Coffee Pot System
Projeto Final - Microprocessadores e Microcontroladores

Afonso Delgado Soares de Souza
Engenharia Eletrônica
Universidade de Brasília
Matrícula: 12/0108151
Email: afonsodelgadoss@gmail.com

Lucas Raposo Souza Carvalho
Engenharia Eletrônica
Universidade de Brasília
Matrícula: 12/0126532
Email: lucas.raposo@apuamaunb.com

I. Objetivos

O projeto final visa aplicar os conhecimentos adquiridos durante o semestre percorrido na matéria de Microprocessadores e Microcontroladores, aplicando os mesmos em um protótipo de um produto.
O projeto sugerido pelos alunos foi o Coffee Pot System, um sistema para controlar uma cafeteira ou qualquer outro eletrodoméstico com o microcontrolador da Texas Instruments, o MSP430g2553, via comunicação UART.

II. Introdução

Recentemente, o tema Internet of Things (IoT) tem sido muito discutido e aplicado, uma vez que a tendência é estabelecer conexões facilitadas com o mundo. Com esse princípio, vem sendo cada vez mais procurado o mercado de empresas ou startups que fornecem soluções de conexões entre as pessoas e sua casa ou seus eletrônicos, o mercado de automação residêncial. Porém, um dos principais problemas da procura por esse mercado é a dificuldade de instalação de um sistema de automação e controle em uma residência e em todos os eletrodomésticos da casa.
A solução proposta é um módulo simples, que liga-se facilmente em eletrodomésticos simples e que permite alguns controles desses dispositivos.
O módulo é aplicado em uma cafeteira, para que seja possível ligá-la, desligá-la e além disso, ajustar um horário no qual deseja-se que a cafeteira seja ligada, dessa forma, estabelecendo controle via UART da cafeteira.

III. Protótipo Desenvolvido

O projeto proposto consiste em realizar o protótipo de um sistema de controle simples de uma cafeteira via conexão UART com o microcontrolador da Texas Instruments, o MSP430g2553. O sistema é constituído por uma software em C para o MSP430g2553 responsável pelo controle da cafeteira, um módulo de ativação para a cafeteira composto por um relé e transistores para que seja possível ativar a cafeteira com o sinal do MSP430g2553 e uma interface para seleção de rotinas e comandos.

A. *Código para MSP430g2553 para controle*

Este código é responsável por estabelecer conexão entre o usuário e o sistema e por realizar o controle da cafeteira.
Na função main() do código, foram configurados o Timer A, a conexão UART e o clock utilizado. As definições encontram-se listadas abaixo:

1) Clock do sistema: 1 MHz.

2) UART: Utilizar SMCLK, com Baud Rate de 9600 bps, com oversampling e interrupção por recebimento de byte habilitada.

3) Timer A: Utilizar SMCLK, em modo up-down, com divisor de frequência por 8 e interrupção habilitada a cada 1 segundo.

O fluxo do código encontra-se explicado no fluxograma simplificado, sem a contagem de tempo, abaixo:

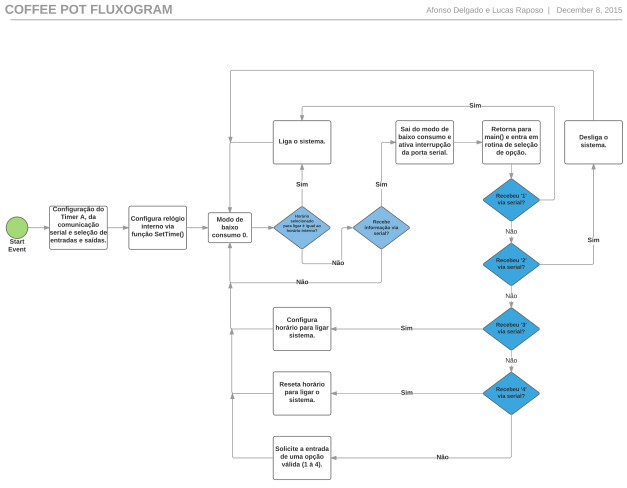

Figura 1. Fluxograma do sistema de controle.

O fluxograma completo, com a contagem de tempo do sistema, será anexado a este relatório, devido ao seu tamanho. Porém, o mesmo pode ser visualizado abaixo:

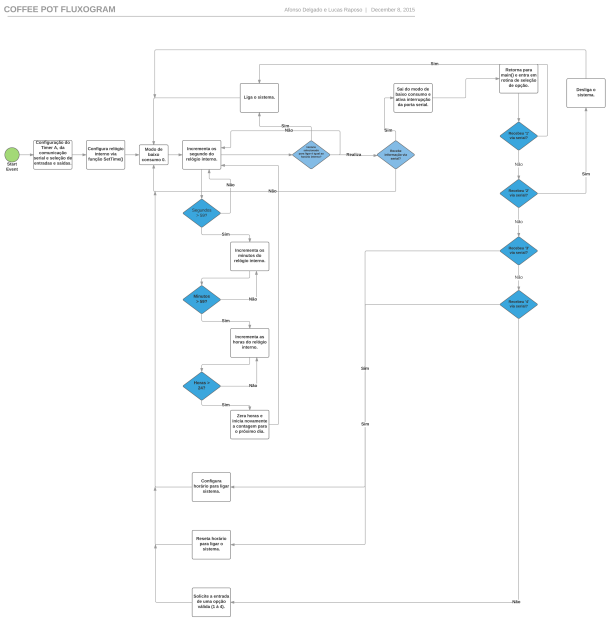

Figura 2. Fluxograma completo do sistema de controle.

B. Módulo de Ativação e Hardware Utilizado

O harware utilizado é compostor por: 1 MSP430g2553, 1 Relé de aticação 5 V e corrente de até 10 A, 1 transistor NPN BC547B para aumento da corrente de saída da porta P1.4 do MSP430g2553.
As saídas utilizadas foram:

1) P1.0 ligado ao LED1 interno da launchpad, para apresentar a dos segundos.

2) P1.6 ligado ao relé, para ativar o sistema de ligação da cafeteira.

3) USB ligado ao computador para recebimento de dados via UART.

O circuito acima é ligado de forma que no fio vermelho, entre a fase da rede elétrica no pino normalmente aberto do relé e, no pino comum (amarelo), seja ligado à fase da cafeteira. O neutro é ligado diretamente entre a rede elétrica e a cafeteira.
Com o acionamento da saída P1.4 do MSP430g2553, a bobina do relé é ativada, fechando a conexão normalmente aberta e possibilitando passagem de corrente pela fase da rede elétrica para a cafeteira, assim, ligando-a.
Abaixo, encontra-se um esquemático do circuito utilizado.

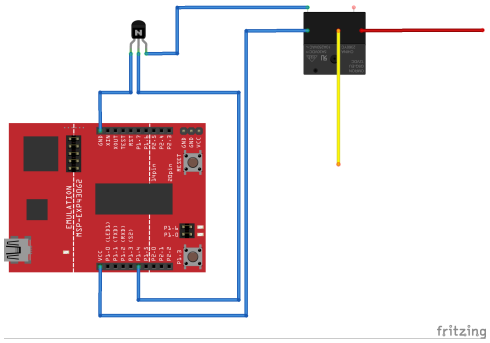

Figura 3. Circuito proposto.

C. Interface

A interface utilizada foi o Serial Monitor do software Energia. Com ele, é possível enviar e receber dados via porta serial para o MSP430g2553. Com a utilização deste meio, foi elaborado um menu de opções, apresentado abaixo.

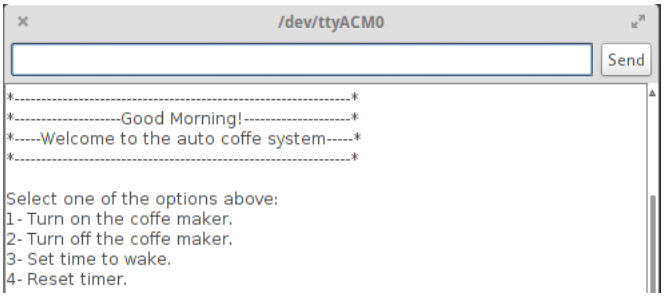

Figura 4. Menu de opções do sistema.

Cada opção é selecionada pelo número que a precede na apresentação do menu. As funcionalidades de cada opção encontram-se explicadas abaixo.

1) *Turn on the coffee maker* : Liga o sistema da cafeteira, iniciando o processo de coar o café. Também mantém a cafeteira ligada após coar, para manter o café aquecido. Caso a cafeteira já encontre-se ligada e o usuário escolha esta opção novamente será mostrada a seguinte mensagem para informar que a cafeteira encontr-ase ligada: *Coffee pot is alredy on! The coffee is heating!*

2) *Turn off the coffee maker*: Desliga o sistema da cafeteira, parando o processo de aquecimento após o café ter sido coado. Caso a cafeteira já encontre-se desligado e o usuário escolha esta opção novamente, a seguinte mensagem será mostrada: *Coffee pot is already off!*

3) *Set time to wake*: Configura um despertador no sistema para que a cafeteira seja ligado no horário selecionado. Ao selecionar essa opção, será inicializada uma nova rotina que para recepção do horário para despertar.

4) *Reset timer*: Reseta o timer configurado na função anterior. A seguinte mensagem será apresentada: *Timer has been reseted!*

Caso uma opção fora destas seja escolhida, será informado ao usuário para inserir uma opção válida com a mensagem: *Insert a valid option!*

IV. Resultados

O sistema proposto foi capaz de ligar e desligar a cafeteira e além disso, foi capaz de configurar um horário para que a mesma seja ligada. Com isso, o protótipo apresentou pleno funcionamento e será posteriormente trabalhado para obter-se um produto final e usual. Os conhecimentos obtidos na disciplina de Microprocessadores e Microcontroladores foram aplicados com êxito e eficiência, e com isso, foi possível realizar um protótipo funcional do produto proposto.

V. Anexos

A. Código proposto

Abaixo, segue o código em linguagem C para o sistema *Coffee Pot* para o MSP430g2553 proposto pelos estudantes:

```
#include <msp430g2553.h>
#include <legacymsp430.h>
#include <intrinsics.h>
#include <stdlib.h>

#define COFFEE BIT6
#define TX BIT2
#define RX BIT1
#define LED1 BIT0

volatile unsigned char status_coffe = -1, cont = 0;
volatile char hours = -1, minutes = -1, seconds = -1;
volatile char TA_hours = '7', TA_minutes = '0', TA_seconds = '0';

void setTime(volatile char hour, volatile char minute, volatile char second){
  TA_hours = hour;
  TA_minutes = minute;
  TA_seconds = second;
}

int print_string_serial(const char *str){
  int status = -1;

  if (str != NULL) {
    status = 0;
    while (*str != '\0') {
      while (!(IFG2 & UCA0TXIFG));
      UCA0TXBUF = *str;
      if (*str == '\n') {
        while (!(IFG2 & UCA0TXIFG));
        UCA0TXBUF = '\r';
      }
      str++;
    }
  }
  return status;
}

int getchar_serial(void){
  int chr = -1;
  if (IFG2 & UCA0RXIFG) {
    chr = UCA0RXBUF;
  }
  return chr;
}

void print_menu_serial(void){
  print_string_serial("\nSelect one of the options above:\n");
  print_string_serial("1- Turn on the coffe maker.\n");
  print_string_serial("2- Turn off the coffe maker.\n");
  print_string_serial("3- Set time to wake.\n");
  print_string_serial("4- Reset timer.\n");
}

void print_welcome_serial(void){
  print_string_serial("*-----------------------------------------*\n");
  print_string_serial("*---------------Good Morning!---------------\n");
  print_string_serial("*-----Welcome to the auto coffe system------*\n");
  print_string_serial("*-----------------------------------------*\n");
}

int main(){
  int cont_on = 0, cont_off = 0;
  WDTCTL = WDTPW + WDTHOLD;

  BCSCTL1 = CALBC1_1MHZ;
  DCOCTL = CALDCO_1MHZ;

  P1DIR |= BIT0;
  TA0CCR0 = 62500-1;
  TA0CTL = TASSEL_2 + ID_3 + MC_3 + TAIE;
  TACCTL0 |= CCIE;

  P1SEL2 = P1SEL = TX+RX;
  P1DIR = COFFEE + LED1;

  P1OUT = 0;

  UCA0CTL0 = 0;
  UCA0CTL1 = UCSSEL_2;
  UCA0BR0 = 6;
  UCA0BR1 = 0;

  UCA0MCTL = UCBRF_8 + UCOS16;

  IE2 |= UCA0RXIE;

  print_welcome_serial();
  print_menu_serial();


  _BIS_SR(LPM0_bits+GIE);

  while(1){
    print_menu_serial();

    _BIS_SR(LPM0_bits+GIE);
    switch(status_coffe){
     case '1':
         if(cont_on == 0){
           print_string_serial("\n\nMaking Coffee!\n\n");
          P1OUT |= COFFEE;
           cont_off = 0;
           cont_on = 1;
         }
         else{
           print_string_serial("\n\nCoffee pot is alredy on! The coffee is heating!\n\n");
         }
         break;

        case '2':
           if(cont_off == 0){
             print_string_serial("\n\nCoffee pot is off!\n\n");
             P1OUT &= 0;
             cont_on = 0;
             cont_off = 1;
           }
           else{
             print_string_serial("\n\nCoffee pot is already off!\n\n");
           }
        break;

         case '3':
           print_string_serial("\n\nInsert the time to wake: ");
                print_string_serial("\nInsira as horas: \n");
                while ((IFG2 & UCA0RXIFG) == 0);
                hours = getchar_serial();
                hours = status_coffe;
                UCA0TXBUF = hours;
                print_string_serial(" Horas\n");


                print_string_serial("Insira os minutos: \n");
                while ((IFG2 & UCA0RXIFG) == 0);
                minutes = getchar_serial();
                minutes = status_coffe;
                UCA0TXBUF = minutes;
                print_string_serial(" Minutos\n");

                print_string_serial("Insira os segundos: \n");
                while ((IFG2 & UCA0RXIFG) == 0);
                seconds = getchar_serial();
                seconds = status_coffe;
                UCA0TXBUF = seconds;
                print_string_serial(" Segundos\n");

           print_string_serial("\n\nTimer has been set!\n\n");
           break;

         case '4':
           hours = -1;
           minutes = -1;
           seconds = -1;
           print_string_serial("\n\nTimer has been reseted!\n\n");
           break;

        default:
        print_string_serial("\n\nInsert a valid option!\n\n");
         }
        }
    return 0;
}

interrupt(USCIAB0RX_VECTOR) Receiver(void){
    status_coffe = getchar_serial();

    LPM0_EXIT;
}



interrupt(TIMER0_A1_VECTOR) TA0_ISR(void){

  P1OUT ^= LED1;
```

```
  if(TA_seconds > 59){
    TA_seconds = 0;
    TA_minutes++;

    if(TA_minutes > 59){
      TA_minutes = 0;
      TA_hours++;

      if(TA_hours > 23){
        TA_hours = 0;
      }
      else;
    }
    else;
  }
  else{
    TA_seconds++;
  }
  if(hours == TA_hours && minutes == TA_minutes && seconds == TA_seconds){
    P1OUT |= COFFEE;
  }
  TA0CTL &= ~TAIFG;
}
```

B. Imagens do sistema.

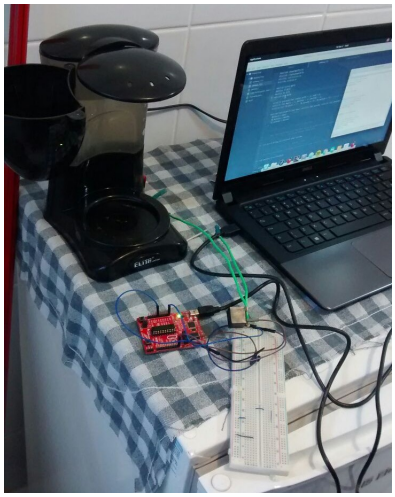

Figura 5. Primeiro teste realizado com o sistema.

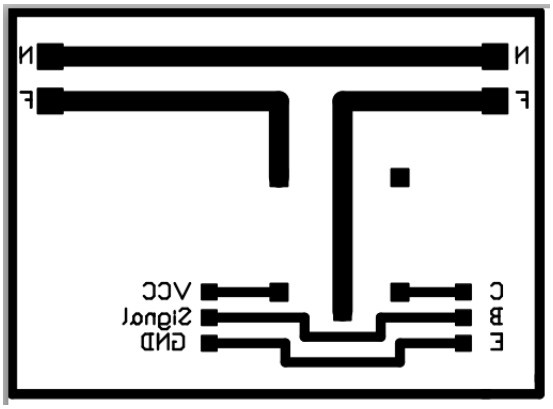

Figura 6. Placa de circuito impresso proposta.

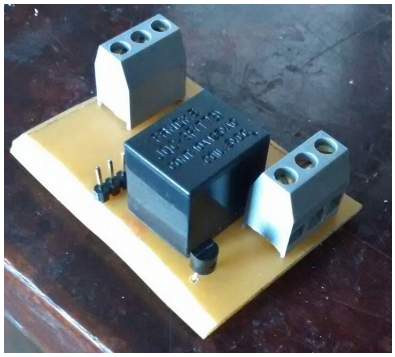

Figura 7. Circuito impresso do módulo de ativação.

REFERÊNCIAS

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] L. Damas, *Linguagem C*, 10rd ed. LTC.

[3] Texas Intruments, *MSP430x2xx Family User's Guide*, Rev. J.