

X'Disciplina: Eletrônica Embarcada **Código:** 120871 **Turma:** A

Professor: Diogo Caetano Garcia

Aluno/Matrícula: Fábio Barbosa Pinto – 11/0116356

Questionário: 10_Dig_Out

1. Projete o hardware necessário para o MSP430 controlar um motor DC de 12V e 4A. Utilize transistores bipolares de junção (TBJ) com $V_{be} = 0,7\text{ V}$, $\beta = 100$ e $V_{ce}(\text{saturação}) = 0,2\text{ V}$. Além disso, considere que $V_{cc} = 3\text{ V}$ para o MSP430, e que este não pode fornecer mais do que 10 mA por porta digital.

Dados:

$$V_{be} = 0,7\text{ V}$$

$$\beta = 100$$

$$V_{CE} = 0,2\text{ V}$$

$$V_{cc} = 3\text{ V}$$

$$I_{\text{max,msp}} = 10\text{ mA}$$

A primeira coisa é obter um transistor que aguarde a corrente de coletor dada pela carga, que nesse caso é o motor DC. A corrente de coletor necessária será de 4 A. A corrente de base será nesse caso

$$I_b = I_c / 100 = 4 / 100 = 40\text{ mA}.$$

Como a corrente de base ultrapassa a corrente que a porta digital pode fornecer, então deve-se utilizar dois transistores, na forma do par Darlington. Então a corrente de base será

$$I_b = I_c / \beta^2 = 4 / 10000 = 0,4\text{ mA}$$

$$\text{O resistor } R_b = (V_{cc} - 2 \cdot V_{be}) / I_b = (3 - 2 \cdot 0,7) / 0,4 \cdot 10^{-3} = 4\text{ k}\Omega$$

2. Projete o hardware necessário para o MSP430 controlar um motor DC de 10V e 1A. Utilize transistores bipolares de junção (TBJ) com $V_{be} = 0,7\text{ V}$ e $\beta = 120$. Além disso, considere que $V_{cc} = 3,5\text{ V}$ para o MSP430, e que este não pode fornecer mais do que 10 mA por porta digital.

Dados:

$$V_{be} = 0,7\text{ V}$$

$$\beta = 120$$

$$V_{CE} = 0,2\text{ V}$$

$$V_{cc} = 3,5\text{ V}$$

$$I_{\text{max,msp}} = 10\text{ mA}$$

Cálculo de I_b :

$$I_c = \beta^2 * I_b;$$

$$1A = 14400 * I_b;$$

$$I_b = 69,44 \text{ uA}.$$

Cálculo de R_b :

$$R_b = (V_{cc} - 2 * V_{be}) / I_b;$$

$$R_b = (3,5 - 2 * 0,7) / 69,44 * 10^{-6};$$

$$R_b = 23,041K \Omega$$

3. Projete o hardware utilizado para controlar 6 LEDs utilizando charlieplexing. Apresente os pinos utilizados no MSP430 e os LEDs, nomeados L1-L6.

Cada pino controla 4 leds. $V_{leds} = 3 * 4 = 12V$ e a amperagem $I = 0,08A$ e $I_b = 10mA$,

$V_{cc} = 3V$, tem-se:

$$I_c = \beta * I_b;$$

$$0,08 = \beta * 10 * 10^{-3};$$

$$\beta = 8;$$

Cálculo de R_b :

$$R_b = (V_{cc} - 0,7) / I_b;$$

$$R_b = (3 - 0,7) / 10 * 10^{-3})$$

$$R_b = 230 \Omega, \text{ para cada pino.}$$

4. Defina a função void main(void){} para controlar 6 LEDs de uma árvore de natal usando o hardware da questão anterior. Acenda os LEDs de forma que um ser humano veja todos acesos ao mesmo tempo.

```
#include<msp430g2553.h>
#define CHPX1 BIT0
#define CHPX2 BIT1
#define CHPX3 BIT2
#define CHPXS (CHPX1 + CHPX2 + CHPX3)

void charlie_on (char CHPX_OUT, char CHPX_ON)
{
    P1OUT &=~CHPX_S;
    P1DIR &=~CHPX_S;
    P1DIR |= CHPX_OUT;
    P1OUT |= CHPX_ON;
}

int main(void)

WDTCTL = WDTPW|WDTHOLD;
while(1)
{
```

```
charlie_on(CHPX1 + CHPX2, CHPX1);
charlie_on(CHPX1 + CHPX2, CHPX2);
charlie_on(CHPX2 + CHPX3, CHPX2);
charlie_on(CHPX2 + CHPX3, CHPX3);
charlie_on(CHPX1 + CHPX3, CHPX1);
charlie_on(CHPX1 + CHPX3, CHPX3);
}
```

5. Defina a função void main(void){} para controlar 6 LEDs de uma árvore de natal usando o hardware da questão 3. Acenda os LEDs de forma que um ser humano veja os LEDs L1 e L2 acesos juntos por um tempo, depois os LEDs L3 e L4 juntos, e depois os LEDs L5 e L6 juntos.

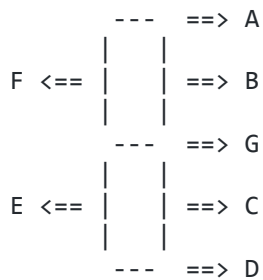
```
#include<msp430g2553.h>
#define CHPX1 BIT0
#define CHPX2 BIT1
#define CHPX3 BIT2
#define CHPXS (CHPX1 + CHPX2 + CHPX3)

void charlie_on (char CHPX_OUT, char CHPX_ON)
{
    P1OUT &=~CHPXS;
    P1DIR &=~CHPXS;
    P1DIR |= CHPX_OUT;
    P1OUT |= CHPX_ON;
}

int main(void)

WDTCTL = WDTPW|WDTHOLD;
while(1)
{
    charlie_on(CHPX1 + CHPX2, CHPX1);
    charlie_on(CHPX1 + CHPX2, CHPX2);
    charlie_on(CHPX2 + CHPX3, CHPX2);
    charlie_on(CHPX2 + CHPX3, CHPX3);
    charlie_on(CHPX1 + CHPX3, CHPX1);
    charlie_on(CHPX1 + CHPX3, CHPX3);
}
```

6. Defina a função void EscreveDigito(volatile char dig); que escreve um dos dígitos 0x0-0xF em um único display de 7 segmentos via porta P1, baseado na figura abaixo. Considere que em outra parte do código os pinos P1.0-P1.6 já foram configurados para corresponderem aos LEDs A-G, e que estes LEDs possuem resistores externos para limitar a corrente.



```
#include <msp430.h>
```

```
/*
 * main.c
 */
```

```
void EscreveDigito(volatile char dig)
```

```
{
#define LEDA BIT0
#define LEDB BIT1
#define LEDC BIT2
#define LEDD BIT3
#define LEDE BIT4
#define LEDF BIT5
#define LEDG BIT6

    if(dig=='0')
    {
        //Digito 0
        //LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 e LEDG = 0
        P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDE + LEDF;
    }
    else if(dig=='1')
    {
        //Digito 1
        //LEDA = 0, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 0, LEDF = 0 E LEDG = 0
        P1OUT |= LEDB + LEDC;
    }
    else if(dig=='2')
    {
        //Digito 2
        //LEDA = 1, LEDB = 1, LEDC = 0, LEDD = 1, LEDE = 1, LEDF = 0 E LEDG = 1
        P1OUT |= LEDA + LEDB + LEDE + LEDG;
    }
}
```

```
else if(dig=='3')
{
//Digito 3
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 0, LEDF = 0 E LEDG = 1
P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDG;
}
else if(dig=='4')
{
//Digito 4
//LEDA = 0, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 0, LEDF = 1 E LEDG = 1
P1OUT |= LEDB + LEDC + LEDG;
}
else if(dig=='5')
{
//LEDA = 0, LEDB = 0, LEDC = 0, LEDD = 0, LEDE = 0, LEDF = 0 E LEDG = 0
//Digito 5
//LEDA = 1, LEDB = 0, LEDC = 1, LEDD = 1, LEDE = 0, LEDF = 1 E LEDG = 1
P1OUT |= LEDA + LEDC + LEDD + LEDF + LEDG;
}
else if(dig=='6')
{
//LEDA = 1, LEDB = 0, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 E LEDG = 1
//Digito 6
P1OUT |= LEDA + LEDC + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='7')
{
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 0, LEDF = 0 E LEDG = 0
P1OUT |= LEDA + LEDC + LEDC;
}
else if(dig=='8')
{
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 E LEDG = 1
P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='9')
{
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 0, LEDF = 1 E LEDG = 1
P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDF + LEDG;
}
else if(dig=='A')
{
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 1, LEDF = 1 E LEDG = 1
P1OUT |= LEDA + LEDB + LEDC + LEDE + LEDF + LEDG;
}
else if(dig=='B')
{
//LEDA = 0, LEDB = 0, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 E LEDG = 1
P1OUT = LEDC + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='C')
{
//LEDA = 1, LEDB = 0, LEDC = 0, LEDD = 1, LEDE = 1, LEDF = 1 e LEDG = 0
```

```

P1OUT = LEDA + LEDD + LEDE + LEDF;
}
else if(dig=='D')
{//LEDA = 0, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 0 e LEDG = 1
P1OUT = LEDB + LEDC + LEDD + LEDE + LEDG;
}
else if(dig=='E')
{//LEDA = 1, LEDB = 0, LEDC = 0, LEDD = 1, LEDE = 1, LEDF = 1 e LEDG = 1
P1OUT = LEDA + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='F')
{//LEDA = 1, LEDB = 0, LEDC = 0, LEDD = 0, LEDE = 1, LEDF = 1 e LEDG = 1
P1OUT = LEDA + LEDE + LEDF + LEDG;
}
}
int main(void) {
WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer
P1DIR |= LEDA + LEDB + LEDC + LEDD + LEDE + LEDF + LEDG;
P1OUT = 0;
for(;;)
{
EscreveDigito('F');
}
return 0;
}

```

7. Multiplexe 2 displays de 7 segmentos para apresentar a seguinte sequência em loop: 00 - 11 - 22 - 33 - 44 - 55 - 66 - 77 - 88 - 99 - AA - BB - CC - DD - EE - FF

```

#include <msp430.h>
#define LEDA BIT0
#define LEDB BIT1
#define LEDC BIT2
#define LEDD BIT3
#define LEDE BIT4
#define LEDF BIT5
#define LEDG BIT6
#define CT1 BIT7 // CT1 - Catodo Comum do Display 1
#define CT2 BIT0 // CT2 - Catodo Comum do Display 2 - Vai no bit 0 da P2OUT
/*
* main.c
*/

void EscreveDigito(volatile char dig)
{

```

```
if(dig=='0')
{
//Digito 0
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 e LEDG = 0
P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDE + LEDF;
}
else if(dig=='1')
{
//Digito 1
//LEDA = 0, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 0, LEDF = 0 E LEDG = 0
P1OUT |= LEDB + LEDC;
}
else if(dig=='2')
{
//Digito 2
//LEDA = 1, LEDB = 1, LEDC = 0, LEDD = 1, LEDE = 1, LEDF = 0 E LEDG = 1
P1OUT |= LEDA + LEDB + LEDE + LEDG;
}
else if(dig=='3')
{
//Digito 3
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 0, LEDF = 0 E LEDG = 1
P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDG;
}
else if(dig=='4')
{
//Digito 4
//LEDA = 0, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 0, LEDF = 1 E LEDG = 1
P1OUT |= LEDB + LEDC + LEDG;
}
else if(dig=='5')
{
//LEDA = 0, LEDB = 0, LEDC = 0, LEDD = 0, LEDE = 0, LEDF = 0 E LEDG = 0
//Digito 5
//LEDA = 1, LEDB = 0, LEDC = 1, LEDD = 1, LEDE = 0, LEDF = 1 E LEDG = 1
P1OUT |= LEDA + LEDC + LEDD + LEDF + LEDG;
}
else if(dig=='6')
{
//LEDA = 1, LEDB = 0, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 E LEDG = 1
//Digito 6
P1OUT |= LEDA + LEDC + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='7')
{
//LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 0, LEDF = 0 E LEDG = 0
P1OUT |= LEDA + LEDC + LEDC;
}
}
```

```
else if(dig=='8')
{
  //LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 E LEDG = 1
  P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='9')
{
  //LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 0, LEDF = 1 E LEDG = 1
  P1OUT |= LEDA + LEDB + LEDC + LEDD + LEDF + LEDG;
}
else if(dig=='A')
{
  //LEDA = 1, LEDB = 1, LEDC = 1, LEDD = 0, LEDE = 1, LEDF = 1 E LEDG = 1
  P1OUT |= LEDA + LEDB + LEDC + LEDE + LEDF + LEDG;
}
else if(dig=='B')
{
  //LEDA = 0, LEDB = 0, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 1 E LEDG = 1
  P1OUT = LEDC + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='C')
{
  //LEDA = 1, LEDB = 0, LEDC = 0, LEDD = 1, LEDE = 1, LEDF = 1 e LEDG = 0
  P1OUT = LEDA + LEDD + LEDE + LEDF;
}
else if(dig=='D')
{
  //LEDA = 0, LEDB = 1, LEDC = 1, LEDD = 1, LEDE = 1, LEDF = 0 e LEDG = 1
  P1OUT = LEDB + LEDC + LEDD + LEDE + LEDG;
}
else if(dig=='E')
{
  //LEDA = 1, LEDB = 0, LEDC = 0, LEDD = 1, LEDE = 1, LEDF = 1 e LEDG = 1
  P1OUT = LEDA + LEDD + LEDE + LEDF + LEDG;
}
else if(dig=='F')
{
  //LEDA = 1, LEDB = 0, LEDC = 0, LEDD = 0, LEDE = 1, LEDF = 1 e LEDG = 1
  P1OUT = LEDA + LEDE + LEDF + LEDG;
}
}

int main(void) {
  WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer
  volatile int i;
  P1DIR |= LEDA + LEDB + LEDC + LEDD + LEDE + LEDF + LEDG + CT1;
  P2DIR |= CT2;
  P1OUT = 0;
  P2OUT = 0;
  for(;;)
  {
    //Aqui foi utilizado a Tabela ASCII para usar os dígitos
    // 0,1,2,3,4,5,6,7,8,9 estão definidos entre 0x30 e 0x39 na tabela
    // A,B,C,D,E,F estão definidos entre 0x41 e 0x46
```



```
for(i=0x30; i<=0x39; i++)
{
    EscreveDigito(i);
    P1OUT ^= CT1;
    P1OUT ^= CT1;
    P2OUT ^= CT2;
    P2OUT ^= CT2;
}
for(i=0x41; i<=0x46; i++)
{
    EscreveDigito(i);
    P1OUT ^= CT1;
    P1OUT ^= CT1;
    P2OUT ^= CT2;
    P2OUT ^= CT2;
}
}
return 0;
}
```