

**UNIVERSIDADE FEDERAL DE VIÇOSA**  
**CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS**  
**DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE PRODUÇÃO**  
**CURSO DE ENGENHARIA ELÉTRICA**

**DESENVOLVIMENTO DE UM SISTEMA DE AQUISIÇÃO E  
MONITORAMENTO UTILIZANDO MICROCONTROLADORES**

**HEVERTON AUGUSTO PEREIRA**

**VIÇOSA**  
**MINAS GERAIS – BRASIL**  
**DEZEMBRO / 2006**

**HEVERTON AUGUSTO PEREIRA**

**DESENVOLVIMENTO DE UM SISTEMA DE AQUISIÇÃO E  
MONITORAMENTO UTILIZANDO MICROCONTROLADORES**

**Trabalho de Conclusão de Curso submetido  
à Universidade Federal de Viçosa para a  
obtenção dos créditos referentes à disciplina  
Monografia e Seminário do curso de  
Engenharia Elétrica.**

**Viçosa – Minas Gerais – Brasil  
20 de Dezembro de 2006**



*Universidade Federal de Viçosa*

**CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS**  
**CURSO DE ENGENHARIA ELÉTRICA**

**Parecer da Banca de Monografia**

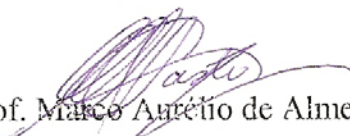
A banca composta com o Prof. Denílson Eduardo Rodrigues, o Prof. José Tarcísio de Resende e o Prof. Marco Aurélio de Almeida Castro, sob a presidência do primeiro, se reuniu no dia 20 de dezembro de 2006 para avaliar a defesa do trabalho de monografia apresentado pelo aluno Heverton Augusto Pereira, com o tema “Desenvolvimento de um Sistema de Aquisição e Monitoramento Utilizando Microcontroladores”.

Após a conclusão dos trabalhos a banca **aprovou** a defesa da monografia apresentada pelo aluno.

Viçosa, 20 de dezembro de 2006.

  
Prof. Denílson Eduardo Rodrigues (presidente) – Orientador

  
Prof. José Tarcísio de Resende (membro)

  
Prof. Marco Aurélio de Almeida Castro (membro)

## DEDICATÓRIA

A todos que pensam em mim.

## **AGRADECIMENTOS**

Fazer um curso superior é como construir um prédio. Antes de começar a fazê-lo é necessário muito planejamento para minimizar os erros durante a construção. Mas, o mais importante tanto na construção de um prédio quanto na graduação no curso superior são as pessoas que ajudam o dono a construí-lo. Muito antes de pensar em fazer o curso de Engenharia Elétrica já tinha uma equipe que gastava horas do seu dia pensando se estava tudo bem comigo. Depois que fui crescendo, essa equipe foi aumentando e pessoas que até então eu nunca tinha visto, com pouco tempo de convívio começaram a se importar com as coisas que eu fazia e se estava tendo as melhores condições tanto físicas quanto psicológicas. Todas as pessoas que de uma maneira direta ou indireta ajudaram na minha formação eu gostaria de agradecer, pois, sem elas eu não teria conseguido superar tantos obstáculos que muitas vezes pareciam intransponíveis.

Eu sei que não terei espaço nesta simples folha para agradecer a todos, mas, se algum dia você perdeu parte do seu tempo por causa de mim, eu lhe agradeço. Agradeço de maneira especial a Deus e Maria, pelas infinitas graças como meu pai Júlio e minha mãe Maria Aparecia, minha avó Zélia que hoje esta no céu e meus outros avós, meu irmão Humberto e minhas irmãs Monaliza e Janaliza e a minha amiga e namorada Liliane juntamente com seus pais. Aos companheiros do quarto 434, André, Zé Pedro e Tiago. Aos meus amigos Marciano, Daiwison, Renato, Elismar, Denis, Tiago e Rommel pelas grandes coisas que fizemos juntos. Aos professores Delly, José Tarcísio, André, Ricardo, Tarcísio e Denílson pelo carinho e atenção. Ao doutor Carlos Alberto por ter acreditado no meu potencial e ter me aberto muitas portas que eu sozinho não conseguiria abrir.

Encerro agradecendo a todos os companheiros da elétrica que fizeram parte desta pequena grande caminhada de minha vida.

## LISTA DE FIGURAS

Figura 1 – Sinal original (a) e sinal distorcido (b).....	12
Figura 2 – Diagrama em blocos de um modulador FSK.....	13
Figura 3 – Formas de onda de um modulador FSK.....	14
Figura 4 – Diagrama em blocos de um demodulador FSK.....	15
Figura 5 – Recursos do microcontrolador PIC 16F877.....	16
Figura 6 – Circuito integrado PIC 16F877 com funcionalidade dos pinos.....	19
Figura 7 – Representação da comunicação serial.....	19
Figura 8 – Níveis Lógicos/ Tensão.....	20
Figura 9 – Seqüência de transmissão serial.....	21
Figura 10 – Ambiente de desenvolvimento do C++ Builder.....	23
Figura 11 – Estrutura do programa do microcontrolador PIC 16F877.....	26
Figura 12 – Estrutura do programa desenvolvido no C++ Builder.....	28
Figura 13 – Arquivo gravado pelo programa.....	30
Figura 14 – Ferramenta Timer do C++ Builder.....	32
Figura 15 – Ambiente de desenvolvimento Proteus.....	33
Figura 16 – Saída do pino de transmissão do microcontrolador.....	34
Figura 17 – Configuração da interface com a porta serial.....	34
Figura 18 – Layout do programa desenvolvido.....	35
Figura 19 – Tela de calibração dos sensores.....	36
Figura 20 – Tela de Análise Estatística.....	37
Figura 21 – Formato do arquivo salvo.....	38
Figura 22 – Exemplo de calibração dos sensores.....	39
Figura 23 – Escolha do arquivo para gravação dos dados.....	40
Figura 24 – Programa em funcionamento com dois sensores de temperatura.....	40
Figura 25 – Tela Análise Estatística do exemplo anterior.....	41
Figura 26 – Erro de não definição da porta serial.....	42
Figura 27 – Erro de não configuração dos sensores.....	42
Figura 28 – Erro de não indicação dos arquivos para gravação dos dados.....	43

## **LISTA DE TABELA**

Tabela 1 – Descrição dos pinos da porta serial.....	20
---	----

## SUMÁRIO

<b>DEDICATÓRIA .....</b>	<b>I</b>
<b>AGRADECIMENTOS .....</b>	<b>II</b>
<b>LISTA DE FIGURAS.....</b>	<b>III</b>
<b>LISTA DE TABELA .....</b>	<b>IV</b>
<b>SUMÁRIO .....</b>	<b>V</b>
<b>RESUMO.....</b>	<b>VI</b>
<b>ABSTRACT .....</b>	<b>VII</b>
<b>1. INTRODUÇÃO .....</b>	<b>8</b>
<b>2. REVISÃO BIBLIOGRÁFICA.....</b>	<b>9</b>
<b>2.1 Automação na agricultura.....</b>	<b>9</b>
<b>2.2 Transmissão de dados .....</b>	<b>9</b>
<b>2.3 Modulação por chaveamento de frequência – FSK .....</b>	<b>13</b>
<b>2.4 Demodulação do sinal FSK.....</b>	<b>14</b>
<b>2.5 Microcontrolador PIC 16F877 .....</b>	<b>15</b>
<b>2.6 Interface Serial .....</b>	<b>19</b>
<b>2.7 O ambiente de desenvolvimento Borland C++ Builder .....</b>	<b>22</b>
<b>3. OBJETIVOS .....</b>	<b>25</b>
<b>4. MATERIAL E MÉTODOS.....</b>	<b>26</b>
<b>4.1 Programa do Microcontrolador.....</b>	<b>26</b>
<b>4.2 Programa no C++ Builder .....</b>	<b>28</b>
<b>4.3 Configuração dos sensores e Conversão do sinal .....</b>	<b>31</b>
<b>4.4 Comparação entre componente Timer e processos com Thread .....</b>	<b>31</b>
<b>5. RESULTADOS E DISCUSSÕES .....</b>	<b>33</b>
<b>Mensagens de erro.....</b>	<b>41</b>
<b>6. CONCLUSÕES .....</b>	<b>44</b>
<b>7. SUGESTÕES PARA TRABALHOS FUTUROS.....</b>	<b>45</b>
<b>8. REFERÊNCIAS BIBLIOGRÁFICAS .....</b>	<b>46</b>



## RESUMO

Com o desenvolvimento do parque industrial e agroindustrial brasileiro, cada vez mais, a automação tem sido uma ferramenta importante para se centralizar as decisões, conhecer os processos de forma a aumentar a produtividade e minimizar as perdas. Mas, muitas vezes esses sistemas não são empregados pelo elevado custo, decorrentes das tecnologias desenvolvidas. Produtos agrícolas têm sua qualidade diretamente ligada às etapas de produção, sendo que, o controle dos fatores físicos como temperatura, umidade, concentração de gases são de vital importância para alcançar a máxima qualidade. O objetivo deste trabalho foi desenvolver um sistema de aquisição e monitoramento de dados sem fio, utilizando microcontroladores PIC 16F877 para fazer a aquisição do sinal analógico de sensores e transmiti-los de forma serial, para um transmissor FM com modulação chaveada. A recepção é feita com um demodulador ligado à porta serial do computador. Um programa desenvolvido utilizando o software C++ Builder é o responsável por mostrar os dados recebidos em gráficos de evolução das medidas dos sensores com o passar do tempo, também sendo responsável por guardar os valores lidos em arquivos de formato \*.txt, os quais, podem ser utilizados por outros programas. Os dados enviados pelo microcontrolador ao computador têm precisão de 10 bits, e toda a parte de ajuste como a faixa de medição e a precisão de exibição dos sinais é feita no próprio programa do C++ Builder, de maneira que o sistema de aquisição feito para o microcontrolador se torna independente do sensor, pois, os dados serão ajustados no computador. Logo, com o sistema desenvolvido, pequenas instalações podem utilizar o sistema de aquisição e monitoração das variáveis do ambiente para em uma segunda etapa, poder automatizar os processos.

Palavras chaves: microcontroladores, C++ Builder, aquisição, transmissão serial, monitoração.

## **ABSTRACT**

With the development of Brazilian industrial park and agro-industry automation has become an important tool to make decisions to know the processes to increase productivity and to minimize losses. But, many times these systems are not used because of high costs, due to the developed technologies. Quality of agricultural products is directly dependent on the stages of production, being that, the control of the physical factors such as temperature, humidity and concentration of gases are of vital importance to reach the maximum quality. The objective of this work was to develop a system of acquisition and observation of data without wire, using microcontrollers PIC 16F877 to make the acquisition of the analogical signal of sensors and to transmit them as a serial form, to a FM transmitter with keyed modulation. The reception is made with a demodulator connected to the serial port of the computer. A developed program using software C++ Builder is responsible for showing the data received in evolution graphs of the real time measures of the sensors, also being responsible for keeping to the values read in format archives \*.txt, which, can be used by other programs. The data sent by the microcontroller to the computer have a precision of 10 bits, and all the measurement band adjustments and the precision of exhibition of the signals is made in the proper program of the C++ Builder, thus the system of acquisition made for the microcontroller becomes independent of the sensor, therefore, the data will be adjusted in the computer. Soon, with the developed system, small installations can use the acquisition system and observation of the environment variable and in one second stage, can automation the processes.

Words keys: microcontrollers, C++ Builder, acquisition, serial transmission, observation.

## 1. INTRODUÇÃO

Na atualidade, a modernização das instalações, tem sido a maneira mais eficiente de reduzir custos e aumentar a produtividade, conseguindo agregar qualidade as mercadorias, e tornar os processos cada vez mais independentes do homem.

Para conhecer um processo tanto industrial quanto agrícola, a monitoração do sistema é de fundamental importância, pois, análises podem ser feitas e problemas podem ser observados e corrigidos.

Mas, não é por falta de sistemas de monitoramento de dados, que muitos processos não são automatizados. O principal problema é o preço desses produtos, que principalmente para pequenos e médios empresários são inviáveis a sua aquisição.

Logo, o projeto que está sendo desenvolvido pode atender a esse espaço existente no setor industrial e principalmente no setor agrícola, onde a qualidade do produto está intimamente ligada às etapas do processo. Sendo que, controles dos diversos fatores climáticos como temperatura, umidade e pressão podem e devem ser feitos para evitar que a qualidade seja perdida e conseqüentemente haja redução de preço dos produtos.

Outro problema encontrado para a aquisição de dados nos setores produtivos é a dificuldade para modificar as plantas industriais, o que ocorre caso seja necessário utilizar sistemas de monitoramento por meio da utilização de fios. Pois, pode ser necessário ter que construir novos eletrodutos para a passagem dos cabos. Uma maneira de minimizar esse problema é através de monitoramento sem fio, sistema conhecido como **wireless**.

Muitos outros problemas, no entanto, surgem com a utilização do sistema de monitoramento sem fio, sendo o principal, as interferências devido a diversas fontes de ruídos. De maneira que recursos como paridade, podem ser utilizados para minimizar o efeito da aquisição de dados de maneira errônea.

## 2. REVISÃO BIBLIOGRÁFICA

Muito tem sido desenvolvido ao longo das últimas décadas a respeito da automação nos processos. De maneira especial, a agricultura brasileira tem conseguido grandes avanços, principalmente, com a necessidade de atender ao mercado externo que é cada vez mais exigente.

### 2.1 Automação na agricultura

Segundo DALLY et al. (1993), o monitoramento automático por meio de sistemas de aquisição de dados tem gerado significativos avanços no controle de ambiente em instalações agrícolas, pois permite rapidez, confiabilidade e menor risco de erros quando comparada ao monitoramento manual, contribuindo para a tomada rápida de decisões.

De acordo com STEIDLE NETO (2003), com o processo de modernização, o setor agrícola passou a buscar, nas instalações, as possibilidades de melhoria do desempenho produtivo dos animais ou vegetais, sendo ela uma das principais responsáveis pelo sucesso ou fracasso, devido à perda da qualidade, do empreendimento agrícola.

Os principais entraves ao monitoramento automático em instalações agrícolas destinadas a produção de animais e vegetais, em escala industrial, são os grandes comprimentos destas instalações, as distâncias entre elas (principalmente em relação às de produção de animais, devido ao aspecto sanitário) e os altos custos dos sistemas de aquisição de dados.

A necessidade de sistemas de monitoramento automático confiáveis, versáteis e de baixo custo, que atendem às exigências das instalações agrícolas, com vistas a otimizar o processo produtivo, é de fundamental importância, principalmente para o pequeno e médio empresário. Desta forma, é desejável um sistema que possibilite a transmissão de dados a grandes distâncias, utilizando um número de sensores suficientes para um monitoramento adequado, com nível de exatidão apropriado à finalidade a que se destina e de custo relativamente baixo, comparado com sistemas tradicionais de aquisição de dados.

### 2.2 Transmissão de dados

**Conceito de canal:** Canal ou meio de comunicação é o meio físico entre o transmissor e o receptor por onde transitam os sinais elétricos ou eletromagnéticos da informação.

### 2.2.1 Tipos e características dos canais

Segundo HAYKIN (2004), a transmissão de informação através de uma rede de comunicação é realizada por meio de um canal de comunicação. Dentre eles podemos destacar:

#### I. Canal fio:

Consiste em pelo menos dois fios condutores elétricos usados para conduzir os sinais da informação.

Uma rede telefônica fixa é feita com fios e cabos. Os cabos são constituídos internamente de múltiplos condutores isolados, podendo abrigar, em seu núcleo, cabos coaxiais e fibras ópticas.

Um cabo pode ser instalado de modo aéreo, enterrado, subterrâneo ou ainda submerso em rio, lago ou oceano. Em casas e edifícios, quando embutidos nas paredes e pisos, passam em condutores destinados exclusivamente à rede telefônica/áudio/TV, separadamente dos condutos com os fios da rede elétrica.

#### II. Canal fibra óptica

A fibra óptica é um elemento monofilar de estrutura cristalina, condutor de luz, que transporta a informação sob forma de energia luminosa. É também chamada de guia de luz. Possui alto índice de refração e a luz, ao se propagar na fibra, sofre atenuação e dispersão. O canal de fibra óptica apresenta algumas vantagens sobre os demais meios, dentre elas:

- Não irradia nem sofre interferências de sinais eletromagnéticos externos.
- A atenuação da luz na fibra, embora exista, é de baixo valor.
- Permite elevadas taxas na transmissão de dados, da ordem de 10Gbits/seg.

Como desvantagens têm o elevado custo, e exigir altas precisões nas suas conexões.

#### III. Canal rádio

Deve ser entendido como um segmento do espectro de frequências ocupado pela onda eletromagnética de um equipamento emissor. A onda de rádio se propaga no espaço livre transportando os sinais elétricos da informação.

O espaço livre é o meio das comunicações de rádio. O canal de rádio é o mais fácil de ocupar. Enquanto os canais fio e fibra óptica precisam ser construídos, para o canal de rádio basta ter em mãos os equipamentos transceptores para o fechamento do enlace. Contudo no

espaço livre, as ondas eletromagnéticas encontram problemas de distúrbios e interferências, que evidenciam a fragilidade do canal.

### **2.2.2 Propriedades dos canais de comunicações**

Cada tipo de canal possui características próprias e utilização específica, mas todos têm algumas propriedades em comum. Dentre elas podem ser citadas:

#### **I. Atenuação da intensidade do sinal**

O sinal elétrico vai perdendo energia ao se propagar no canal de comunicação e chega ao receptor com intensidade atenuada.

#### **II. Limitação em largura de faixa**

Todo canal é limitado em faixa ou banda passante. Sendo a largura do canal limitada, o espectro do sinal da fonte deve ser menor ou, no máximo, igual à largura do canal. Se essa imposição for contrariada, fatalmente ocorrerá distorção do sinal com implicações na inteligibilidade da mensagem.

#### **III. Retardo**

Retardo é o nome dado ao tempo gasto para o sinal atravessar o canal de comunicações. O tempo de retardo  $\Delta t$  é calculado dividindo-se a distância percorrida pela onda no enlace dos dois pontos, pela velocidade de propagação da onda no meio de propagação.

Também cada tipo de equipamento integrante do sistema impõe ao sinal um tempo de retardo próprio do circuito eletrônico, quase sempre de pequeno valor, mas nem sempre desprezível. Em alguns casos, grupos de frequência do sinal sofrem um retardo maior que outras frequências.

### **2.2.3 Principais distúrbios dos canais de comunicações**

Os sinais estão sujeitos à ocorrência de distúrbios no canal, promovidos por fenômenos que prejudicam e, por vez, até inviabilizam a recepção. Os principais distúrbios que ocorrem nos canais são:

#### **I. Ruído elétrico**

O ruído elétrico existe na forma de corrente elétrica, quando gerado internamente em dispositivos eletrônicos e na forma de onda eletromagnética, no espaço livre. É o resultado da agitação térmica dos elétrons existentes na matéria. Na realidade, a onda de rádio viaja num mar de ruídos gerados por diversas fontes.

Segundo HAYKIN, (2004), o ruído refere-se a sinais indesejados que tendem a perturbar a presença a transmissão e o processamento da mensagem em um sistema de comunicação.

## II. Distorção do sinal

A distorção do sinal consiste na alteração da forma do sinal. Os diferentes valores de atenuação imposta às diferentes frequências que formam os sinais geram distorções. Por exemplo, no canal fio, ao se aplicar pulsos das comunicações digitais diretamente a um par de fios ou a um cabo, devido à capacitância e à indutância distribuídas, a partir de alguns poucos metros da fonte geradora começará a ocorrer distorções. Na figura 1 pode se observar os pulsos transmitidos e distorcidos.

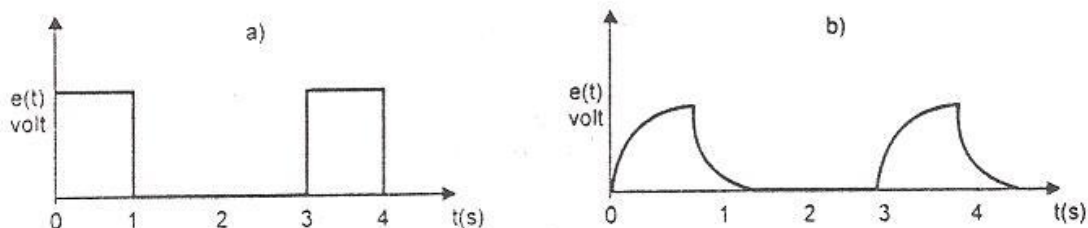


Figura 1 – Sinal original (a) e sinal distorcido (b)

### 2.2.4 As ondas de rádio

As ondas de rádio ou hertzianas são campos eletromagnéticos de alta frequência não audíveis e não visíveis pelo homem, irradiados pela antena do transmissor.

- I. O rádiotransmissor:** é o aparelho eletrônico gerador da onda alternada senoidal, numa certa frequência, que alimenta a antena.
- II. A antena:** feita de condutor metálico tem por função na transmissão, converter a corrente elétrica em energia radiante e, inversamente na recepção captar a onda de rádio e convertê-la em corrente elétrica.

As ondas hertzianas podem ser geradas em qualquer frequência, entretanto são mais usadas em telecomunicações acima dos 100 kHz.

Gerada pelo transmissor, em sua forma original, sem modulação, a onda portadora é usada nas comunicações chaveadas. Quando modulada pelo sinal da informação, em amplitude (AM) ou em frequência (FM), no transmissor, pode transportar a voz, a música, a imagem e os dados das comunicações digitais.

**III. A demodulação:** é a operação realizada na recepção que retira o sinal de informação da onda portadora.

### 2.3 Modulação por chaveamento de frequência – FSK

Segundo GOMES (1998), a modulação por chaveamento de frequência, FSK (Frequency Shift Keying), apresenta como principal característica a boa imunidade a ruídos, quando comparada a outros tipos de modulação. Como ponto negativo a modulação FSK apresenta a maior largura de faixa dentre as modulações chaveadas.

A modulação FSK pode ser obtida pela aplicação do sinal digital, com a banda de frequência limitada, na entrada de um VCO (circuito que oscila conforme a amplitude), conforme figura 2.

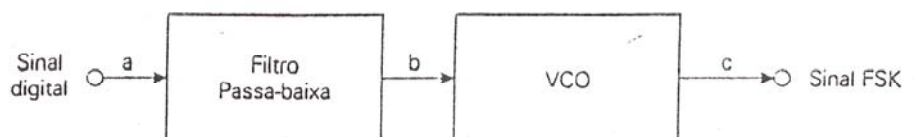


Figura 2 – Diagrama em blocos de um modulador FSK

As variações de amplitude do sinal digital forçam o VCO a variar sua frequência entre dois valores diferentes. A frequência correspondente ao bit 0 é chamada de frequência espaço e a correspondente ao bit 1, de frequência marca, conforme figura 3.



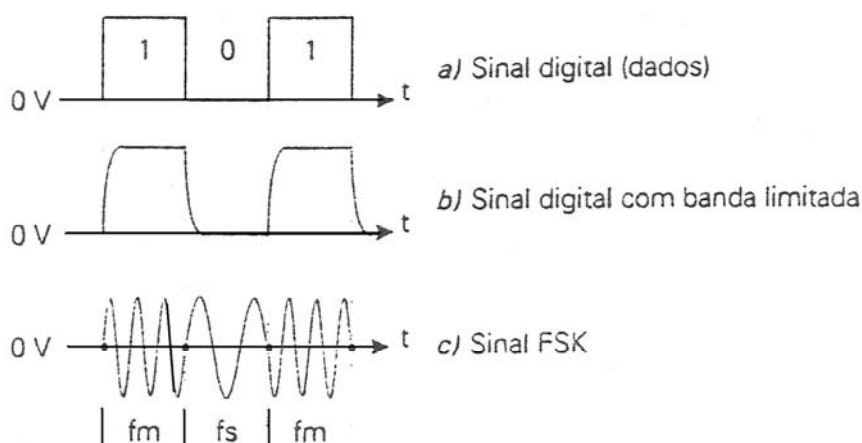


Figura 3 – Formas de onda de um modulador FSK

A largura de faixa do sinal FSK depende da velocidade de transmissão e da diferença entre as frequências marca,  $f_m$ , e espaço,  $f_s$ . A equação 1 é usada para o cálculo da largura de faixa do sinal.

$$BW(\text{FSK}) = V_m (1 + R) + (f_m + f_s) \quad (\text{Equação 1})$$

Onde:  $BW(\text{FSK})$  = largura de faixa do sinal FSK, em Hz;

$V_m$  = velocidade de transmissão em bps;

$R$  = fator de filtragem passa-baixa, em Hz;

$f_m$  = frequência marca, em Hz;

$f_s$  = frequência espaço, em Hz.

O desvio de frequência utilizado, que é a diferença entre a frequência marca e a frequência espaço, está relacionada com a velocidade de transmissão. Normalmente, se usa um desvio de frequência, em Hz, entre a metade e o dobro da velocidade de transmissão, em bps.

## 2.4 Demodulação do sinal FSK

Segundo GOMES (1998), a demodulação do sinal pode ser feita com o circuito mostrado na figura 4. O amplificador limitador tem a finalidade de amplificar o sinal FSK aplicado na entrada do demodulador e eliminar as variações de amplitude e ruídos eventualmente

presentes em a. Na saída do amplificador limitador, em b, teremos um sinal de amplitude constante, que será aplicado aos FPFs do circuito marca e espaço. O amplificador limitador é o maior responsável pela boa imunidade contra ruídos da modulação FSK.

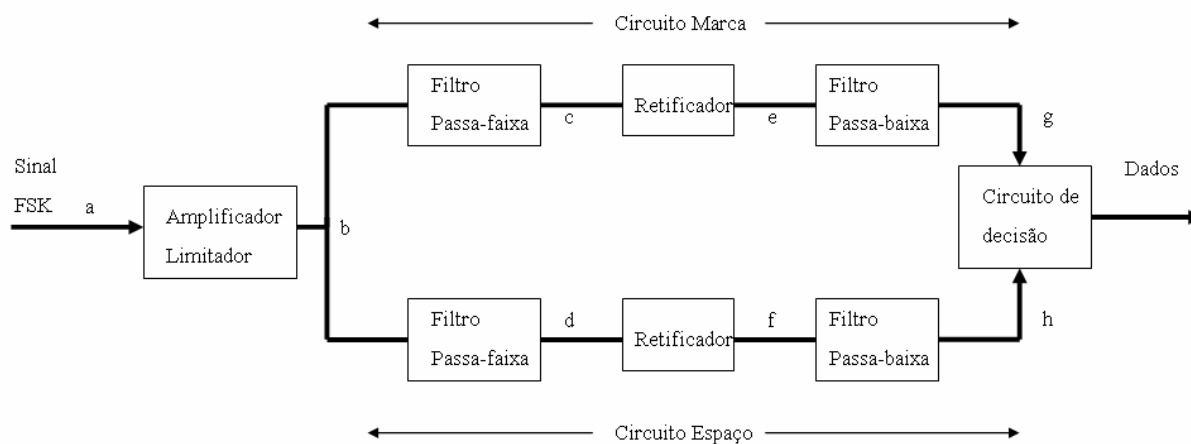


Figura 4 – Diagrama em blocos de um demodulador FSK

Outra razão para a boa imunidade a ruídos deve-se ao modo como funciona o circuito de decisão usado no demodulador. O circuito de decisão determina o nível de saída em função da amplitude dos sinais em sua entrada, pontos g e h. A saída irá para nível alto se a tensão no ponto g for mais elevada que no ponto h. Caso contrário, a saída irá para nível baixo.

Quando a frequência do sinal recebido for igual à frequência espaço, aparecerá sinal na saída do filtro passa-faixa do circuito de espaço, d, o sinal será retificado, f, que é filtrado pelo filtro passa-baixa, aparecendo uma tensão em h. Como a tensão em h será maior que a em g, o circuito de decisão coloca a saída em nível baixo.

## 2.5 Microcontrolador PIC 16F877

O microcontrolador PIC 16F877 da Microchip® é um dispositivo programável capaz de realizar diferentes atividades que requerem de processamento de dados digitais, controle e comunicação digital de diferentes dispositivos.

Segundo PEREIRA (2004), os microcontroladores possuem uma memória interna que armazena dois tipos de dados: as instruções, que corresponde ao programa que se executa, e os registradores, para manipular os dados. Eles podem ser programados nas linguagens computacionais C, Basic e Assembler.

As partes principais são: uma ALU (Unidade Lógica Aritmética), memória de programa, memória de registros, e pinos I/O (entrada/ saída). A ALU é a encarregada de processar os dados dependendo das instruções que se execute (ADD, OR, AND). Alguns pinos se encarregam de comunicar o microcontrolador com o meio externo. A função dos pinos pode ser de transmissão/ recepção de dados ou alimentação de corrente para o funcionamento.

As características do Microcontrolador PIC16F877 Microchip® podem ser vistos na figura 5 e são as seguintes:

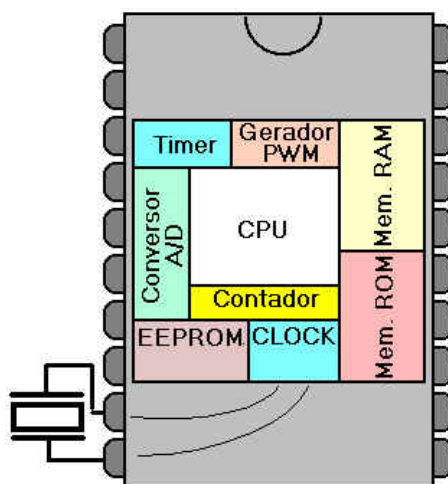


Figura 5 – Recursos do microcontrolador PIC 16F877

- Memória FLASH de programa com 8kbytes.
- 368 bytes de RAM.
- 256 bytes de EEPROM.
- 32 pinos de I/O com dreno de corrente na ordem de 20 mA.
- 1 conversor analógico (AD) de 10 bits .
- Dois “timers” de 8 bits e um de 16 bits.
- Dois canais CCP (Capture, Compare and PWM).
- Um canal USART para comunicação RS-232.
- Comunicação SPI ou I2C.
- Várias opções para oscilador:
  - RC
  - Cristal ou ressonador
  - Clock externo
- Watch Dog Timer: (cão de guarda) interno.

- Controle de “Power-on Reset” e “Power-up Timer”.
- Proteção de código contra cópias.
- Encapsulamento DIP com 40 pinos.

**Quantidade de I-Os (entradas e saídas):** Refere-se à quantidade de pinos de entrada e saída existentes no microcontrolador. Estes pinos geralmente são conhecidos por agrupamentos (portos). As I-Os estão agrupadas em "Ports". No PIC 16F877 temos o Port A (com seis I-Os), o Port B, Port C e Port D (cada um com oito I-Os), e o Port E (com três I-Os). Cada I-O pode funcionar como entrada e como saída ao mesmo tempo. Os sinais são digitais em níveis de tensão zero e cinco volts. Quando um pino é acionado, ele envia um sinal de cinco volts, e quando é desligado, este nível de tensão baixa para zero volts. Importante lembrar que existem limitações de corrente que devem ser observadas. Se for submetida uma corrente maior do que o limite do microcontrolador, o mesmo poderá queimar.

**Entradas analógicas:** Alguns microcontroladores são providos de conversores A/D (analógico/digital) permitindo que uma determinada variação de tensão sobre um pino (zero a cinco volts) seja transformada em um valor digital (geralmente de oito a dezesseis bits). No caso do PIC16F877, são oito entradas conectadas a um conversor AD de dez bits.

**Saídas pulsantes (PWM):** permite que determinados pinos "simulem" uma saída analógica, por intermédio de "modulação por largura de pulso", ou PWM.

**Timer ou Temporizador / Contador:** Permite a contagem de tempo (através do Clock) ou de eventos externos (pulsos em um pino específico). No caso de necessitar uma contagem de tempo precisa, esta é uma boa solução.

**RAM:** Os "registradores" são áreas da memória RAM do microcontrolador utilizadas diretamente no processamento, assim como a área para memória de dados. Esta área de memória RAM interna é necessária, e em um projeto devemos levar em consideração a capacidade do microcontrolador (geralmente são alguns bytes).

**ROM:** O programa desenvolvido é gravado na memória ROM. Quanto mais complexo o problema, maior deve ficar o programa. Esta memória é somente de leitura, e pode ser

gravada muitas vezes. A capacidade desta memória é relativamente pequena, e em alguns casos o programa desenvolvido precisa ser otimizado para poder caber nesta memória.

**Memória ROM tipo FLASH:** Permite a gravação rápida, utilizando-se um circuito especial geralmente ligado ao microcomputador. Bom para ser utilizado em protótipos ou para fins didáticos.

**CLOCK:** Quem dita a velocidade como as instruções serão executadas é um circuito pulsante conhecido por "circuito oscilador". No entanto, grande parte deste circuito já está embutido no microcontrolador, restando somente para o desenvolvedor a adição de um cristal externo, de frequência variável, dependendo do microcontrolador utilizado. Por exemplo, um PIC 16F877A pode usar um cristal de até vinte MHz sem problemas, mas tradicionalmente é empregado um de quatro Mhz.

**Registradores Especiais:** Algumas áreas da memória RAM do microcontrolador são utilizadas para configuração de periféricos e informações de controle dos circuitos embutidos. Estas áreas de memória estão reservadas para estas funções. Para uso com variáveis e armazenamento temporário de valores, são utilizados os REGISTRADORES DE USO GERAL.

**Interrupção:** É um recurso que permite que uma determinada tarefa seja disparada por um determinado evento (interno ou externo). Por exemplo, uma rotina de emergência pode ser configurada como um evento externo, associada a um pino específico do sistema.

**Instrução:** Um programa é composto de várias instruções, executadas seqüencialmente. Uma instrução é uma unidade de programa que realiza uma determinada tarefa, como por exemplo, adicionar um valor, ativar uma saída ou desativar um recurso.

Na figura 6 é possível observar como é o formato e quais são as funções dos pinos do microcontrolador PIC16F877.

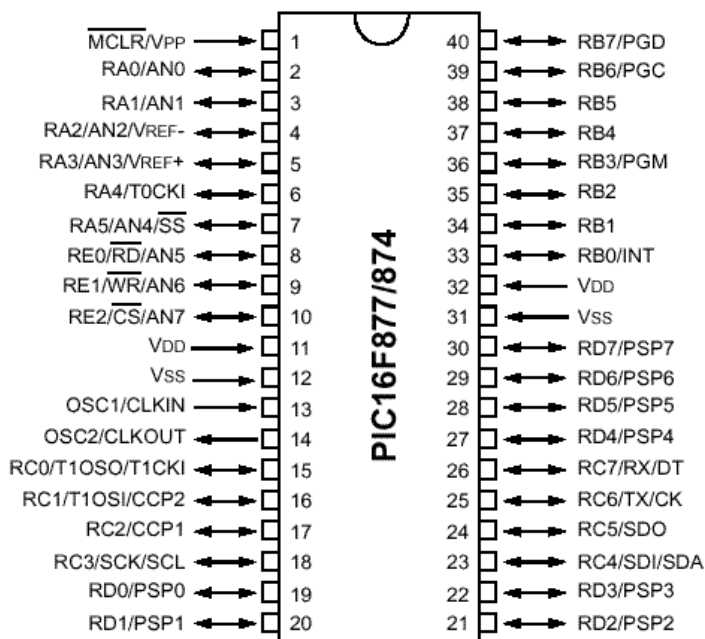


Figura 6 – Circuito integrado PIC 16F877 com funcionalidade dos pinos

## 2.6 Interface Serial

Segundo MATEUS (2000), a comunicação serial é aquela em que os dados são transmitidos por apenas um par de fios (TX e RX), tendo uma maior imunidade a ruídos. A figura 7 ilustra a comunicação serial.

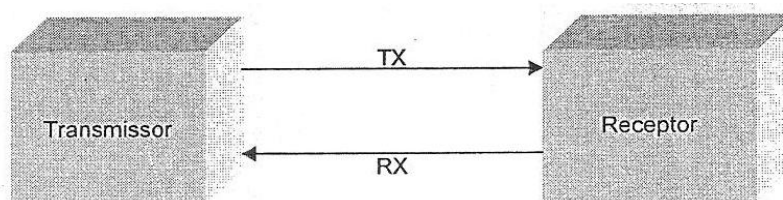


Figura 7 – Representação da comunicação serial

Na tentativa de tornar possível a interconexão de equipamentos diferentes, foi elaborada pela EIA (Electronic Industries Association) em 1969, o padrão RS-232C que especifica as características elétricas e físicas da comunicação entre dois equipamentos, o DTE (Data Terminal Equipment) e o DCE (Data Communication Equipment).

O padrão RS-232 define:

- Nível lógico baixo (bit 0) – Tensão entre +3V e + 15V.

- Nível lógico alto (bit 1) – Tensão entre -3V e -15V.

Na figura 8 está exibido a representação nível lógico/ tensão.

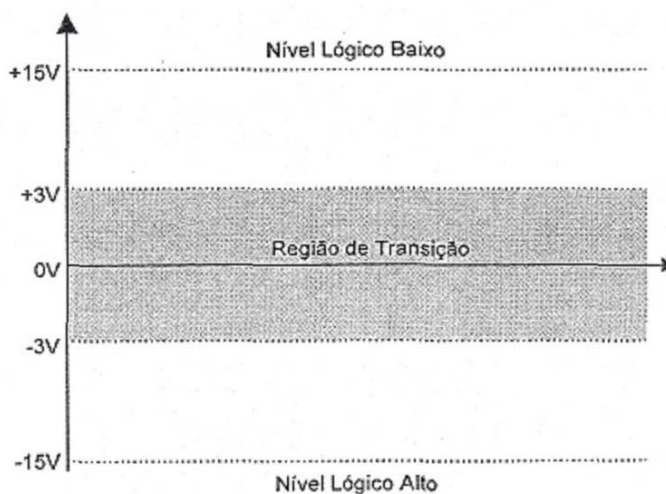


Figura 8 – Níveis Lógicos/ Tensão

E na tabela 1 a especificação dos pinos da porta serial de um computador.

Pino	Função
1	CD- Carrier Detect
2	Recepção (RX)
3	Transmissão (TX)
4	DTR – Data Terminal Ready
5	Terra
6	DSR- Data Set Ready
7	RTS- Request To Send
8	CTS- Clear To Send
9	RI- Ring Indicator

Tabela 1 – Descrição dos pinos da porta serial.

O principal elemento de uma porta de comunicação serial é um chip chamado UART (Universal Asynchronous Receiver Transmitter), que é responsável por transformar qualquer informação paralela em uma seqüência de bits seriais, e também executar a operação inversa.

Ainda segundo MATEUS (2000), para usar a porta serial é preciso definir alguns parâmetros da comunicação como:

**Taxa de Transmissão (Baud Rate):** É a velocidade com que os dados serão transmitidos ou recebidos pela interface serial. A taxa de transmissão pode variar de 300 bps (bits por segundo) até 115200 bps.

**Número de Bits de dados:** É a quantidade de bits a serem transmitidos/ recebidos. Os valores de bits podem variar de 5 a 8 bits.

**Start Bit:** Bit que marca o início do envio de um caractere.

**Stop Bit:** Define a quantidade de bits de parada, ou seja, quantidade de bits que indicam o final de um caractere. Podemos definir 1 ou 2 como a quantidade de Stop Bits.

**Paridade:** É um bit usado para detecção de erros na transmissão. Sempre que um dado é enviado, ele está sujeito a sofrer alterações no meio de transmissão. Tais alterações podem ser provocadas por fenômenos como interferências ou ruídos no canal de transmissão.

As opções de configuração da UART para paridade podem ser: nenhuma, par ou ímpar. Quando for definida paridade “par”, a soma dos bits 1 do dado mais o bit de paridade sempre resulta em um número par. Quando a paridade é “ímpar”, a soma dos bits de dados mais a paridade resultam em um número ímpar e quando for definida “nenhuma”, o bit de paridade não é enviado.

A configuração feita em um dos equipamentos (transmissor, por exemplo) deve ser idêntica às configurações feitas para o outro equipamento (receptor, por exemplo). Caso sejam feitas configurações diferentes para cada um dos equipamentos, não teremos comunicação ou os dados recebidos não terão relação com o que foi enviado.

Configurada corretamente, a porta serial vai enviar os dados segundo a seqüência de sinais da figura 9.

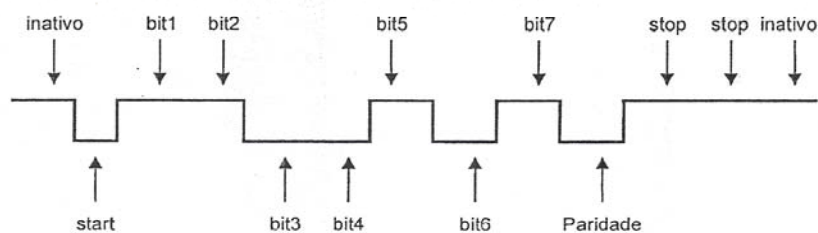


Figura 9 – Seqüência de transmissão serial



## 2.7 O ambiente de desenvolvimento Borland C++ Builder

O programa computacional para o gerenciamento do sistema foi desenvolvido de acordo com o protocolo de comunicação RS-232, sendo implementado em linguagem C++, utilizando-se a plataforma de programação C++ Builder versão 6.0.

De acordo com STEIDLE NETO (2003), na plataforma C++ Builder, a programação é orientada a objeto, ou seja, enfatiza os dados manipulados. Neste tipo de programação, os procedimentos, as funções e toda a conformação do programa dependem de estruturas denominadas classes. As classes armazenam objetos que possuem atributos semelhantes. É possível afirmar que as classes, os objetos e os seus atributos são abstrações do mundo real e por meio deles é possível criar relações de forma hierárquica.

As vantagens da programação orientada a objetos envolvem a reutilização de códigos existentes de maneira facilitada e flexível, diminuição do tempo de desenvolvimento e minimização do número de procedimentos de verificação e validação.

Segundo ALMEIDA (2003), uma das grandes vantagens do Borland C++ Builder é o seu Ambiente integrado de Desenvolvimento ou simplesmente IDE (Integrated Development Environment). É através deste ambiente que o programador encontra todas as ferramentas para desempenhar as funções necessárias ao desenvolvimento de sua aplicação. Com ele, por exemplo, o programador poderá criar, executar, depurar e testar seu aplicativo sem necessidade de deixar o seu ambiente de desenvolvimento. A figura 10 exibe como é subdividido o programa.

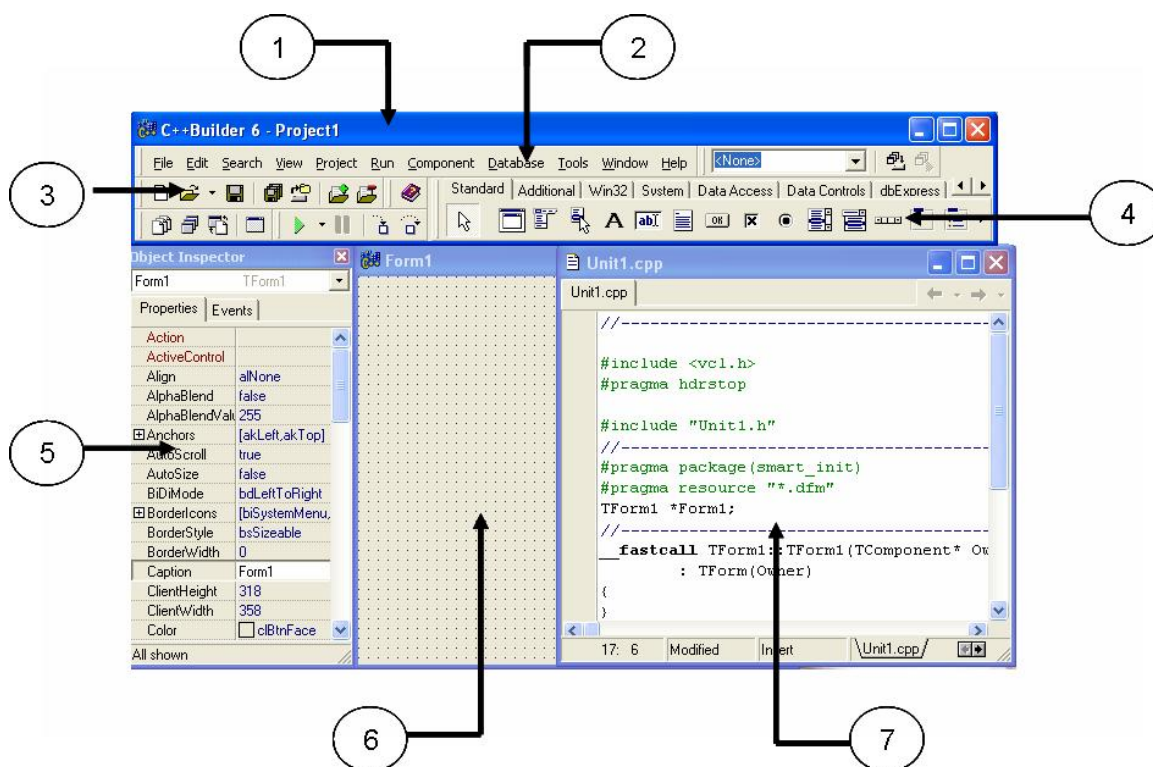


Figura 10 – Ambiente de desenvolvimento do C++ Builder

1. **Barra de títulos:** Exibe o nome do programa em execução, seguido do nome do projeto.
2. **Barra de Menus:** É onde encontra as opções através das quais se consegue acessar a maioria dos recursos do C++ Builder.
3. **Barra de Ferramentas:** Possui alguns botões que têm por finalidade a execução das tarefas mais repetitivas durante a criação de um projeto, como por exemplo: o salvamento e abertura de arquivos e projetos, a inclusão e execução de arquivos, a execução da aplicação, bem como a parte de depuração.
4. **Palheta de Componentes:** É a ferramenta mais utilizada na criação de uma aplicação, pois é nesta palheta que encontramos todos os componentes, que fazem parte do ambiente de desenvolvimento, agrupados de forma ordenada, lembrando um fichário. Assim, podemos selecionar a aba que desejamos acessar e em seguida o componente que será utilizado para construção de uma aplicação.

**5. Object Inspector:** Ou simplesmente Inspector de objetos é responsável pela alteração das propriedades e eventos associados a cada objeto de uma aplicação.

**6. Formulário:** É a base onde se constrói um aplicativo. Um formulário em C++ Builder possui as mesmas características que podem ser encontradas em qualquer outra janela do Windows. Ele possui um menu de controle e uma barra de títulos. É nesse formulário que inserimos os componentes que farão parte da interface com o usuário em uma aplicação.

**7. Janela de Edição:** É onde se encontram os códigos do programa, boa parte do código é inserido pelo próprio C++ Builder.

### 3. OBJETIVOS

Este trabalho foi desenvolvido com o intuito de alcançar dois tipos de objetivos, alguns genéricos e outros específicos.

Os objetivos genéricos são:

- Aprender a tecnologia de automação enfatizando a comunicação via porta serial do computador.
- Aprimorar os conhecimentos de programação em microcontroladores.

Os objetivos específicos são:

- Desenvolver um sistema de aquisição e monitoramento, utilizando microcontroladores da Microchip PIC 16F877A, e com transmissão dos dados via ondas de rádio.
- Desenvolver o sistema com baixo custo.
- Estudar a possibilidade de acoplar qualquer sensor que tem sinal de saída entre zero e cinco volts, tornando o sistema genérico e sem gastos adicionais com componentes, pois, todo ajuste das medições deve ser feito via software.

#### 4. MATERIAL E MÉTODOS

Este trabalho foi desenvolvido no laboratório de Eletrônica do Curso de Engenharia Elétrica do Departamento de Engenharia Elétrica e de Produção da Universidade Federal de Viçosa, MG.

O método pode ser descrito pelas seguintes operações: (1) Programa do microcontrolador, (i) Etapa de configuração, (ii) Etapa de aquisição, (iii) Etapa de transmissão, (2) Programa no C++ Builder, (i) Etapa de configuração, (ii) Etapa de recepção, (iii) Etapa de exibição, (iv) Etapa de gravação, (3) Conversão dos sensores e (4) Comparação entre componente Timer e processos Threads.

##### 4.1 Programa do Microcontrolador

O programa do microcontrolador foi feito na linguagem computacional C, utilizando funções para realizar as tarefas, pela maior facilidade de utilização e por proporcionar um maior reaproveitamento de código. O programa foi estruturado conforme visto na figura 11, com as seguintes etapas:

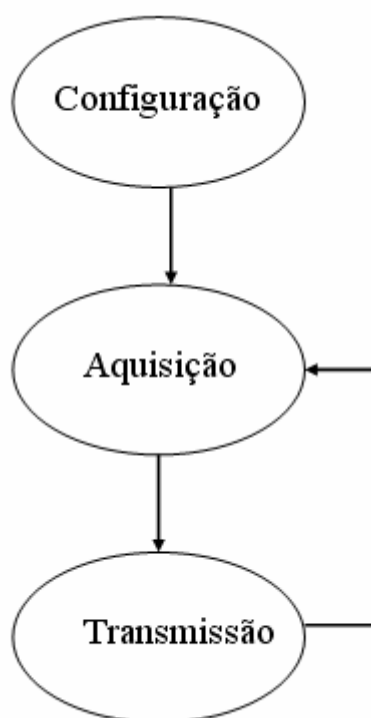


Figura 11 – Estrutura do programa do microcontrolador PIC 16F877

#### **4.1.1 Etapa de configuração:**

A etapa de configuração só ocorre na primeira vez que o programa do microcontrolador é executado. Sendo que, são configuradas as variáveis de endereço dos sensores, que são responsáveis por identificar qual sensor está transmitindo a informação e declarados quantos sensores serão utilizados.

Em seguida são definidas as variáveis referentes à conversão analógica para digital, a qual é feita com a precisão de dez bits, utilizando o clock externo com cristal de 4 MHz.

Também é inicializada a função usart, que é a responsável por fazer a transmissão dos dados, e definido a velocidade de transmissão para o computador, que é de 9600 bits por segundo.

#### **4.1.2 Etapa de aquisição:**

Faz parte da etapa cíclica do programa, ou seja, acontece toda vez que o programa executa sua rotina. Como o microcontrolador PIC 16F877 tem apenas um conversor analógico digital, tem-se a idéia de que só um pino dele pode ser usado para realizar a conversão analógica para digital. Mas, pode-se utilizar o conversor de forma multiplexada em qualquer um dos oito pinos do port A. Logo, podem-se colocar sensores nos oito pinos da porta A com apenas um conversor.

A função do programa de aquisição é definir o primeiro pino que será feita a aquisição e em seguida fazer a aquisição, mudar a configuração para o próximo pino e fazer a aquisição, repetindo o processo até que todos os pinos sejam lidos. Agora, com todos os endereços dos sensores configurados na etapa de configuração e todos os valores referentes a medições dos sensores efetuados na etapa de aquisição, basta transmití-los para o computador.

#### **4.1.3 Etapa de transmissão:**

O computador é rápido o suficiente para receber todos os valores enviados com a velocidade de 9600 bits por segundo, mas, ele não é inteligente o suficiente para saber de qual sensor é a medida que ele recebeu. Assim, a transmissão é feita enviando primeiro o endereço do sensor e em seguida o valor de sua medição, pois, no programa do computador cada sensor tem um local reservado de memória para ir acumulando os dados recebidos.

Ressalta-se que esta é a etapa crítica do sistema, pois, um erro pode causar uma má interpretação dos dados pelo usuário, e não pode ser reparado, pois, no próximo ciclo outros valores serão transmitidos. Isso ocorre pelo fato de o microcontrolador não armazenar os dados, simplesmente fazendo a leitura e transmitindo.

Utilizou-se de um modulador e transmissor FSK, cuja frequência da portadora foi de 914,5 Mhz. O microcontrolador envia o sinal para um pino do transmissor que tem a função embutida de modular e transmitir.

#### 4.2 Programa no C++ Builder

Como já foi descrito anteriormente o programa escolhido para fazer a interface com o usuário e a aquisição dos dados foi o C++ Builder. Para se desenvolver o programa etapas com menor complexidade foram feitas, principalmente para se entender melhor como funciona cada recurso utilizado e qual a melhor maneira de fazer a interação entre eles. A estrutura do programa está exibida na figura 12.



Figura 12 – Estrutura do programa desenvolvido no C++ Builder

#### **4.2.1 Etapa de configuração:**

A etapa de configuração realizada no computador é mais complexa que a realizada no microcontrolador. Como a aquisição é feita pela porta serial do computador deve-se primeiramente definir quais são as portas disponíveis e qual é a que está sendo utilizada. Após a definição da porta, devem-se configurar quais as características do sensor que está sendo utilizado. A metodologia para esta configuração será descrita de maneira detalhada posteriormente.

A próxima configuração é de como os dados serão recebidos, para isso, deve-se ter a mesma configuração do microcontrolador. Sendo configurado no começo do programa: a velocidade de recepção que é de 9600 bits por segundo, a quantidade de bits que serão recebidos, que pode variar de cinco a oito bits, a presença ou não de paridade, e quantos bits de parada.

Depois de configurado a porta serial e qual a característica dos bits, define-se como serão os arquivos onde os dados serão armazenados. Todos os arquivos para armazenamento de dados serão configurados para escrita e devem ser do tipo \*.txt.

#### **4.2.2 Etapa de recepção:**

Assim como na transmissão, na recepção um receptor e demodulador de 914,5 Mhz é utilizado em conjunto com o circuito integrado MAX232 que é ligado diretamente na porta serial do computador.

O programa como já descrito anteriormente, não pode “adivinhar” de quais sensores são os valores recebidos, assim, primeiro ele deve receber um endereço de sensor válido, e depois faz a aquisição do valor de medição do sensor. Sempre, a “chave” para o programa registrar um valor de medição do sensor é seu endereço.

Como no microcontrolador a aquisição é feita com o conversor analógico para digital de 10 bits e o máximo de bits que a porta serial do computador pode receber é de oito bits, não sendo possível pode fazer à aquisição de uma única vez, sendo necessário que a aquisição seja feita em duas etapas. Primeiro recebendo os oito primeiros bits e numa segunda aquisição recebendo os dois bits restantes completados de zeros. Uma conversão matemática é feita em seguida para se obter o número de dez bit transmitido novamente.



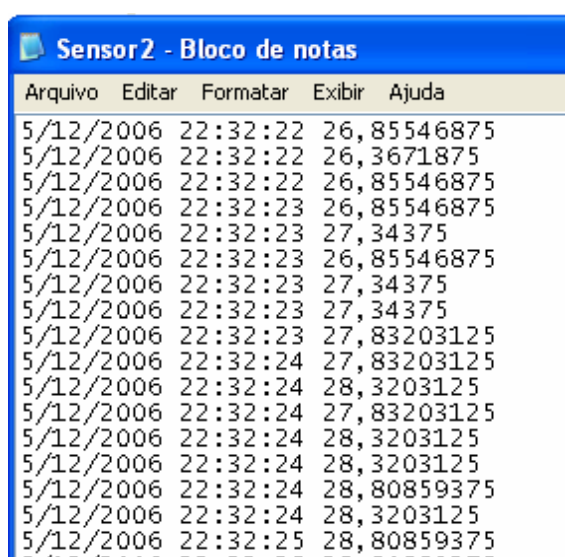
### 4.2.3 Etapa de exibição:

Como um dos objetivos do trabalho é analisar o comportamento de determinadas variáveis físicas, químicas ou biológicas por meio de sua evolução, é necessário que gráficos sejam gerados para se acompanhar o seu comportamento. O fato do C++ Builder não possuir bibliotecas prontas para a exibição de gráficos é um “entrave” á sua utilização. Para reverter este problema, uma biblioteca adicional foi instalada para permitir que se gerassem gráficos de maneira simples e dinâmica.

Logo após o valor de um sensor ser recebido, ele é exibido no seu gráfico tornando o processo de exibição de ponto a ponto.

### 4.2.4 Etapa de gravação:

Para comparações futuras e até mesmo utilizar outros programas como o Matlab para fazer análises dos dados, é de fundamental importância que os dados sejam arquivados. Como já mencionado anteriormente, os dados dos sensores são armazenados em arquivos do formato \*.txt. No programa foi definido que cada sensor possui seu arquivo e antes de começar a fazer a aquisição é necessário indicar um arquivo \*.txt para cada sensor utilizado. Além dos valores referentes à medição dos sensores serem gravados, é gravado a data com dia, mês e ano e também a hora da aquisição com hora, minuto e segundo. Na figura 13 se pode ver um exemplo de como os dados são gravados nos arquivos \*.txt .



Arquivo	Editar	Formatar	Exibir	Ajuda
5/12/2006 22:32:22	26,85546875			
5/12/2006 22:32:22	26,3671875			
5/12/2006 22:32:22	26,85546875			
5/12/2006 22:32:23	26,85546875			
5/12/2006 22:32:23	27,34375			
5/12/2006 22:32:23	26,85546875			
5/12/2006 22:32:23	27,34375			
5/12/2006 22:32:23	27,34375			
5/12/2006 22:32:23	27,83203125			
5/12/2006 22:32:24	27,83203125			
5/12/2006 22:32:24	28,3203125			
5/12/2006 22:32:24	27,83203125			
5/12/2006 22:32:24	28,3203125			
5/12/2006 22:32:24	28,3203125			
5/12/2006 22:32:24	28,80859375			
5/12/2006 22:32:24	28,3203125			
5/12/2006 22:32:25	28,80859375			

Figura 13 – Arquivo gravado pelo programa

### 4.3 Configuração dos sensores e Conversão do sinal

Como o microcontrolador faz a aquisição do sinal do sensor em um valor binário de 10 bits (VALORLIDO), e transmite para o computador, este tem a função de transformar o dado recebido no valor relativo à medida do sensor (VALORCONVERTIDO). Para realizar a conversão do valor do sinal binário de dez bits recebido para o valor real de medição, é feita a conversão baseada nos dados de configuração do sensor. Os dados são: valor máximo em volts (MAXVOLTS) do sensor e respectivo valor máximo de escala do sensor (MAXESCALA). Através desses dois dados, é feita a conversão do sinal binário recebido do microcontrolador, em sinal decimal representando o valor de medição. Para isso, define-se o fator de escala (FATORESCALA), que é dado pela equação 2.

$$\text{FATORESCALA} = 1024 \frac{\text{MAXVOLTS}}{5} \quad (\text{Equação 2})$$

A constante 1024 refere ao valor dos 10 bits, pois  $2^{10} = 1024$ . A constante 5 refere-se ao fator de referência da alimentação do microcontrolador que é de 5V. Com o fator de escala, obtém o VALORCONVERTIDO, conforme equação 3.

$$\text{VALORCONVERTIDO} = \frac{\text{VALORLIDO} * \text{MAXESCALA}}{\text{FATORESCALA}} \quad (\text{Equação 3})$$

Para entender melhor, o processo de conversão dos sinais binários obtidos pelo conversor analógico digital do microcontrolador para o valor referente à medição do sensor, toma-se como exemplo o sensor de temperatura LM35 com as seguintes características:

$$\text{MAXVOLTS} = 1\text{V} \text{ e } \text{MAXESCALA} = 100^{\circ}\text{C}$$

O fator de escala torna-se igual a:

$$\text{FATORESCALA} = 204,8$$

Assim, para uma leitura do conversor do microcontrolador como sendo igual a 70, por exemplo, tem-se como valor correspondente à temperatura de:

$$\text{VALORCONVERTIDO} = \frac{70 * 100}{204,8} = 34,179^{\circ}\text{C}$$

### 4.4 Comparação entre componente Timer e processos com Thread

Existem várias maneiras de resolver um problema de automação, podendo ser escolhido inúmeras opções desde o programa a ser usado até a linguagem de programação para resolver

o problema. Na literatura encontram-se aplicações de automação utilizando C++ Builder, com a ferramenta timer que é exibida na figura 14.

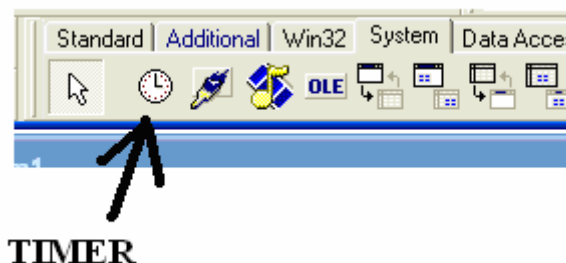


Figura 14 – Ferramenta Timer do C++ Builder

O objetivo do componente timer é cronometrar eventos em um intervalo regular de tempo. Os problemas trazidos com a utilização deste componente são basicamente dois. O primeiro é o fato de o programa parar o que está fazendo para executar a rotina programada. O segundo é que o programa gasta tempo atendendo a chamada causada pelo tempo estipulado no timer não sabendo se ocorreu ou não um evento como, por exemplo, a chegada de dados na porta serial do computador.

Para resolver esse problema optou-se por realizar uma programação baseada em threads. Segundo SEIXAS FILHO (2003), threads são processos com pesos leves. O conceito de thread surgiu a fim de propiciar maior grau de paralelismo nas ações executadas por um processo. Assim, evita-se parar uma rotina em execução para atender um evento que pode ou não acontecer, ganhando tempo e tornando o processo extremamente eficiente, pois, consegue-se capturar todos os sinais enviados a porta serial do computador, e enquanto nenhum sinal for recebido outras rotinas são executadas.

## 5. RESULTADOS E DISCUSSÕES

Os resultados do projeto proposto foram: o circuito de aquisição para transmissão de sinais serial, o programa desenvolvido para fazer a monitoração dos dados dos sensores, juntamente com a placa de recepção do sinal que é ligada ao computador.

Na construção do sistema, a parte de aquisição é extremamente simples por contar basicamente com um microcontrolador PIC 16F877, um cristal, dois capacitores, um resistor e um transmissor. Além dos sensores que podem ser ligados em oito pinos do microcontrolador.

O programa utilizado para fazer os testes antes da construção física das ligações foi o Proteus. Na figura 15, é exibido como é parte do ambiente de desenvolvimento e como são ligados os componentes. Verifica-se que o componente mais a direita da figura é um osciloscópio, sendo ele utilizado para saber se os dados estão sendo transmitidos. Na figura 16 é exibido a saída do osciloscópio no momento em que um sinal foi transmitido.

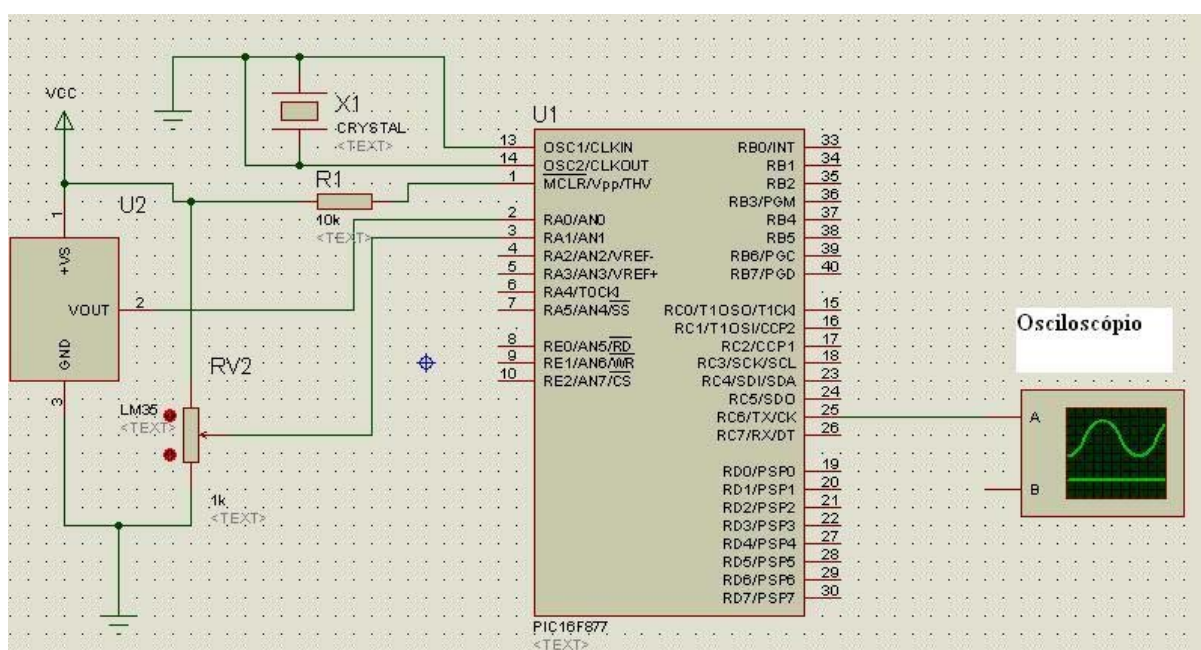


Figura 15 – Ambiente de desenvolvimento Proteus

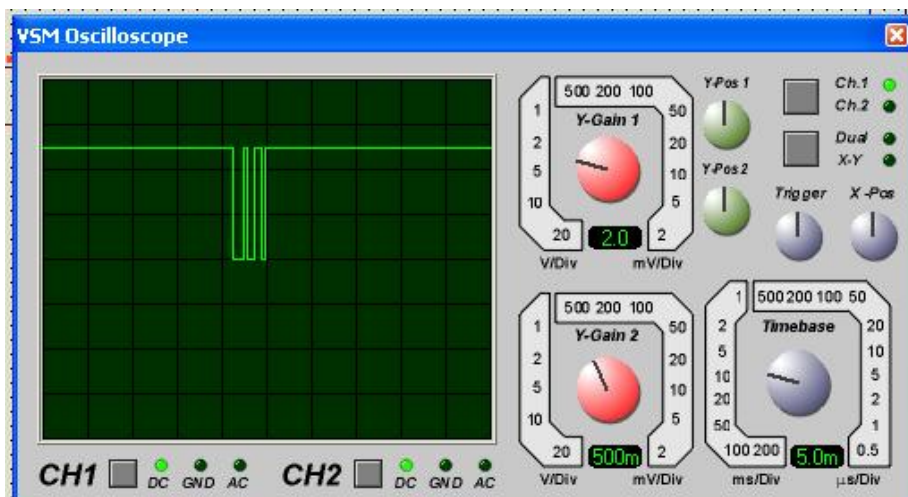


Figura 16 – Saída do pino de transmissão do microcontrolador.

A parte do circuito responsável por fazer a aquisição dos dados transmitidos é formada por um receptor, um circuito integrado MAX 232, quatro capacitores e o cabo de comunicação com a porta serial do computador. Na figura 17 é ilustrado o processo de comunicação com a porta serial do computador.

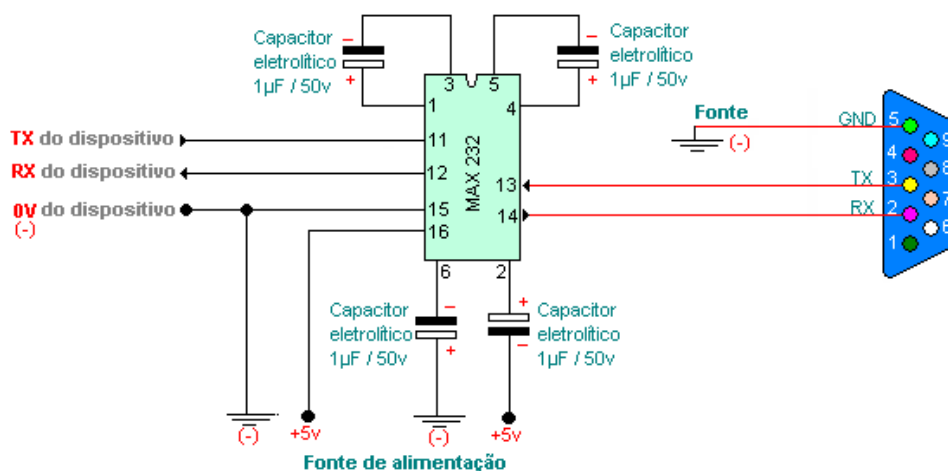


Figura 17 – Configuração da interface com a porta serial

Toda a parte de hardware desenvolvida anteriormente faz parte da aquisição do sinal para que o computador possa receber de maneira eficiente os dados relativos às medições dos sensores. Toda lógica desenvolvida, a partir do momento em que os dados são coletados podem ser facilmente modificadas através de linhas de programa. Assim, alterações podem

ser feitas sem implicar em alterações nas estruturas das placas de recepção, o que é extremamente vantajoso, pois, alterações no sistema não implicam em sua substituição.

A figura 18 mostra a tela principal do aplicativo desenvolvido, onde pode ser observado alguns dos recursos e funções do programa.

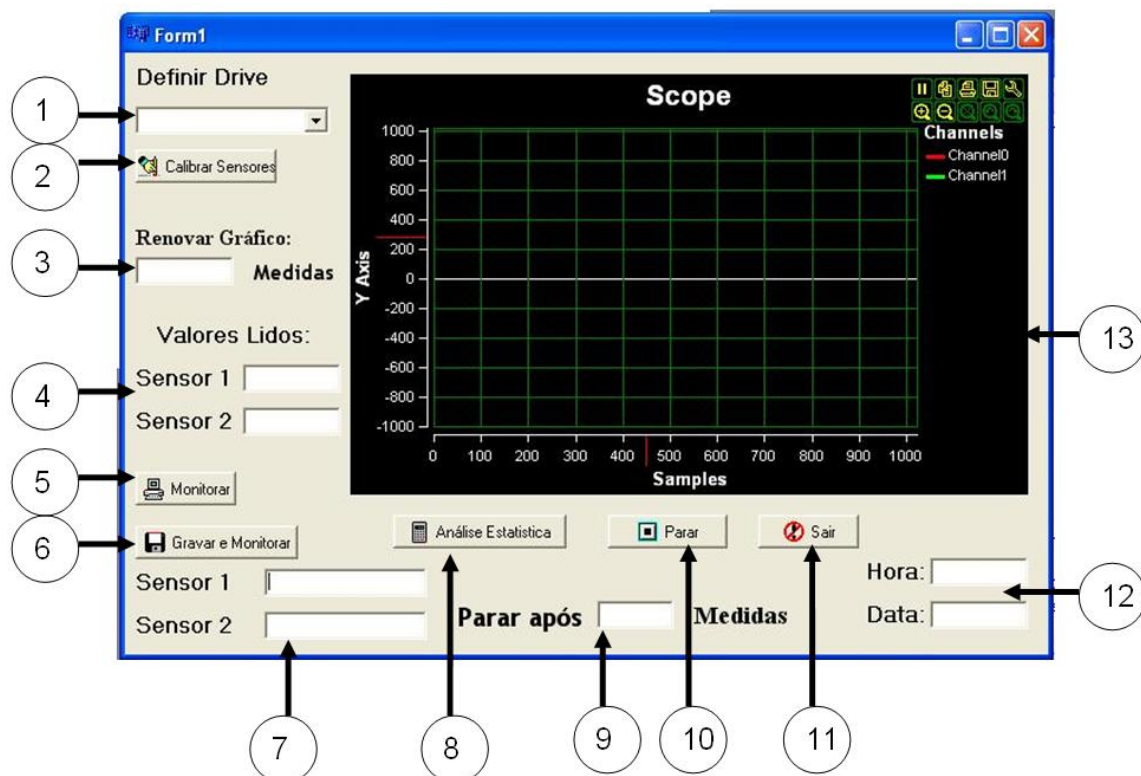


Figura 18 – Layout do programa desenvolvido

**1. Definir porta:** Um computador pode ter mais de uma porta serial, assim, quando se inicia o programa, uma função interna do programa faz a leitura das portas presentes e as exibe para escolha.

**2. Botão de Calibrar Sensor:** Este botão é responsável por abrir a tela de configuração dos sensores. A tela é exibida na figura 19.

**2.1-MAXESCALA:** Valor que faz parte do cálculo de conversão do valor binário para o valor de medição do sensor. Indica qual é o maior valor que o sensor pode medir em sua respectiva escala.

**2.2-MAXVOLTS:** Valor que faz parte do cálculo de conversão do valor binário para o valor de medição do sensor. Indica qual é o maior valor em volts que o sensor emite quando está em sua MAXESCALA.

**2.3-Botão voltar:** Depois de digitados os valores referentes à calibração do sensor, deve-se apertar o botão voltar para retornar a tela principal.

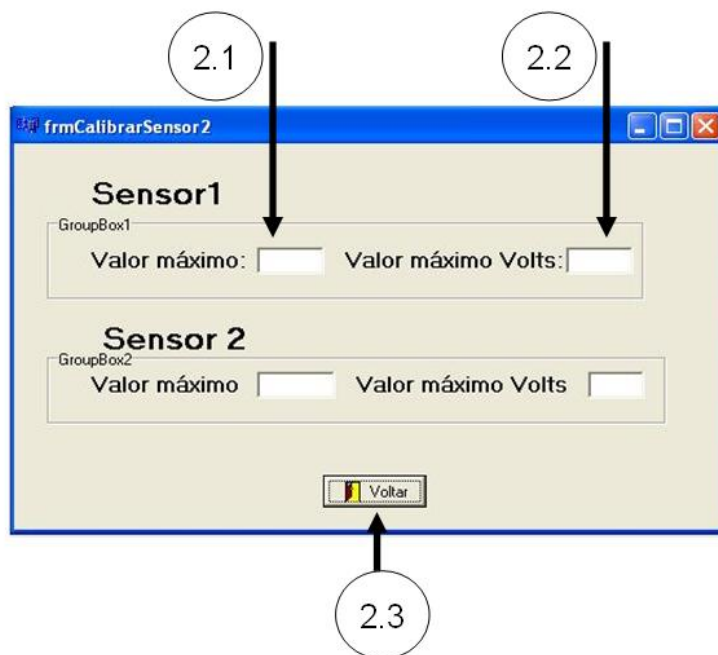


Figura 19 – Tela de calibração dos sensores

**3. Número de pontos exibidos no gráfico:** Para tornar o programa mais flexível, fica a critério do usuário definir até quantos pontos o eixo x, ou eixo da abscissa do gráfico, deve comportar. Por exemplo, se o usuário digitar 100, após 100 pontos coletados o gráfico será limpo e iniciado do ponto zero novamente, quando coletar mais 100 pontos, o gráfico será limpo novamente.

**4. Espaço de exibição dos valores recebidos:** Além da exibição gráfica o programa exibe em tempo real os valores coletados da comunicação com o microcontrolador.

**5. Botão Monitorar:** É o botão responsável por iniciar a recepção dos dados pela porta serial e exibir os valores dos sensores nos seus respectivos gráficos. Quando este botão é apertado uma thread é iniciada.

**6. Botão Gravar e Monitorar:** Apresenta as mesmas funções do botão Monitorar, acrescidas das funções de gravar os dados dos sensores em arquivos \*.txt. Para isso são abertas telas quando o botão é clicado, onde deve se escolher o arquivo em que os dados serão salvos. Além dos valores medidos nos sensores também é gravado a data e a hora da leitura dos dados.

**7. Espaço de abertura de arquivos:** Após indicar os arquivos onde serão gravados os dados, o caminho dos arquivos no diretório do computador fica exibido nos espaços indicados.

**8. Botão Análise Estatística:** É o responsável por exibir alguns parâmetros dos sensores. Quanto o botão é clicado a tela da figura 20 é exibida.

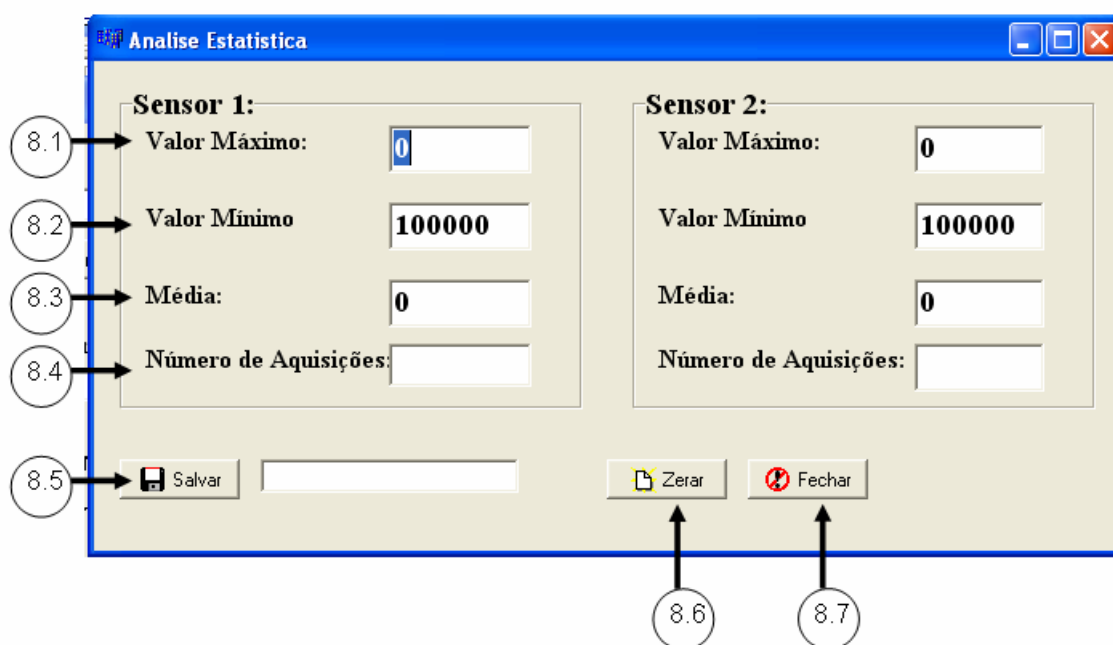


Figura 20 – Tela de Análise Estatística

**8.1-Valor Máximo:** A cada aquisição o programa compara o valor recebido com o máximo recebido. Caso o valor recebido seja maior, ele substitui o valor máximo pelo valor recebido.

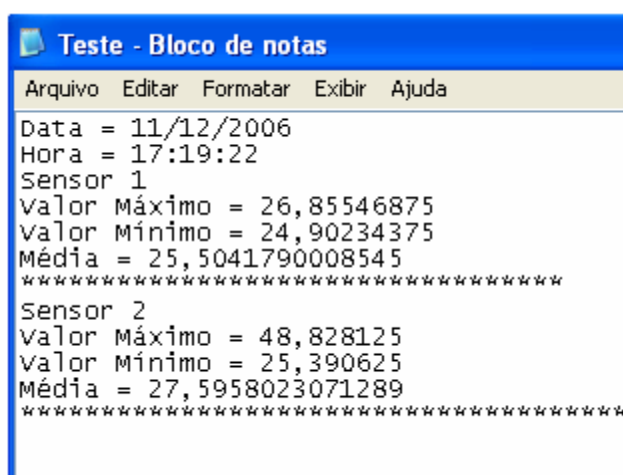
**8.2-Valor Mínimo:** A cada aquisição o programa compara o valor recebido com o mínimo recebido. Caso o valor recebido seja menor, ele substitui o valor mínimo pelo valor recebido.



**8.3-Média:** A cada aquisição o programa calcula a média de todos os valores recebidos.

**8.4-Número de Amostras:** A cada aquisição o programa incrementa o número de amostras recebidas e a exibe.

**8.5-Botão Salvar:** Caso em algum momento da aquisição o usuário queira salvar os valores listados anteriormente, basta clicar no botão Salvar e indicar um arquivo do formato \*.txt para que o programe grave os valores. Na figura 21 é exibido como os dados são salvos.



```

Teste - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda
Data = 11/12/2006
Hora = 17:19:22
Sensor 1
Valor Máximo = 26,85546875
Valor Mínimo = 24,90234375
Média = 25,5041790008545
*****
Sensor 2
Valor Máximo = 48,828125
Valor Mínimo = 25,390625
Média = 27,5958023071289
*****

```

Figura 21 – Formato do arquivo salvo

**8.6-Botão Fechar:** Fecha a tela Análise Estatística.

**9. Para após coletar amostras:** Caso o usuário queira fazer uma aquisição de um número específico de dados, basta digitar a quantidade no campo especificado. Quando a quantidade de dados for coletado o programa para a execução. Caso a opção for deixada em branco o programa faz aquisições de maneira continua até ser fechado ou o botão Parar for clicado.

**10. Botão Parar:** É responsável por parar a execução da thread.

**11. Botão Sair:** Fecha a tela principal e encerra todos os recursos utilizados pelo programa.

**12. Exibição de Data e Hora:** Além de ser gravadas nos arquivos a data e a hora ficam em exibição para tornar o aplicativo com mais funcionalidades.

**13. Gráfico de exibição das curvas:** É a parte da tela do aplicativo onde são exibidos os gráficos de cada sensor. Esta aplicação não é disponível no C++ Builder, sendo encontrada na internet, devendo ser instalada como um aplicativo adicional.

Para entender melhor o funcionamento do programa, foi feito um exemplo de uso detalhado de como utilizar o programa e como são as características do programa em execução. Foram utilizados dois sensores de temperatura LM 35, um para medir a temperatura do ambiente e outro para medir a temperatura do corpo humano como se fosse um termômetro.

O aplicativo foi aberto aparecendo à tela da figura 20. O primeiro passo foi escolher a porta serial de comunicação, que no computador onde foi realizado o teste foi a porta serial COM1. Depois se optou pela opção Calibrar Sensores, aparecendo à tela da figura 22.



A imagem mostra a interface de usuário do aplicativo 'frmCalibrarSensor2'. O título da janela é 'frmCalibrarSensor2'. O conteúdo da tela é dividido em duas seções principais, 'Sensor 1' e 'Sensor 2', cada uma contendo um grupo de controles (GroupBox).  
- Na seção 'Sensor 1', há um grupo de controles 'GroupBox1' com dois campos de entrada: 'Valor máximo:' com o valor '100' e 'Valor máximo Volts:' com o valor '1'.  
- Na seção 'Sensor 2', há um grupo de controles 'GroupBox2' com dois campos de entrada: 'Valor máximo' com o valor '100' e 'Valor máximo Volts' com o valor '1'.  
- Na parte inferior da tela, há um botão com o texto 'Voltar' e um ícone de uma seta amarela apontando para a esquerda.

Figura 22 – Exemplo de calibração dos sensores

Como já mostrado anteriormente em um exemplo, o máximo valor medido pelo sensor LM35 é de 100 °C e com máximo valor de 1 V. Como os dois sensores são iguais, as configurações são iguais. Após entrar com os dados clicou-se no botão voltar que levou de volta para a tela principal.

Na tela principal foi apertado o botão Gravar e Monitorar, abrindo a tela da figura 23. Foi escolhido o arquivo de texto Sensor1 e depois o arquivo Sensor2. Esses arquivos são para gravar os valores medidos pelos sensores, a data e a hora da medição.

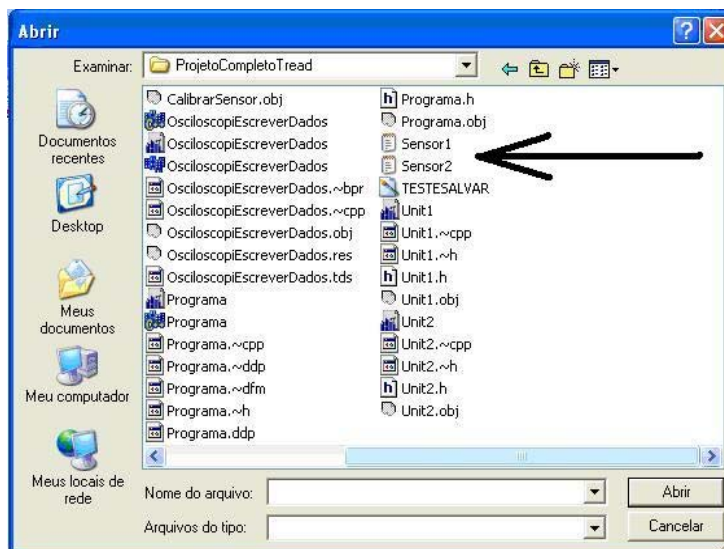


Figura 23 – Escolha do arquivo para gravação dos dados

Assim que é feita a escolha dos arquivos para gravação dos dados, uma thread é iniciada e o programa começa a executar. Na tela da figura 24 é mostrado o arquivo em execução.

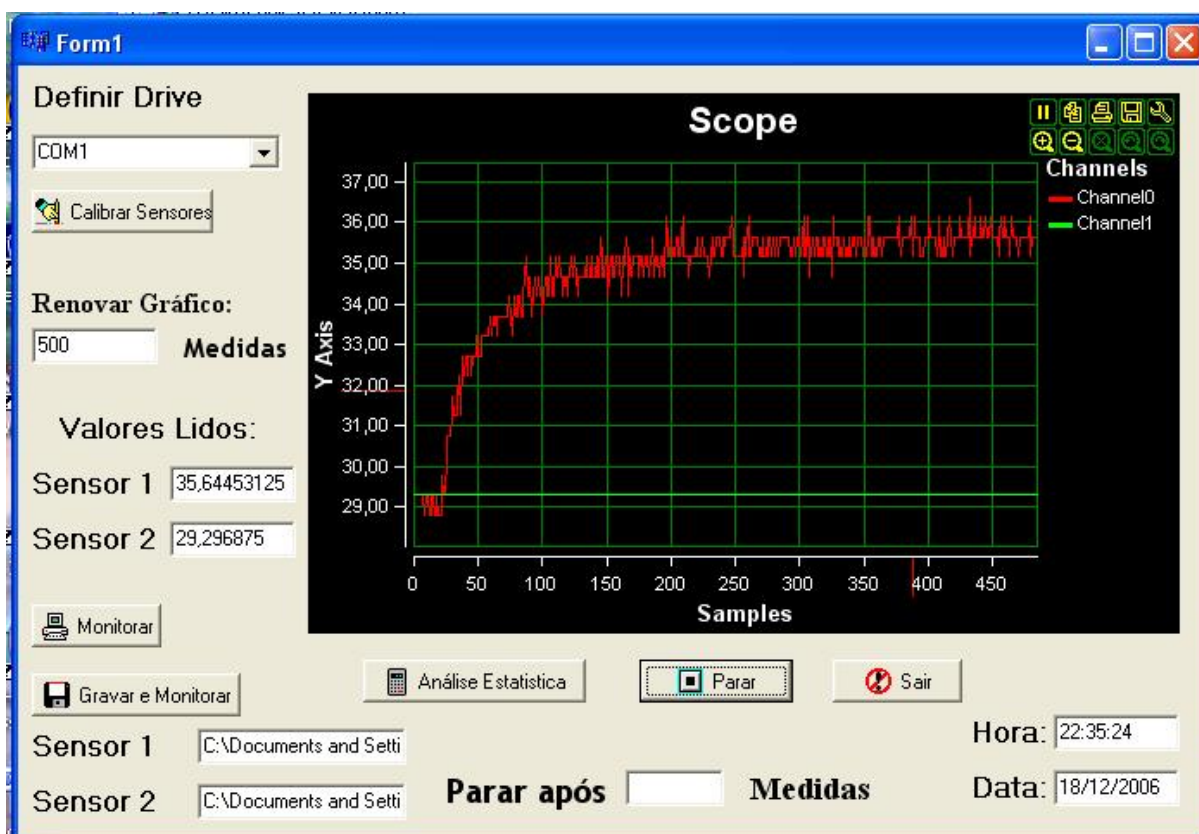


Figura 24 – Programa em funcionamento com dois sensores de temperatura

Observa-se que a temperatura ambiente indicado pelo canal 1 em verde teve medições em torno de 29°C. Com o outro sensor, canal 0 em vermelho, colocado debaixo das axilas de uma pessoa, no momento em que começou a ser executado o programa, atingiu em torno de 36°C com 485 aquisições. Este valor pode ser obtido clicando no botão, Análise Estatística, que exhibe a tela da figura 25.

Como o sensor se encontrava na temperatura ambiente no momento em que foi colocado para fazer a medição da temperatura do corpo humano, e o tempo em contato com o corpo foi pequeno, a temperatura média ficou em 34,6 °C, o que pode parecer um erro, mas, a média leva em considerações todas as medidas inclusive quando a temperatura era a ambiente.

As oscilações no gráfico da temperatura da figura 24 são devidas ao fato de o sensor ter um tempo de acomodação e como a taxa de leitura foi muito alta, na ordem de seis aquisições por segundo, o sinal sofreu influência desse tempo de acomodação que provocou o aumento da sua oscilação. Uma maneira de resolver o problema da oscilação do sinal é diminuir a taxa de leitura o que tornaria o gráfico mais suave.

The screenshot shows a window titled "Analise Estatistica" with a blue title bar. The window is divided into two columns, "Sensor 1:" and "Sensor 2:". Each column contains four data points in a table-like format. At the bottom, there are three buttons: "Salvar" (Save), "Zerar" (Reset), and "Fechar" (Close). The "Fechar" button has a red 'X' icon.

Sensor 1:		Sensor 2:	
Valor Máximo:	36,621093	Valor Máximo:	29,29687
Valor Mínimo:	28,320312	Valor Mínimo:	29,29687
Média:	34,689067	Média:	29,29687
Número de Aquisições:	485	Número de Aquisições:	485

Buttons: Salvar, Zerar, Fechar

Figura 25 – Tela Análise Estatística do exemplo anterior

### Mensagens de erro

O programa foi feito de maneira a gerar mensagens de erro para o usuário caso ele esqueça de algum detalhe na configuração do programa. Caso um usuário aperte o botão para Monitorar sem configurar a porta serial será exibida a tela da figura 26.

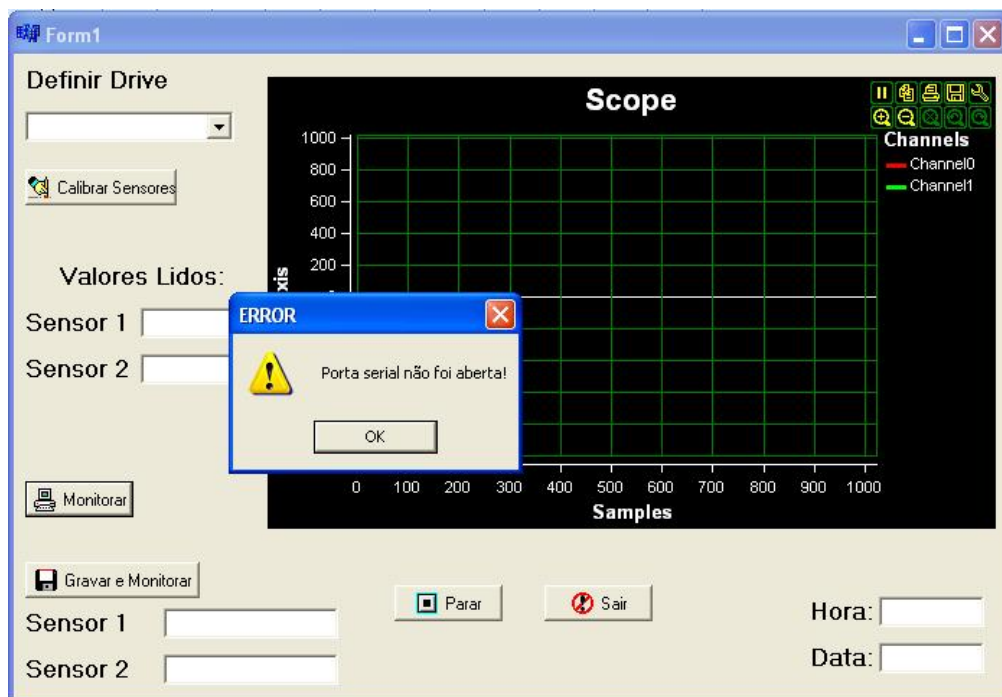


Figura 26 – Erro de não definição da porta serial

Outro erro possível é esquecer de configurar os sensores e apertar o botão para Monitorar ou o botão Gravar e Monitorar. A seguinte mensagem de erro é exibida na figura 27.

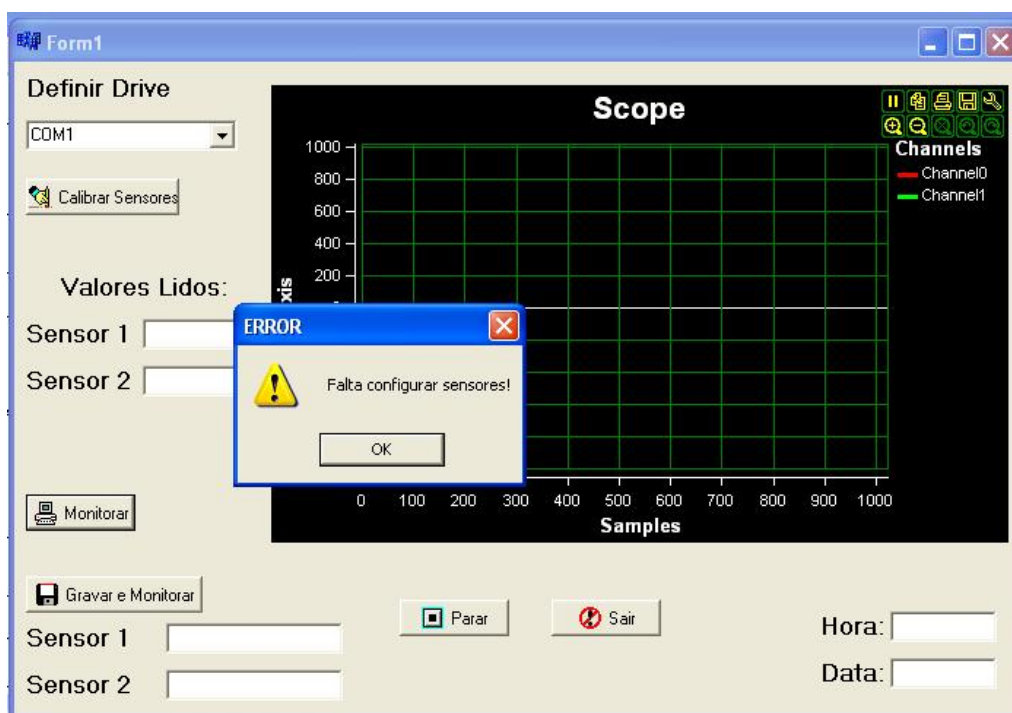


Figura 27 – Erro de não configuração dos sensores

Caso se configure os sensores, depois se indique qual a porta serial certa para fazer a aquisição e aperte o botão Gravar e Monitorar, não indicando os arquivos onde serão salvos os valores medidos, aparecerá a mensagem de erro mostrada na figura 28.

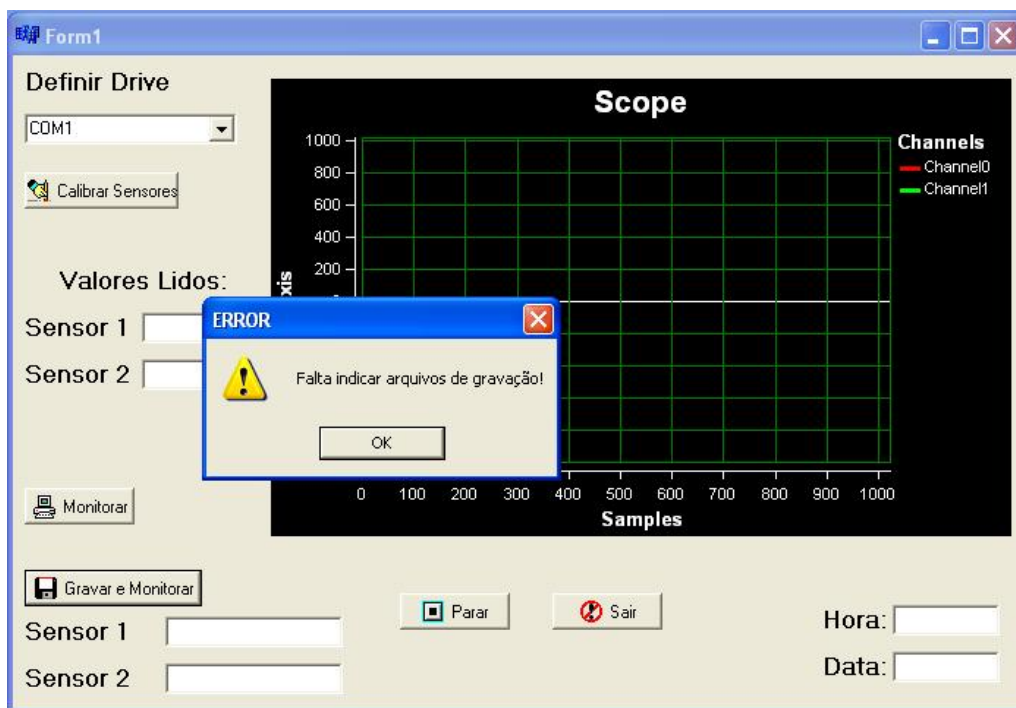


Figura 28 – Erro de não indicação dos arquivos para gravação dos dados

Com essas mensagens de erro alertando os possíveis erros na utilização do programa, tornar-se muito difícil esquecer de algum detalhe. Não se pode esquecer, no entanto de saber as características dos sensores que estão sendo utilizados, pois, um valor errado na configuração implicará em valores errados de exibição e gravação.

## 6. CONCLUSÕES

Após a finalização das etapas traçadas durante o desenvolvimento do trabalho, foi possível perceber que com o sistema desenvolvido muitas outras funcionalidades podem ser acrescentadas, para melhorar o sistema em termos de flexibilidade e recursos. O domínio da tecnologia de automação através da ligação entre microcontroladores programáveis e computador foi realizado de maneira eficiente, tanto em termos de velocidade quanto em termos de qualidade.

O microcontroladore PIC 16F877 da Microchip mostrou-se um excelente dispositivo para realizar a aquisição de sensores e a transmissão de sinais de forma serial. Através de funções simples como a do conversor analógico para digital, pôde-se capturar o dado de maneira rápida e com precisão de dez bits, o que é bem vantajoso quando se trabalha com sensores com faixa de sinal entre zero e cinco volts.

Outra grande vantagem do sistema é o baixo custo envolvido e a simplicidade das ligações entre os componentes. O módulo de aquisição e transmissão tem custo aproximado de R\$ 40 (quarenta reais) mais os sensores que se quer acoplar. O modulo de recepção possui custo aproximado de R\$ 22 (vinte e dois reais). O programa desenvolvido não teve custo materiais, tendo apenas o custo humano gasto no desenvolvimento das funções.

Uma vantagem que deve sempre estar presente em sistemas de comunicação, monitoramento ou aquisição é a flexibilidade e a possibilidade de fazer incrementos no sistema sem ter que fazer a sua substituição. Esta característica um ponto forte no sistema desenvolvido.

Comparações com outros sistemas de aquisição e monitoramento de dados podem ser realizadas, mas, no mercado a maioria dos sistemas encontrados é de uma robustez e de um preço muito elevado, sendo difícil estabelecer critérios para comparar qual é o custo benefício de um sistema simples e barato para comparar com um sistema caro e muitas vezes complexo.

## **7. SUGESTÕES PARA TRABALHOS FUTUROS**

No sistema atual o microcontrolador só transmite dados ao computador e não recebe nenhuma informação do computador, ou seja, a comunicação é unidirecional o microcontrolador transmite e o computador recebe. Para que o sistema ganhe mais flexibilidade é necessário que o microcontrolador trabalhe em função do computador e para isso ocorrer é necessário fazer a comunicação bidirecional para que o microcontrolador seja capaz de receber dados do computador. Assim, o computador poderia processar os dados recebidos e enviar comandos de controle como acionar uma válvula ou um motor. Podendo com a comunicação bidirecional desenvolver um sistema centralizado, de maneira que através de uma cabine central o operador pudesse controlar todas as suas variáveis, ou até mesmo, programando-as para que situações indesejadas não ocorressem.

Desenvolvimentos de antenas para a comunicação de maneira mais eficiente também é de vital importância, pois, como o sistema é sem fio, uma boa comunicação é fundamental para que os dados cheguem de maneira íntegra ao destino.

No sistema atual os sensores utilizados devem ter resposta na faixa de zero a cinco volts. Outras metodologias podem ser desenvolvidas para que outros sensores também possam ser utilizados, de maneira a ampliar a utilização do sistema tornando-o mais flexível e conseqüentemente mais vantajoso para o uso.



## 8. REFERÊNCIAS BIBLIOGRÁFICAS

- ALMEIDA, Waldir de. Conhecendo o C++ Builder 6, Editora Visual Books, Florianópolis, 2003.
- C++ BUILDER. Disponível: <http://www.mitov.com> [capturado em 02 out. 2006].
- CURTIS, S. E. Environmental management in animal agriculture. AMES: The Iowa University, 1983.
- DALLY, J. W.; WILLIAM, F. R.; McCONNELL, K. G. Instrumentation for engineering measurements. New York: John Wiley & Sons, 1993.
- GOMES, Alcides Tadeu. Telecomunicações: Transmissão e Recepção: AM-FM, Editora Érica, 14ª Edição, Taubaté, 1998.
- HAYKIN, Simon. Sistemas de Comunicação. Tradução: José Carlos Barbosa dos Santos. Editora Bookman, 4ª Edição, Porto Alegre, 2004.
- MATEUS, César Augusto. C++ Builder 5-Guia Prático, Editora Érica, 2ª Edição, Tatuapé, 2000.
- NETO, Walter Goldberg. Utilizando C++ BUILDER, Editora Érica, São Paulo, 1997.
- PEREIRA, Fábio. Microcontroladores PIC, Programação em C. Editora Érica, 3ª Edição, Tatuapé, 2004.
- PORTA SERIAL. Disponível: <http://www.rogercom.com> [capturado em 10 set. 2006].
- SEIXAS FILHO, Constantino; SZUSTER, Marcelo. Programação Concorrente em Ambiente Windows. Editora UFMG, Belo Horizonte, 2003.
- STEIDLE NETO, A. J. Avaliação do sistema 1-wire para aquisição de dados de temperatura em instalações agrícolas. Tese de Mestrado apresentada à Universidade Federal de Viçosa, Viçosa, 2003.