Projeto Final
Irrigador Inteligente

Alice Fazzolino
Matrícula: 12/0108747
Universidade de Brasília
E-mail: afazzolino@gmail.com

Jackson Paz
Matrícula: 13/0028789
Universidade de Brasília
E-mail: jackson.paz@gmail.com

***Resumo*— Com a escassez de água cada vez maior, hoje não só o Distrito Federal mas outras regiões vem sofrendo com a crise hídrica, existe uma demanda cada vez maior por recursos hídricos. A irrigação do solo demanda uma quantidade substancial de água para a produção de alimentos, quando esta não ocorre de forma satisfatória e otimizada temos um alto desperdício de água, a irrigação realizada de forma automatizada possibilita o melhor uso deste recurso natural. A automatização do Sistema de irrigação envolve o controle sistêmico de toda uma plantação ou até mesmo de uma simples planta tornando este processo amplo e de difícil implementação. A proposta de trabalho vem com o uso de um microcontrolador, no intuito de acionar uma bomba d'água e assim efetuar a irrigação em um determinado tempo necessário. Neste processo de controle, utiliza-se a sinergia eletrônica da solenóide, sensor de umidade, bem como a MSP430 para o circuito programável.**

Palavras-Chaves—MSP430, irrigação

1. OBJETIVOS

Este projeto tem como principal objetivo desenvolver um sistema de irrigação automatizado, que consiga detectar o nível de umidade do solo e assim irrigar de forma racional determinadas plantas. Assim como outros objetivos:

1. Estudar e analisar as formas de irrigação;
2. Estudar sensores de detecção de umidade do solo;
3. Estudar o microcontrolador MSP430;
4. Testar e verificar o funcionamento do protótipo.

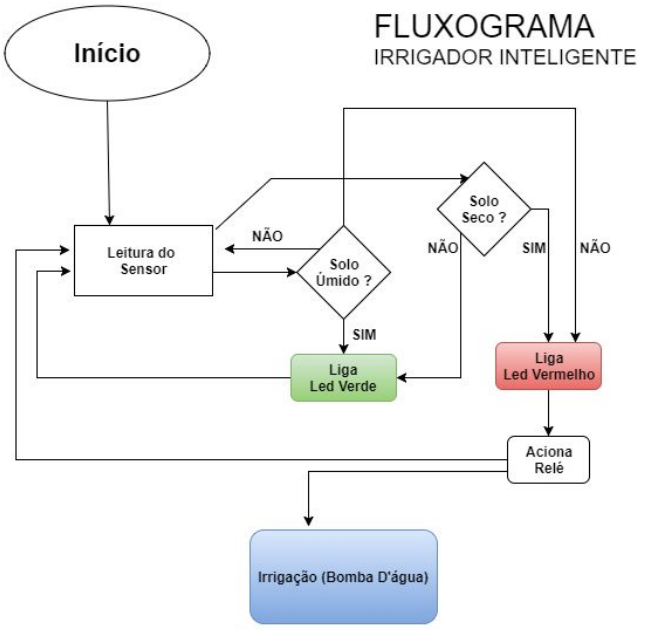
2. Introdução

A essência deste projeto é colaborar com usuário no processo de irrigação do solo quando o mesmo é necessário, realizando isso de modo automatizado e com exatidão, não permitindo que o solo fique extremamente seco e nem encharcado, sendo economicamente acessível, e ambientalmente sustentável.

É de suma importância salientar que cada tipo de cultura necessita de uma pesquisa detalhada, uma vez que o grau de umidade pode ser distinto para cada uma, como também pode haver uma variação conforme o tipo de solo e outros fatores importantes. Contudo, o projeto não entra nesses assuntos, visto que este é de caráter universitário de

um curso de engenharia eletrônica e se enfatiza no desempenho entre o microcontrolador e o sensor usando para isso o mesmo tipo de solo para o teste do circuito.

O sistema de irrigação apresentado não se adentra em temas ambientais, nem na seleção das diferentes técnicas de irrigação, como topografia, clima, solo, quantia de água e diferentes métodos de plantação. O foco é na parte da engenharia, que apresenta a comunicação desta com a agricultura. Para o projeto, não foi selecionado nenhum procedimento de irrigação, entretanto, o circuito têm a capacidade de utilizar qualquer método, já que o objetivo está na monitoração do solo e na comunicação entre o sensor e o microcontrolador , que determina o momento do acionamento da válvula, conforme as particularidades da plantação.



3. Desenvolvimento do protótipo

3.1 Descrição do sistema de irrigação sugerido

O sistema utiliza um sensor de umidade (higrômetro),onde este sistema possibilita o monitoramento da quantidade de água (umidade) presente no solo.

Este sensor higrômetro leva ao MSP430 um sinal que varia entre 0 a 5V, assim sendo capaz de especificar alcances de umidade relativa do solo e assim decretar se o solo está úmido ou não.

Quando o solo estiver seco,o sistema executará essa informação através de um led vermelho. Quando o solo estiver úmido acenderá um led verde, em seguida teremos a leitura do sensor novamente.

Quando o solo estiver seco,será acionado uma bomba d'água para efetuar a irrigação do local.

A bomba d'água será desligada quando a umidade se normalizar e só será acionada novamente se o solo indicar que está seco.

O acionamento da bomba d'água é feito por um relé, isolada da parte eletrônica.

3.2. Descrição do hardware

3.2.1 Materiais utilizados e Orçamento

Materiais	**Valor**
1 Protoboard	**20,00**
1 Relé	**12,00**
1 Bomba de aquário	**17,00**
1 Sensor de umidade	**12,00**
1 MSP430	**50,00**
1 Led vermelho	**0,20**
1 Led verde	**0,20**
1 Chave	**0,50**
330ohms Resistor	**0,15**
Total	**112,05**

3.2.2 Montagem do sistema de irrigação

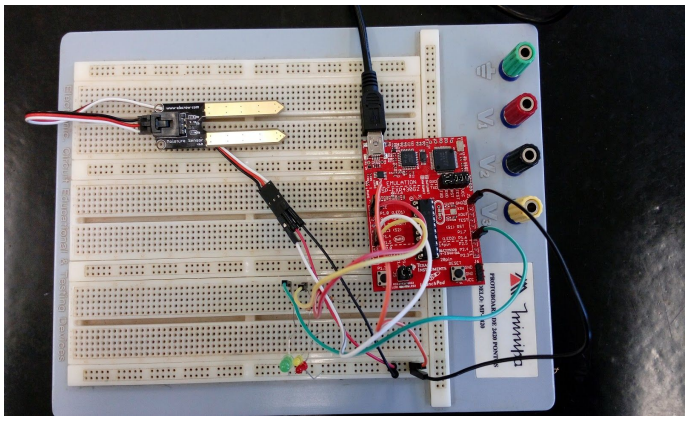


Figura 2: Ponto de controle 2: Teste do sensor de umidade

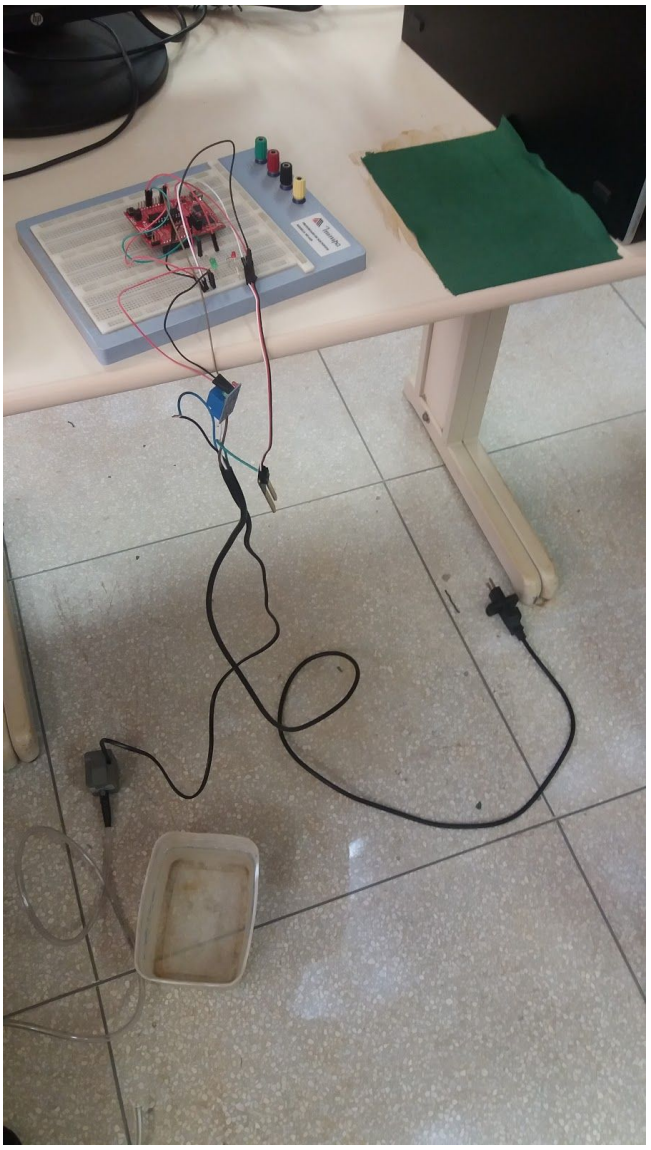


Figura 3: Ponto de controle 3 -Montagem do circuito

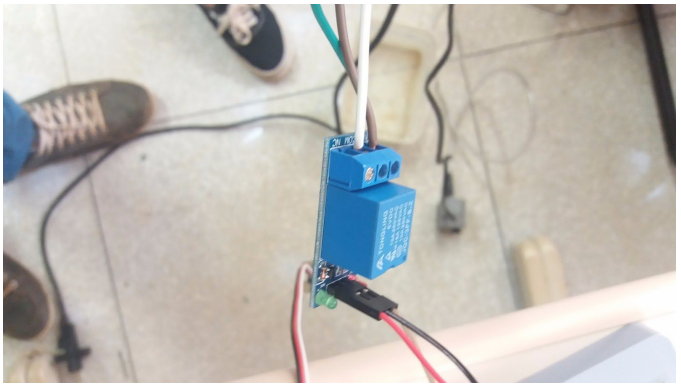


Figura 4: Ponto de controle 3 - Relé, utilizado para ativar a bomba

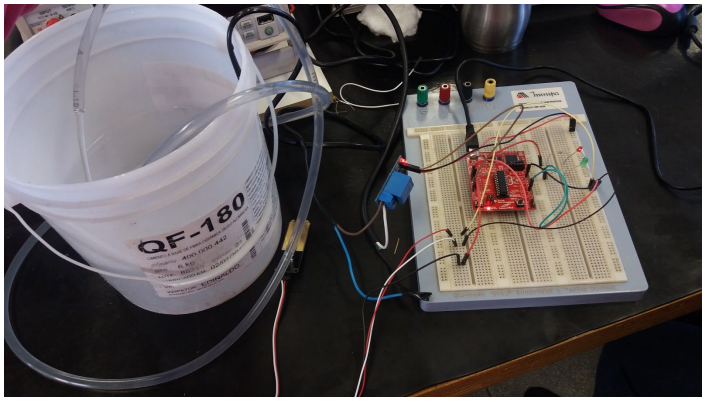


Figura 5: Ponto de controle 4- Solo seco, led vermelho aceso e irrigação acionada

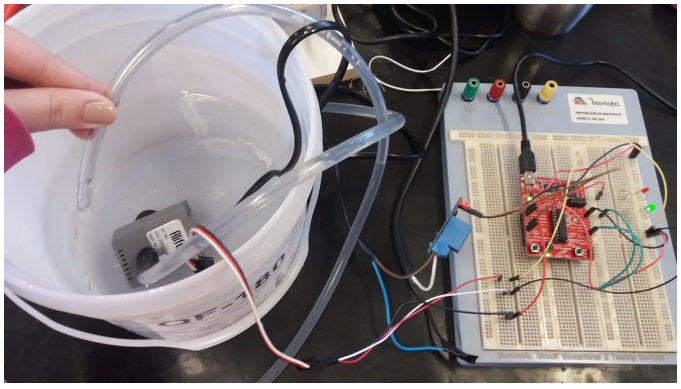


Figura 6: Ponto de controle 4- Solo úmido, led verde aceso e irrigação desativada

3.3. Desenvolvimento do software

Em todo o projeto foi utilizado o microcontrolador MSP430. Como plataforma de desenvolvimento foi usado o IAR Embedded Workbench , que é constituído de um software com linguagem de programação C/C++.

A leitura do sensor de umidade está presente dentro da estrutura while(1), essa estrutura tem a finalidade de efetuar laços repetitivos dentro do código, ou seja, a plantação será controlada todo o tempo. O código possui apenas duas bibliotecas, msp430g2553.h e intrinsics.h.

No loop que o MSP430 fica efetuando, é analisado se o nível baixo (solo seco) do sensor está acionado e, caso esteja, o led vermelho é aceso e o sistema envia um comando para acionar a bomba de água. Quando o nível é alto (solo úmido) , o led verde é aceso e o vermelho apagado, e logo em seguida, o sistema envia um comando para desativar a bomba, e assim parar de irrigar.

Para leitura do sensor foi usado no código o conversor A/D, pois precisávamos fazer a conversão de um valor de tensão analógica para um valor de tensão digital.

3.3.1 Descrição de Software

Para o respectivo projeto foi usado apenas duas bibliotecas. Também foi utilizado o recurso Define para definir o nome das variáveis utilizadas no desenvolvimento do código.

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

#define VALVULA BIT0
#define SENSOR BIT5
#define LEDRED BIT4
#define LEDGREEN BIT2
```

Figura 7: Bibliotecas e Recuso Define

Na Figura 8 temos a função convAdc responsável por configurar o conversor A/D do MSP430 . E também temos o timer para as varreduras do sensor.

```
void ConvAdc(void)
{
  //Config. Timer para o sensor (delay entre as leituras do sensor)
   TACCTL1 = OUTMOD_7;
   TACCR0 = 400-1;
   TACCR1 = TACCR0/2;
   TACTL = TASSEL_2 + ID_0 + MC_1;

   /* Configuração do canal do ADC */
   ADC10CTL1 = INCH_5 + ADC10DIV_3 ; // Canal 5, ADC10CLK/4
   ADC10CTL0 = SREF_0 + ADC10SHT_3 + ADC10ON + ADC10IE; //Vcc & Vss como referência
   ADC10AE0 |= SENSOR; //P1.5 bit pra leitura adc
}
```

Figura 8: Configuração do Conversor A/D e do Timer

Logo após a configuração do conversor A/D , iniciamos a parte principal do código. Aqui é desligado o WDT,

chamada a função convAdc e inciado um um loop infinito onde é ativado o conversor e o modo de baixo consumo do MSP430.

```
int main(void)
{
   WDTCTL = WDTPW + WDTHOLD;
   BCSCTL1 = CALBC1_1MHZ; // configuração do MCLK e SMCLK
   DCOCTL = CALDCO_1MHZ;
   BCSCTL2 &= ~(DIVS_3);

   P1DIR |= VALVULA + LEDRED + LEDGREEN;
   P1SEL |= SENSOR; //Entrada ADC pino P1.5
   P1OUT &= ~(VALVULA + LEDRED + LEDGREEN);

   ConvAdc();

   _BIS_SR(GIE); // habilita as interrupções

   while(1)
   {
      ADC10CTL0 |= ENC + ADC10SC; // Inicio da amostragem e conversão

      umidade = ADC10MEM; //valor que o conversor tá lendo
```

Figura 9: main

E abaixo, na figura 10, podemos visualizar a lógica do projeto.

```
      umidade = ADC10MEM; //valor que o conversor tá lendo

      if (umidade > 190) // solo úmido
      {
         P1OUT &= ~(VALVULA + LEDRED + LEDGREEN);
         P1OUT |= LEDGREEN + VALVULA;
      }
      else // solo seco
      {
         P1OUT &= ~(VALVULA + LEDRED + LEDGREEN);
         P1OUT |= LEDRED;
      }
   }
}
```

Figura 10: Lógica do projeto

E por último, na figura 10, temos a interrupção.

```
#pragma vector=ADC10_VECTOR
__interrupt void ADC10_ISR (void)
{
   __bic_SR_register_on_exit(CPUOFF); // Retorna ao modo ativo
}
```

4. Resultados e Protótipo Final

O projeto foi simulado no software fritzing conforme a figura 11, tendo o mesmo como maior objetivo, proporcionar o layout final do circuito em ambiente real após ser montado.

Os resultado após conclusão da etapa de montagem do circuito em questão se deu de forma satisfatória, tendo sua aplicabilidade na manutenção de plantas e como fator preponderante o ensino do curso de microcontroladores de uma maneira efetiva e prática.

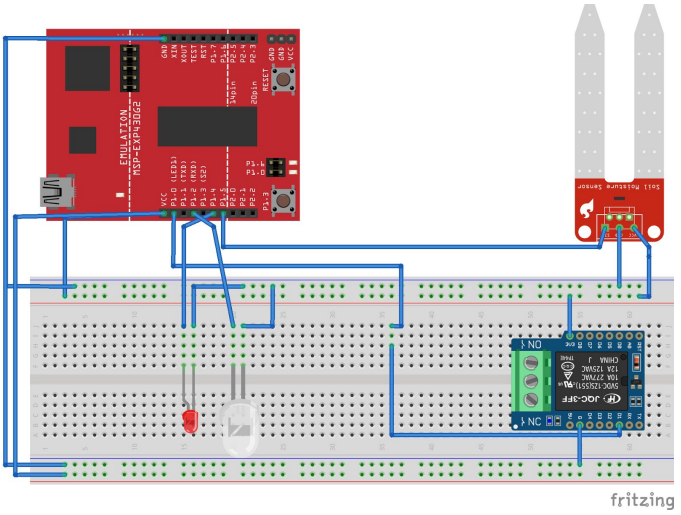


Figura 11: Simulação do projeto no Fritzing

5.Conclusão

O projeto executado foi de suma relevância para a dupla, pois foi utilizado basicamente toda a matéria passada no decorrer do semestre, que inclui a comunicação de circuitos externos ao MSP, conversão analógica-digital, e vários outros. E tudo foi aplicado em um sistema real, elevando mais o grau de aprendizado. Além disso, como o projeto

necessita de um sensor de umidade, houve um aprofundamento no estudo de conversão analógica-digital que foi de extrema necessidade para o projeto funcionar.

6. Revisão Bibliográfica

[1] SISTEMA AUTOMATIZADO PARA IRRIGAÇÃO DE ESTUFAS, MADALOSSO, EMANOELI.TCC UTFPR 2014.

[2] Davies, J., MSP430 Microcontroller Basics, Elsevier, 2008.

[3] Sistema de Monitorização da Humidade do Solo para Gestão Eficiente da Irrigação, de Brito Neves, Helder Filipe. Dissertação Engenharia Eletromecânica, Covilhã e UBI, Agosto de 2009.

- **APÊNDICE**

- CÓDIGO DO PROJETO

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

#define VALVULA BIT0
#define SENSOR BIT5
#define LEDRED BIT4
#define LEDGREEN BIT2

unsigned int umidade;

void ConvAdc(void)
{
   //Config. Timer para o sensor (delay entre as leituras do sensor)
  TACCTL1 = OUTMOD_7;
  TACCR0 = 400-1;
  TACCR1 = TACCR0/2;
  TACTL = TASSEL_2 + ID_0 + MC_1;

  /* Configuração do canal do ADC */
  ADC10CTL1 = INCH_5 + ADC10DIV_3 ; // Canal 5, ADC10CLK/4
ADC10CTL0 = SREF_0 + ADC10SHT_3 + ADC10ON + ADC10IE; //Vcc & Vss como referência
 ADC10AE0 |= SENSOR; //P1.5 bit pra leitura adc
}

int main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  BCSCTL1 = CALBC1_1MHZ; // configuração do MCLK e SMCLK
  DCOCTL = CALDCO_1MHZ;
  BCSCTL2 &= ~(DIVS_3);

 P1DIR |= VALVULA + LEDRED + LEDGREEN;
 P1SEL |= SENSOR; //Entrada ADC pino P1.5
P1OUT &= ~(VALVULA + LEDRED + LEDGREEN);

  ConvAdc();

  _BIS_SR(GIE); // habilita as interrupções

  while(1)
  {
     ADC10CTL0 |= ENC + ADC10SC; // Inicio da amostragem e conversão

          umidade = ADC10MEM; //valor que o conversor tá lendo

    if (umidade > 190) // solo úmido
    {
          P1OUT &= ~(VALVULA + LEDRED + LEDGREEN);
      P1OUT |= LEDGREEN + VALVULA;
    }
    else // solo seco
    {
          P1OUT &= ~(VALVULA + LEDRED + LEDGREEN);
```

```
        P1OUT |= LEDRED;
      }
    }
}

// ADC10 rotina
#pragma vector=ADC10_VECTOR
__interrupt void ADC10_ISR (void)
{
   __bic_SR_register_on_exit(CPUOFF); // Retorna
ao modo ativo
}
```