ScTec – Automação e Projetos Especiais

www.sctec.com.br

AN501 Termômetro digital com display LCD, utilizando a Microlab X1 e o MSP430F449

05/2006 Autor: Fábio Pereira – ScTec

Esta nota de aplicação vai demonstrar como utilizar o display LCD estático que integra a Microlab X1. Para tanto, iremos apresentar um simples termômetro digital implementado utilizando o sensor de temperatura que integra o módulo ADC12 do MSP430F449.

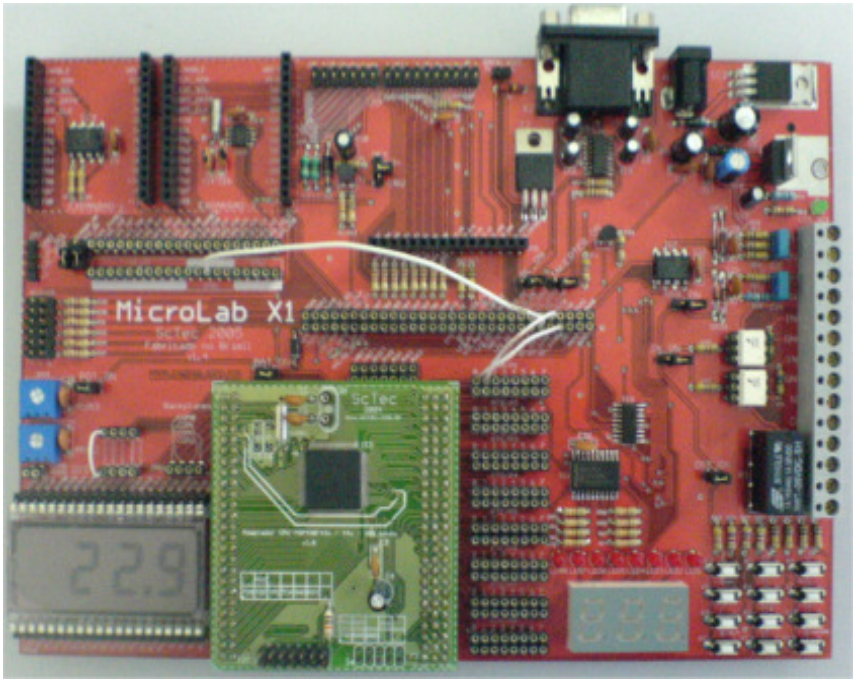

Figura 1

A implementação do hardware deste exemplo é extremamente simples e constitui-se apenas da conexão dos pinos P1.0 ao sinal TCL_LIN0, P1.1 ao sinal TCL_LIN1 e TCL_COL0 ao GND.

O software, por outro lado, implementa diversas técnicas de baixo consumo, utilizando intensivamente as capacidades do MSP430.

O laço principal do programa é extremamente simples: apenas a uma chamada à função de inicialização (que configura o clock do MSP430F449 para uma freqüência de 1MHz, o ADC12 para o modo de conversão única com 1024 ciclos de amostragem, o controlador de LCD para o modo estático e o timer básico para gerar duas interrupções por segundo), seguida pela entrada no modo LPM3 de baixo consumo.

Durante o funcionamento do termômetro, a CPU permanece operando em modo LPM3, sendo acordada periodicamente por um dos seguintes eventos:

- Timer Básico
- Interrupção da porta 1
- Término de uma conversão do ADC12

No primeiro caso, o timer básico, configurado para gerar interrupções a 2Hz, é o responsável pelos seguintes procedimentos dentro do software:

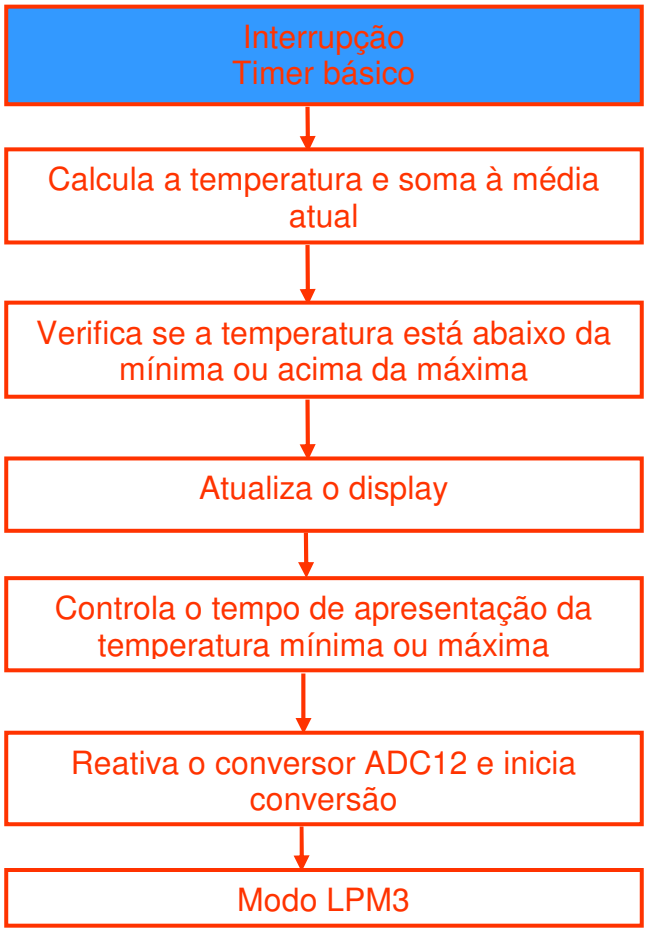

Figura 2

O controle de tempo de apresentação da temperatura mínima ou máxima consiste apenas num temporizador que é decrementado até zero pela função. No instante da seleção de um novo modo, este temporizador é inicializado com o valor 5 (equivalente a 5 segundos). Enquanto o temporizador possui um valor não-zero, o modo selecionado é utilizado. No instante em que o temporizador atinge o valor zero, o software seleciona o modo normal (apresentação da temperatura atual).

Observe que após o processamento da interrupção, a CPU é novamente colocada no modo LPM3 (o que acontece automaticamente quando, no retorno da interrupção, o estado prévio do SR é retirado da pilha e armazenado no SR novamente).

A interrupção da porta 1 é responsável pela detecção do pressionamento das teclas K2 e K5 do teclado. A função de tratamento de interrupção irá determinar o modo atual de exibição do display com base na tecla pressionada.

Observe que se pressionarmos as duas teclas simultaneamente, iremos comandar o apagamento das memórias de temperatura mínima e máxima.

A última função de tratamento de interrupção, a do ADC12, é utilizada apenas para provocar a desativação do mesmo, de forma a reduzir o consumo do sistema. O ADC12 somente será novamente ativado quando a interrupção do timer básico o comandar.

As figuras 3 e 4 apresentam a distribuição dos diversos elementos utilizados neste exemplo.

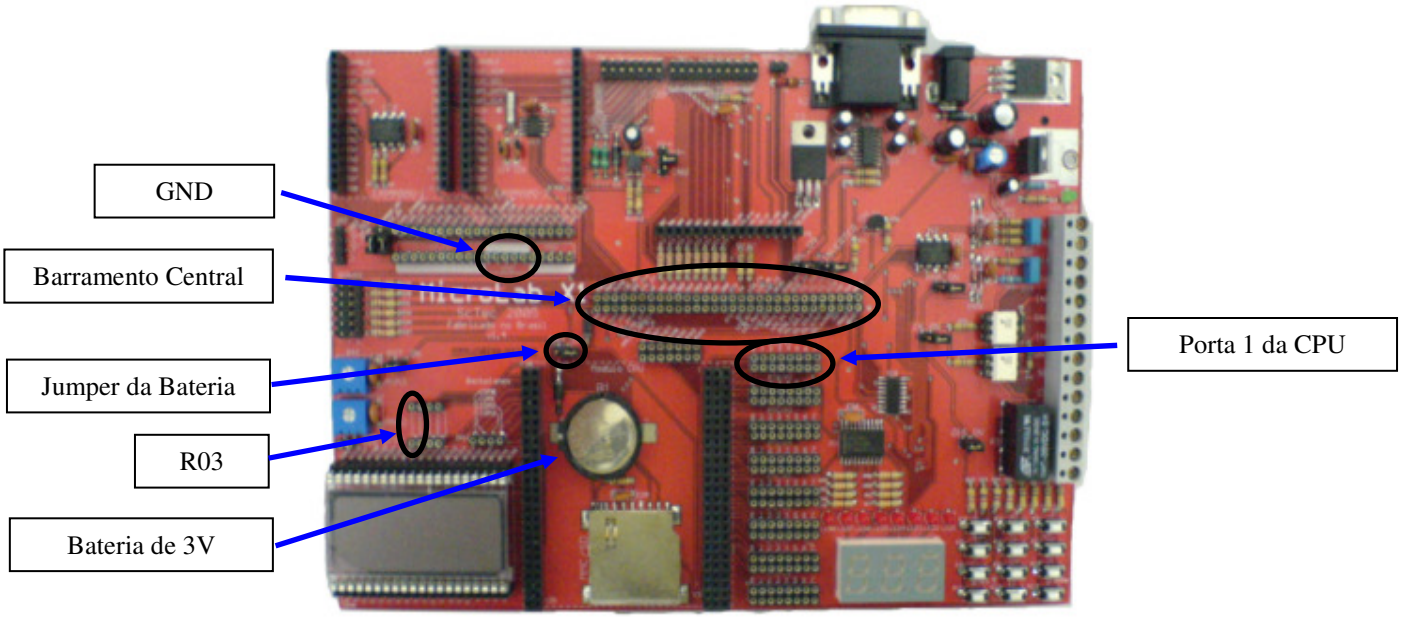

Figura 3

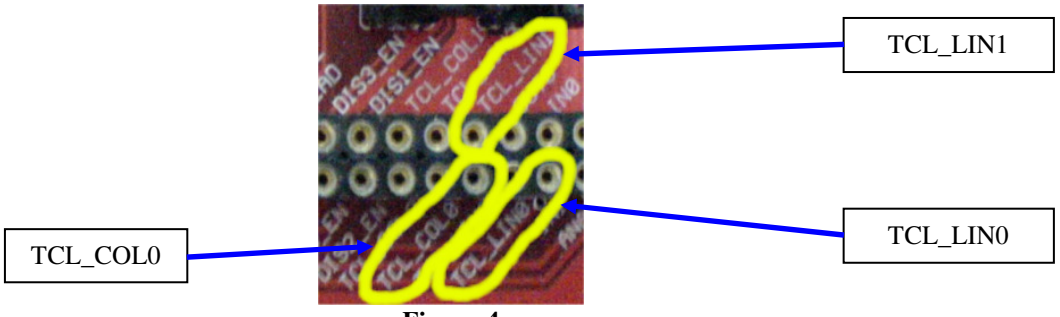

Figura 4

O diagrama esquemático do módulo adaptador de CPU MSP430F449, o manual da Microlab X1 podem ser baixados no site: www.sctec.com.br.

O arquivo "lcd_estatico.c" pode ser baixado diretamente em: www.sctec.com.br/downloads/lcd_estatico.c .

```
/*****************************************************************************
 ************************** ScTec 2006 ***************************************
 ************************ www.sctec.com.br ***********************************
 *****************************************************************************
 *                                                                           *
 * Termômetro digital na Microlab X1, utilizando um display LCD estático e um  *
 * MSP430F449.                                                               *
 *                                                                           *
 *****************************************************************************
 Conexões Necessárias:

 1 - fio conectando o pino R01 do MSP430 ao terra (contraste do LCD)
 2 - P1.0 ao sinal TCL_LIN0 (temperatura máxima)
 3 - P1.1 ao sinal TCL_LIN1 (temperatura mínima)
 4 - TCL_COL0 ao sinal GND

 O arquivo lcd_estatico.c pode ser baixado do site da sctec

 Para maiores informações consulte o manual da Microlab X1, o esquemático do
 módulo de CPU, o datasheet do componente e o livro MSP430: Teoria e Prática

 *****************************************************************************/

#include <io430x44x.h>
#include <stdio.h>
#include <intrinsics.h>
#include "lcd_estatico.c"

#define numero_medias 16  // média de 16 amostras

char string[5];
unsigned char modo=0, tempo_apresentacao=0;
signed long int temperatura, temperatura_media;
signed int maxima=-999, minima=1999;

void atualiza_display(void);

// RTI para tratamento da interrupção da porta 1
#pragma vector = PORT1_VECTOR
__interrupt void trata_tecla(void)
{
  modo = P1IFG; // atualiza o modo de apresentação
  P1IFG = 0;  // apaga a interrupção
  if (!(P1IN & 0x03)) // se as duas teclas forem pressionadas
  {
    maxima = -999;  // inicializa a máxima em -99.9 graus
    minima = 1999;  // inicializa a mínima em 199.9 graus
    modo = 0; // modo normal
  }
  tempo_apresentacao = 5; // mostra durante cinco segundos
  atualiza_display(); // atualiza o display
}

#pragma vector = ADC12_VECTOR
__interrupt void trata_adc12(void)
{
  ADC12CTL0 = 0; // o primeiro apagamento do ADC12CTL0 apenas apaga o ENC
```

```
  ADC12CTL0 = 0; // agora apaga todos os bits do ADC12CTL0 (desliga o conversor)
  ADC12IFG = 0; // apaga a interrupção do ADC12
}

// RTI para tratamento do timer básico
#pragma vector = BASICTIMER_VECTOR
__interrupt void trata_basictimer(void)
{
  static unsigned char contagem;
  static signed long int mediaX;
  // lê o resultado da conversão e calcula a temperatura
  temperatura = (((long)ADC12MEM0*4230)>>12)-2780;
  temperatura_media = mediaX/numero_medias;
  // faz a média das últimas 4 leituras
  mediaX = mediaX + temperatura - temperatura_media;
  // a cada segundo, calcula e atualiza a temperatura atual
  if (++contagem>2)
  {
    // se a temperatura atual for 0,5 graus acima ou abaixo da média, recalcula a média
    if (temperatura>(temperatura_media+5) || temperatura<(temperatura_media-5))
    {
      mediaX = temperatura * numero_medias;
      temperatura_media = mediaX/numero_medias;
    }
    // atualiza a máxima e a mínima
    if (temperatura_media>maxima) maxima = temperatura_media;
    if (temperatura_media<minima) minima = temperatura_media;
    temperatura = temperatura_media;
    // calcula o módulo da temperatura
    if (temperatura<0) temperatura=-temperatura;
    atualiza_display(); // atualiza o display
    contagem = 0;
    // decrementa o tempo de apresentação da máxima e mínima, quando chega a zero,
    // retorna ao modo normal
    if (tempo_apresentacao) tempo_apresentacao--; else modo = 0;
  }
  // reativa o ADC12 e inicia outra conversão
  ADC12CTL0 = SHT0_15 + REFON + ADC12ON + ENC + ADC12SC;
}

void atualiza_display(void)
{
  signed int display;
  if (modo>2) modo = 0; // evita que o modo saia da faixa permitida
  switch (modo)
  {
  case 0: // modo normal, indica a temperatura
    display = temperatura;
    break;
  case 1 : // indica a máxima temperatura
    display = maxima;
    break;
  case 2 : // indica a mínima temperatura
    display = minima;
    break;
  }
  sprintf(string,"%4u",display);  // converte a temperatura para decimal
  // insere um zero à esquerda para temperaturas menores que 1 grau
  if (temperatura<10) string[2]='0';
  // apresenta o valor no display
```

```
  mostra_lcd(string[0]-'0',string[1]-'0',string[2]-'0',string[3]-'0');
  // se a temperatura for negativa
  if (temperatura_media<0)
  {
    // e maior que -9.9 coloca o sinal negativo no segmento 29,
    // senão coloca no segmento 31
    if (temperatura_media>-99) {seg29;} else seg31;
  }
  seg_p1; // liga o ponto decimal
}

void inicializacao(void)
{
  WDTCTL = WDTPW + WDTHOLD;   // desativa o watchdog
  // todos os pinos como saídas e em nível 0
  P1OUT = P2OUT = P3OUT = P4OUT = P5OUT = P6OUT = 0;
  P2DIR = P3DIR = P4DIR = P5DIR = P6DIR = 0xFF;
  P1DIR = 0xFC;
  // configura o FLL
  SCFI0 = FLLD_1; // fator D = 1
  SCFQCTL = 0x1E; // fator N = 30
  FLL_CTL0 = DCOPLUS; // fFLL = 1*(30+1)*32768 ~= 1MHz
  FLL_CTL1 = SMCLKOFF + XT2OFF; // desliga o SMCLK e o XT2
  // configura o ADC12 para clock interno e divisor por 8
  ADC12CTL1 = SHP + ADC12DIV_7;
  // seleciona o diodo de temperatura e a referência Vref+ para a memória 0
  ADC12MCTL0 = SREF_1 + INCH_10;
  // seleciona um período de amostragem de 1024 ciclos de clock do ADC12,
  // modo de conversão única
  ADC12CTL0 = SHT0_15 + REFON + ADC12ON + ENC + ADC12SC;
  ADC12IE = BIT0;
  // posiciona o cursor na primeira coluna da primeira linha
  // basic timer: fcnt1 = ACLK/32, fcnt2 = 32768 / 256 / 64 = 2Hz
  BTCTL = BTDIV + BT_fCLK2_DIV64;
  LCDCTL = LCDON + LCDSTATIC + LCDSG0_7; // ativa o LCD, modo estático
  IE2 = BTIE; // liga a interrupção do timer básico
  // configura as teclas
  P1IES = BIT1 + BIT0;  // interrupção na borda de descida de P1.0 e P1.1
  P1IFG = 0; // apaga os flags de interrupção
  P1IE = BIT1 + BIT0; // habilita a interrupção dos pinos P1.0 e P1.1
  apaga_lcd();  // apaga o display LCD
  seg29; seg17; seg9; // liga os traços centrais: ---
  __enable_interrupt(); // habilita as interrupções
}

void main(void)
{
  inicializacao();  // inicializa os registradores e variáveis
  __low_power_mode_3(); // entra em LPM3 e aguarda uma interrupção
}
```