {
        myled = 1;
        wait(0.2);
        myled = 0;
        wait(0.2);
    }
}
```

Sumário

1. LED RGB.
2. **Módulo PWM.**
3. Comunicação serial.
4. Acelerômetro.

Módulo PWM

- PWM significa Modulação por Largura de Pulso (Pulse Width Modulation);
- Sinal frequência constante e largura de pulso (ciclo ativo ou *duty cycle*) variável.
- Tensão Média:

$$V_M = \frac{1}{T}\int_0^T V(t)dt$$ ➔ onde *T* é o período do sinal.

- Sinal PWM:

$$V(t) = \begin{cases} V_{pulso}, & 0 \le t \le t_p \\ 0, & t_p \le t \le T \end{cases}$$ ➔ onde tp e Vp são a duração e a tensão do pulso em nível alto.

Módulo PWM

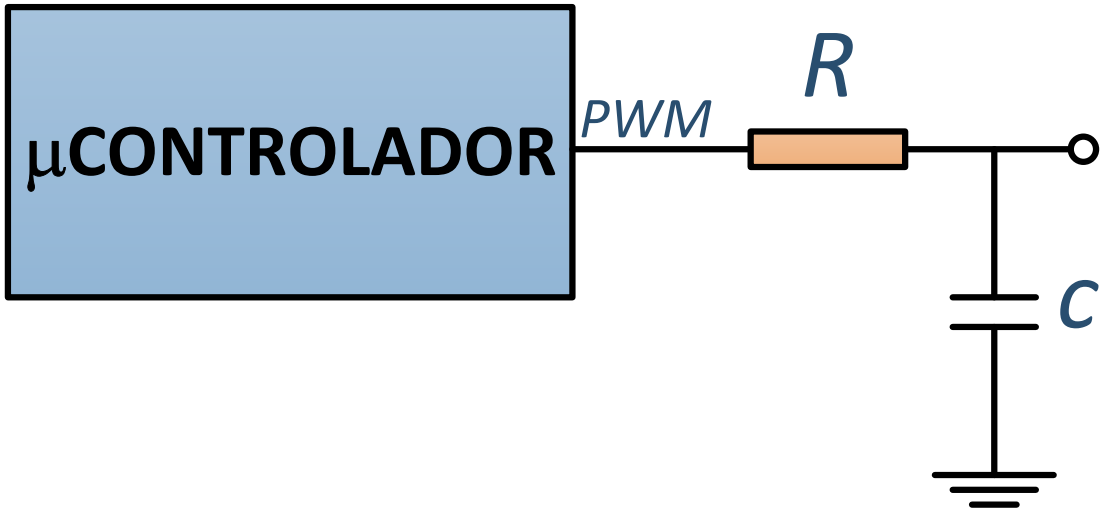
- Assim:

$$V_M = \frac{1}{T}\left[\int_0^{t_p} V(t)dt + \int_{t_p}^{T} 0dt\right] = \frac{t_p}{T}V_{pulso}$$ ➔ t_p/T é o *duty cycle*

- O pulso da onda PWM apresenta tensão fixa, porém o valor médio da tensão varia em função do *duty cycle*.
- A tensão média (V_M) é diretamente proporcional ao ciclo ativo e, como este varia entre 0 e 1, a tensão média pode variar entre 0 e V_{pulso}.

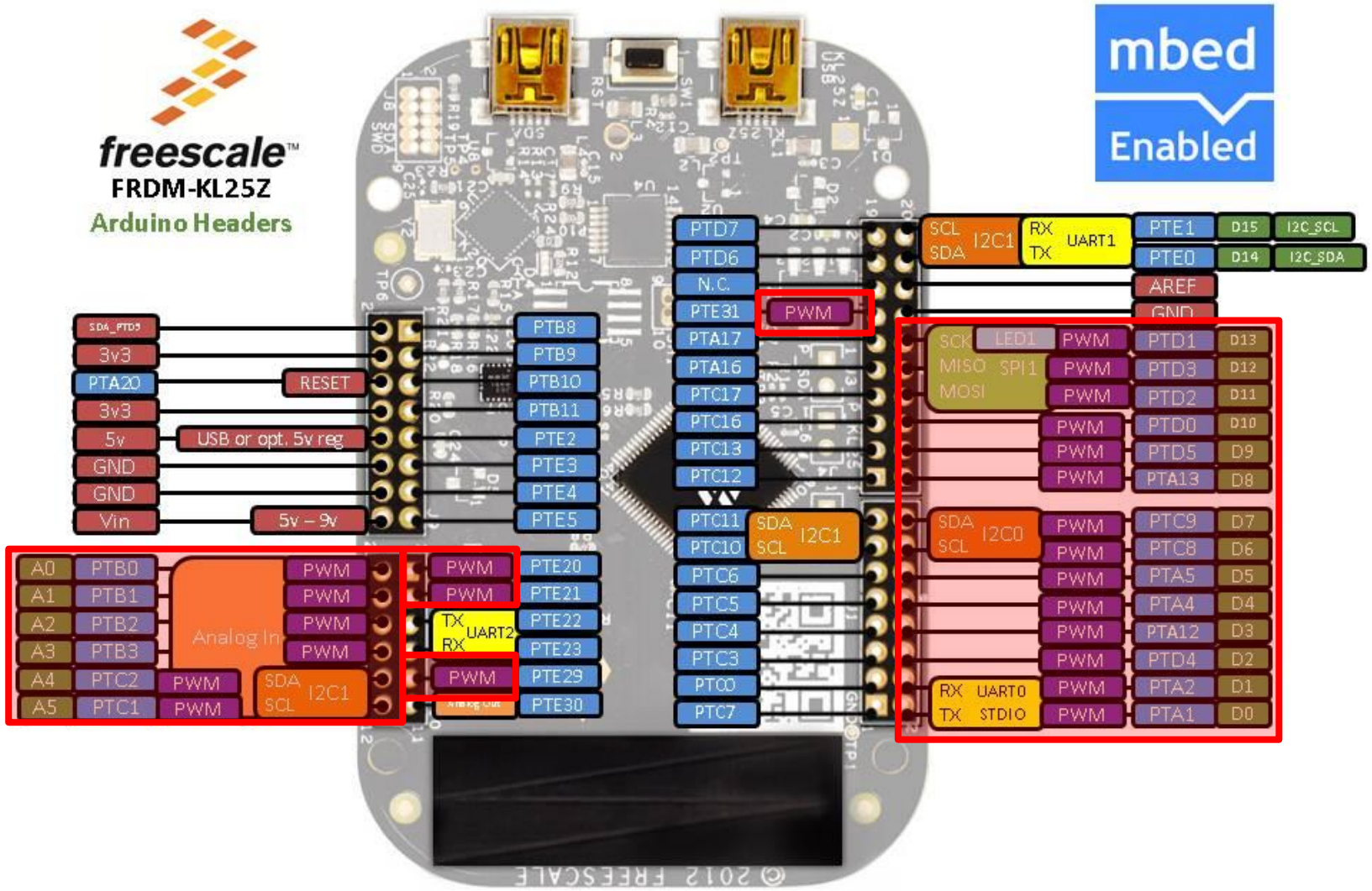
Módulo PWM

- Para obter um conversor D/A a partir do de uma saída PWM, basta adicionar à saída um filtro passa-baixas com frequência de corte menor que a frequência do PWM.



$$f_c = \frac{1}{2\pi RC} \leq \frac{f_{PWM}}{10}$$

Módulo PWM


freescale™
FRDM-KL25Z
Arduino Headers
mbed
Enabled
SDA_PTD5
3v3
PTA20
RESET
3v3
5v
USB or opt. 5v reg
GND
GND
Vin
5v – 9v
A0 PTB0
A1 PTB1
A2 PTB2
A3 PTB3
A4 PTC2
A5 PTC1
Analog In
PWM
SDA
SCL
I2C1
TX
RX
UART2
PTB8
PTB9
PTB10
PTB11
PTE2
PTE3
PTE4
PTE5
PTE20
PTE21
PTE22
PTE23
PTE29
PTE30
PTD7
PTD6
N.C.
PTE31
PTA17
PTA16
PTC17
PTC16
PTC13
PTC12
PTC11
PTC10
PTC6
PTC5
PTC4
PTC3
PTC0
PTC7
SCL
SDA
I2C1
RX
TX
UART1
PTE1 D15 I2C_SCL
PTE0 D14 I2C_SDA
AREF
GND
SCL
MISO
MOSI
SPI1
LED1
PTD1 D13
PTD3 D12
PTD2 D11
PTD0 D10
PTD5 D9
PTA13 D8
SDA
SCL
I2C0
PTC9 D7
PTC8 D6
PTA5 D5
PTA4 D4
PTA12 D3
PTD4 D2
PTA2 D1
PTA1 D0
RX
TX
UART0
STDIO
© 2012 FREESCALE

Módulo PWM

```
#include "mbed.h"
PwmOut r(LED_RED); //Pino PT18 (LED RED) como saída PWM
PwmOut g(LED_GREEN); //Pino PT18 (LED RED) como saída PWM
int main()
{   int aux = 0;
    r.period(0.001); //Define o periodo do PWM da saída ligada ao LED RED
    g.period(0.001); //Define o periodo do PWM da saída ligada ao LED GREEN

    while (true) {
        for (float i = 0; i < 1.0 ; i+= 0.1) {
            if (aux ==0){//muda intensidade do LED RED
                g = 1;
                r = i;
            }
            else{ //muda intensidade do LED GREEN
                r = 1;
                g = i;
            }
            wait (0.1);
        }
        aux = !aux;
    }
}
```

Sumário

1. LED RGB.
2. Módulo PWM.
3. **Comunicação serial.**
4. Acelerômetro.

Comunicação Serial

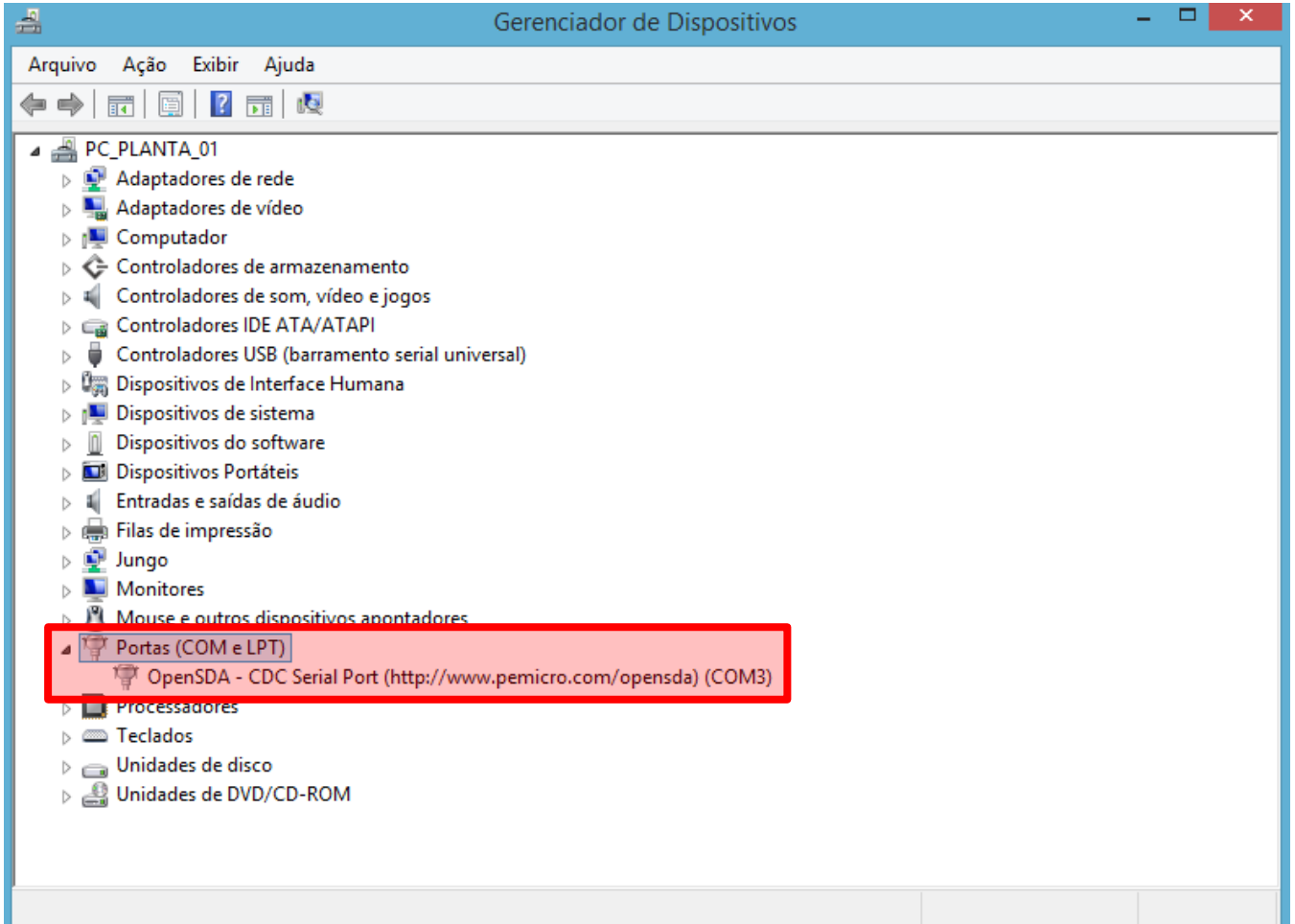
- Comunicação com um PC host através de um "USB Virtual Serial Port" através do mesmo cabo USB que é usado para a programação. Isso permite que você:
  - Imprima mensagens para um terminal host PC (útil para depuração!)
  - Leia a entrada do teclado do PC host
  - Comunique-se com aplicativos e linguagens de programação em execução no PC host que podem se comunicar com uma porta serial, por exemplo, Perl, Python, Java, Matlab, etc.
  - Pode auxiliar no *debug* de variáveis do programa.

Comunicação Serial

- No mbed:
  - **`Serial name(USBTX, USBRX)`** : define a comunicação serial com TX e RX via USB SDA. Define o label name.
  - **`name.baud(9600)`** :define a taxa (baud rate) em 9600. Aceita outros valores (19200, 38400, etc).
  - **`name.printf`**: imprime *string* KL25Z → PC*;*
  - **`name.putc`**: envia um caractere KL25Z → PC*;*
  - **`name.getc`**: recebe um caractere PC → KL25Z.

Comunicação Serial

- Ao conectar o cabo USB, aparece enumerada no computador uma porta OpenSDA – CDC.

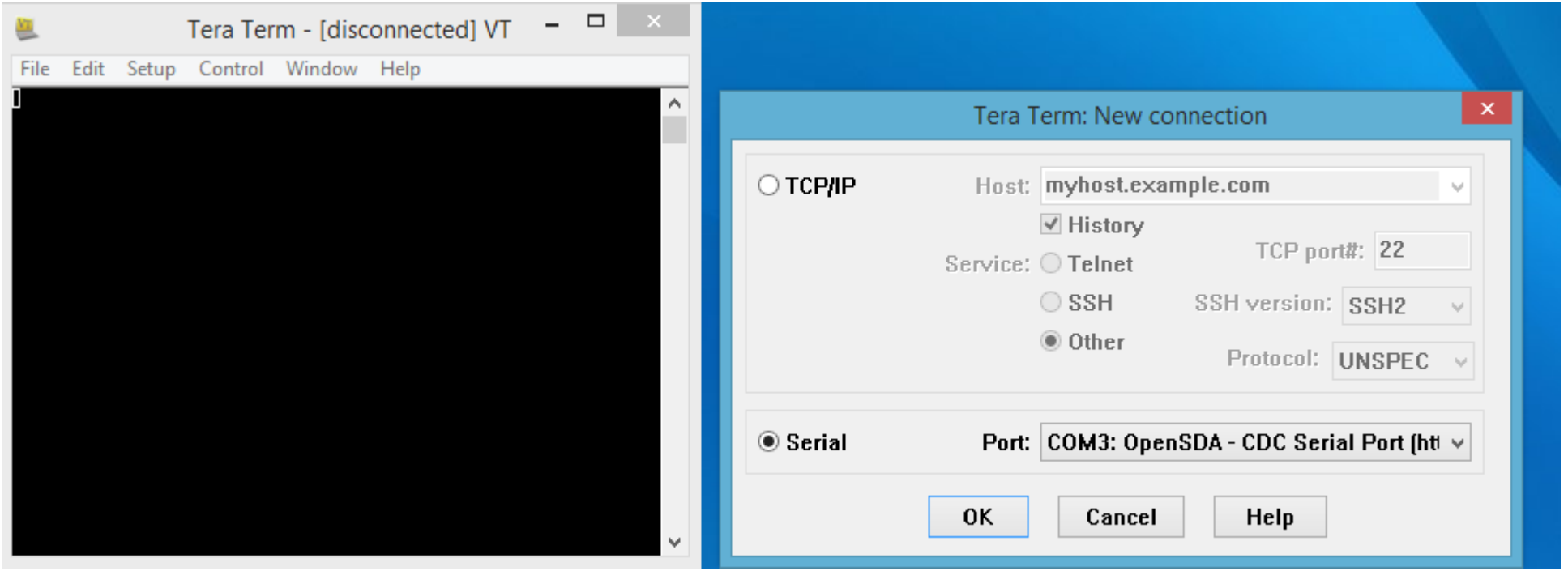


Comunicação Serial

- Caso não apareça no sistema windos, é necessário instalar o driver. Busque por **Download the mbed Windows serial port driver** no *Dashboard* do mbed.
- Para o curso, vamos utilizar um aplicativo terminal denominado Tera Term (http://ttssh2.sourceforge.jp/index.html.en).
  - OBS: tais arquivos de instalação estão postados no Moodle do minicurso.

Comunicação Serial

- Escolher a opção serial e a porta enumerada.



Comunicação Serial

- **Hello World!**

```
#include "mbed.h"
Serial pc(USBTX, USBRX); // tx, rx
int main() {
pc.printf("Hello World!\n");
}
```

- ***Echo back characters you type***

```
#include "mbed.h"
Serial pc(USBTX, USBRX);
int main() {

pc.printf("Echoes back to the screen anything you type\n");
   while(1) {
        pc.putc(pc.getc());
   }
}
```

Comunicação Serial

- Controla brilho do LED (‘u’ p/ UP e ‘d’ p/ DOWN).

```
#include "mbed.h"
Serial pc(USBTX, USBRX); // tx, rx
PwmOut led(LED1);
float brightness = 1.0;
int main() {
    pc.printf("Press 'u' para aumentar brilho e 'd' para diminuir \n\r");
    while(1) {
        char c = pc.getc();
        if((c == 'u') && (brightness > 0.0)) {
            brightness -= 0.1;
            led = brightness;
        }
        if((c == 'd') && (brightness < 1.0)) {
            brightness += 0.1;
            led = brightness;’
        }
        pc.printf("Brilho  = %2.0f %% \r", (1.0-led)*100);
     }
}
```

Comunicação Serial

- Muda cor do led RGB (‘r’ p/ RED, ‘g’ p/ GREEN, ‘b’ p/ BLUE. Qualquer outra tecla apaga tudo).

```
#include "mbed.h"
Serial pc(USBTX, USBRX); // tx, rx
PwmOut r(LED1); PwmOut g(LED2); PwmOut b(LED3);
int main() {
    r=1; g=1; b=1;
    pc.printf("Pressione a inicial da cor desejada (r ou g ou b)");
    while(1) {
        char c = pc.getc();
        switch(c){
        case 'r':
            r=0; g=1; b=1;
            break;
        case 'g':
            r=1; g=0; b=1;
            break;
        case 'b':
            r=1; g=1; b=0;
            break;
        default:
            r=1; g=1; b=1;
            break;
        }   }  }
```

Sumário

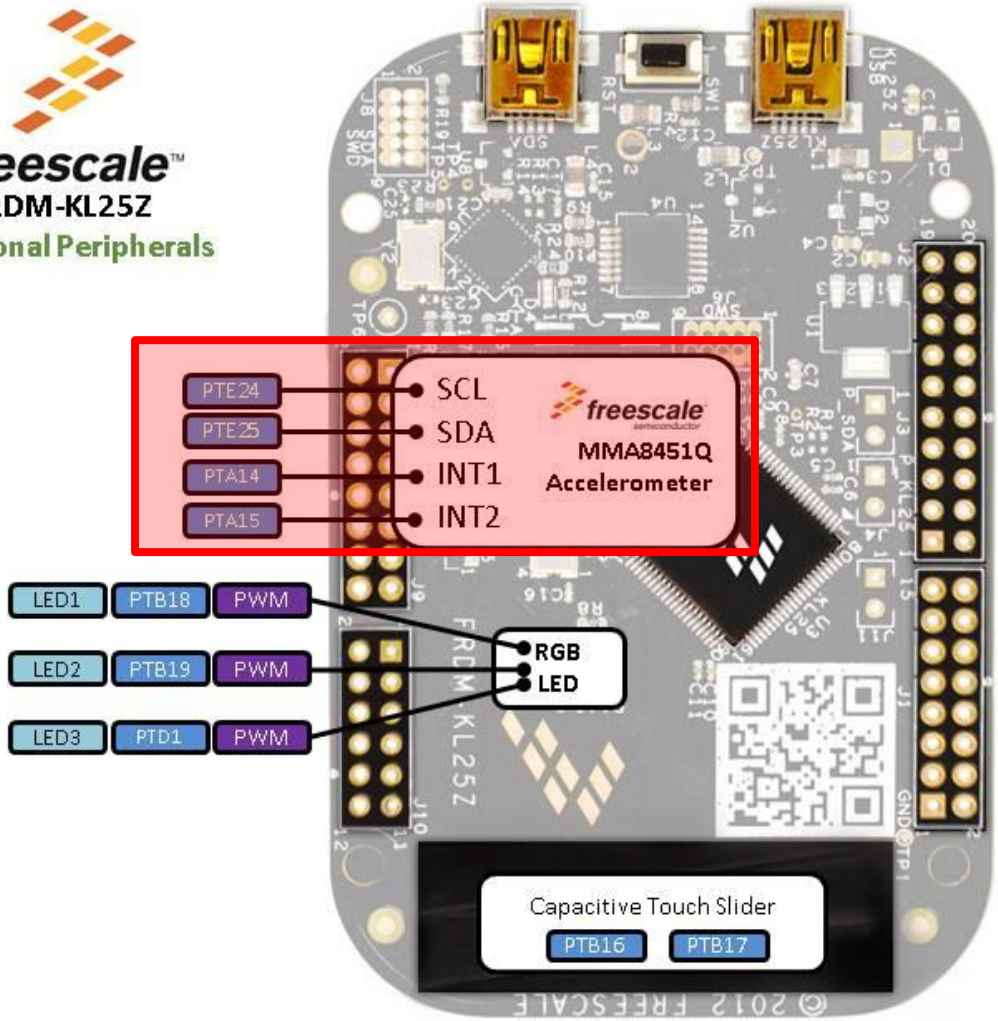
1. LED RGB.
2. Módulo PWM.
3. Comunicação serial.
4. **Acelerômetro.**

Acelerômetro

- Dispositivo para medição de aceleração própria.
- O kit FRDM KL25Z possui um acelerômetro triaxial, do tipo capacitivo, modelo MMA8451Q.
- O MMA8451Q é conectado ao KL25Z via interface de comunicação serial I2C, com a seguinte pinagem:
  - SCL = PTE24
  - SDA = PTE25

Acelerômetro

freescale™
FRDM-KL25Z
Additional Peripherals



mbed Enabled

Acelerômetro

```
#include "mbed.h"
#include "MMA8451Q.h" //biblioteca do Acelerometro
// Definição dos pinos de acordo com o modelo do KIT
#define SDA PTE25
#define SCL PTE24
#define MMA8451_I2C_ADDRESS (0x1d<<1)

int main(void)
{   //ativa o periférico e aloca os pinos da com. I2C
    MMA8451Q acc(SDA, SCL, MMA8451_I2C_ADDRESS);
    PwmOut rled(LED1);
    PwmOut gled(LED2);
    PwmOut bled(LED3);
    printf("MMA8451 ID: %d\n", acc.getWhoAmI());
    while (true) {
        float x, y, z;
        x = acc.getAccX(); //valor de acordo com a posição do eixo x
        y = acc.getAccY(); //valor de acordo com a posição do eixo y
        z = acc.getAccZ(); //valor de acordo com a posição do eixo z
        rled = 1.0 - abs(x); //intensidade da componente x na cor vermelha
        gled = 1.0 - abs(y); //intensidade da componente y na cor verde
        bled = 1.0 - abs(z); //intensidade da componente z na cor azul
        wait(0.2);
        printf("X: %1.2f, Y: %1.2f, Z: %1.2f\r", x, y, z);
    }
}
```