PSI2662 – Projeto em Sistemas Eletrônicos Embarcados: Sensores e Atuadores

Temporizadores e Interrupção, AD/DA, Display, Integração com MatLab

Escola Politécnica da Universidade de São Paulo

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

Segundo Semestre de 2015

Sumário

1. **Temporizadores e Interrupção.**
2. Entradas Analógicas e Saída Analógica.
3. Interface display 7 segmentos.
4. Integração com o MatlabR2014.
5. Acessórios.

Temporizadores e Interrupção

- **Timers**
- O módulo FRDM-KL25Z possui como base de clock um cristal de 8 MHz.
- A interface Timer é usado para criar, iniciar, parar e ler um temporizador para medir pequenos períodos de tempo (entre microssegundos e segundos).
- Qualquer quantidade de Timer pode ser criado, e pode ser ligado e desligado de forma independente.

Temporizadores e Interrupção

- São baseados em 32 bits → pode contar até 2^{31}-1 microssegundos, ou seja, 30 minutos.
- Comandos importantes:
  - **Timer** t **:** define um timer com o *label* t.
  - t.**start**(): inicializa o timer t.
  - t.**stop**(): para o timer t.
  - t.**reset**(): reseta o timer t para 0.
  - t.**read**(): lê o tempo do timer t em segundos.
  - t.**read_ms**(): lê o tempo do timer t em milisegundos.
  - t.**read_us**(): lê o tempo do timer t em microsegundos.

Temporizadores e Interrupção

```
#include "mbed.h"
Timer t;
int main()
{
    t.start();
    printf("Hello World!\n");
    wait(0.2);
    t.stop();
    printf("The time taken was %f seconds\n", t.read());
}
```

Temporizadores e Interrupção

- **Timeout:**
- Usado para configurar uma interrupção para chamar uma função após um atraso especificado.
- Qualquer número de objetos timeout pode ser criado.

Temporizadores e Interrupção

- Comandos Importantes:
  - **Timeout** name: define um evento do tipo timeout.
  - name**.attach** (&nome_rotina, t): define que a execução do programa será desviada para a rotina nome_rotina após t segundos.
  - name**.attach_us** (&nome_rotina, t): define que a execução do programa será desviada para a rotina nome_rotina após t microsegundos.

Temporizadores e Interrupção

```
#include "mbed.h"
Timeout flipper;
DigitalOut led1(LED_RED);
DigitalOut led2(LED_GREEN);
int aux = 0; //variavel global
void flip() {
    aux = !aux;
}
int main() {
    led1 = 1;
    led2 = 1;
    flipper.attach(&flip, 5.0);//chama a funçao flip apos 5 segundos
    while (true) {
        if (aux == 0)
            {   led2=1;
                led1 = !led1;
                wait(0.2);
            }
        else
            {   led1=1;
                led2 = !led2;
                wait(0.2);
            }
    }
}
```

Temporizadores e Interrupção

- **Ticker:**
- Usado para configurar uma interrupção recorrente para chamar repetidamente uma função a uma taxa especificada.
- Qualquer número de objetos Ticker pode ser criado, permitindo que várias interrupções pendentes ao mesmo tempo.
- A sintaxe é a mesma do **Timeout**.

Temporizadores e Interrupção

```
#include "mbed.h"
Ticker flipper;
DigitalOut led1(LED_RED);
DigitalOut led2(LED_GREEN);
int aux = 0; //variavel global
void flip() {
    aux = !aux;
}
int main() {
    led1 = 1;
    led2 = 1;
    flipper.attach(&flip, 5.0);//chama a funçao flip apos 5 segundos
    while (true) {
        if (aux == 0)
            {   led2=1;
                led1 = !led1;
                wait(0.2);
            }
        else
            {   led1=1;
                led2 = !led2;
                wait(0.2);
            }
    }
}
```

Temporizadores e Interrupção

- **InterruptIn:**
- Usado para disparar um evento quando um há mudanças em um pino de entrada digital.
- Comandos importantes:
  - **Interruptin** name(pin): cria uma interrupção associada a um pino de entrada.
  - name.**rise**(&nome_rotina_interrupt): define que a rotina de interrupção nome_rotina_interrupt é chamada quando acontecer borda de subida.
  - name.**fall**(&nome_rotina_interrupt): define que a rotina de interrupção nome_rotina_interrupt é chamada quando acontecer borda de subida.

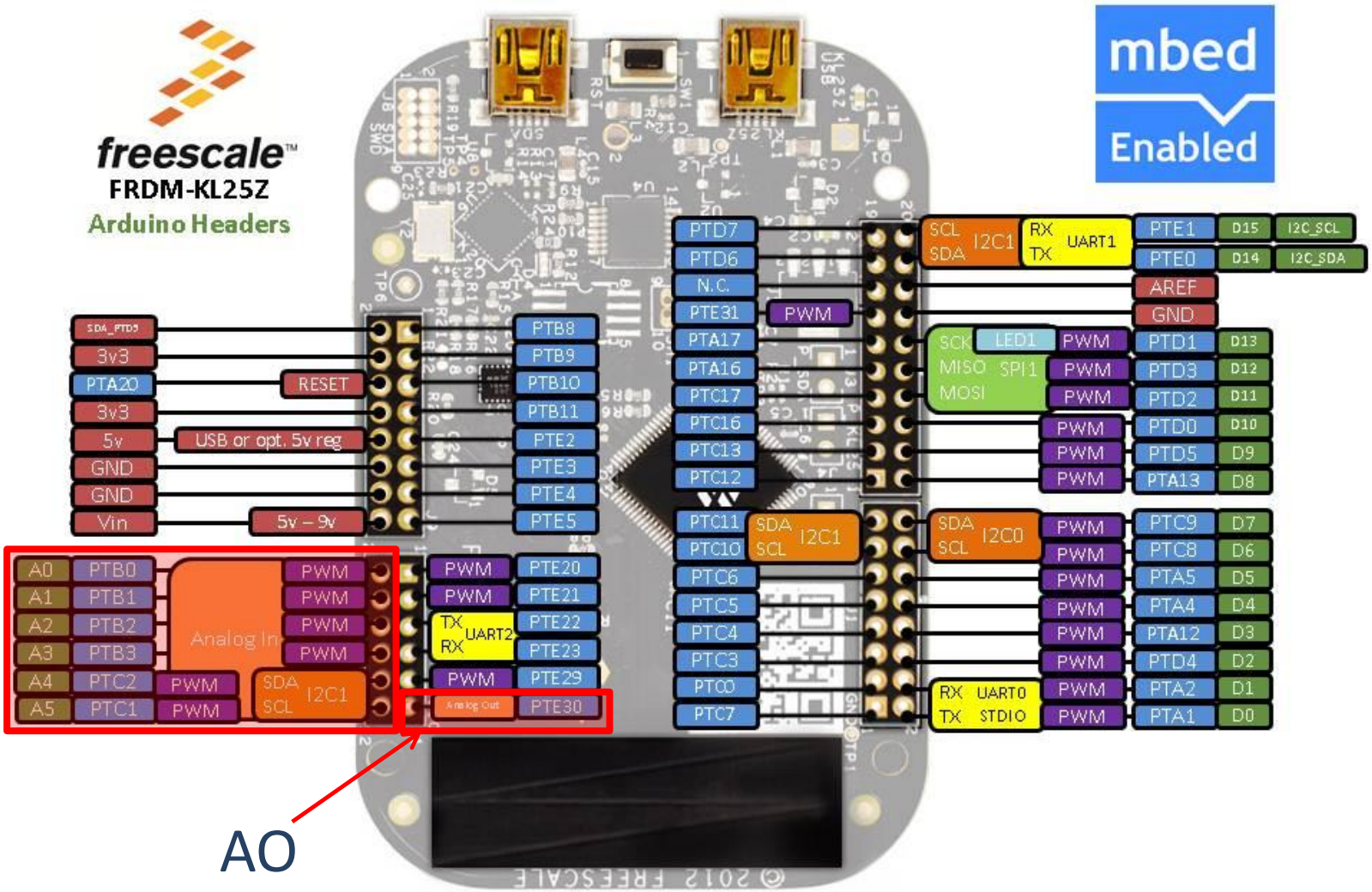
Temporizadores e Interrupção

```
#include "mbed.h"
Serial pc(USBTX, USBRX); //tx, rx
DigitalOut led_red(LED_RED);
DigitalOut led_green(LED_GREEN);
InterruptIn sw(PTA1); //define o pino de interrupcao externa
int aux = 0; //variavel global
void sw_release(void) //quando o botão for solto na configuracao pull-up
{
    aux = !aux;
    pc.printf("Interrupcao Gerada\n\r");
}
int main()
{   sw.mode(PullUp); //habilita bota para modo pull-up interno
    sw.rise(&sw_release); //desvia para a iqr com borda de subida
    while (true) {
        if (aux == 0){
            led_red = 0; led_green=0;
        }
        else{
            led_red = 1; led_green=0;
        }
    }
}
```

Sumário

1. Temporizadores e Interrupção
2. **Entradas Analógicas e Saída Analógica.**
3. Interface display 7 segmentos
4. Integração com o MatlabR2014.
5. Acessórios.

Entradas Analógicas e Saída Analógica



Entradas Analógicas e Saída Analógica

- AnalogIn PinName(pin): define uma entrada analógica, com nome PinName, conectada ao pino pin.
- AnalogOut PinName(PTE30): define uma saída analógica, com nome PinName, conectada ao pino PTE30 (único).
- OBS: Como já visto, uma saída PWM pode ser utilizada para gerar uma saída analógica.

Entradas Analógicas e Saída Analógica

- Exemplo de conexão da saída analógica em uma entrada analógica

```
// Saida dente de serra. Ler entrada analogica e mostrar na serial
#include "mbed.h"
AnalogOut tri(PTE30);
AnalogIn ain(PTB0);
int main()
{
    float temp;
    while(1) {
        tri = tri+0.1;
        wait(0.5);
        if(tri == 1) {
            tri = 0;
        }
        temp = ain;
        printf("%.2f\n\r",temp);
    }
}
```

Sumário

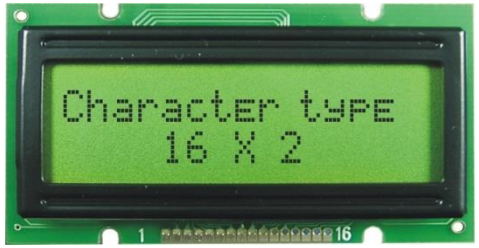
1. Temporizadores e Interrupção
2. Entradas Analógicas e Saída Analógica.
3. **Interface display 7 segmentos.**
4. Integração com o Simulink do MatlabR2014.
5. Acessórios.

Interface display 7 segmentos

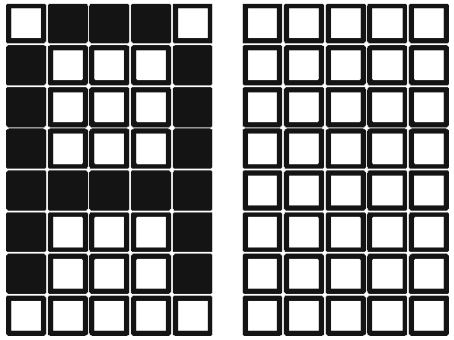
- Há diversos tipos e tamanhos de LCD alfanuméricos disponíveis comercialmente.
- Eles são sempre especificados em número de caracteres exibidos, no formato de colunas e linhas.
- Mais comuns: 08x02 (oito colunas por duas linhas), 16x01 (16 colunas por 1 linha), **16x02 (16 colunas por 2 linhas)**, 16x04 (16 colunas por 4 linhas), 20x01 (20 colunas por 1 linha), 20x02 (20 colunas por 2 linhas) e 20x04 (20 colunas por 4 linhas).

Introdução

- Exemplo de um display LCD 16x02:



- Cada “célula”(caractere) do LCD é possui 8 pixels na vertical e de 5 pixels na horizontal.
- Os caracteres ocupam 57 pixels, pois a linha inferior é normalmente reservada para o cursor.



Introdução

- Os LCDs mais comuns são os gerenciados por um chip controlador Hitachi HD44780.
- O barramento de dados pode ser de quatro bits (modo *nibble*) ou oito bits (modo *byte*).
- No modo *nibble*, a ser considerado, apenas as quatro linhas mais significativas de dados (D4 a D7).

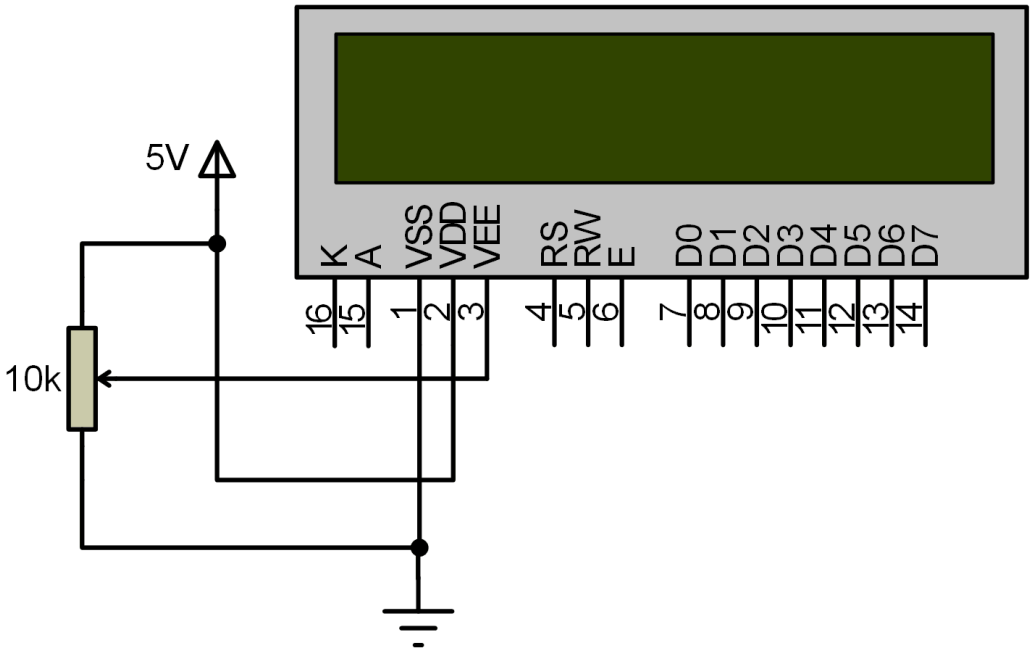
Interface display 7 segmentos

- Na maioria dos displays, a pinagem está impressa.

PINO	SÍMBOLO	FUNÇÃO
1	VSS	GND (0V)
2	VDD	DC +5V
3	Vo	Ajuste de contraste
4	D/I	Dados/Instruções
5	R/W	Leitura / Escrita
6	E	Enable
7	DB0	Linha de dados 0
8	DB1	Linha de dados 1
9	DB2	Linha de dados 2
10	DB3	Linha de dados 3
11	DB4	Linha de dados 4
12	DB5	Linha de dados 5
13	DB6	Linha de dados 6
14	DB7	Linha de dados 7
15	A	LED +
16	K	LED –

Interface display 7 segmentos

- O pino 1 (V_{ss}) é ligado ao terra e o pino 2 (V_{dd}) na tensão de 5V.
- No pino 3 (*Vo*) deve-se ter uma tensão entre 0 e 5V para ajuste do contraste. Isso pode ser feito como:



Interface display 7 segmentos

- O display reconhece dois tipos de informação na via de dados: instruções (comandos) e dados.
- RS=0: modo comandos; RS=1: modo dados.
- O pino RW controla a operação. RW=0: escrita; RW=1: leitura.

Interface display 7 segmentos

- Biblioteca TextLCD: disponível no MBED para utilização do display LDC no kit FR25Z, modo *nibble*.
- A mesma deve ser incluída no cabeçalho do programa principal: `#include "TextLCD.h"`
- O seguinte comando define a pinagem utilizada do kit para interface com o display LCD:

```
TextLCD lcd(RS, Enable, DB4, DB5, DB6, DB7);
```

- Como exemplo, o comando:

```
TextLCD lcd(PTC12, PTC13, PTC5, PTC6, PTC10, PTC11);
```

faz as seguintes atribuições:

Interface display 7 segmentos

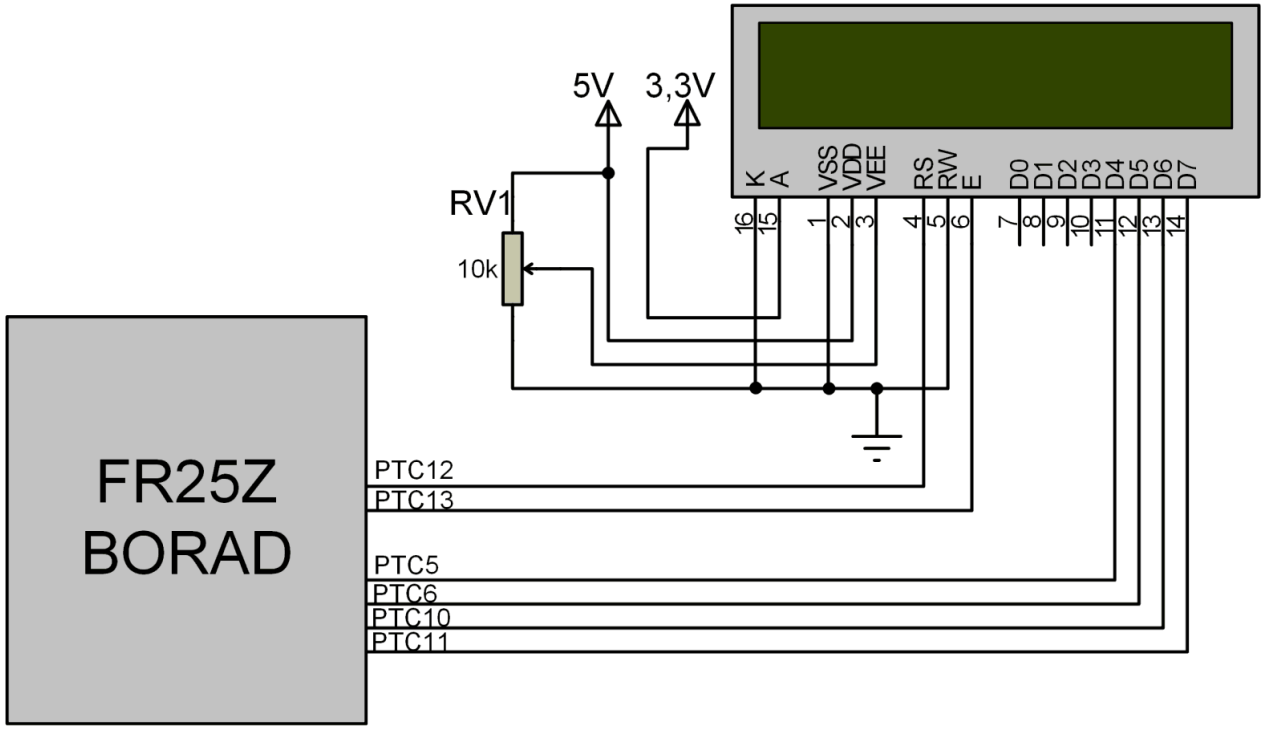
```
/* PINAGEM DA LIGAÇÃO FR25Z - LDC **
-----------------------------------------
 *  - DB4              : d4  -> PTC5
*  - DB5               : d5  -> PTC6
*  - DB6               : d6  -> PTC10
*  - DB7               : d7  -> PTC11
*  - RS                : d8  -> PTC12
*  - Enable            : d9  -> PTC13
*  - RW                : GND (only W mode)
-----------------------------------------
***/
```

Interface display 7 segmentos

- O comando `lcd.cls()` limpa os caracteres do display;
- O comando `lcd.printf()` imprime *strings* constantes e variáveis no display.

Interface display 7 segmentos

- Utilizar o display LCD 16x02 para mostrar o valor do acumulador de um contador cíclico de 0 a 59, com incremento a cada 0,5 segundos.
  - O seguinte esquema de ligação é efetuado:



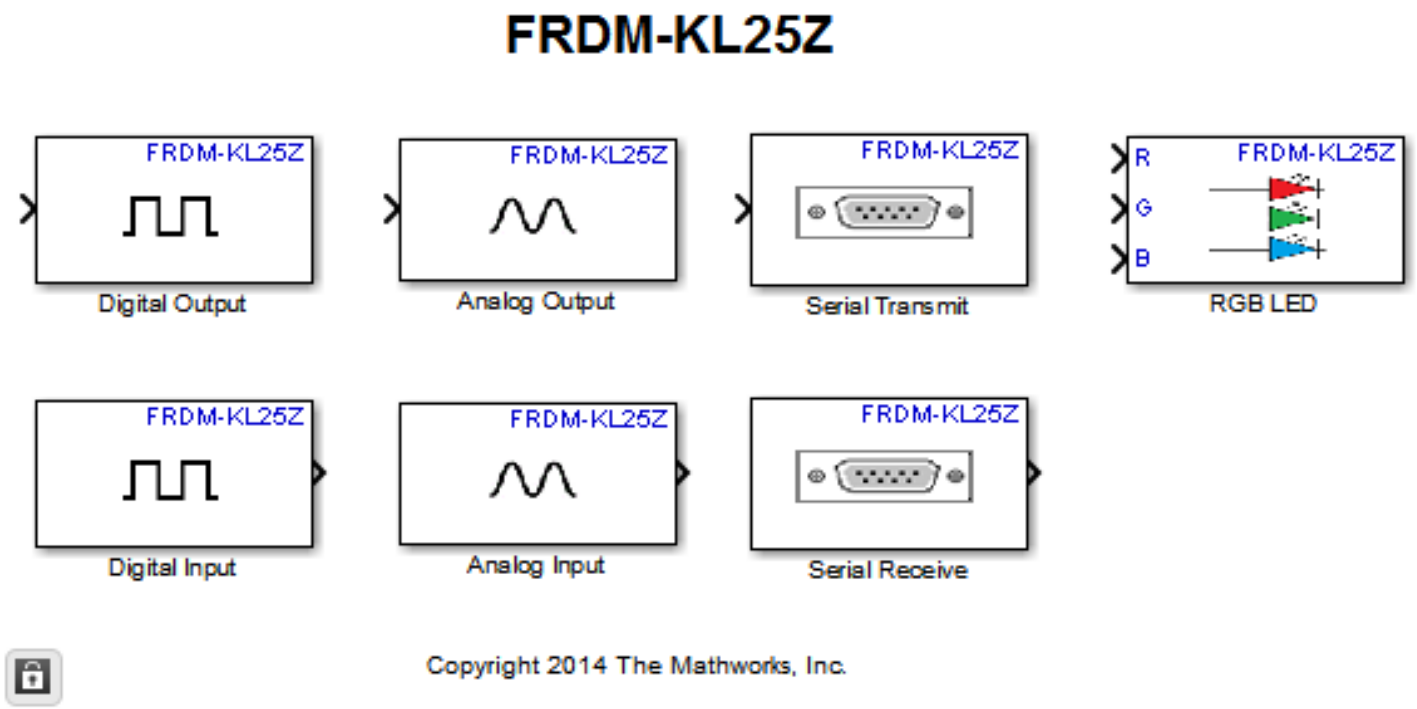
Interface display 7 segmentos

```
#include "mbed.h"
#include "TextLCD.h"
TextLCD lcd(PTC12, PTC13, PTC5, PTC6, PTC10, PTC11);
int main()
{
    int count = 0;
    lcd.cls();
    lcd.printf("CONTA DE 0 A 59");
    wait(0.5);
    while(1)
    {
        lcd.locate(0,1);
        lcd.printf("CT=%2d", count);
        count = count + 1;
        wait(0.5);
        if (count == 59) count=0;
    }
}
```

Sumário

1. Temporizadores e Interrupção
2. Entradas Analógicas e Saída Analógica.
3. Interface display 7 segmentos.
4. **Integração com o Simulink do MatlabR2014.**
5. Acessórios.

Integração com o MatlabR2014



- Vídeo explicativo da instalação:

  http://www.mathworks.com/videos/freescale-cup-installing-the-freescale-frdm-kl25z-embedded-coder-support-package-94853.html

Sumário

1. Temporizadores e Interrupção
2. Entradas Analógicas e Saída Analógica.
3. Interface display 7 segmentos.
4. Integração com o Simulink do MatlabR2014.
5. **Acessórios.**

Acessórios

- Alguns “shields” padrão arduino:

“Shield” Ethernet

“Shield” Ethernet

“Shield” Bluetooth - Serial

“Shield” Cartão SD e Dataflash

“Shield” com dois relés NA/NF

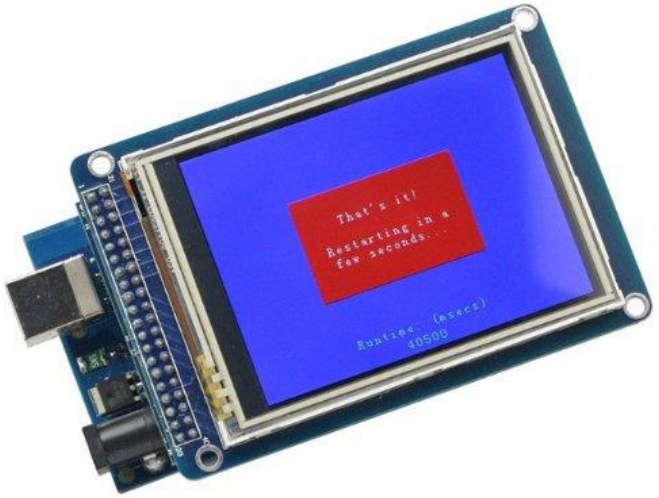
“Shield” com dois relés NA/NF

Acessórios

Display LCD gráfico 64x128
matriz de pontos

“Shield” serial RS-232

“Shield” com driver /ponte H para
acionamento de motores



“Shield” Display LCD matriz de
pontos