Sistema Embarcado para Aplicação em Monitoramento e Controle de Temperatura e Umidade

Vlademir de Jesus Silva Oliveira
Curso de Engenharia Elétrica, FACET
UNEMAT, Sinop, Brasil
vlademir.oliveira@unemat.br

Eduardo Rodrigo Thiesen
Curso de Engenharia Elétrica, FACET
UNEMAT, Sinop, Brasil
thiesen.edu@gmail.com

Douglas Aguiar do Nascimento
Faculdade de Engenharia Elétrica e de Computação
Unicamp, Campinas, Brasil
eng.douglas.a@ieee.org

***Resumo*— Neste trabalho é apresentada a implementação de um sistema embarcado de aquisição de dados (DAQ) para monitoramento e controle de temperatura e umidade em estufas por meio de placas de circuito impresso customizadas. O sistema possui duas saídas e dois atuadores controlados por relés. Uma saída controla a temperatura através de um ventilador exaustor da estufa, de acordo com o *set-point* pré-definido. A outra saída atua na irrigação, de acordo com eventos programados. No desenvolvimento do sistema utilizaram-se um microcontrolador de 8-bits, com comunicação entre periféricos via interface serial, e uma interface em ferramenta computacional, de modo que o usuário pudesse ajustar os parâmetros e registrar os dados no computador. Verificou-se que o projeto possuiu desempenho satisfatório demonstrando ser uma opção viável de baixo custo como alternativa aos modelos comerciais encontrados em mercado.**

Keywords—aquisição de dados; microcontrolador; LabVIEW; monitoramento de temperatura e umidade

I. INTRODUÇÃO

Placas de circuito impresso microcontroladas e computadores pessoais são utilizados de forma ampla na aquisição de dados (DAQ) e controle de variáveis [1]. Porém, uma placa ou sistema de aquisição de dados comercial tem um custo elevado para aplicações específicas. Por isso, o custo torna-se um aspecto importante no desenvolvimento de uma placa customizada utilizando microcontroladores – componentes eletrônicos de baixo custo com uma vasta gama de funcionalidades, programadas através de firmware – o que permite a redução de custo e melhoria de desempenho em aplicações diversas. Dessa forma, vários trabalhos apresentam aplicações de aquisição de dados, nas quais utilizam-se microcontrolador e Circuitos Integrados (CIs) de interface para comunicação com sensores e atuadores em substituição de placas de aquisição de dados [2], [3]. Entretanto, verificou-se uma literatura escassa quanto ao desenvolvimento de sistemas DAQ customizados e de baixo custo, utilizados essencialmente a pequenos consumidores e com aplicação voltada a controle de parâmetros ambientais. Portanto, no presente trabalho é investigado o uso de microcontrolador de 8-bits para aplicação no monitoramento e controle e de temperatura e de umidade em estufas para plantações.

II. DESCRIÇÃO DO PROJETO

A implementação de um sistema de aquisição de dados, ou sistema DAQ (*Data Acquisition*) depende do desempenho e características do hardware utilizado, podendo constituir-se de uma ampla variedade de blocos de hardwares de diferentes fabricantes [4]. Quando esses sistemas necessitam de controle, o hardware inclui dispositivo de processamento e atuadores para o controle de processos. Esses sistemas podem possuir: Conversores Analógico-Digitais, computadores, microcontroladores e CIs de interface. O sistema proposto é a aplicação de aquisição de dados de temperatura e umidade em uma estufa vegetal de teste, o qual se utilizou um hardware customizado e software para interface homem-máquina. O sistema permite o controle das variáveis aferidas pelo sensor e do armazenamento do registro da aquisição de dados. Na Fig. 1 mostram-se os principais componentes do sistema desenvolvido. Para se efetuar os testes e obter resultados, o sistema permite ajustar um valor desejado para que uma das saídas (Atuador 1) atue de acordo com o set-point de temperatura, também se programa eventos para que outra saída (Atuador 2) atue de acordo com o horário desejado. Os dados serão registrados em intervalos desejados e salvos em planilhas do Microsoft Office Excel® para serem analisados posteriormente.

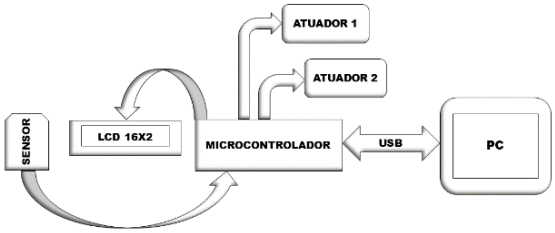

Figura 1. Diagrama funcional do sistema.

O sistema é composto por duas placas de circuito impresso desenvolvidas em fenolite e denominadas: 1. placa principal e; 2. placa auxiliar. No sistema tem-se o CI FT232R fixado no interior de uma caixa de montagem opaca de PVC, além de um mostrador LCD e um botão fixados na parte superior da caixa. O sensor de temperatura e umidade DHT22 é conectado à caixa, o qual é lido através de uma entrada digital do microcontrolador. Na tabela 1 são descritas as especificações técnicas do sensor e faixas de medição.

Tabela 1. Especificações técnicas do sensor DHT22 [5].

Power supply	3.3-6V DC
Output signal	digital signal via single-bus
Sensing element	Polymer capacitor
Operating range	humidity 0-100%RH; temperature -40~80Celsius
Accuracy	humidity +-2%RH(Max +-5%RH); temperature <+-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH; temperature 0.1Celsius
Repeatability	humidity +-1%RH; temperature +-0.2Celsius
Humidity hysteresis	+-0.3%RH
Long-term Stability	+-0.5%RH/year
Sensing period	Average: 2s

A calibração do sensor não é necessária, sendo já calibrado e compensado pelo fabricante, sua comunicação com o microcontrolador pode ser realizada em longas distâncias sendo de até 100 metros. O sensor é um módulo que possui internamente um sensor de umidade capacitivo, um termistor e um conversor A/D para se comunicar com o microcontrolador.

Para montagem das placas utiliza-se: Soquetes para entradas analógicas; soquetes para pinos digitais que podem ser configurados como entrada ou saída; soquete para comunicação UART entre o microcontrolador e a placa com CI FT232R; soquetes para alimentação da placa; transistores e relés para acionar cargas a partir das saídas digitais. O diagrama elétrico e detalhes construtivos estão apresentados na figura 2, com destaque ao microcontrolador (Microchip PIC 16F877A) e ao mostrador (LCD 16x2). Na ilustração pode ser observado o detalhe da tampa colocada na parte superior da caixa, no qual está fixado um mostrador LCD e um botão tipo *push button*. O botão é usado para alternar entre os dados de temperatura e umidade e o horário do relógio configurado pelo usuário. Para que o botão possa ser lido em tempo instantâneo foi necessário gerar uma interrupção externa através do pino RB0 do microcontrolador.

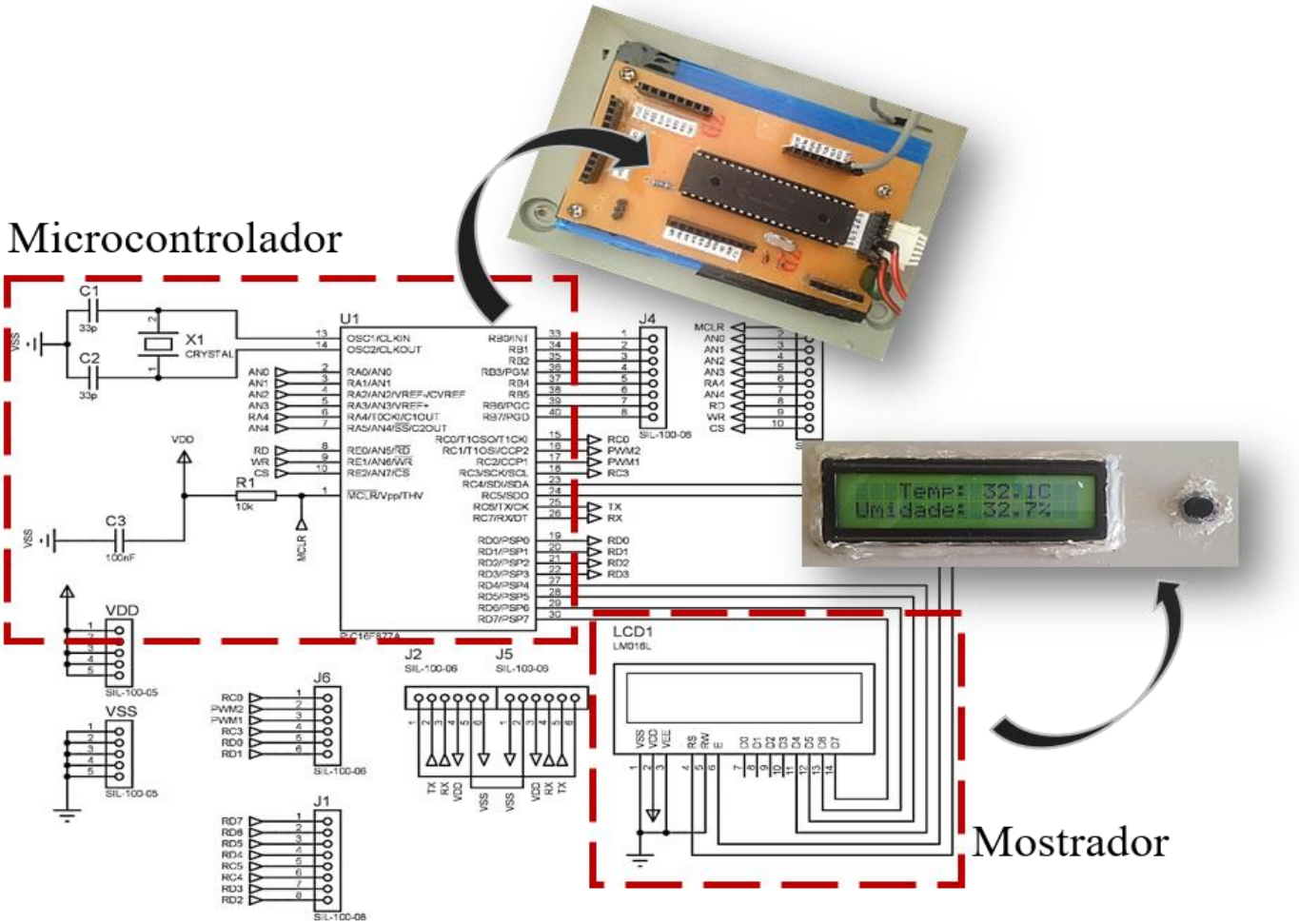

Figura 2. Ilustração da placa principal.

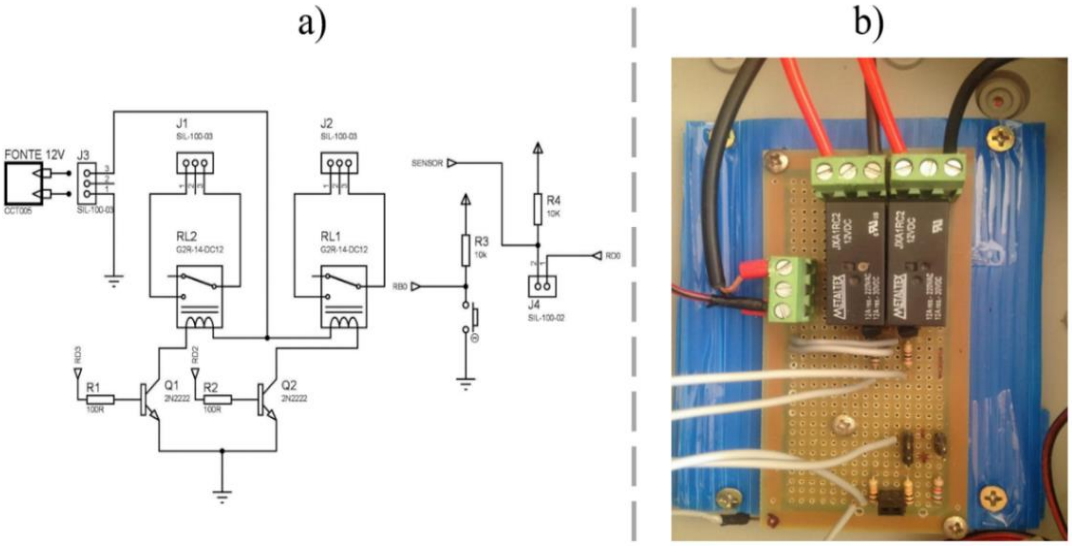

Figura 3. Detalhamento da placa auxiliar. a) esquemático elétrico. b) componentes montados na placa.

Para possibilitar o acionamento dos atuadores, projetou-se uma placa auxiliar para comutação de cargas com corrente elevada. A placa possui dois relés miniatura com um contato reversível com capacidade de corrente de comutação nominal de 12 A em corrente continua com tensão de até 30 V. Na Figura 3 observa-se o diagrama esquemático elétrico (Figura 3a) e os componentes montados sobre a placa (Figura 3b), dadas pelas conexões: a placa com CI FT232R está conectada nos soquetes J2 e J5 da placa principal. O CI FT232R, fabricado pela empresa FTDI (*Future Technology Devices International*) emula a comunicação serial *Universal Asynchronous Receiver/Transmitter* (UART), sendo uma boa opção, pois possui drivers atualizados e empresta seu *vendor ID* para o usuário [6].

III. PROJETO DO CÓDIGO

Empregou-se o compilador *mikroC PRO for PIC* da empresa MikroElektronika para o desenvolvimento do *firmware* inserido no microcontrolador. Com isso, foi possível utilizar a linguagem C padrão e obter muitas bibliotecas, além de configurações através de registradores. O fluxograma da lógica principal está ilustrado na Figura 4.

Na Figura 5 são apresentadas as configurações de interrupções. Na linha 186 a atribuição habilita a interrupção externa, a qual possibilita o botão interromper o laço infinito do código. Para implementar o relógio no microcontrolador foi necessário utilizar o *timer1*. Nas linhas 188 a 190 do código inicializa-se a contagem do timer1 para um intervalo de 100ms, através dos registradores TMR1H e TMR1L.

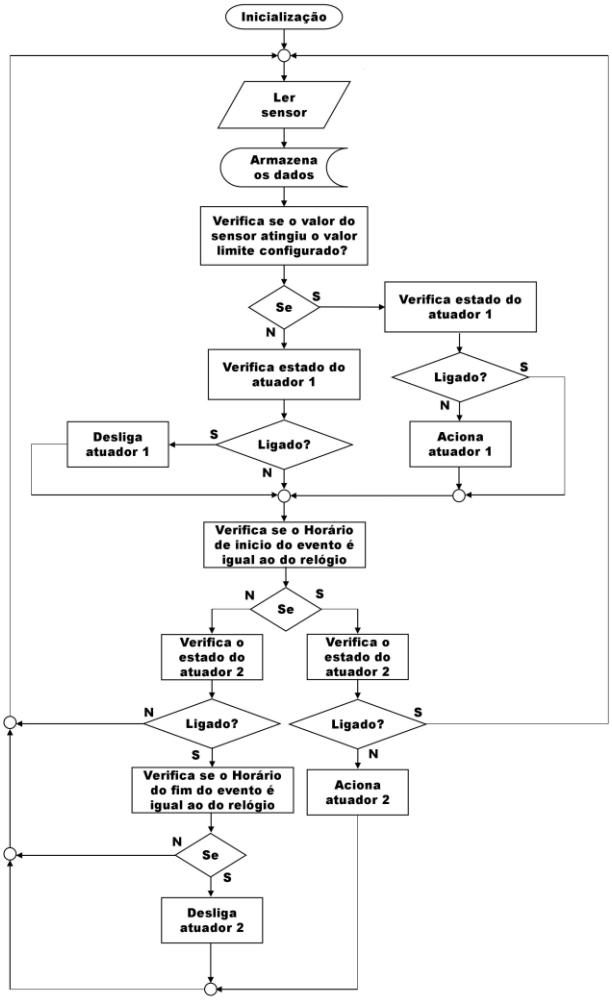

Figura 4. Fluxograma de lógica da programação.

```
GIE bit    = 0x01;    //Habilita Interrupção global
PEIE bit   = 0x01;    //Habilita Interrupção por periférico
INTE bit   = 0x01;    //Habilita a interrupção externa
INTEDG bit = 0x0C;    //Configura a interrupção borda de descida
T1CON      = 0x31;    //Configura Prescaler 1:8 TMR1, TMR1 on
TMR1L      = 0xDC;    //Inicializa o TMR1 em 3036 para
TMR1H      = 0x0E;              // uma contagem de 62500

TMR1IE_bit = 0x01;    //Habilita a interrupção do TMR1
```

Figura 5. Trecho do código com configuração das *interrupções*.

Na Fig. 6 apresenta-se o trecho do código comentado com a rotina de interrupção. Para incrementar um contador de acordo com o estouro do *timer1*, os segundos são incrementados a cada 10 estouros do *timer1*, ou seja, a cada 100ms uma variável de contagem é incrementada, quando contar 10 alcança-se 1 segundo e a variável de contagem é zerada. O botão da Fig. 7 conectado ao pino RB0/INT aciona a interrupção externa. O bit INTE_bit serve para habilitar a interrupção e o bit INTF_bit serve como *flag* de sinalização da interrupção.

```
// --- Rotina de Interrupção ---
void interrupt()
{
     if(INTF bit)
    {                          //Sim..
        Botao = ~Botac;        //Inverte o estado da variável
        INTF_bit = 0x0C;       //Limpa a flag
    }
    if(TMR1IF bit)             //Houve overflow do Timer1?
    {                          //Sim...
       TMR1IF bit = 0x00;      //Limpa a flag
       TMR1L = 0xDC;           //Reinicia o TMR1 em 3036
       TMR1H = 0x0B;
       clk cont++;             //Incrementa variável de controle
       if(clk_cont == 0x0A)    //clk_cont igual a 10?
       {
          clk_cont = 0x00;     //Reinicia clk_cont

          segundos++;          //Incrementa segundos

       } //end if clk_cont
    } //end if Timer1

} //end interrupt
```

Figura 6. Rotina de interrupção

O sensor DHT22 possui protocolo proprietário *one-wire*, o qual pode ser usado para comunicação com o microcontrolador, através de um pino digital. As funções para comunicação com o sensor foram desenvolvidas seguindo os passos descritos em [5]. Como ilustrado na Fig. 7, quando o microcontrolador envia o sinal de início (*StartSignal*) o sensor comuta do modo de espera para o modo de execução. A função *StartSignal* enviará um sinal de resposta de dados para que o sensor retorne o valor de umidade e temperatura. Para garantir o retorno dos dados, a função (*CheckResponse*) é necessária para checar o recebimento dos dados.

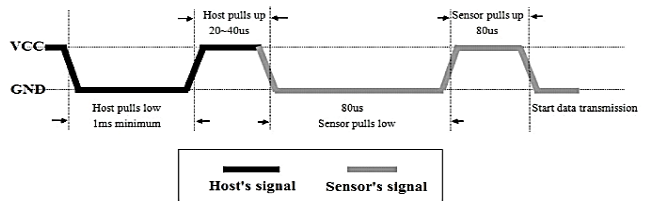

Figura 7. Diagrama de tempo da inicialização da comunicação.

Na Figura 8 mostra-se as funções desenvolvidas em linguagem C [7], onde o bit s1 representa RD0_bit. Para iniciar a comunicação entre o MCU (Microcontrolador) e o sensor, a função *StartSignal* é executada, então o pino s1 é configurado como saída e passa para nível baixo, este estado deve ultrapassar 10ms para garantir que o sensor possa detectar o sinal do MCU, nesse caso foi usado um atraso de 18ms, então s1 passa para nível alto e deve esperar em torno de 20 a 40us para que o sensor responda, onde foi usado 30us e no fim do processo o pino s1 é configurado como entrada. Se o sensor passar para o nível lógico baixo por um tempo de 80us significa que está detectando a função *StartSignal*, então ocorre uma nova transição de até 80us para o sensor enviar os dados lidos. Na função *CheckResponse* a *flag check* é zerada e após um atraso de 40us verifica-se s1, caso esteja em nível baixo aguarda mais 80us, então se s1 estiver em nível alto, a *flag* é marcada e um novo atraso de 40us gerado.

```
void StartSignal()
{
TRISD.F0 = 0;
s1 = 0;
delay ms(18);
s1 = 1;
delay us(30);
TRISD.F0 = 1;
 }
void CheckResponse()
{
    Check = 0;
    delay us(40);
    if (s1 == 0){
    delay us(80);
    if (s1 == 1)
     Check = 1;   delay_us(40);}
}
```

Figura 8. Funções de inicialização para comunicação com sensor.

Quando o sensor está enviando dados para o MCU, a transmissão de cada *bit* começa em nível baixo, que ultrapassa os 50us e dependendo do comprimento do sinal de nível alto, o *bit* é 0 ou 1. Se esse comprimento estiver na faixa entre 26 e 28us, o valor é 0 para este *bit*, caso o comprimento seja na faixa de 70us, o valor é 1 para o bit. A função *ReadData* aguarda até a entrada do MCU ficar em nível baixo, após um atraso de 30us é verificado se continua em nível baixo, se verdadeiro o MCU irá receber o valor 0 para esse *bit*, caso contrário o MCU receberá o valor 1. Na função foram criadas duas variáveis "i" e "j", onde a variável "i" retorna o valor do *byte* transmitido para o MCU e a variável "j' controla a posição dos *bits*. Por fim foi criado as variáveis que armazenam os *bits* recebidos sendo necessário dois *bytes* para cada dado e um *byte* para a soma, que serve de paridade. Nas Figuras 9 e 10 visualizamos a função *ReadData* e o trecho do código para armazenar os dados do sensor nas 5 variáveis criadas.

```
char ReadData()
{
char i, j;
for(j = 0; j < 8; j++)
{
while(!s1);
delay us(30);
if(s1 == 0)   i&= ~(1<<(7 - j));
else {i|= (1 << (7 - j));
while(s1);}
}
return i;
}
```

Figura 9. Funções para leitura de 1 byte na variável i.

```
if(Check == 1){
RH byte1 = ReadData();
RH byte2 = ReadData();
T byte1 = ReadData();
T byte2 = ReadData();
Sum = ReadData();
if(Sum == ((RH_byte1+RH_byte2+
      T_byte1+T_byte2) & 0XFF)){
Temp = T byte1;
Temp = (Temp << 8) | T_byte2;
RH = RH byte1;
RH = (RH << 8) | RH_byte2;
```

Figura 10. Trecho do código que armazena os bytes lidos do sensor.

Pode-se resumir a programação através das variáveis adotadas e configurações. Nesse código foram usadas várias variáveis tipo char, como “text” e seus componentes: “horas”, “minutos” e “segundos”, a variável “str” que recebe os caracteres lidos do LabVIEW® através do registrador RCREG, o vetor “texto[32]”, para enviar os dados e status das variáveis para o LabVIEW® e unsigned char, como RH_byte1, RH_byte2, T_byte1, T_byte2 e Sum, para ler os dados do sensor, os contadores: Evh1, Evh2, Evh3, Evh4, Evm1, Evm2, Evm3, Evm4, Tmax, que armazenam as configurações dos botões do LabVIEW® de horário dos eventos e temperatura máxima. Por fim, as variáveis tipo bit ‘b_min’, ‘b_hor’, para incrementar “horas” e “minutos”, ‘botao’ que aciona a escrita da temperatura/umidade no LCD e ‘Evento1’ e ‘Evento2’ que marcam o acionamento dos eventos.

A comunicação é configurada através dos registradores, seguindo o *datasheet* [8]. No modo assíncrono SPBRG=0x2A configura 28800 *baud rate*. No registrador TXSTA, TXEN_bit=1 habilita a transmissão, BRGH_bit=1 seleciona baud rate em velocidade alta, SYNC_bit=0 configura para modo assíncrono. No registrador RCSTA, SPEN_bit=1 habilita a porta serial (nos pinos RC7/RX/DT e RC6/TX/CK), CREN_bit=1 habilita recepção contínua. Após receber os dados pela USB utiliza-se o comando while(!TRMT_bit) para aguardar o buffer esvaziar.

IV. RESULTADOS

Na Figura 11 pode-se observar como os componentes do sistema foram montados em uma caixa de montagem opaca de PVC e suas conexões externas para o sensor: LCD, atuadores e computador. O sistema foi testado em uma mini estufa contendo o sensor de temperatura e umidade, bomba de irrigação e exaustor.

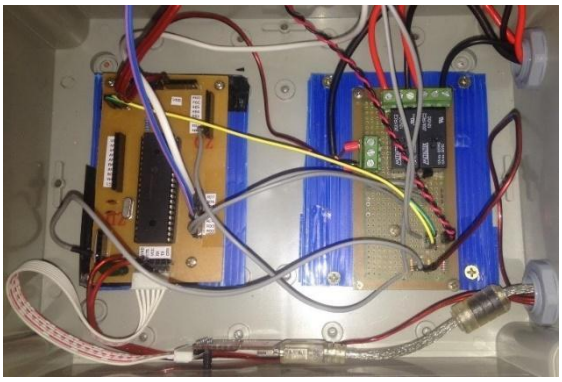

Figura 11. Detalhe da instalação das placas do sistema de aquisição de dados.

O sistema é controlado através de uma aplicação feita em LabVIEW®, a qual permite programar eventos e armazenar os dados no computador manipulando-se os comandos virtuais na tela do computador. Na Fig. 12 é apresentado o painel elaborado na ferramenta computacional.

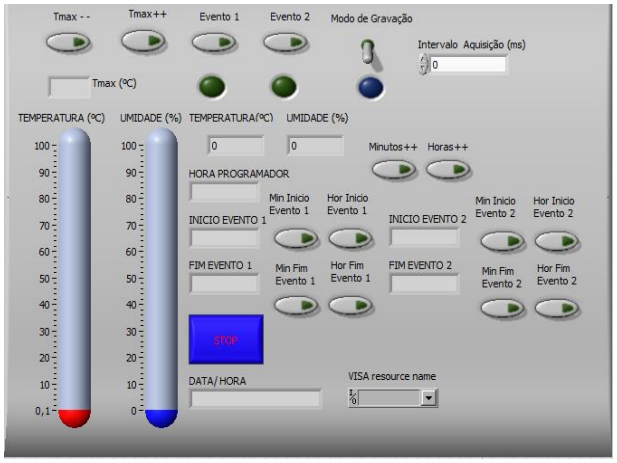

Figura 12. Painel frontal elaborado em LabVIEW®.

O valor das variáveis da umidade e temperatura são apresentados em duas barras de progresso na escala de 0 a 100, como também seus dígitos numéricos com uma casa decimal após a vírgula. Os valores são atualizados a cada um segundo no painel, exceto quando ligado o modo de gravação para armazenamento dos dados em intervalos maiores que um segundo. As funções “Evento 1”, “Evento 2” e “Modo de Gravação” possuem um indicador luminoso para sinalização no painel. Para realizar a aquisição de dados de temperatura e umidade é necessário ativar o modo de gravação, assim a aplicação registra os dados em um arquivo do Excel®, em intervalos definidos pelo valor de “intervalo de gravação”. O usuário escolhe o intervalo de registro dos dados e inicia a gravação, comutando o botão “Modo de Gravação”. Quando interrompido a gravação um arquivo é salvo. Após retomada a gravação um novo arquivo pode ser gerado. A comunicação usa o conversor serial/USB FT232R, sendo necessário abrir o aplicativo e selecionar a porta COM disponível para efetuar a conexão, sempre que o cabo de comunicação era conectado na entrada USB.

Os botões denominados “Evento 1/ 2” são usados para ativar eventos diários quando desejado, mas é necessário que o botão esteja ativo para ocorrer o evento horário que foi configurado. Só foi utilizado o atuador da bomba para essa aplicação. Assim, foi possível monitorar/registrar as alterações de temperatura e umidade durante os eventos de irrigação. Os dados adquiridos foram registrados em planilha do MS Excel®, e um exemplo pode ser visualizado na Tabela 2. Nesse exemplo, o intervalo de aquisição foi de aproximadamente um segundo, no entanto foi realizado testes com intervalos de até 3 minutos, não havendo erros. Com intervalo de aquisição próximo a 1 segundo, notou-se que alguns dados podem ter sido corrompidos. Uma solução para isso é usar filtros no processamento dos dados da planilha.

Tabela 2. Aplicação em Funcionamento durante teste.

	A	B	C
1	Time	Temp	Umid
2	15:57:02	31,8	32
3	15:57:04	31,9	32,2
4	15:57:05	31,8	32
5	15:57:06	31,9	32
6	15:57:07	31,9	31,9
7	15:57:08	31,9	31,8
8	15:57:09	31,8	31,7
9	15:57:10	31,8	31,7
10	15:57:11	31,8	31,6
11	15:57:12	31,9	31,6
12	15:57:13	31,9	31,5

Na Figura 13 pode-se visualizar o painel frontal da aplicação durante um teste.

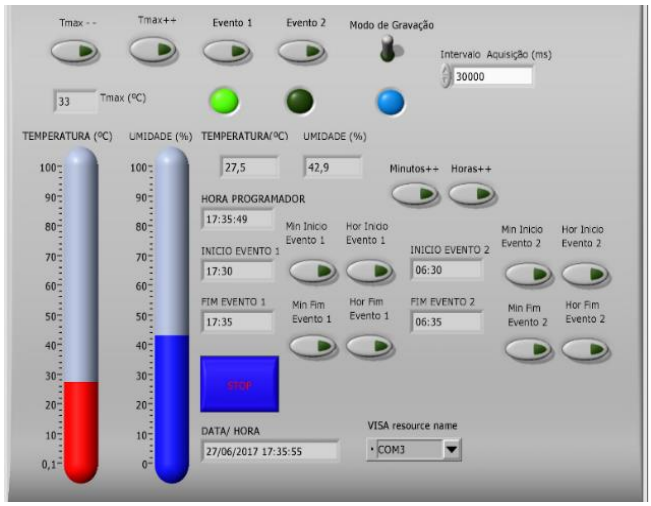

Figura 13. Aplicação em Funcionamento durante teste.

Nesse teste a irrigação foi programada no evento 1, iniciando-se 17:30 horas com duração de 5 minutos. O evento 2, de acordo com a figura está desabilitado, ou seja, a saída atuaria apenas uma vez ao dia de acordo com o evento 1. O controle de temperatura não foi testado e o *set-point* foi configurado para que não atuasse. Os dados de temperatura e umidade foram adquiridos em intervalos de 30 segundos. Na Tabela 1 visualiza-se os dados coletados durante a irrigação com seu respectivo intervalo. No entanto com o intervalo de aquisição escolhido de 30 segundos o sistema ficou instalado por horas não havendo erros nos dados.

Tabela 3. Dados adquiridos durante irrigação.

Horário	Temperatura	Umidade
17:30:26	27,7	42,9
17:30:56	27,7	43
17:31:26	27,6	44,9
17:31:56	27,6	44,6
17:32:26	27,5	49,9
17:32:56	27,4	49,6
17:33:26	27,3	49,7
17:33:56	27,2	48.9
17:34:26	27,1	49.1
17:34:56	27,1	49.8
17:35:26	27,2	48.3
17:35:56	27,5	48.1

V. CONCLUSÃO

Utilizando-se o conceito de DAQ foi possível desenvolver um sistema de aquisição de dados e controle através de placas customizadas de baixo custo que continham sensor de temperatura e umidade, LCD, relés, softwares, microcontrolador e computador. O objetivo é além da aquisição de dados, permitir o controle das variáveis ambientais sob estudo. A comunicação com o sensor feita diretamente com uma porta digital possibilita o microcontrolador estabelecer comunicação usando o protocolo UART. Essa vantagem pode ser aproveitada em outros projetos com outros protocolos. O código utiliza apenas um timer. O Timer 1 é utilizado apenas na rotina do relógio, na temporização da comunicação com o sensor utilizou-se rotinas de *delay*. Uma das dificuldades desse trabalho foi conciliar a aquisição de dados com o tempo de resposta do sensor. Esse tempo depende de vários fatores, se for analógico, depende das constantes de tempo do circuito, no caso desse projeto foi usado um sensor digital com tempo de resposta médio de 2 segundos. Mesmo os sensores digitais podem precisar de circuitos adicionais, dependendo da distância. Essas informações são encontradas no *datasheet* do sensor. Verifica-se, portanto, que o sistema se comportou adequadamente e dentro das condições estabelecidas de monitoramento e controle de variáveis ambientais em estufa vegetal sendo, nesse caso, adequado a uso nesses tipos de aplicações.

AGRADECIMENTOS

Os autores são gratos à Fundação de Amparo a Pesquisa do Estado de Mato Grosso - FAPEMAT pelo apoio financeiro concedido na realização do projeto.

REFERENCES

1. D. G. Alciatore e M. B. Histand, Introdução à Mecatrônica e aos Sistemas de Medições. AMGH: São Paulo, 2014.
2. F. L. M. C. Filho, Desenvolvimento de Sistema de Controle e Aquisição Microcontrolado - Monografia, Universidade Federal do Ceará – UFC, 2010, 76p.
3. A. V. Kale, S. A. Bankar e S. R. Jagtap, "Design of PIC Microcontroller-based Data Acquisition Module with Lab VIEW Interfacing," In: 2014 IEEE International Conference on Communications and Signal Processing (ICCSP), p. 858-861.
4. J. Park and S. Mackay, Practical Data Acquisition for Instrumentation and Control Systems. Newnes: New Jersey, 2003.
5. T. Liu, Digital Relative Humidity & Temperature Sensor AM2302/DHT22. Disponível em: <http://www.electroschema-tics.com/11293/am2302-dht22-datasheet/>. Acesso em: 27 maio 2017.
6. A. S. Oliveira e F. S. Andrade, Sistemas Embarcados: Hardware e Firmware na Prática. Érica: São Paulo, 2010.
7. Embedded Lab, Measurement of Temperature and Relative Humidity using DHT11 Sensor and PIC Microcontroller. Disponível em: <http://embedded-lab.com/blog/measurement-of-temperature-and-

relative-humidity-using-dht11-sensor-and-pic-microcontroller/>.
Acesso em: 10 Agos 2018.

8. MICROCHIP, PIC16F87X Data Sheet; _MP, USERs GUIDE. Microchip Technology Inc. 2001.

APÊNDICE I – DIAGRAMA DE BLOCOS EM LABVIEW®