ScTec – Automação e Projetos Especiais

www.sctec.com.br

AN900 Timer com Resolução em Minutos ou Segundos na Microlab X1 com um MSP430F149

06/2006 Autor: Edmilson S. Teixeira – ScTec

Esta nota de aplicação vem reforçar o que foi descrito na aplicação AN800 (com relação aos recursos do teclado da placa), que demonstrou um contador up/down utilizando 3 teclas, na experiência que aqui segue, utilizaremos todas as 12 teclas do teclado matricial, implementando um timer com menu e cinco modos de programação.

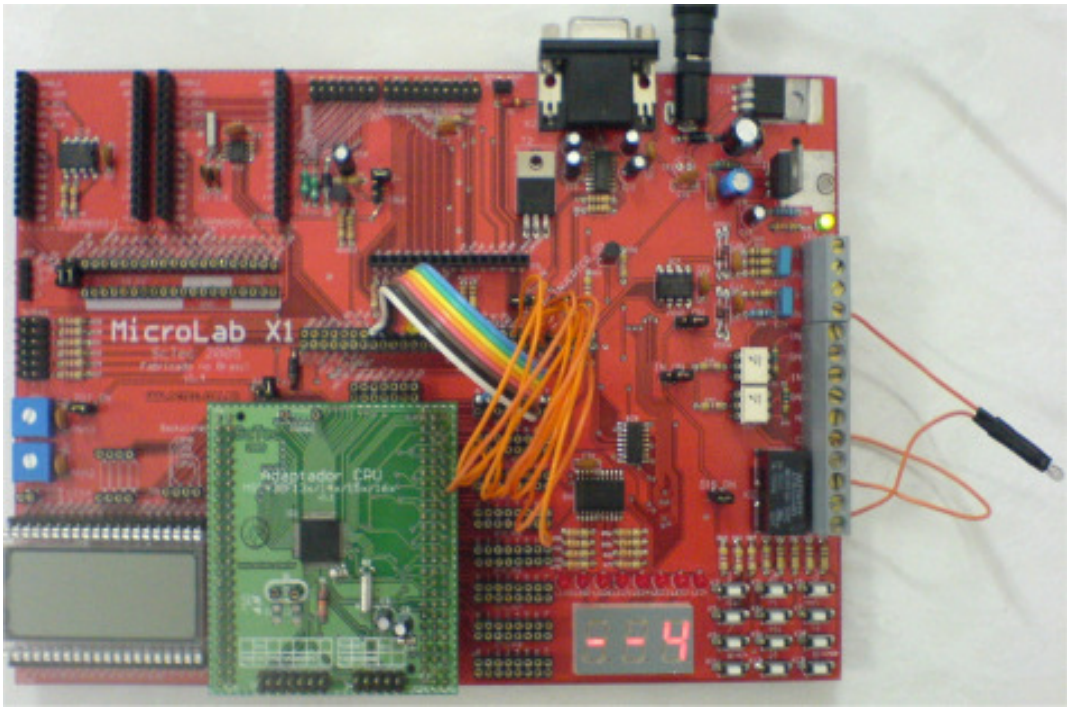


Figura 1

O teclado incluso na placa é composto de 12 teclas dispostas no formato de matriz multiplexada (figura 2).

A utilização do teclado é feita simplesmente pela colocação seqüencial das colunas (sinais TCL_COL0 a TCL_COL2) em nível "0" e a leitura dos sinais presentes nas linhas (sinais TCL_LIN0 a TCL_LIN3). Caso uma tecla da coluna esteja pressionada, a linha correspondente apresentará num nível "0", caso contrário, apresentará nível "1".

Não é recomendada a utilização das colunas com pinos da CPU configurados como entrada, pois os mesmos podem ficar flutuando causando instabilidades no programa. Tocar com os dedos na parte de baixo da placa onde se encontra o teclado também pode causar instabilidades.

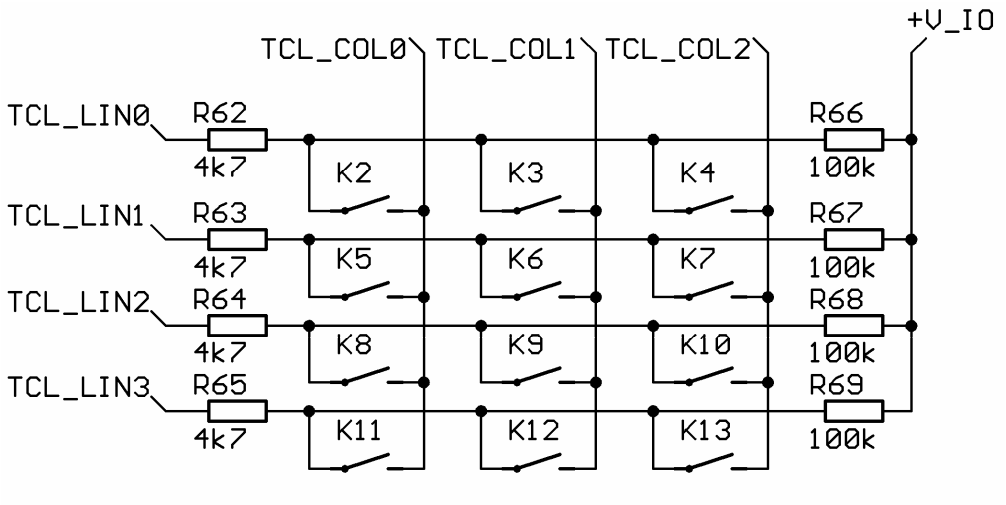


Figura 2

Conforme dito anteriormente, o timer aqui implementado funciona em cinco modos selecionáveis pela tecla * (asterisco). O primeiro, é o estado de reset, ou seja, o timer não pode acionar a saída à relé e também não pode iniciar nenhuma temporização, nesse modo é apresentado "000" nos displays de LEDs.

Pressionando a tecla * uma vez, aparece a tela "- - 1" nos displays indicando modo 1, nesse modo o timer tem resolução em minutos e conta regressivamente até zero acionando o relé.

Pressionando a tecla * uma segunda vez, aparece "- - 2" nos displays indicando modo 2, nesse modo o timer tem resolução em segundos com contagem regressiva e somente ao zerar aciona o relé.

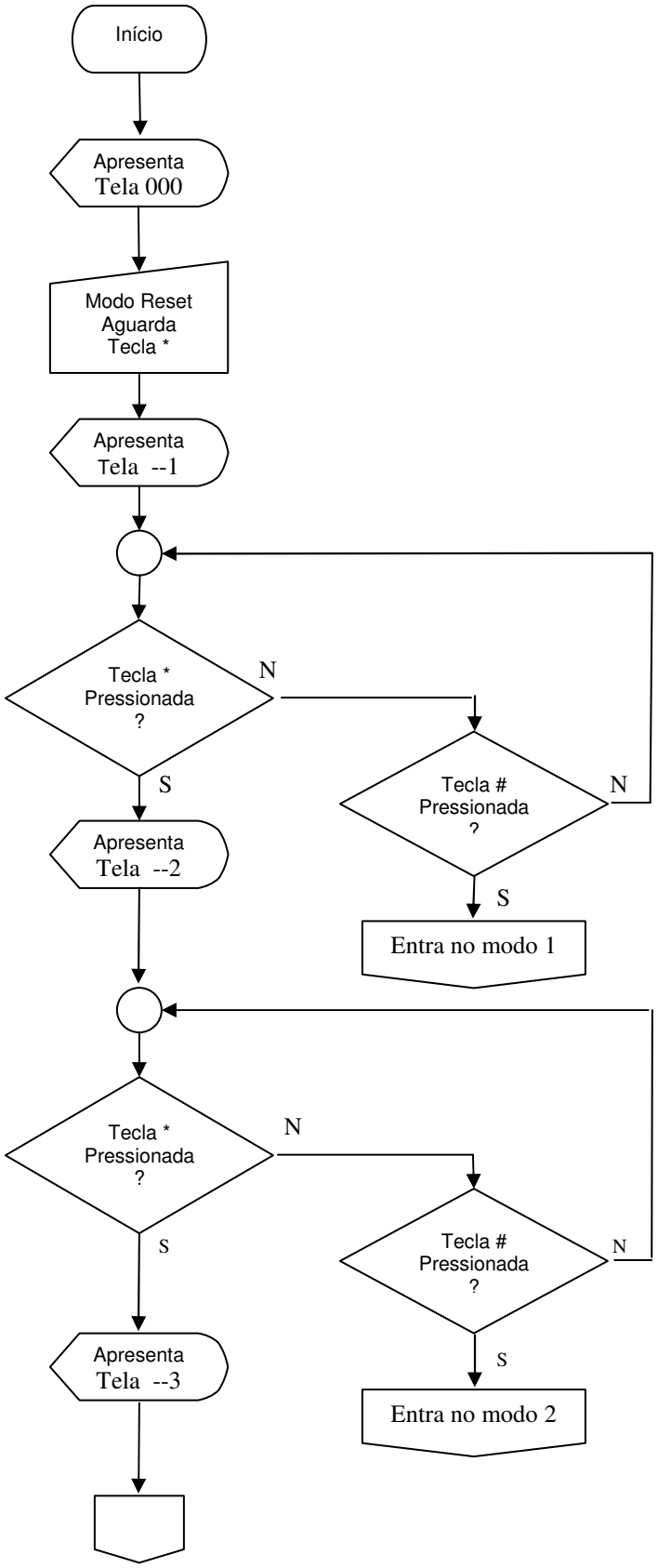
Pressionando a tecla * mais uma vez aparece a tela "- - 3", nesse modo o timer tem resolução em minutos com contagem regressiva também, mas o relé é acionado no início da contagem do tempo e é desligado quando a contagem é zerada.

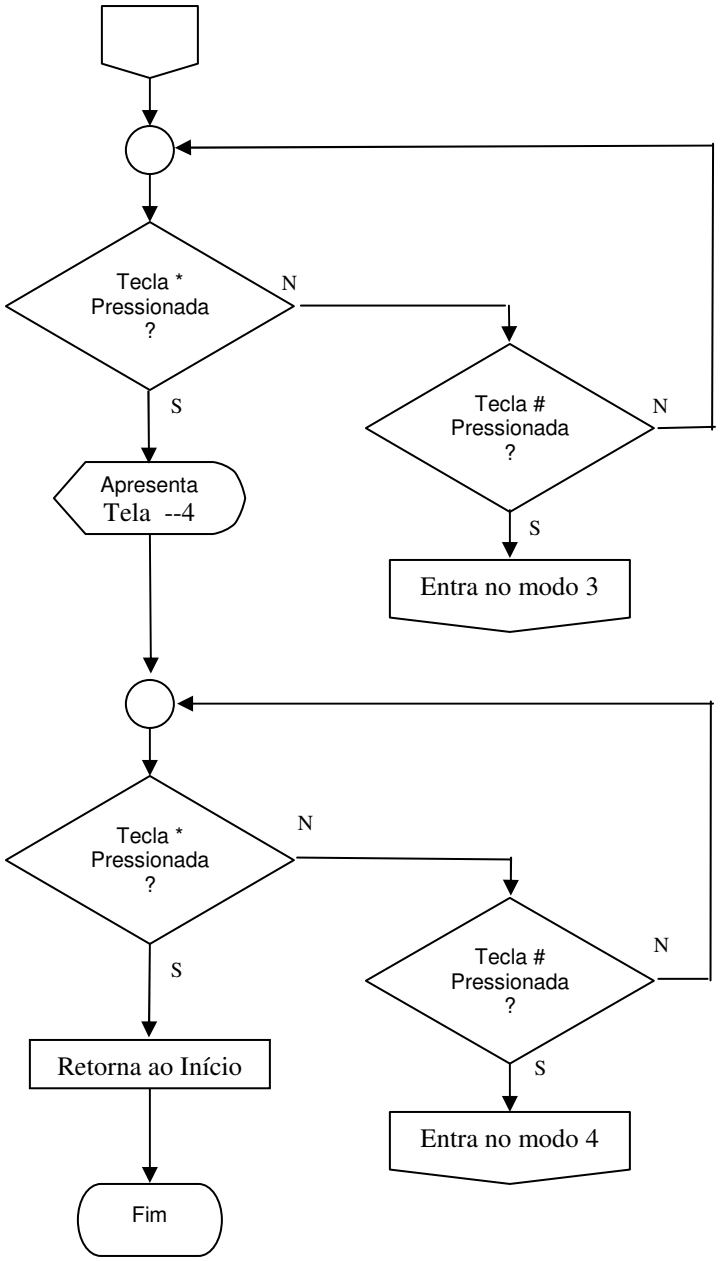
Finalmente, na quarta vez que for pressionada a tecla *, aparece a tela "- - 4", nesse modo o timer funciona como no modo 3, mas com resolução em segundos. Ainda nessa tela, se for pressionada a tecla * mais uma vez o timer retorna ao começo no modo reset "000".

Em qualquer um dos quatro modos de temporização, pressionando a tecla # (sustenido) uma vez, deverá aparecer "000" nos displays nessa tela é possível digitar o valor do tempo (entre 0 e 999). Após digitar o valor do tempo é só pressionar a tecla # mais uma vez para iniciar a temporização.

Durante a contagem do tempo ou mesmo durante a digitação do seu valor, se for pressionada a tecla *, ocorre um reset da temporização ou da digitação.

O processo do menu pode ser visualizado no fluxograma 1 e o processo de um dos modos pode ser visualizado no fluxograma 2, os outros modos diferenciam na resolução e/ou quando a saída é acionada.



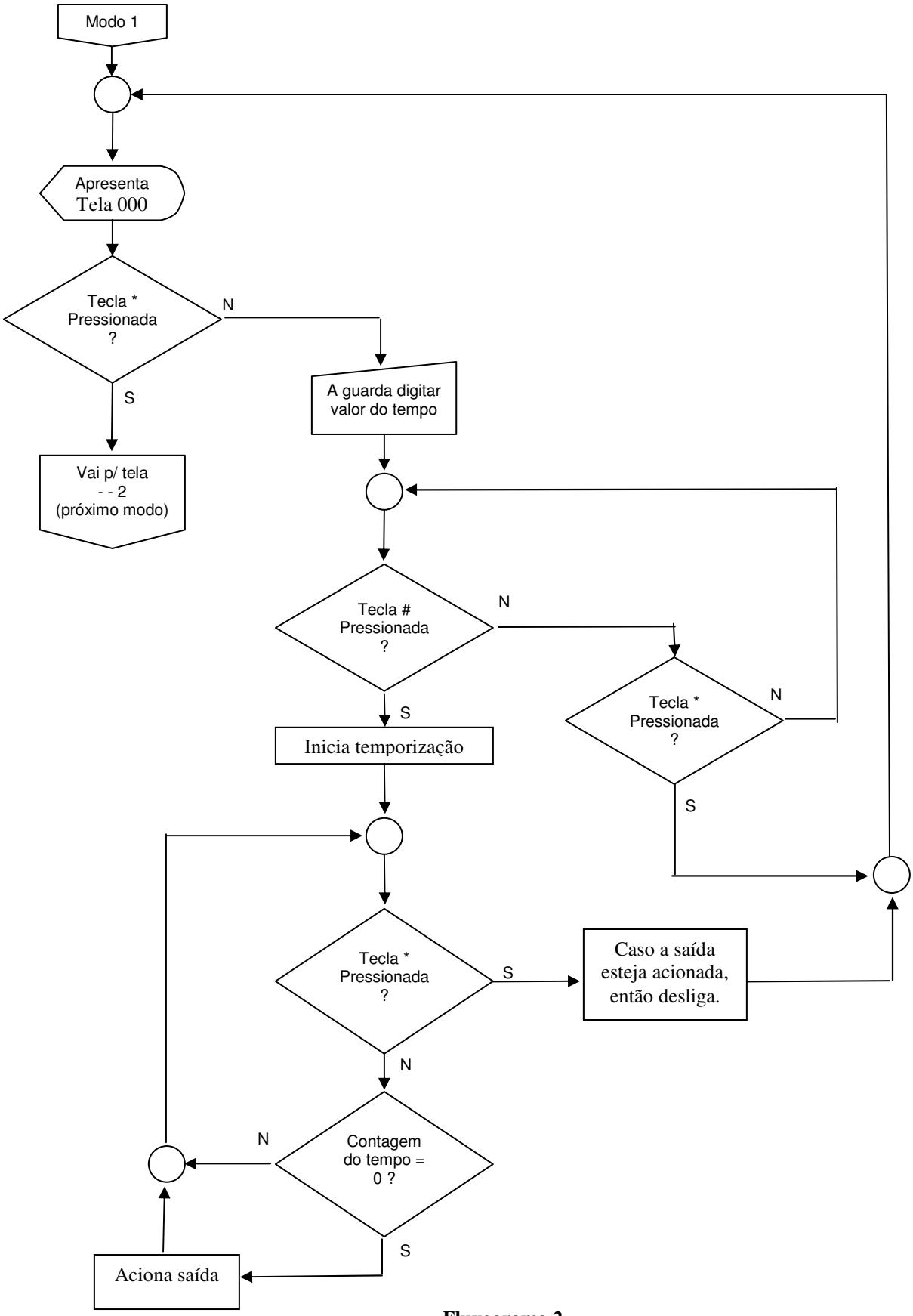


Fluxograma 1

Para compilar o software desta nota de aplicação, é necessário incluir na pasta (diretório) onde está o projeto, uma biblioteca de manipulação de teclado chamada teclado3x4.c que pode ser baixado no site: www.sctec.com.br.

O diagrama esquemático do módulo adaptador de CPU MSP430F149 e o manual da Microlab X1 podem ser baixados no mesmo site.

Lembre-se que o limite de corrente de carga do relé é de 1A x 125VCA ou 2A x 30VDC.



Fluxograma 2

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

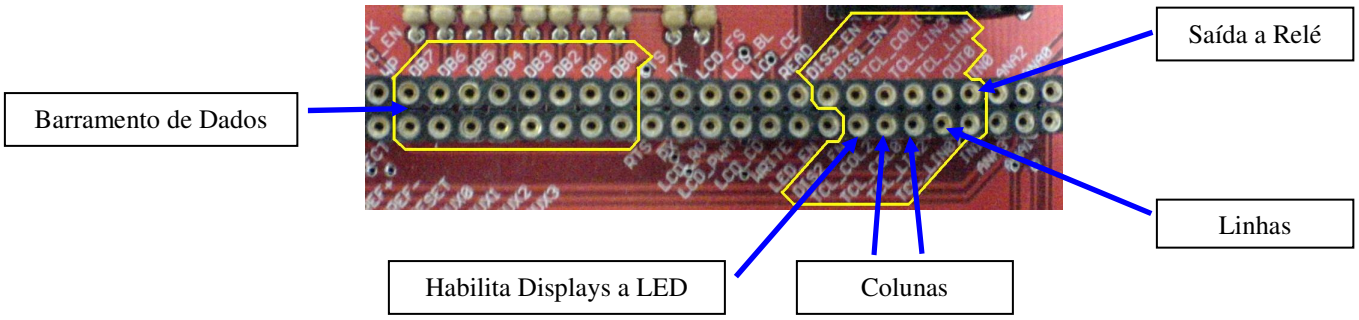


Figura 3

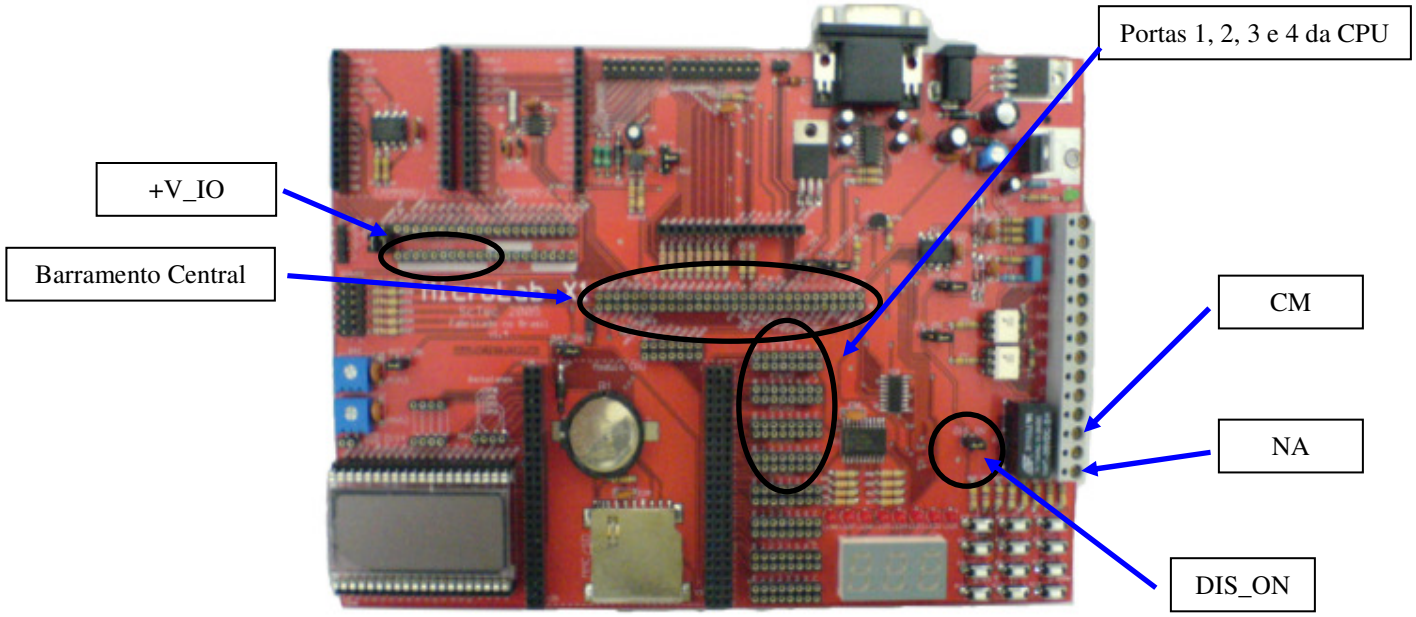


Figura 4

Para obter o correto funcionamento da experiência é necessário configurar a otimização do compilador para Low, em: "Project >> Options >> C/C++ compiler >> Optimizations".

```
/************************ ScTec 2006 ****************************
********************** www.sctec.com.br ************************
****************************************************************
********** AN900 - Temporizador com Minutos e Segundos ***********
********** Utilizando o Teclado Matricial 3 x 4 Teclas ***********
****************************************************************
  Esta aplicação demonstra como utilizar o teclado  matricial 3x4,
os displays de LED e a saída a relé da Microlab X1.
  Este programa faz um temporizador digital em que o usuário esco-
lhe o modo (vide nota abaixo) e digita o tempo. Utilizando uma CPU
MSP430F149 com um cristal oscilador externo  de  32,768KHz,  conta
regressivamente desde o tempo digitado até  zero,  então  liga  ou
desliga o relé conforme o modo.
```

```
      ****************************************************************
       Modos de temporização oferecidos:
       "000" Reset - não faz nada;
       "--1" em Minutos - liga saída no fim da contagem do tempo;
       "--2" em Segundos - liga saída no fim da contagem do tempo;
       "--3" em Minutos - desliga saída no fim da contagem do tempo;
       "--4" em Segundos - desliga saída no fim da contagem do tempo;
       Teclas:
       # - Enter - confirma modo / confirma valor do tempo;
       * - Reset / Mode - muda o modo / reseta temporização ou digitação;
       0 a 9 - Tempo entre 000 e 999 (em segundos ou minutos, depende  do
      modo).
      ****************************************************************
      *************** Conexões na Microlab X1: ************************
        Para que o programa funcione corretamente é necessário:
      * Retirar o Módulo LCD de Caracteres (16x2) da placa;
      * Fechar o jumper DIS_ON;
      * Conectar  os  pinos da porta 1 ao barramento de 8  bits  (P1.0  a
      DB0, P1.1 a DB1 e assim  sucessivamente  até o bit 7);
      * Conectar o sinal DIS1_EN ao pino P2.0;
      * Conectar o sinal DIS2_EN ao pino P2.1;
      * Conectar o sinal DIS3_EN ao pino P2.2;
      * Conectar o sinal OUT0 ao pino P4.7;
      * Conectar o sinal TCL_COL0 ao pino P3.0;
      * Conectar o sinal TCL_COL1 ao pino P3.1;
      * Conectar o sinal TCL_COL2 ao pino P3.2;
      * Conectar o sinal TCL_LIN0 ao pino P3.3;
      * Conectar o sinal TCL_LIN1 ao pino P3.4;
      * Conectar o sinal TCL_LIN2 ao pino P3.5;
      * Conectar o sinal TCL_LIN3 ao pino P3.6;
        Para maiores detalhes, favor consultar o manual da Microlab X1.
      ****************************************************************/

#include <io430x14x.h>
#include <intrinsics.h>
#include "teclado3x4.c"    // Biblioteca de manipulação do teclado matricial

// Constantes que definem os dígitos (0 1 2 3 4 5 6 7 8 9)
const char numero[] = {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x67};

// Define caractere -
#define caractere_ 0x40

// Constantes que definem os displays
#define disp_1 1
#define disp_2 2
#define disp_3 3

// Outras constantes
#define min 60
#define seg 1
#define saida P4OUT_bit.P4OUT_7

// Variáveis utilizadas em várias partes do programa
unsigned char digito_1, digito_2, digito_3;
// enter - tecla #, reset_mode - tecla *, tecla - guarda leitura do teclado,
// t_on - energiza na temporização, t_off - desenergiza na temporização,
// ctl - controla tecla enter, lib - libera contagem do tempo
unsigned char enter, reset_mode, tecla, t_on, t_off, ctl, lib;
unsigned int resol = 1;     // Guarda resolução em segundos ou minutos
```

```
unsigned int tempo = 0;     // Variável que guarda o valor do tempo em segundos
unsigned char mode = 0;     // Modos --1, --2, --3, --4 e reset "000"

// Protótipos de funções
void inicializa(void);
void menu(void);
unsigned int concatena_digitos(void);
void separa_digitos(unsigned int numero);
void mostra_disp(unsigned char dg, unsigned char disp);
void atraso(unsigned int temp);

// Função de tratamento de interrupção do Timer A
#pragma vector = TIMERA1_VECTOR
__interrupt void trata_timer(void)
{
  switch(TAIV)
    case 0x0A :                  // Caso haja estouro de contagem:
    {
    if(tempo > 0)                // Se temporização iniciada:
    {
      tempo--;                   // Decrementa tempo
      if(tempo == 0)             // Se zerar:
      {
        if(!t_off)t_on = 1;      // Liga t_on ou desliga t_off
        else {t_off = 0; t_on = 0;}
        digito_1 = digito_2 = digito_3 = lib = 0;
      }
    }
  }
}

void main(void)
{
  inicializa();
  while(1)
  {
    menu();                      // Entra nas configurações
    if(tempo)                    // Se temporização iniciada:
    {                            // Mostra tempo restante nos displays
      switch(resol)
      {
        case min :
        {
          if((tempo / 60 ) < 1)  // Se resolução está em minutos e tempo < 1min
          {                      // mostra em segundos
            digito_1 = digito_2 = digito_3 = 0;
            separa_digitos(tempo);
          }else
          {                      // Mostra tempo em minutos
            digito_1 = digito_2 = digito_3 = 0;
            separa_digitos(tempo/60);
          }
          break;
        }
        case seg :               // Se resolução está em segundos
        {                        // Mostra tempo em segundos
          digito_1 = digito_2 = digito_3 = 0;
          separa_digitos(tempo); break;
        }
      }
```

```
    }
    if(t_on)  saida = 1;             // Coloca saída em "1"
    else if(t_off && tempo > 0) saida = 1;
    if(!t_on && !t_off) saida = 0;   // Coloca saída em "0"
    mostra_disp(digito_1, disp_1);   // Apresenta primeiro digito e aguarda um tempo
    atraso(500);
    mostra_disp(digito_2, disp_2);   // Apresenta segundo digito e aguarda um tempo
    atraso(500);
    mostra_disp(digito_3, disp_3);   // Apresenta terceiro digito e aguarda um tempo
    atraso(500);
  }
}


void inicializa(void)
{
  WDTCTL = WDTPW + WDTHOLD;          // Desativa o watchdog
  P1DIR = 255;                       // Configura todos os pinos da porta 1 como saída
  P2DIR = 255;                       // Configura todos os pinos da porta 2 como saída
  P4DIR = 255;                       // Configura todos os pinos da porta 4 como saída
  P2OUT = 0;                         // Apaga os bits da porta 2;
  P4OUT = 0;                         // Apaga os bits da porta 4;
  // Configura DCO para 4.13MHz
  DCOCTL = DCO2 + MOD4 + MOD3 + MOD2;
  // MCLK = SMCLK = 4.13MHz, ativa LFXT1 em baixa frequência - clock dividido por 1
  BCSCTL1 = XT2OFF + RSEL2 + RSEL1 + RSEL0 + DIVA_0;
  BCSCTL2 = 0x00;
  // Configura Timer A, TASSEL_1 - clock auxiliar ACLK, ID_0 - clock dividido por 1
  // MC_1 - contagem até valor de TACCR0, TAIE - habilita interrupção por estouro do TA
  TACTL = TASSEL_1 + ID_0 + MC_1 + TAIE;
  // Módulo de contagem = frequência do cristal = 1 interrupção por segundo
  TACCR0 = 32767;
  __enable_interrupt();               // Habilita interrupções
  digito_1 = digito_2 = digito_3 = 0;
  enter = reset_mode = t_on = t_off = ctl = lib = 0;// Inicializa variáveis
}

// Função que mostra menu, escolhe um dos 4 modos de temporização e define valor do
// tempo
void menu(void)
{
  tecla = varre_teclado();          // Faz varredura de teclado
   if(tecla == 254) return;         // A mesma tecla está pressionada
   if(tecla == 255) return;         // Nenhuma tecla pressionada
   if(tecla == 10)                  // Se foi pressionada a tecla "*" (reset_mode):
    {
      if(ctl || lib)                // Caso esteja em temporização, reseta tudo
      {
        digito_1 = digito_2 = digito_3 = 0;
        t_on = tempo = lib = ctl = 0;
        saida = 0;
        return;
      }else
      {
        switch(mode)             //Se não estiver em temporização, escolhe outros modos
        {
          case 0 :
          {
            digito_2 = digito_3 = 10;// Apresenta "--1" modo: minutos
            digito_1 = 1;
```

```
        mode = 1;
      } break;
      case 1 :
      {
        digito_2 = digito_3 = 10;// Apresenta "--2" modo: segundos
        digito_1 = 2;
        mode = 2;
      } break;
      case 2 :
      {
        digito_2 = digito_3 = 10;// Apresenta "--3" modo: minutos
        digito_1 = 3;
        mode = 3;
      } break;
      case 3 :
      {
        digito_2 = digito_3 = 10;// Apresenta "--4" modo: segundos
        digito_1 = 4;
        mode = 4;
      } break;
      case 4 :
      {
        digito_1 = digito_2 = digito_3 = 0; // Apresenta "000"
        mode = 0;
      } break;
    }
  }
}else
if(tecla == 11)                 // Se for pressionado a tecla "#" enter:
{
  if(!ctl && lib)               // Se ainda não houve enter e já foi digitado tempo
  {
     tempo = concatena_digitos();
     tempo = tempo * resol;     // Inicia contagem do tempo
     ctl = 1;                   // Proibe 2 enters consecutivos
  }else if(ctl) return;         // Senão, se já houve um enter, retorna
  switch(mode)
  {                            // Se ainda não houve enter e está em escolha de modo
    case 1 :                    // Em modo 1 (minutos), tempo x 60s :
    {                           // liga no fim da contagem do tempo
      resol = min;
      t_on = 0;
      t_off = 0;
    } break;
    case 2 :                    // Em modo 2 tempo em segundos:
    {                           // liga no fim da contagem do tempo
      resol = seg;
      t_on = 0;
      t_off = 0;
    } break;
    case 3 :                    // Em modo 3 (minutos), tempo x 60s :
    {                           // desliga no fim da contagem do tempo
      resol = min;
      t_on = 0;
      t_off = 1;
    } break;
    case 4 :                    // Em modo 4 tempo em segundos:
    {                           // desliga no fim da contagem do tempo
      resol = seg;
      t_on = 0;
```

```
          t_off = 1;
        } break;
      }
    digito_1 = digito_2 = digito_3 = 0;
    }else
    if(mode)                      // Se já estiver em algum dos modos e
    {                             // for pressionado uma tecla entre 0 e 9:
      if(digito_3 > 0) return;    // Retorna se já foram digitados os 3 dígitos
      if(ctl) return;             // ou se já foi teclado enter
      digito_3 = digito_2;        // Faz o cursor mudar para próximo dígito
      digito_2 = digito_1;
      digito_1 = tecla;           // Atualiza último dígito teclado
      lib = 1;                    // Libera mas não inicia contagem do tempo
    }
}

// Função que concatena os 3 dígitos formando um único número
unsigned int concatena_digitos(void)
{
    unsigned int numero = 0;
    numero = digito_1;                    // Unidade
    numero = (numero + (digito_2 * 10));  // Dezena
    numero = (numero + (digito_3 * 100)); // Centena
    return (numero);
}

// Função que separa os dígitos do número de contagem
// Argumento de chamada: numero entre 0 e 999
void separa_digitos(unsigned int numero)
{
    unsigned int b = 0;
    if(numero < 1000)
    {
      for(b=99; b < numero; )           // Separa dígito da centena
      {
        numero = numero - 100;
        digito_3++;
      }
      for(b=9; b < numero; )            // Separa dígito da dezena
      {
        numero = numero - 10;
        digito_2++;
      }
      if(numero < 10) digito_1 = numero;// Separa dígito da unidade
    }
}


// Função que apresenta nos displays os dígitos
// Argumentos de chamada: valor do dígito, qual display
void mostra_disp(unsigned char dg, unsigned char disp)
{
    P1OUT = 0;
    switch(disp)                      // Define qual display ligar
    {
      case disp_1 : P2OUT = 4; break;
      case disp_2 : P2OUT = 2; break;
      case disp_3 : P2OUT = 1; break;
    }
    switch(dg)                        // Define qual dígito mostrar
```

```
    {
      case 0 :  P1OUT = numero[0]; break;
      case 1 :  P1OUT = numero[1]; break;
      case 2 :  P1OUT = numero[2]; break;
      case 3 :  P1OUT = numero[3]; break;
      case 4 :  P1OUT = numero[4]; break;
      case 5 :  P1OUT = numero[5]; break;
      case 6 :  P1OUT = numero[6]; break;
      case 7 :  P1OUT = numero[7]; break;
      case 8 :  P1OUT = numero[8]; break;
      case 9 :  P1OUT = numero[9]; break;
      case 10 : P1OUT = caractere_; break;
   }
}

void atraso(unsigned int temp)       // Função para delay
{
  unsigned int t = 0;
  for(t = 0; t < temp; t ++);
  for(t = 0; t < temp; t ++);
}
```