Relatório Projeto Final

Sistema de controle de temperatura

Schneider
Engenharia Eletrônica
Universidade de Brasília
Taguatinga, Brasil
julio$_s chneider@hotmail.com$

Mateus Vasconcelos
Engenharia Eletrônica
Universidade de Brasília
Gama, Brasil
mateusvasconcelos.araujo@gmail.com

***Abstract*—Este relatório contém informações sobre o desenvolvimento de um sistema de controle de temperatura de um ambiente, que utiliza o microcontrolador MSP430 para realizar o controle da aferição e mudança da temperatura.**

***Index Terms*—sistema, MSP430, controle, temperatura**

I. Introduction

Este projeto tem como objetivo a criação de um sistema para controlar a temperatura em uma estufa, mantendo a temperatura desejada, utilizando o microcontrolador, sensor de temperatura e modificadores de temperatura. O projeto foi apresentado indicando as medições do sensor de temperatura, e que, ao se alterar o parâmetro de temperatura desejada, o atuador era ativado. Foi utilizado o microcontrolador MSP430G2553 para realizar o controle do sistema, com implementação de interrupção e modo “low power”.

II. Desenvolvimento

A. Desenvolvimento do Hardware

O funcionamento do hardware consiste na ativação do canal específico do relé de acordo com a temperatura medida dentro da ”estufa”, esse valor é comparado com um valor de setpoint, que pode ser alterado por meio de botões, para determinar qual canal do relé deve ser ativado. Essa operação é feita pelo controlador MSP430G2553. São utilizados dois atuadores, um para aumento, outro para diminuição da temperatura, foi utilizada uma lâmpada de 12V e 55W para o aquecimento. Para o resfriamento foi utilizado uma placa Peltier, para a utilização da Peltier é necessário resfriar o lado quente da placa para que tenha um melhor desempenho, para isso foi utilizado um dissipador de calor e uma ventoinha. É indicado na Figura 1 o diagrama de blocos do Hardware, e na Tabela 1 o Bill of Material.

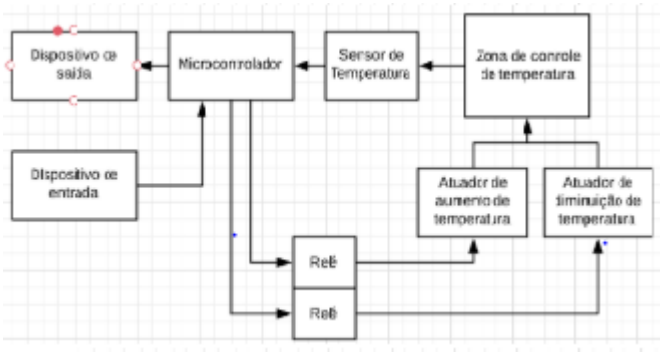

Fig. 1. Diagrama de Blocos em Hardware.

TABLE I
Bill of Material

Materiais	**Informações**		
	Descriação	***Preço Total***	***Quantidade***
MSP430G2553	Controlador	100	1
Peltier	Resfriador	20	1
Lampada 12v50W	Aquecedor	40	1
Jumpers	Conexões	10	40
Lm35	Sensor	10	1
Push Button	Entrada	0.5	2
Display Lcd 16x2	Saída	15	1
Módulo relé 2 canais	Multiplexador	15	1
Caixa de Isopor	Estufa	30	1
Ventoinha	Resfriador	10	1
Dissipador de Aluminio	Resfriador	10	1

[a]Preços aproximados em reais.

B. Desenvolvimento do Software

O software consiste em iniciar as entradas e saídas, o adc10, o lcd e o watchdog timer. Então o loop deve começar, no começo do loop é iniciado o ”low power mode 3”, o watchdogtimer cria uma interrupção que retira o controlador do modo ”low power”, então é necessário avaliar o estado dos botões, e de acordo com eles aumentar ou diminuir o setpoint de temperatura, após isso, é iniciada a conversão pelo adc10, novamente é iniado o ”low power mode 3”, porém desta vez o a interrupção do adc10 pela conclusão da conversão retira o controlador do modo ”low power”, o resultado da conversão é então processado para definir a temperatura. A temperatura obtida é comparada com o setpoint, o resultado dessa comparação liga o devido atuador, por último a temperatura e o setpoint são mostrados no display. O algoritmo do software é indicado por meio de um diagrama na figura 2. O algoritmo utiliza o modo de ”low power” para reduzir gastos desnecessários com o controlador. O Primeiro ingresso no modo é necessária para evitar leituras desnecessárias, criando espaçamentos entre cada conversão, visto que o controlador

consegue realizar leituras em um ritmo muito elevado, para isso se define o intervalo com o Watchdog timer que causa a interrupção para retirar o controlador do modo ”low power”, já o segundo se deve a necessidade de esperar o fim da conversão para realização do cálculo, assim o quando o Adc10 termina a conversão ele gera uma interrupção que retira o controlador do modo ”low power”.

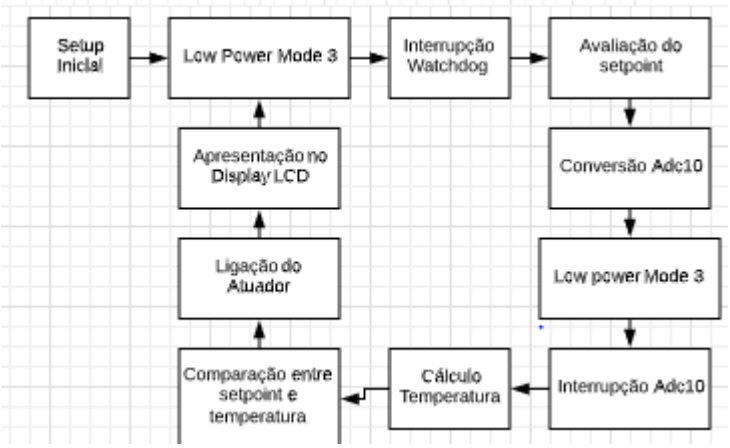

Fig. 2. Diagrama do algoritmo.

C. Descrição da Estrutura

Na contrução da estrutura foi utilizada uma caixa de isopor que atuaria como nossa estufa, foi optado por esse matérial deviado a ser um ótimo isolante térmico. Ela foi vedada com EVA para reforçar o isolamento e não deixar que o ar do ambiente externo entrasse na caixa.

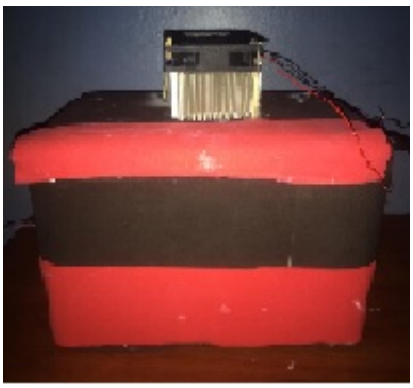

Fig. 3. Estrutura completa

O atuador de resfriamento foi colocado na parte central da tampa da caixa, devido aos conceitos da física, onde o ar frio tende a descer, fazendo com que o ambiente fosse sendo resfriado por completo. Junto da Peltier na parte de dentro da tampa, foi colocado um dissipador de calor pequeno e uma ventoinha que iria espalhar o ar frio com maior facilidade.

Na parte externa da tampa foi colocado um dissipador maior de calor para poder aumentar a melhoria do resfriamento

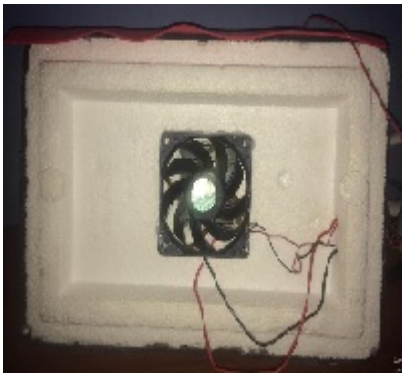

Fig. 4. Parte interna da tampa

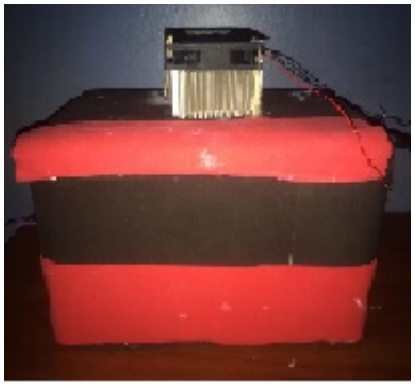

Fig. 5. Parte externa da tampa

interno, dissipando mais calor para fora da estufa, onde outra ventoinha foi colocada para resfriar este dissipador.

O atuador de aquecimento foi colocado na parte lateral inferior da caixa, como mencionado, o conceito físico diz que a ar quente tende a subir, fazendo com que o calor se propague mais uniformimente na caixa. Outro ponto é que para que o atuador não tivesse contato com o solo da estufa, para que não houvesse problema caso a temperatura variasse tanto que criasse uma camada liquida no fundo.

III. Resultados

A implementação em Software foi parcialmente funcional. No que tange o controle de temperatura estava operante, utilizando a plataforma de debug do software ”Code Composer Studio” foi possível verificar as aferições do sensor de temperatura e o comportamento correto dos atuadores e a mudança no setpoint de acordo com os botões utilizados. Foi encontrada dificuldade de implementação na parte de software que controlaria o display LCD, assim, o mesmo não foi implementado. A implementação do hardware foi completamente funcional, foi possível montar os circuitos e realizar os testes sem dificuldades. Mesmo sem o display LCD que não pode ser testado por meio da MSP430G2553, o mesmo foi testado com a utilização de um Arduino Mega 2560R3 e os resultados obtidos foram os esperados.

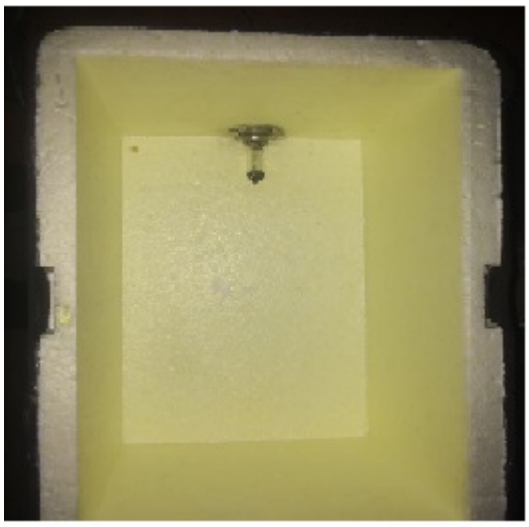

Fig. 6. Parte interna da caixa

IV. Conclusão

O projeto tinha o objetivo de criar um ambiente de temperatura controlada. Foi utilizado o microcontrolador MSP430G2553 com auxílio de um sensor e atuadores de temperatura para controlar a temperatura do ambiente. O objetivo central do projeto foi atingido visto que a aferição e modificação da temperatura através dos botões foi atingida. Porém ainda é necessário um desenvolvimento ainda maior, visto que não mostrar ao usuário os dados sobre a temperatura, o que dificulta na utilização do produto, pois não se sabe a temperatura atual da estufa.

References

[1] J. Davies, "MSP430 Microcontroller Basics" Elsevier Science. USA, pp. 582, 2008.

Appendix

A. *Código em C*

```
#include <stdio.h>
#include <msp430.h>
#include <intrinsics.h>
#include <stdint.h>
#include <string.h>
#include "lcd.h"

#define quente  BIT0
#define frio    BIT2
#define aumenta BIT1
#define diminui BIT5
#define sensor  BIT5
/**
 * main.c
 */
extern uint16_t calcular(uint16_t x);

int main(void)
{
    //lcdBegin();
    uint16_t temperatura, setpoint;
    //char temporario[10];
    P2OUT &= ~frio;
    P2OUT &= ~quente;
    P2DIR = quente | frio;
    WDTCTL = WDTPW | WDTTMSEL | WDTCNTCL |
        WDTSSEL;  //watchdog para interromper
        entre medicoes
    //WDTCTL = WDTPW | WDTHOLD;
    IE1 |= WDTIE;
    P1SEL |= BIT0;
    ADC10CTL0 = ADC10SHT_0 | ADC10IE | SREF_1;
    ADC10CTL1 = INCH_0 | SHS_0 | ADC10DIV_3 |
        ADC10SSEL_0 | CONSEQ_0;
    ADC10AE0 = BIT0;
    setpoint = 100;

    while(1){
        __low_power_mode_3();
        if((P2IN & aumenta) == 1){
            setpoint++;
        }
        if((P2IN & diminui) == 1){
            setpoint--;
        }
        ADC10CTL0 |= REFON + ADC10ON + ENC +
            ADC10SC;
        __low_power_mode_3();
        ADC10CTL0 &= ~(REFON | ADC10ON | ENC);
        temperatura = ADC10MEM;
        temperatura =  temperatura >> 2;
        calcular(temperatura);

        if (temperatura < setpoint+2){
            P2OUT &= ~frio;
            P2OUT |= quente;
        }
        else if (temperatura > setpoint-2){
            P2OUT &= ~quente;
            P2OUT |= frio;
        }
        else{
            P2OUT &= ~frio;
            P2OUT &= ~quente;
        }

        /*lcdSetCursor(1,0);
        lcdPrint("Temperatura:");
        itoa(temperatura, temporario);
        lcdPrint(itoa(temporario));
        lcdSetCursor(2,0);
        lcdPrint("SetPoint:");
        itoa(setpoint, temporario);
        lcdPrint(temporario);*/
    }
}

//-------------------------------------------
//Rotina de interrupcao do watchdog timer
#pragma vector = WDT_VECTOR
__interrupt void WDT_ISR ( void )
{
__low_power_mode_off_on_exit();
}
//Rotina de interrupcao do ADC10
#pragma vector = ADC10_VECTOR
__interrupt void ADC10_ISR ( void )
```

```
{
__low_power_mode_off_on_exit();
}
```

B. Código em Assembly

```
;-------------------------------------------
.cdecls C,LIST,"msp430.h"
;-------------------------------------------

.global calcular
.sect ".text"

calcular
        rra.w  R12
        rra.w  R12
        ret
```