UNIVERSIDADE

ESTADUAL DE MARINGÁ

PÓS-GRADUAÇÃO EM FÍSICA

Andre Luiz de Lima Ponzoni

Desenvolvimento de um Sistema para Caracterização de Sensores de Gases

Orientadora: Profª. Drª. Suzana Nóbrega de Medeiros
Co-orientador: Prof. Dr. Jusmar Valentin Bellini

Maringá - PR
2007

UNIVERSIDADE

ESTADUAL DE MARINGÁ

PÓS-GRADUAÇÃO EM FÍSICA

Andre Luiz de Lima Ponzoni

Dissertação apresentada à Universidade Estadual de Maringá para obtenção do título de mestre.

Orientadora: Profª. Drª. Suzana Nóbrega de Medeiros
Co-orientador: Prof. Dr. Jusmar Valentin Beliini

Maringá - PR
2007

DEDICO ESTE TRABALHO

À minha esposa Raquel pelo amor, incentivo e por estar sempre ao meu lado,

ao meu sobrinho e afilhado Gabriel,

e aos meus pais Celito e Jeanine pelo amor e exemplo de vida.

AGRADECIMENTOS

Ao professor Dr. Jusmar Valentin Bellini pela orientação, apoio e incentivo desde o início da graduação.

À professora Drª. Suzana Nóbrega de Medeiros pela orientação e atenção durante todo o mestrado.

Ao professor Dr. Antônio Carlos Saraiva da Costa e ao Msc. Ivan Granemann de Souza Jr. pela ajuda na realização da liofilização e da espectroscopia de absorção atômica.

À CAPES pela bolsa concedida.

Ao pessoal da oficina mecânica pela ajuda com a montagem física do sistema.

Agradeço também a todas as pessoas que de alguma forma ajudaram no desenvolvimento deste trabalho.

Pra mim, é muito melhor
compreender o universo como ele
realmente é do que persistir no
engano, por mais satisfatório e
tranqüilizador que possa parecer.
(Carl Sagan)

RESUMO

Um sistema para caracterização de amostras por termoeletroresistometria (TER) foi desenvolvido, incluindo-se também o código fonte do programa de aquisição de dados para tal sistema. TER consiste na medida da resistência elétrica em função da temperatura. O sistema foi testado para amostras constituídas de misturas de óxidos obtidas após tratamentos térmicos de pós liofilizados. Misturas de pós de hematita (Fe_2O_3) e acetato de manganês tetrahidratado [$(CH_3COO)_2Mn{\cdot}4H_2O$], com uma razão molar dos cátions Fe:Mn ≅ 2:1, foram obtidas por liofilização. Os pós liofilizados foram tratados termicamente em ar por duas horas, em três diferentes temperaturas. Para a amostra tratada termicamente a 400 ºC/2h, os espectros de difração de raios X (DRX) indicaram a presença das fases hematita e hausmanita (Mn_3O_4). Para as amostras tratadas a 700 e 1000 ºC, DRX indicou a presença das fases óxido de manganês (Mn_2O_3) e hematita, e óxido de manganês ferro ($FeMnO_3$) e hematita, respectivamente. TER mostrou que a amostra tratada a 400 ºC é sensível ao gás metano (CH_4). Porém, esta sensibilidade só ocorre na condição específica de baixa pressão atmosférica e na presença exclusiva do gás.

ABSTRACT

A system for samples characterization by thermoelectroresistometry (TER) was developed, including the code for the acquisition data program for such system. TER consists on the measure of the electric resistance in function of the temperature. The system was tested for samples of oxides mixtures achieved after a thermal treatment of the freeze-dried powders. Mixtures of powders of hematite (Fe_2O_3) and manganese acetate tetra-hydrated [$(CH_3COO)_2Mn{\cdot}4H_2O$], with a molar ratio for cations of Fe:Mn ≅ 2:1, were obtained by freeze-drying. The freeze-dried powders were thermal treated at 400 ºC/2h, the X-ray diffraction (XRD) spectra indicate the presence of hematite and hausmanite (Mn_3O_4). For the samples treated at 700 and 1000 ºC, XRD indicated the presence of manganese oxide (Mn_2O_3) and hematite, and iron manganese oxide ($FeMnO_3$) and hematite phases, respectively. TER showed that the sample treated at 400 ºC is sensitive to methane gas (CH_4). Nevertheless, the sensitivity only occurs in the specific condition of low atmospheric pressure and in the exclusively presence of the gas.

SUMÁRIO

1 INTRODUÇÃO

Atualmente sensores de gás constituem uma ferramenta necessária para a detecção e controle de gases tóxicos e inflamáveis. O uso de sensores em refinarias de petróleo, indústrias químicas e siderúrgicas já é bem estabelecido, entretanto recentemente tem-se percebido um gradativo aumento do uso dos mesmos no comércio, e até mesmo em modernas residências. Esta demanda por sensores de gás tem incentivado a pesquisa na área, a qual vem sendo desenvolvida com o intuito de obter novos tipos de sensores, visando sempre melhoria na qualidade e baixo custo.

Existem diferentes tipos de sensores de gás, os mais comuns são sensores eletroquímicos, catalíticos, infravermelhos, de foto-ionização e do estado sólido. Vários são os tipos de gases detectados por esses sensores, em especial o dióxido de nitrogênio (NO_2), o monóxido de carbono (CO) e o dióxido de enxofre, os quais são oriundos de fábricas (papel e cimento) e da queima do diesel, os quais são os maiores fatores de poluição da atmosfera. Na literatura, há principalmente progressos no desenvolvimento de sensores do estado sólido, os quais são fabricados a partir de materiais nanoestruturados a base de metal-óxidos, mistura de óxidos e ferritas.

O objetivo deste trabalho é a construção e automatização de um aparato experimental para medidas de termoeletroresistometria com o intuito de testar novos sensores do estado sólido. Nos vários testes realizados foram utilizados sensores obtidos da mistura aquosa de hematita com acetato de manganês tetrahidratado via liofilização, os quais basicamente consistem de misturas de óxidos.

Secagem por liofilização trata-se de uma técnica eficiente para desidratação de misturas aquosas, obtendo um produto final homogêneo e livre de impurezas. É uma técnica muito utilizada na ciência, nas indústrias farmacêuticas e até alimentícias. A liofilização foi empregada neste trabalho para obter uma mistura de óxido, com qualidade acima mencionada, entre os precursores utilizados.

Por questões didáticas, o presente trabalho foi dividido em 5 capítulos.

Uma revisão bibliográfica sobre sensores de gás e síntese por liofilização é abordada no capítulo 2. No capítulo 3 explica-se detalhadamente à montagem do sistema, a construção do software e a preparação das amostras. Os resultados obtidos e discussão são apresentados no capítulo 4. E o capítulo 5 é dedicado às conclusões.

2 REVISÃO DA LITERATURA

2.1 SENSORES DE GASES

Um canário dentro de uma mina de carvão, caindo do seu poleiro, historicamente indicava aos mineiros que suas vidas estavam ameaçadas por metano, monóxido de carbono e outros gases tóxicos, mas sem cheiro (AZEVEDO-RAMOS *et. al.*, 2003).

Nos anos recentes, diferentes tipos de gases têm sido usados em diferentes áreas. De fato, muitos gases industriais tornaram-se importantes também como matérias primas. Por esta razão, entre outras, tornou-se muito importante desenvolver detectores altamente sensíveis a gases para prevenir acidentes devido a vazamento de gás, e assim, salvar vidas e equipamentos. Tais detectores devem continuamente monitorar a concentração de um gás particular no ambiente de maneira quantitativa e seletiva.

Até hoje nenhum sensor de gás existente é 100% seletivo a um único gás. Para alcançar tal seletividade requer-se o uso de instrumentos que empregam técnicas analíticas para identificar os gases. Contudo estes instrumentos requerem operadores altamente qualificados e geralmente são muito caros, além de ter um tamanho grande e possuir um lento tempo de resposta.

Um sensor de gás é um dispositivo que detecta moléculas de gases e produz um sinal elétrico com magnitude proporcional à concentração do gás. Essa leitura da concentração de um ou mais gases será feita de acordo com a necessidade do local, ou seja, o equipamento pode medir quantitativa ou qualitativamente a concentração de um único gás ou de diversos gases ao mesmo tempo (CHOU, 2000).

2.1.1 Tipos de sensores de gases

Existem diferentes tecnologias e equipamentos atualmente utilizados para a detecção de gases, cada qual com suas vantagens e desvantagens. Grande parte dos sensores comercialmente disponíveis é destinada à detecção e quantificação de contaminantes do ar.

Os tipos de sensores mais adequados e largamente utilizados para proteção contra gases combustíveis e tóxicos na área da qualidade do ar e segurança são apresentados a seguir: eletroquímicos; catalíticos de gases combustíveis; infravermelhos; de fotionização; do estado sólido (CHOU, 2000).

Uma característica comum destes sensores é que eles não são especializados em detectar qualquer gás específico. Cada sensor é sensível a um grupo ou família de gás. Em outras palavras, ele é sujeito a interferência de outros gases, por exemplo, um detector de fumaça em uma casa não consegue distinguir entre a fumaça causada pela queima de um móvel e a fumaça causada pelo cozimento de alimentos em um fogão. Em casos restritos um filtro químico pode ser instalado para filtrar interferências químicas, permitindo que apenas o gás desejado passe para o sensor.

2.1.1.1 Sensores eletroquímicos

Os sensores eletroquímicos mais antigos datam da década de 50 e eram utilizados para monitorar níveis de oxigênio. Eram extremamente grandes e pesados. Mais recentemente, foram desenvolvidos instrumentos melhores e menores, alguns deles portáteis, com uma seletividade e sensibilidade mais adequada, utilizados para segurança de diversos ambientes em diferentes edificações. Diferentes sensores eletroquímicos podem parecer muito similares, mas são construídos com materiais diferentes, que incluem elementos críticos, tais como: eletrodos de detecção, composição do eletrólito e porosidade das membranas.

Os sensores eletroquímicos operam reagindo com o gás a ser detectado, produzindo um sinal elétrico proporcional à concentração do gás (CHOU, 2000). Um sensor eletroquímico típico consiste em um eletrodo de detecção e um eletrodo reagente separados por uma fina camada de eletrólito (Figura 2.1). O gás que entra no sensor reage com a superfície do eletrodo reagente envolvendo um mecanismo de oxidação e redução. Com o resistor conectado transversalmente entre os eletrodos, uma corrente proporcional à concentração de gás passa entre o anodo e o catodo (Figura 2.2). A corrente pode ser medida para determinar a concentração de gás.

Mudanças de pressão afetam muito pouco os sensores eletroquímicos, por outro lado, os mesmos são bastante sensíveis a variações na temperatura. Portanto, é importante manter o aparelho dentro de uma pressão e temperatura estáveis.

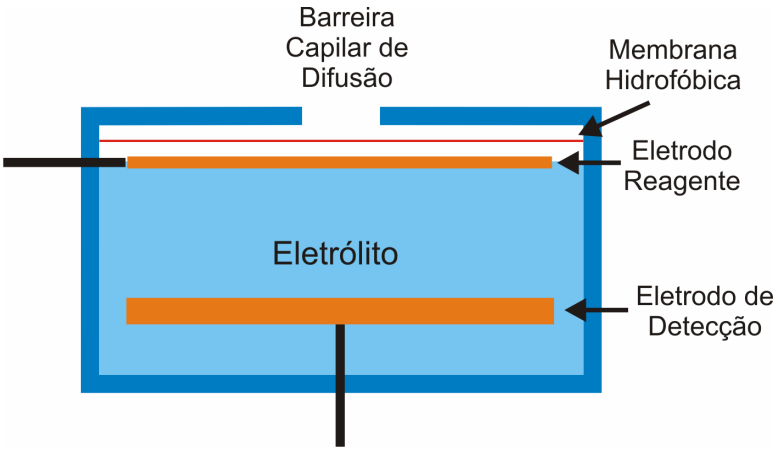

Figura 2.1 - Esquema de um sensor eletroquímico típico.

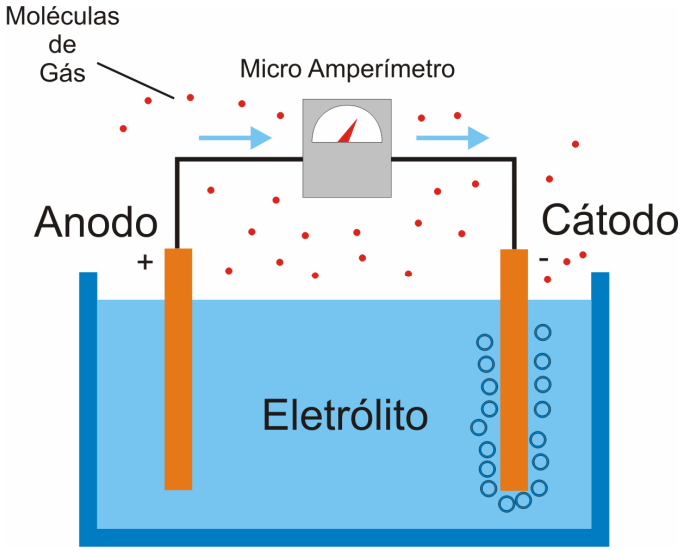

Figura 2.2 - Esquema do mecanismo de oxidação e redução.

Embora existam diversos modelos de sensores eletroquímicos, para detecção dos mais diversos gases, eles consistem basicamente dos seguintes componentes: i) *barreira permeável ao gás* (membrana hidrofóbica) (Figura 2.3): é usada para cobrir o eletrodo de detecção do sensor, e em alguns casos controla a quantia de moléculas de gás que alcançam a superfície do eletrodo. O material mais utilizado é o PTFE (Poli Tetra Flúor Etileno); ii) *eletrodo*: a seleção do material do eletrodo é muito importante e depende do gás a ser detectado, normalmente são utilizados metais nobres como platina ou ouro; iii) *eletrólito*: o eletrólito deve facilitar a

reação e transportar a carga iônica de maneira eficaz através dos eletrodos; iv) *filtro*: na maioria das vezes é instalado na frente do sensor e serve para filtrar gases externos indesejados deixando atravessar apenas o gás de interesse (CHOU, 2000).

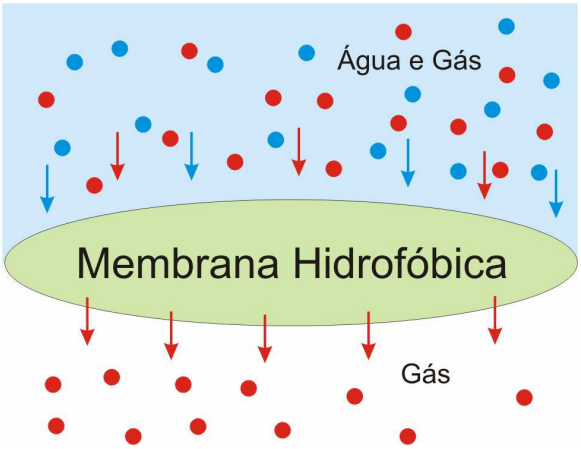

Figura 2.3 - Membrana hidrofóbica.

2.1.1.2 Sensores catalíticos de gases combustíveis

Os sensores catalíticos detectam unicamente gases combustíveis e foram utilizados por mais de 50 anos. Inicialmente foram utilizados para monitorar o gás em minas de carvão, onde substituíram os canários que estavam em uso por um longo período. O próprio sensor é bastante simples e fácil de ser fabricado, em sua forma mais primitiva, utilizando um único arame de platina.

Misturas gasosas combustíveis não queimam até que alcancem sua temperatura de ignição. Porém, na presença de certas condições químicas, o gás começa a inflamar em temperaturas mais baixas. Esse fenômeno é conhecido como combustão catalítica (CHOU, 2000). Materiais eletricamente condutores mudam sua condutividade com mudanças de temperatura. O coeficiente de resistência de temperatura (C_t) é a mudança da condutividade por grau de variação da temperatura. A platina é um dos materiais mais utilizados em sensores catalíticos por apresentar um grande C_t em comparação a outros metais. Além de ter propriedades químicas e mecânicas excelentes, a platina é resistente à corrosão e pode ser operada em altas temperaturas por um longo período de tempo. Produz também um sinal seguro e proporcional à concentração do gás no ambiente (CHOU, 2000).

Os sensores catalíticos são compostos por dois filamentos de platina: um filamento ativo e o de referência. Quando a mistura combustível entra em contato com o filamento ativo, os gases combustíveis da mistura são queimados cataliticamente. Isto provoca um aumento da temperatura no detector e consequentemente uma mudança na resistência elétrica.

Há vários fatores que afetam a operação e a confiabilidade dos sensores catalíticos: a) *contaminação do catalisador*: existem substâncias químicas que desativam o sensor, como as que contêm silício. Elas causam a perda de sensibilidade do sensor e eventualmente este não responderá mais à presença de gases no ambiente; b) *inibidores do sensor*: substâncias químicas como combinações de halogênios utilizados em extintores de incêndio, inibem o sensor catalítico e causam perda temporária de funcionamento do sensor. Normalmente, após 24 horas de exposição ao ar o sensor volta a funcionar novamente; c) *deterioração do sensor*: a exposição em excessivas concentrações de gases, bem como aquecimento excessivo podem inutilizar o sensor catalítico.

2.1.1.3 Sensores infravermelhos

Analisadores infravermelho de gás tem a reputação de serem complicados, incômodos e caros, apesar de sua tecnologia bem desenvolvida. Entretanto, recentes avanços tecnológicos abriram uma nova fronteira para análise infravermelha do gás. Estes avanços resultaram em um aumento na demanda no setor comercial e esta, provavelmente, continuará provendo o avanço desta tecnologia (CHOU, 2000).

Os gases a serem detectados são frequentemente corrosivos e reativos. Na maioria dos sensores, o próprio sensor é exposto diretamente ao gás, o que, frequentemente, causa perda parcial ou total das habilidades do sensor de modo prematuro. A principal vantagem dos sensores infravermelhos (IR) é que o sensor não entra em contato direto com o gás (ou gases) a ser detectado. Seus componentes funcionais ficam protegidos, e somente um fluxo luminoso interage com as moléculas de gás.

Em geral, para gases tóxicos e combustíveis os instrumentos de detecção infravermelhos são os mais utilizados e requerem menos manutenção. Os sensores infravermelhos são altamente seletivos e oferecem uma ampla gama de sensibilidades, desde algumas partes por milhão até cem por cento de concentração do gás.

O principio de detecção infravermelho incorpora somente uma pequena porção de um amplo espectro eletromagnético. A complexidade das moléculas do gás determina o número

de pontos de absorção da radiação. Quanto mais átomos formam uma molécula, maior a faixa de absorção que esta terá. A região na qual esta absorção acontece, a quantidade absorvida, e o caráter específico da curva de absorção são únicos para cada gás. As moléculas de gás podem ser identificadas usando suas características e estas podem ser arquivadas para análise do gás para propósitos de identificação. Uma biblioteca dessas curvas pode ser armazenada na memória do instrumento, assim quando um determinado gás é identificado pelo detector, seu gráfico é comparado com as curvas armazenadas na memória do equipamento para identificar as moléculas do gás.

Quando a radiação interage com moléculas de gás, parte da energia tem a mesma freqüência que a freqüência natural das moléculas do gás e é absorvida enquanto o resto da radiação é transmitida. Como as moléculas de gás absorvem esta radiação elas ganham energia e vibram mais vigorosamente. Esta vibração resulta em uma elevação na temperatura das moléculas do gás. A temperatura aumenta em proporção com a concentração do gás, e é então detectada pelo sensor (CHOU, 2000).

O detector infravermelho é essencialmente um sensor de temperatura e é, então, potencialmente sensível a mudanças na temperatura do local em que está instalado. Porém, ele trabalha normalmente bem em temperaturas ambientes, pois estas variam lentamente. A umidade ambiente tem muito pouco efeito sobre o detector; entretanto, altas umidades podem ocasionar erros. Os detectores infravermelhos têm uma longa expectativa de vida útil, superior a cinco anos.

Há muitos outros tipos de detectores infravermelhos e cada um deles oferece uma extensiva gama de características.

2.1.1.4 Sensores de gases por fotionização

Os detectores de fotionização (PID) utilizam iluminação ultravioleta pra ionizar moléculas de gás, é normalmente empregado na detecção de combinações orgânicas voláteis. O coração do detector de fotionização é a fonte ultravioleta. Nos anos 80 com o advento da tecnologia do circuito integrado, foram desenvolvidas técnicas para processar e melhorar o pequeno sinal fornecido pelo sensor em dados úteis e seguros, utilizáveis para detectar a presença de um gás específico.

A principal falha destes instrumentos é que a fonte luminosa requer limpeza freqüente, isto porque ela é exposta diretamente ao fluxo do gás. A limpeza influenciará diretamente na

leitura do sensor, ou seja, os resultados num sensor com lâmpada suja serão muito diferentes dos obtidos em outro com ela limpa. Devido a isto, e ainda por ter uma expectativa de vida relativamente pequena, aproximadamente 6000 horas, estes sensores não são utilizados na monitoração de gases que exigem uma leitura constante, normalmente eles são utilizados em aplicações que requerem leituras periódicas (CHOU, 2000).

A fonte luminosa, também chamada de fonte de eletrodos, do detector de fotionização, contrariando o nome, não tem eletrodos, e sim está cheia de um gás inerte a baixa pressão. Quando o gás da fonte luminosa é energizado, com energia em ressonância com a freqüência natural das moléculas do gás, uma radiação espectral ultravioleta é produzida. Um par de eletrodos fica localizado próximo da janela da lâmpada, de onde a luz ultravioleta é emitida. Conforme as moléculas do gás a ser detectado movem-se em direção da janela da lâmpada, elas são ionizadas e os elétrons livres são coletados pelos eletrodos, resultando em uma corrente que é proporcional à concentração do gás (Figura 2.4).

Pelo fato dos sensores de fotionização serem suscetíveis à umidade é recomendável calibrar o sensor nas mesmas condições a que ele será submetido trabalhar, caso contrário, o mesmo sensor pode apresentar respostas diferentes quando exposto à ambientes distintos (CHOU, 2000).

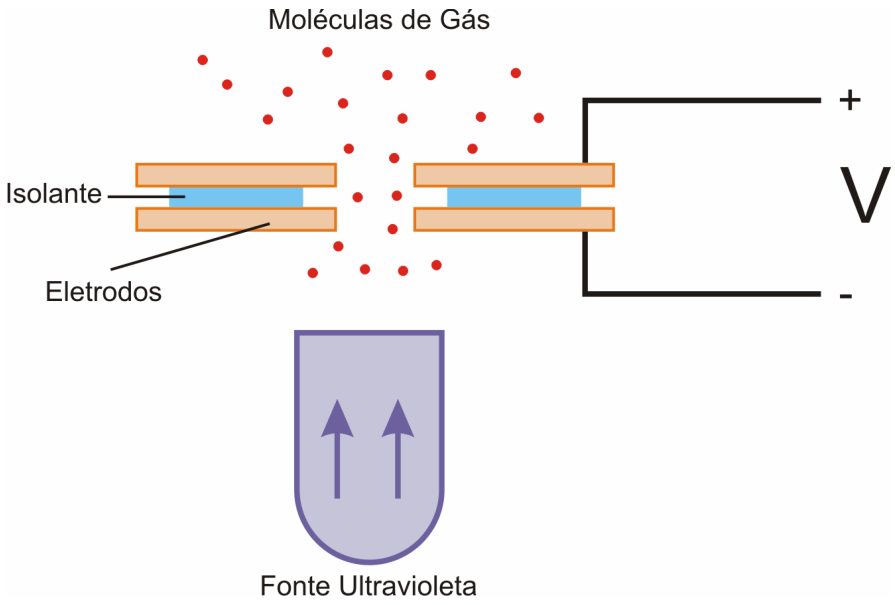

Figura 2.4 - Configuração típica de um detector de fotionização.

2.1.1.5 Sensores de gases de estado sólido

Enquanto os cientistas trabalhavam em pesquisas relacionadas às junções *p-n* em semicondutores, descobriram que elas eram sensíveis aos gases do ambiente. Esse comportamento foi a principio considerado um problema. Este problema, contudo, foi solucionado encapsulando o chip do semicondutor de maneira que não ficasse mais exposto ao ambiente. Subsequentemente, tentativas sem sucesso foram realizadas para utilizar a sensibilidade do semicondutor de junção como um detector de gás.

Em 1968, N. Taguchi desenvolveu um simples semicondutor, ou um sensor de estado sólido para a detecção de hidrocarbonetos. A intenção era fornecer uma alternativa ao popular sensor catalítico, que sofria de vários problemas, incluindo perda da sensibilidade com o tempo e envenenamento do sensor quando exposto em grandes concentrações de gás (CHOU, 2000).

Em 1972 a IST (International Sensor Technology) em Irvine, Califórnia introduziu um sensor de estado sólido para a detecção de sulfeto de hidrogênio em uma escala de 0 – 10 ppm. Alguns anos mais tarde, a IST desenvolveu um sensor do estado sólido para a detecção de mais de 100 diferentes tipos de gases tóxicos em baixos níveis de ppm (Figura 2.5).

Hoje, os sensores de estado sólido estão disponíveis para a detecção de mais de 150 gases diferentes, incluindo sensores para os gases que de outra maneira só poderiam ser detectados usando caros instrumentos analíticos. Existem agora diversos fabricantes de sensores do estado sólido, mas cada sensor tem características diferentes e os diversos fabricantes oferecem diferentes níveis de desempenho e qualidade. Corretamente fabricados, os sensores do estado sólido possuem uma longa expectativa de vida. Não é difícil encontrar sensores completamente funcionais que foram instalados há mais de 30 anos.

No começo dos anos 80, o Japão aprovou uma lei que requeria a instalação de detectores de gás em apartamentos residenciais onde eram usados botijões de gases. Para esse enorme mercado, a competição se travou entre sensores do estado sólido e sensores catalíticos.

Figura 2.5 - Sensor de estado sólido usado para detectar mais de 100 gases tóxicos (CHOU, 2000).

As reclamações iniciais a respeito dos alarmes falsos produzidos pelos sensores do estado sólido foram compensadas pela longa expectativa de vida do sensor. Os sensores catalíticos queimam o gás a ser detectado, eventualmente mudando as características ou ainda queimando o sensor (CHOU, 2000). Os sensores do estado sólido, por outro lado, simplesmente absorvem o gás na superfície do sensor, mudando a resistência elétrica do material usado como sensor. Quando o gás some, o sensor volta a sua condição original. O material do sensor não é consumido no processo e, por isso, oferece uma longa vida útil. Após alguns anos de uso, os sensores catalíticos, perderam popularidade para tais aplicações devido a freqüente necessidade de troca dos sensores.

Um sensor de estado sólido consiste em um ou mais óxidos de metal dos metais de transição, tais como óxido de estanho, óxido de alumínio, etc. Um elemento de aquecimento é usado para regular a temperatura do sensor. Então o sensor é elevado a altas temperaturas que determinam as características finais do sensor. Na presença de gás, o óxido metálico dissocia o gás em íons carregados que resultam na transferência de elétrons. A fonte de aquecimento interna que aquece o sensor, a uma temperatura operacional para o gás ser detectado, é regulada e controlada por um circuito específico (CHOU, 2000).

Um par de eletrodos é introduzido no óxido metálico para medir sua mudança de condutividade. Esta mudança no sensor é resultado da interação do óxido metálico com as moléculas de gás. Normalmente um sensor de estado sólido produz um sinal muito forte, especialmente em altas concentrações de gás. Os sensores do estado sólido estão entre os mais versáteis, pelo fato de detectarem uma grande variedade de gases, e poderem ser usados em diversas aplicações. Diferentes características de resposta são obtidas variando-se os materiais do semicondutor, a temperatura operacional do sensor e as técnicas de processamento.

Dentre as qualidades dos sensores de estado sólido, estão as habilidades para detectar tanto baixos como altos níveis de concentração de gases, além de ter uma probabilidade de vida útil longa, 10 anos ou mais. Esta é a maior vantagem se comparado com outros tipos de sensores, tais como, sensores catalíticos ou eletroquímicos que geralmente duram apenas um ou dois anos. Porém esses sensores são mais suscetíveis a gases de interferência que outros tipos de sensores. Assim, em aplicações onde outros gases estão presentes, este sensor pode ativar falsos alarmes. Estas interferências podem ser minimizadas usando-se filtros apropriados que absorvem todos os outros gases, exceto o gás a ser detectado. Por exemplo, um sensor do estado sólido para monitoração de monitoração de hidrogênio e monóxido de carbono pode ser equipado com um filtro de carvão vegetal que elimina a maioria dos gases de interferência. Desta maneira o sensor trabalha muito bem e torna-se muito seletivo para estes dois gases (CHOU, 2000).

Outra grande vantagem do sensor é sua versatilidade. Frequentemente, as mais baixas concentrações de um gás precisam ser monitoradas, enquanto que simultaneamente o mesmo gás precisa ser monitorado em concentrações mais elevadas. O sensor de estado sólido é capaz de detectar ambas as concentrações, isto simplifica o sistema e a manutenção porque elimina ou minimiza o uso de tecnologias de sensores múltiplos, os quais devem ser projetados e mantidos diferentemente.

2.1.1.5.1 Óxidos como sensores de gases de estado sólido

Óxidos são frequentemente utilizados como sensores de gases do estado sólido. Uma boa parte dos óxidos utilizados como sensor faz uso da hematita (Fe_2O_3) misturada com algum outro material, podendo ser outro óxido. Ferritas são óxidos importantes que também são utilizados como sensores de gás. A mais utilizada é a ferrita de níquel, porém também se encontram na literatura trabalhos feitos com ferritas de cobre, cobalto, zinco e bismuto. Não existem referências publicadas sobre o uso da ferrita de manganês como sensor de gás.

Os métodos utilizados para misturar estes materiais são os mais diversos possíveis, como moagem de alta energia, co-precipitação, sol-gel e etc.

A sensibilidade S de um sensor é definida (Eq. 2.1) como a razão entre ΔR e R_a.

$$S = \frac{\Delta R}{R_a} = \frac{\left(\left| R_a - R_g \right| \right)}{R_a} \qquad (2.1)$$

Onde R_a é o valor de resistência elétrica medido em ar, R_g o valor de resistência elétrica medido com gás e ΔR é a diferença entre R_a e R_g (GOPAL REDDY *et al.*, 1999).

Em 1999, C. V. Gopal Reddy e colaboradores publicaram que a ferrita de níquel preparada por co-precipitação mostrou boa sensibilidade para detecção de gás cloro (GOPAL REDDY *et al.*, 1999). L. Satyanarayana também estudou a ferrita de níquel como sensor. A ferrita de níquel, preparada por síntese hidrotermal, quando dopada de paládio, mostrou melhoras na sensibilidade, tempo de resposta e seletividade para o gás GLP (SATYANARAYANA *et al*., 2003).

No ano 2000, Shanwen Tao e colaboradores pesquisaram sobre a ferrita de cobre produzida pelo método da co-precipitação. Dos gases testados: C_2H_5OH, gasolina, C_2H_2, H_2, CO e GLP, o álcool (C_2H_5OH) mostrou a melhor resposta, enquanto H_2 e CO mostraram as menores respostas entre os gases testados. A diferença na resposta para os vários gases testados pode ser atribuída a reação entre os gases e o oxigênio que foi absorvido durante a preparação da amostra (TAO *et al*., 2000).

XINGQIN LIU *et. al.* (1998) desenvolveram um novo sensor de etanol baseado no sistema de óxidos semicondutores $CdO\text{-}Fe_2O_3$. As amostras foram preparadas por co-precipitação e foi estudada a performance do sensor, incluindo sensibilidade, seletividade, estabilidade, bem como, tempo de resposta e de restabelecimento. Os resultados mostraram que todas as características do sensor são excelentes, o que indica um bom sensor para etanol. O sensor mostrou boa seletividade para o etanol contra outros gases comuns que podem coexistir com o etanol, tais como H_2, CO, GLP e outros gases de hidrocarbonetos.

TAN *et. al.* (2004) estudaram a aplicação, como sensor de etanol, da solução sólida, preparada por moagem de alta energia, de $\alpha\text{-}Fe_2O_3$ misturada separadamente com diferentes moles de SnO_2, ZrO_2 e TiO_2, que foram sintetizados em filmes finos. No caso do sensor de etanol, os materiais $\alpha\text{-}Fe_2O_3\text{–}SnO_2$ e $\alpha\text{-}Fe_2O_3\text{–}ZrO_2$ mostraram-se os mais adequados, enquanto que para o sensor de gás o material $ZrO_2\text{–}\alpha Fe_2O_3$ é útil para aplicações em temperaturas baixas.

ARSHAK & GAIDAN (2006) também estudaram filmes finos como sensores de gases. O trabalho deles investigou o uso de NiO/Fe_2O_3 como um filme fino sensor de gás em temperatura ambiente. Várias razões de NiO e Fe_2O_3 foram utilizadas. Os filmes foram usados

para detectar metanol, etanol, propano, tolueno e acetona em temperatura ambiente. Todos os sensores mostraram maior sensibilidade para o tolueno, seguido do propano. Foi descoberto que um elemento de aquecimento comumente usado em óxidos como sensores, não era necessário. Assim, esses sensores podem ser particularmente empregados na monitoração de gases, visando uma melhora na segurança de ambientes.

JING (2006[a]) estudou as propriedades do sensor de gás de γ-Fe_2O_3 dopado com Ni e preparado por método químico. Os resultados revelaram que ambas amostras, não dopada e γ-Fe_2O_3 dopada com 15 mol % de Ni exibiram boa sensibilidade para acetona e etanol e baixa sensibilidade para CH_4, H_2 e CO na temperatura de 270 ºC. Comparada com a amostra não dopada, a amostra dopada com 15 mol % de Ni apresenta maior sensibilidade, menor tempo de resposta e de restabelecimento e melhor estabilidade a longo prazo.

JING (2006[b]) estudou o comportamento do sensor de gás de γ-Fe_2O_3 dopada e não dopada com Mg, sintetizada por método químico. A amostra foi testada para os gases C_2H_5OH, CH_4, NH_3, H_2 e CO. Ele percebeu que o sensor de γ-Fe_2O_3 dopado com 30 %-mol Mg apresentou a maior sensibilidade para o gás C_2H_5OH, mas baixa sensibilidade para os outras gases testados, na temperatura de 270 ºC. Comparado com o não dopado, a melhora no desempenho do sensor de gás, tal como sensibilidade, seletividade e tempo de resposta é principalmente atribuída a adição de Mg.

2.2 SÍNTESE POR LIOFILIZAÇÃO E TERMOELETRORESISTOMETRIA

O método de secagem por liofilização tem sido utilizado para a preparação de precursores de óxidos metálicos. Normalmente, misturas aquosas de sais solúveis em água como citratos, nitratos ou acetatos são primeiramente liofilizados e em seguida são tratados termicamente em temperaturas acima daquela onde os sais se decompõem para seus respectivos óxidos. Esta técnica denomina-se síntese por liofilização. Mais especificamente, os acetatos metálicos são sais solúveis em água e facilmente liofilizáveis. Geralmente, esses sais têm fórmula geral $(CH_3COO)_2M{\cdot}xH_2O$, onde M pode ser algum dos cátions metálicos como, por exemplo, Co^{2+}, Mn^{2+}, Cu^{2+}, Zn^{2+}, etc. e x corresponde ao número de moléculas de água de cristalização. Para fins práticos esses acetatos são denotados por $Ac_2M{\cdot}xH_2O$. Assim, para $Ac_2M{\cdot}xH_2O$ (M = Co^{2+} ou Mn^{2+}) eles são denominados acetato de Co ou Mn

tetrahidratado (x = 4) e para (M = Cu ou Zn) eles são denominados de acetato de Cu ou Zn monohidratado (x = 1).

A síntese por liofilização permite a obtenção de misturas de pós de óxidos metálicos altamente homogêneos, livres de contaminação por impurezas e altamente reativos. Com esta técnica, BELLINI (2001) e BELLINI *et al* (2002[a,b], 2003[a,b]) obtiveram misturas homogêneas de pós de ZnO + $Ac_2Cu{\cdot}H_2O$ + (frita de vidro) para a produção de varistores de ZnO e para a produção de cerâmicas de ZnO dopados com Cu (ZnO:Cu). Mais recentemente, BELLINI *et al.* (2007) sintetizaram ferritas de manganês a partir de pós liofilizados de Fe_2O_3 + $Ac_2Mn{\cdot}2H_2O$ tratados termicamente em atmosfera de N_2.

Termoeletroresistometria (TER) é uma técnica de caracterização que consiste na medida da resistência elétrica de um material em função da sua temperatura. Por exemplo, esta técnica em conjunto com medidas de calorimetria diferencial de varredura (DSC) e termogravimetria (TG) permite o estudo dos eventos térmicos que ocorrem em um material sensível à temperatura. Recentemente, esta técnica foi utilizada por BELLINI *et al.* (2006) para o cálculo da energia de ativação para a decomposição térmica de $Ac_2Cu{\cdot}H_2O$ utilizando-se misturas de pós liofilizados de ZnO + $Ac_2Cu{\cdot}H_2O$. TER pode ser considerada como uma técnica complementar de análise térmica.

3 PROCEDIMENTO EXPERIMENTAL

3.1 MONTAGEM FÍSICA DO SISTEMA

3.1.1 Controle de temperatura

O aquecimento da amostra, bem como o controle da temperatura em rampas e patamares é feito utilizando-se uma resistência elétrica de 127 V e 400 W de potência. A resistência elétrica está dentro de um cilindro metálico de 135 mm de comprimento e 26 mm de diâmetro (Figura 3.1 a). Para a maximização da condução térmica foi utilizado cobre na fabricação deste cilindro.

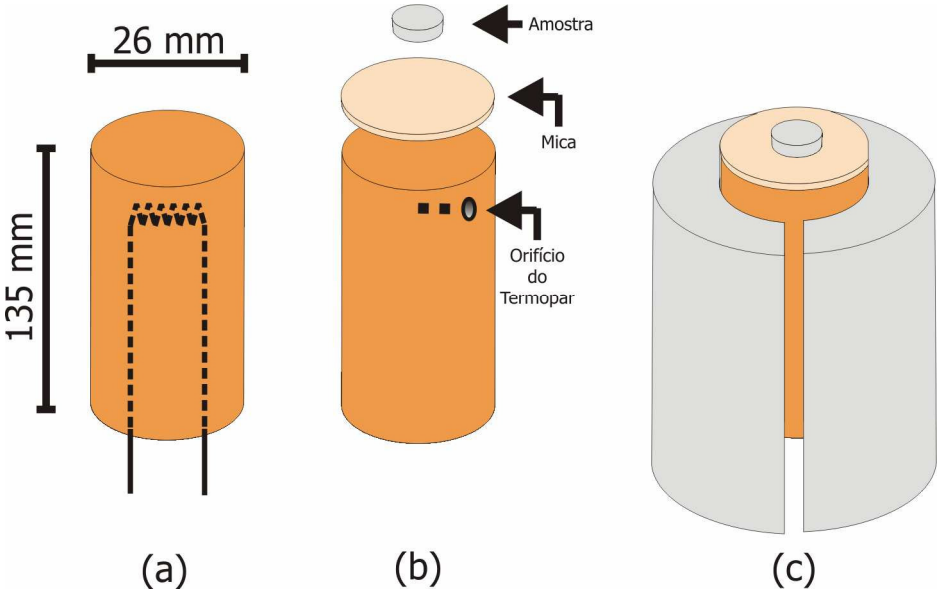

Figura 3.1 - Sistema de aquecimento: (a) resistência elétrica dentro do cilindro metálico; (b) lâmina de mica e orifício do termopar; (c) cilindro metálico envolto em um molde de cimento.

A leitura da temperatura é realizada por um termopar do tipo K (Cromel/Alumel) que tem uma boa sensibilidade e suporta temperaturas de até 1200 ºC. Este termopar é inserido dentro do cilindro de cobre em um orifício que dista aproximadamente 1 mm do plano onde se encontra a amostra. Para evitar o contato elétrico entre a amostra e o cilindro de cobre é utilizada uma lâmina de mica, pois se trata de um mau condutor elétrico (Figura 3.1b).

Para minimizar a perda de calor do cilindro para o ambiente, foram construídos dois moldes que atuam como envoltório para o cilindro. Estes moldes foram fabricados com um tipo de cimento especial o qual é utilizado em fornos de altíssimas temperaturas (Figura 3.1c).

A resistência elétrica do elemento de aquecimento e o termopar são ligados ao controlador de temperatura (DHACEL, modelo DH100B) (Figura 3.2). Trata-se de um controlador monocanal com interface de comunicação com o PC via porta RS-232. Este modelo aceita diversos tipos de termopares bem como diversos tipos de programação de rampas e patamares.

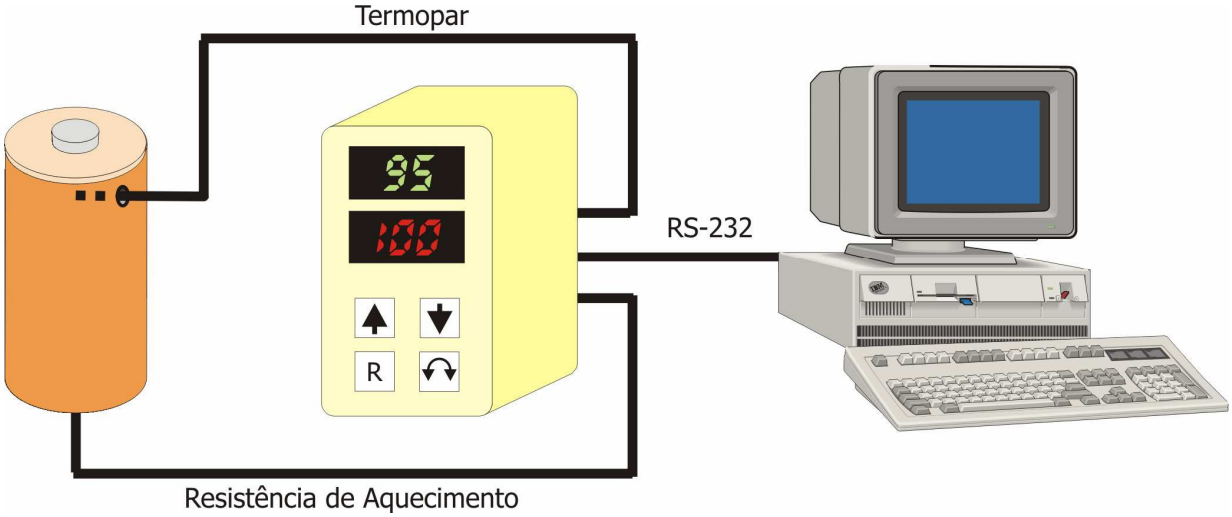

Figura 3.2 - Conexões do controlador de temperatura.

A conexão entre o PC e o controlador é feita utilizando um adaptador fornecido pelo fabricante. Este adaptador (que precisa ser alimentado por uma fonte DC 9V) é conectado diretamente ao computador e possui três saídas enumeradas, como mostra a figura 3.3.

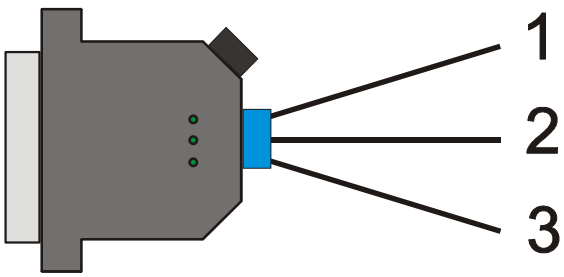

Figura 3.3 – Adaptador utilizado na conexão entre o controlador e computador.

A saída 1 do adaptador não é utilizada, enquanto a saída 2 e 3 devem ser ligadas diretamente nas saídas 7 e 6 do controlador de temperatura, respectivamente.

3.1.2 Medida de resistência elétrica

Para a medida da resistência elétrica da amostra é utilizado um multímetro digital (AGILENT, modelo 34401A). Este multímetro pode comunicar-se com o computador através da porta RS-232 ou GPIB. Outra vantagem deste aparelho é ter um bom limite superior na medida de resistência elétrica, que é 100 MΩ. Porém, como na maioria das vezes a resistência elétrica da amostra é superior ao valor alcançado pelo multímetro, então é necessário ligar um resistor em paralelo com a amostra utilizada. Este resistor pode ter diferentes valores conforme o tipo de material a ser estudado, mas em geral utiliza-se um resistor de 50 MΩ com tolerância de 1%. A Figura 3.4 mostra como são ligados os instrumentos ao multímetro.

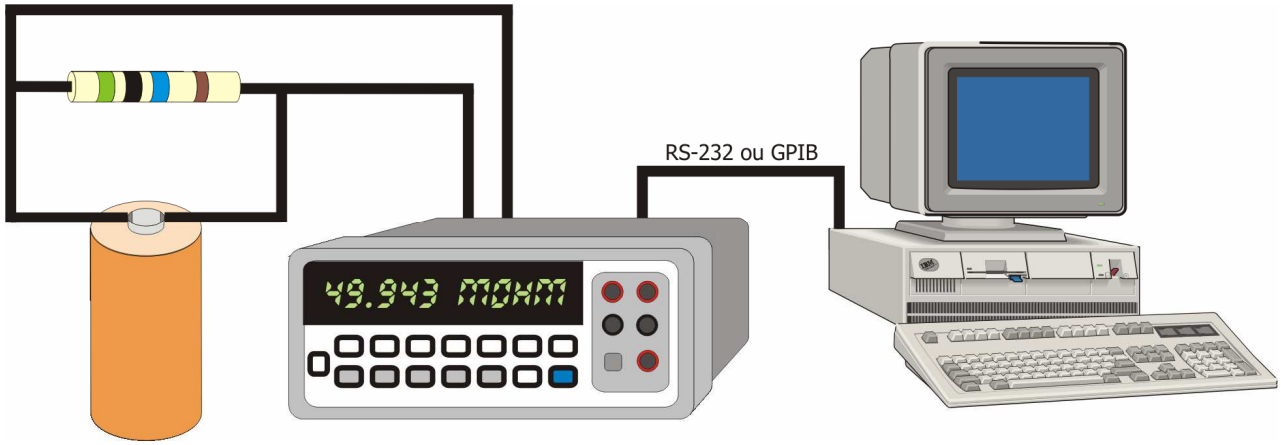

Figura 3.4 - Resistência elétrica em paralelo com a amostra, e conexão entre o computador e o multímetro.

A conexão entre o multímetro e o computador é feita utilizando um cabo que em cada uma de suas pontas possui um conector do tipo DB9 fêmea. Este cabo foi fornecido pelo fabricante do multímetro e a Figura 3.5 mostra como é configuração deste cabo.

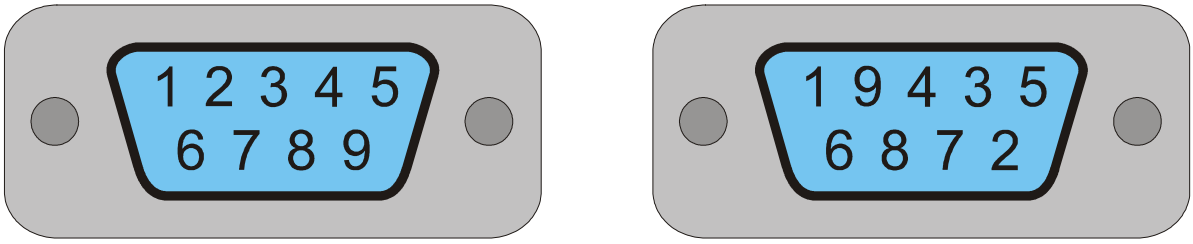

Figura 3.5 – Esquematização dos pinos do cabo do multímetro.

3.1.3 Conexões

Para medidas em que se faça necessário o uso de uma atmosfera específica é necessário uma câmara cilíndrica metálica que envolve todo o sistema de aquecimento (ANDRADE, 1998). A existência desta câmara cilíndrica, bem vedada, possibilita a utilização de gases altamente tóxicos como o monóxido de carbono (CO) e altamente inflamáveis como o metano (CH_4).

A câmara cilíndrica metálica se encaixa perfeitamente a uma base metálica com um O-ring de vedação. Na parte superior se encontra a entrada de gases (Figura 3.6a). Um suporte metálico fixo à base da câmara suporta o cilindro de cobre onde se encontra a amostra. Neste

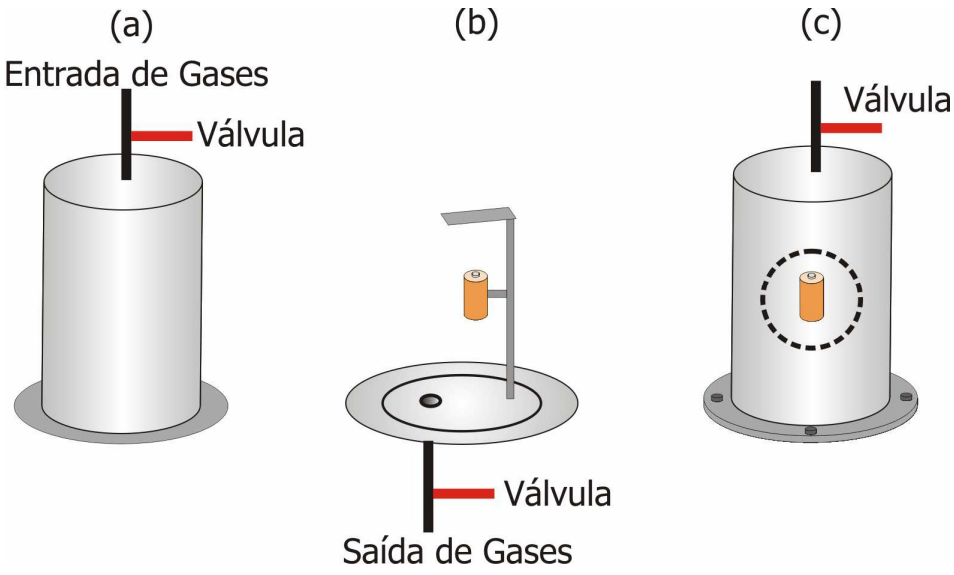

Figura 3.6 - Câmara cilíndrica: (a) cilindro superior; (b) base do cilindro; (c) sistema fechado e pronto para uso.

mesmo suporte, acima do cilindro de cobre, há uma chapa metálica para impedir que o fluxo de gás incida diretamente sobre a amostra. A saída de gases encontra-se na base inferior. Tanto na entrada quanto na saída de gases, existem válvulas para o controle do fluxo de gases a qualquer momento (Figura 3.6b). A Figura 3.6(c) mostra a câmara encaixada na parte inferior, pronta para uso.

Acoplado ao sistema existe ainda uma bomba de vácuo mecânica (EDWARDS, modelo E2M2) para retirar todo o ar existente dentro da câmara antes de inserir a atmosfera desejada, ou mesmo para medidas sob vácuo. Também faz parte do sistema um medidor de pressão (EDWARDS, modelo PIRANI 77/2) com sensor (EDWARDS, modelo PR10-S) acoplado à saída de gases. A Figura 3.7 ilustra as conexões de gases e a Figura 3.8 ilustra o sistema completo. Uma foto do sistema encontra-se nos apêndices deste trabalho.

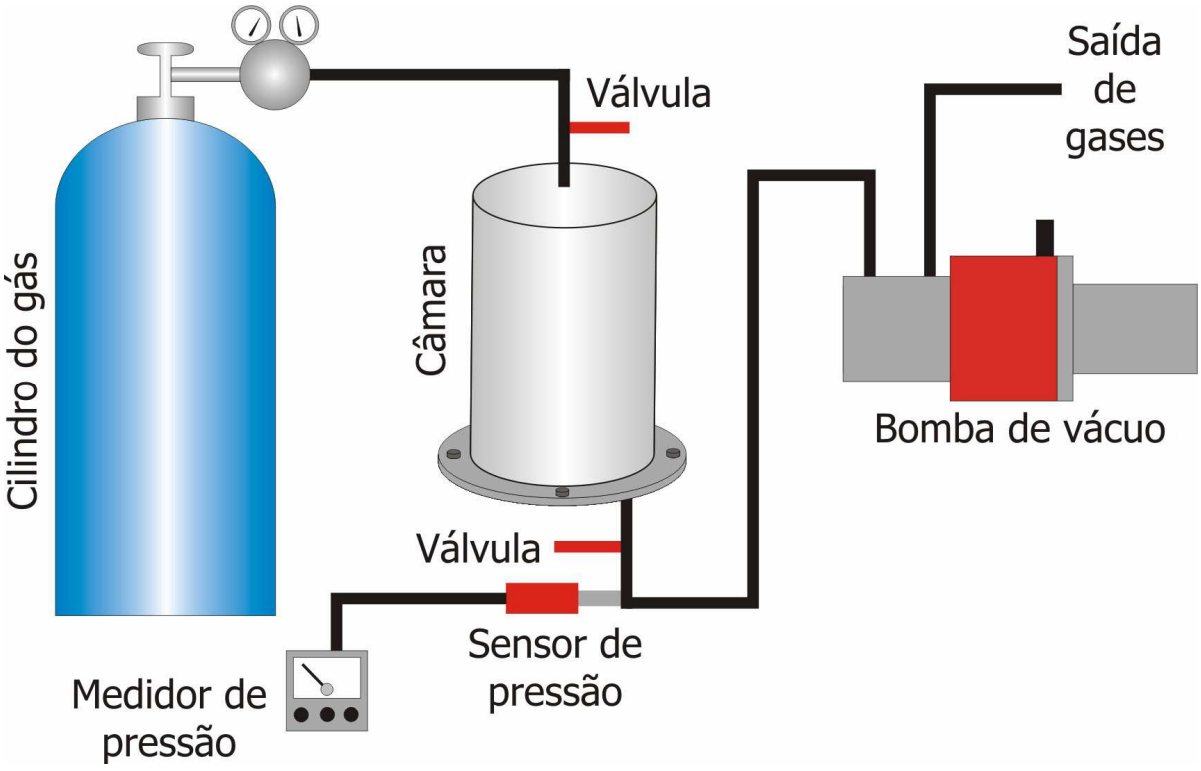

Figura 3.7 - Conexões Gasosas: entrada e saída de gases.

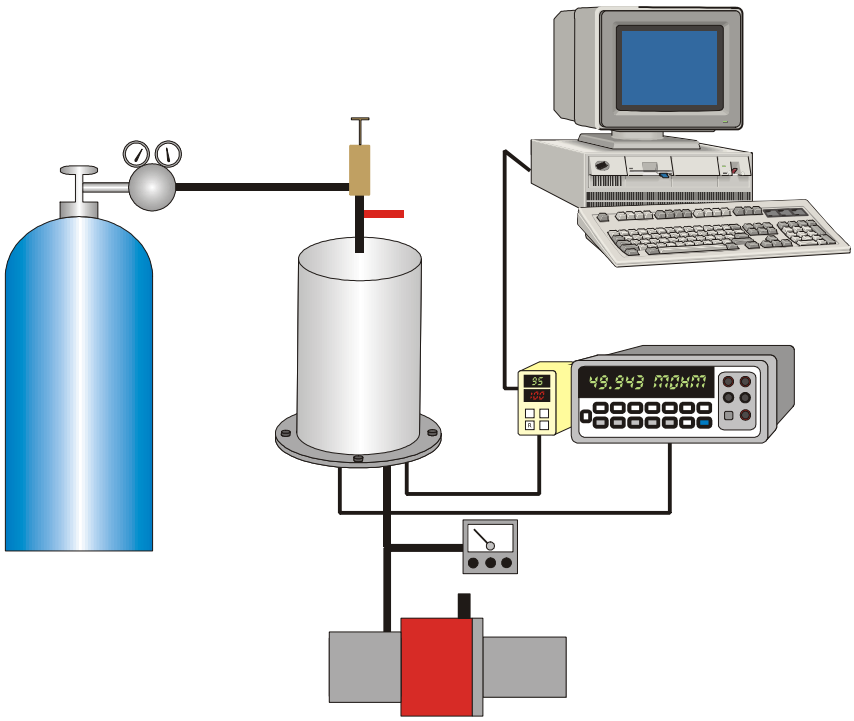

Figura 3.8 – Sistema Completo.

3.2 MONTAGEM DO SOFTWARE PARA O SISTEMA

Para testes de novos sensores, faz-se necessário a utilização de um microcomputador para aquisição de dados. Esta exigência é de suma importância, pois é extremamente inviável registrar manualmente em uma tabela valores de resistência elétrica e de temperatura a cada 2 ou 3 segundos, por um período que pode chegar até 20 horas.

Porém, de nada adianta dispor de um computador recente se não existir um software para fazer o trabalho acima mencionado. Cada sistema de caracterização de sensores é único, pois utilizam diferentes aparelhos, de diferentes marcas e modelos e com diferentes tipos de comunicação com o PC. Esta unicidade torna impossível conseguir um programa pronto na internet, ou encontrá-lo comercialmente. Com tantos fatores discriminando o sistema resta uma única alternativa: a criação do próprio software.

Decidiu-se utilizar a plataforma Windows para o funcionamento deste software, pois além de ser um sistema operacional presente na maioria dos computadores, é um sistema multitarefa, com muitas opções de linguagem para programação e de domínio da maioria dos usuários.

O programa foi elaborado em pascal orientado a objeto, pois se trata de uma linguagem amigável, orientada a objeto, para o sistema operacional escolhido e tem-se uma relativa facilidade para se obter informações sobre programação em sites especializados.

O software comunica-se frequentemente com o controlador de temperatura e com o multímetro utilizando as portas seriais RS-232 (COM1 e COM2). Para caracterizar as amostras é preciso que o software faça uma medida de resistência elétrica seguida de uma medida de temperatura. Esta medida deve se repetir em um intervalo de 2 a 40 segundos a ser definido pelo usuário. Além de registrar os valores de temperatura e resistência elétrica, o software também precisa gravar o instante de tempo t em que foram feitas as leituras.

Fazer o *looping* básico da aquisição de dados é o menor dos problemas. As dificuldades começam em descobrir como funciona a comunicação serial em pascal e entender a linguagem de comunicação tanto do multímetro quanto do controlador.

A linguagem do multímetro é detalhadamente descrita no manual do usuário, o qual demonstra quais *strings* devem ser enviadas para se obter a resposta desejada (neste caso, o valor de resistência elétrica). O mesmo não ocorreu com o controlador de temperatura. Com ele veio apenas um resumido manual de duas folhas ensinando o básico da utilização do controlador. No manual não foram encontradas opções avançadas como instalação, comunicação, protocolo, etc.

A solução para este problema foi utilizar o software fornecido pelo fabricante do controlador e monitorar a comunicação serial entre PC e controlador. Esta comunicação é feita em hexadecimal, o que dificulta o processo de decodificação. Após muitos testes foi possível escrever a Tabela 3.1. Esta tabela demonstra quais *strings* devem ser enviadas para que o controlador execute a tarefa desejada. É importante ressaltar que os valores estão em hexadecimal e o símbolo # é utilizado apenas como separador desses números.

Tabela 3.1 – Linguagem de comunicação do controlador.

Tarefa	***String* a ser enviada**	***String* recebida**
Obter temperatura	#02#03#00#4A#00#00#64#2F	#02#03#00#4A**#T#T**#E5#E4
Definindo o programa N em uso	#02#05#B6#20#00**#N**#2B#BB	#02#05#B6#20#00**#N**#2B#BB
Definindo o Set Point 1 do programa 1	#02#06#B6#71**#T#T**#7F#E9	#02#06#B6#71**#T#T**#7F#E9

Tarefa	***String* a ser enviada**	***String* recebida**
Definindo o CC1 do programa 1	#02#06#B6#73#00**#N**#9F#AA	#02#06#B6#73#00**#N**#9F#AA
Definindo o tempo 1 do programa 1	#02#06#B6#75**#t#t**#BE#64	#02#06#B6#75**#t#t**#BE#64
Definindo o Set Point 2 do programa 1	#02#06#B6#79**#T#T**#7E#D6	#02#06#B6#79**#T#T**#7E#D6
Definindo o CC2 do programa 1	#02#06#B6#7B#00**#N**#1E#68	#02#06#B6#7B#00**#N**#1E#68
Definindo o tempo 2 do programa 1	#02#06#B6#7D**#t#t**#BF#A1	#02#06#B6#7D**#t#t**#BF#A1
Definindo o Set Point 3 do programa 1	#02#06#B6#81**#T#T**#5E#1F	#02#06#B6#81**#T#T**#5E#1F
Definindo o CC3 do programa 1	#02#06#B6#83#00**#N**#9F#99	#02#06#B6#83#00**#N**#9F#99
Definindo o tempo 3 do programa 1	#02#06#B6#85**#t#t**#3E#5F	#02#06#B6#85**#t#t**#3E#5F
Definindo o Set Point 4 do programa 1	#02#06#B6#89**#T#T**#7D#F7	#02#06#B6#89**#T#T**#7D#F7
Definindo o CC4 do programa 1	#02#06#B6#8B#00**#N**#1E#5B	#02#06#B6#8B#00**#N**#1E#5B
Definindo o tempo 4 do programa 1	#02#06#B6#8D**#t#t**#3F#84	#02#06#B6#8D**#t#t**#3F#84
Definindo o Set Point 5 do programa 1	#02#06#B6#91**#T#T**#FD#F0	#02#06#B6#91**#T#T**#FD#F0
Definindo o CC5 do programa 1	#02#06#B6#93#00**#N**#9E#5C	#02#06#B6#93#00**#N**#9E#5C
Definindo o tempo 5 do programa 1	#02#06#B6#95**#t#t**#BF#92	#02#06#B6#95**#t#t**#BF#92
Definindo o Set Point 6 do programa 1	#02#06#B6#99**#T#T**#7E#B6	#02#06#B6#99**#T#T**#7E#B6
Definindo o CC6 do programa 1	#02#06#B6#9B#00**#N**#DE#5E	#02#06#B6#9B#00**#N**#DE#5E

A comunicação é feita enviando e recebendo 8 bytes (ou seja, oito valores em hexadecimal) para cada tarefa. Nas linhas da tabela, as *strings* em negrito (T, t e N) representam os valores em hexadecimal respectivos a tarefa solicitada.

Para a obtenção da temperatura lida pelo controlador (primeira linha da tabela), o valor lido é retornado no quinto e sexto byte recebido. Supondo que a *string* recebida seja #02#03#00#4A#01#27#E5#E4, então o quinto e o sexto byte são, respectivamente, 01 e 27. Juntando este dois valores temos 0127 ou 127 que no sistema decimal é igual a 295. Quanto o quinto byte for nulo, basta transformar diretamente o sexto byte para a obtenção da temperatura.

Na segunda linha da tabela ensina como definir um programa em uso no controlador. N neste caso simboliza o número do programa. Quando N for igual a zero o controlador entende que nenhum programa é para ser executado.

O Set Point indica em qual temperatura o sistema deverá alcançar, e o valor deve ser informando em dois bytes (indicados por T na tabela). Por exemplo, para definir o set point 1 em 500 ºC basta transformar este valor para hexadecimal (neste caso obtemos 01F4) e substituir estes valores na string de comando, que nesse caso será #02#06#B6#71#01#F4#7F#E9.

A sigla CC significa Continua Ciclo, e através dela o controlador verifica se deve ou não continuar o programa. Ou seja, o controlador verifica se deve ir até o próximo set point. N poderá assumir apenas dois valores, 0 ou 1, e estes significam NÃO e SIM respectivamente.

O tempo entre os *set points* devem ser informados em minutos, utilizando dois bytes (simbolizados por **t** na tabela). Por exemplo, para definir o tempo entre os *set points* 1 e 2 como 120 minutos, precisa-se transformar 120 em hexadecimal (o que nos resulta 78 ou 0078) e substituir na *string* mencionada na tabela. Neste caso teremos #02#06#B6#75#00#78#BE#64.

A comunicação entre controlador de temperatura e multímetro com o computador é feita utilizando-se as seguintes configurações:

Porta:	COM1 (Controlador) e COM2 (Multímetro);
Velocidade:	9600bps;
Bits de dados:	8;
Bits de parada:	1;
Paridade:	Sem paridade.

Entretanto, os problemas não se encerraram na comunicação. Este software desenvolvido para o sistema realiza diversas tarefas ao mesmo tempo, como comunicação, cálculos matemáticos, plotagem e armazenamento de dados. Nessas tarefas foram utilizadas ferramentas de programação como *Delay* (faz o computador esperar um tempo t), *looping* (faz o computador repetir uma tarefa diversas vezes) e comandos estruturados (*if*, *for* e *case*, que executam uma tarefa se uma determinada condição for satisfeita).

Tais ferramentas de programação são eficazes, porém com um problema: não é possível utilizá-las ao mesmo tempo, como o programa de aquisição exige. Então, surge a necessidade da utilização de *threads*. Trata-se de um comando de programação avançado que torna o software multitarefa. Em outras palavras, ele possibilita a utilização de diversos comandos ao mesmo tempo.

No software desenvolvido neste trabalho foram projetadas outras funções não indispensáveis como a aquisição de dados em si, mas que poupam muito trabalho para o usuário no dia-a-dia. Essas funções são descritas a seguir.

3.2.1 Programação do controlador via software

O controlador de temperatura normalmente é programado (rampas e patamares) manualmente, utilizando seus quatro botões frontais. Porém o software elaborado faz a programação completa do controlador. Para isto, basta que o usuário informe todos os detalhes de seu programa de aquecimento, como número de rampas e patamares, taxa de aquecimento/resfriamento, tempo e temperatura de patamares. A figura 3.9 mostra a janela de programação do controlador. O software oferece a opção para salvar o programa criado, caso seja necessário utilizá-lo posteriormente.

3.2.2 Plotagem de dados em tempo real

Ao iniciar a aquisição de dados, o software inicia imediatamente a plotagem dos dados em um gráfico 2D. Nesta janela, pode ser definido o tipo de escala a ser utilizada para cada eixo (logarítmica ou linear). O software desenvolvido determina a escala automaticamente de acordo com os dados obtidos. Porém, também é possível definir os valores iniciais e finais de cada eixo (X,Y), bem como alterar o tamanho e o formato dos pontos do gráfico.

Baseado nas informações do programa de aquecimento, o software determina automaticamente quando se trata de um gráfico em função do tempo ou da temperatura. É

possível salvar os gráficos para uma possível análise posterior. A Figura 3.10 mostra a janela da plotagem de dados em tempo real.

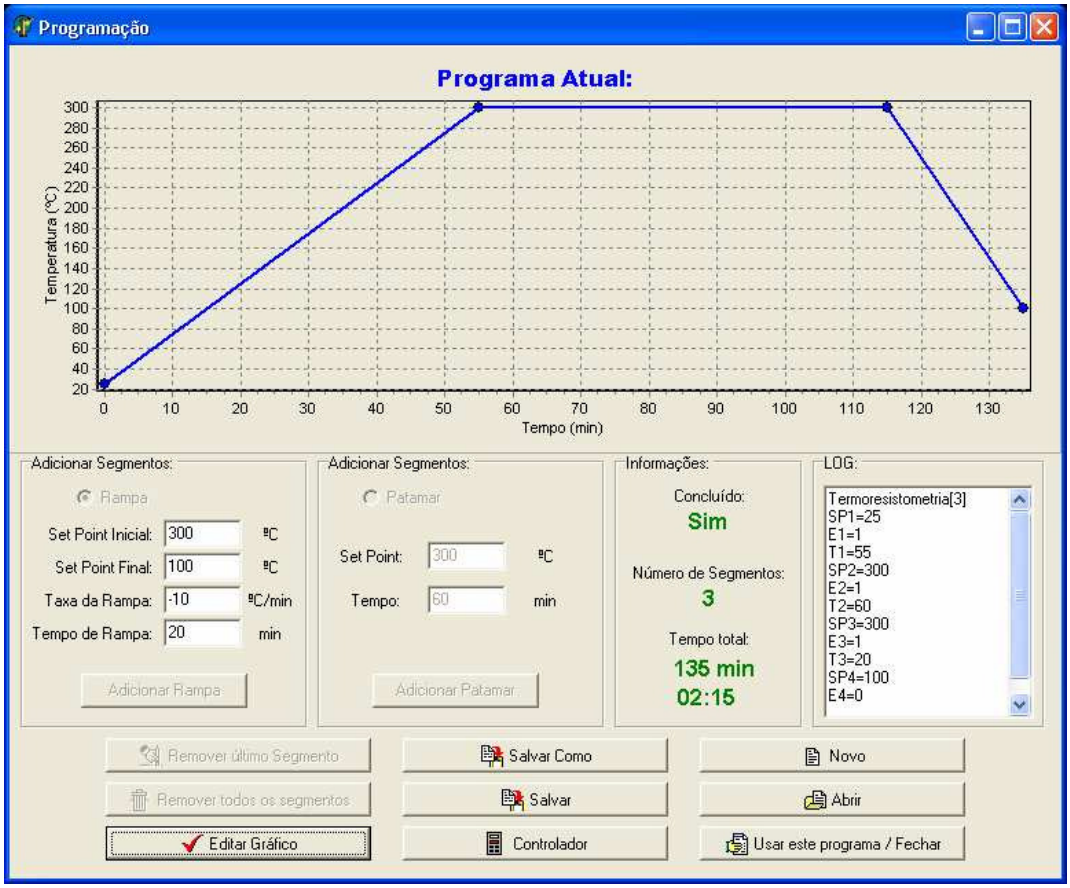

Figura 3.9 - Janela da programação do controlador de temperatura.

3.2.3 Cálculo automático de resistência real

Conforme descrito anteriormente, certas amostras podem ter um valor de resistência elétrica muito elevada. Isto torna impossível a leitura direta da resistência elétrica, pois foge do alcance do multímetro (100 MΩ). Neste caso, utiliza-se uma resistência elétrica de precisão conhecida a qual é ligada em paralelo com a amostra, gerando assim resistência equivalente que fique dentro da faixa de leitura alcançada pelo multímetro.

Como o multímetro lê a resistência equivalente da amostra, o software pode determinar automaticamente a resistência real da amostra. Para que o software realize o cálculo da resistência real basta informar o valor da resistência elétrica utilizada em paralelo

com a amostra. A Figura 3.11 mostra a janela principal do programa, onde é possível informar ao software se o usuário irá utilizar ou não uma resistência em paralelo.

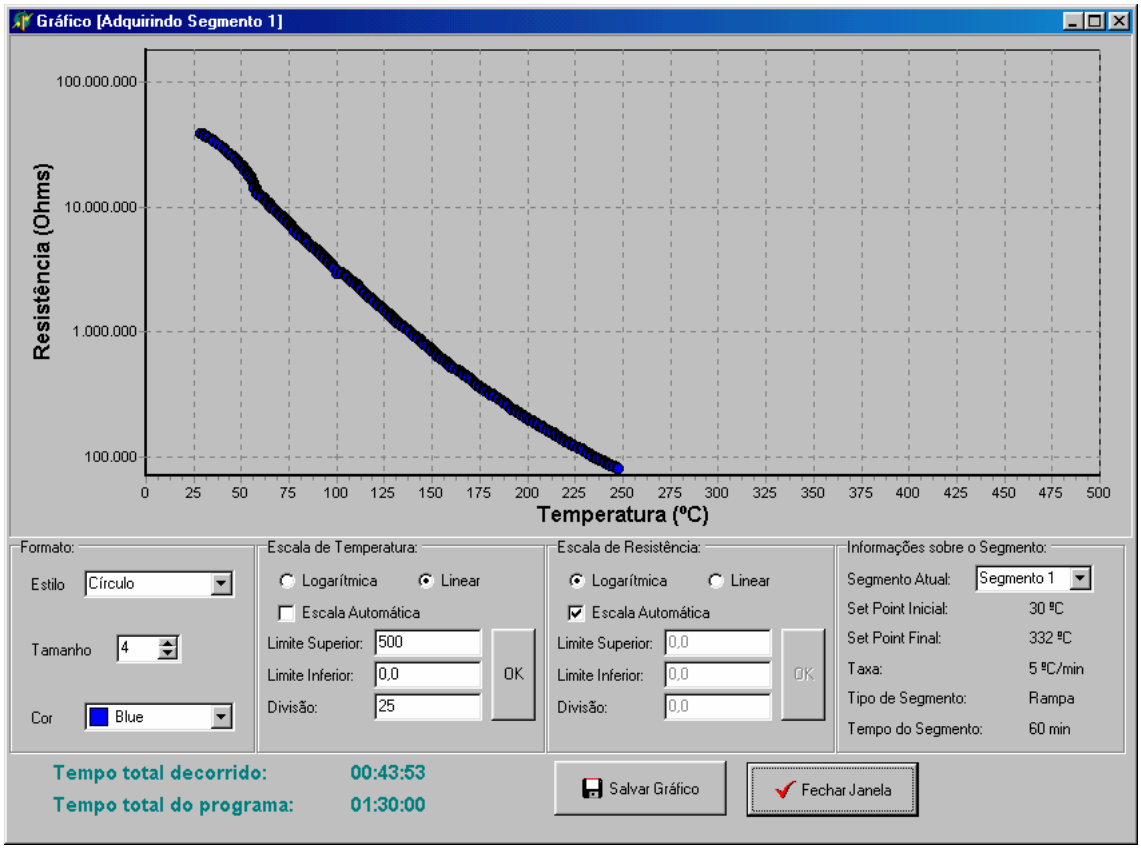

Figura 3.10 - Plotagem de dados em tempo real.

3.2.4 Intervalo de aquisição

Com este software é possível determinar o intervalo de tempo entre uma leitura e outra. O valor mínimo é de dois segundos e o máximo é de 40 segundos. Esta opção é muito útil em medidas mais longas como 15 ou mais horas. Nessas medidas mais longas, que se utilizam baixíssimas taxas de aquecimento (p. ex., 1 ºC/min), as transformações no material ocorrem muito lentamente. Portanto, adquirir um ponto a cada 2 segundos irá apenas aumentar o tamanho da base de dados e sobrecarregar o software, sem nenhum benefício adicional. A Figura 3.11 mostra a janela principal do programa onde é possível determinar o intervalo em segundos, entre uma aquisição e outra.

3.2.5 Filtragem de dados

Quando se utiliza uma resistência elétrica em paralelo com a amostra, dependendo da tolerância desta resistência, em certas ocasiões, o software pode determinar um valor de resistência elétrica negativa da amostra. Este ponto de resistência negativa (obtido matematicamente), se for plotado em um gráfico de escala logarítmica resultará num erro que finalizará a aplicação, pois não é possível plotar um gráfico em escala logarítmica com pontos negativos.

Portanto, antes de plotar e salvar a resistência elétrica o programa confere se o valor de resistência elétrica lido é maior que zero. O programa também faz a contagem de quantos pontos foram descartados e quantos foram utilizados na caracterização. A Figura 3.11 mostra a janela principal do programa, onde é possível verificar a contagem dos pontos descartados e utilizados.

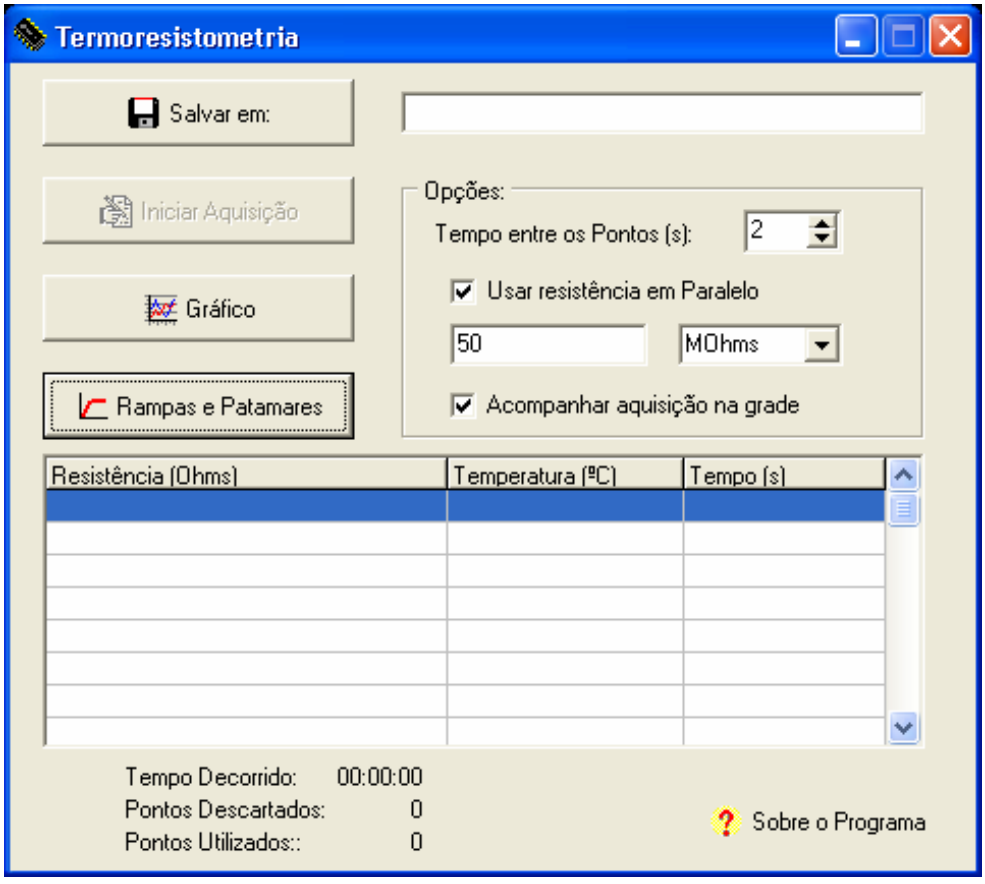

Figura 3.11 - Janela principal do programa.

3.2.6 Interface simples e útil

O software é apresentado em janelas simples e amigáveis. Sempre com informações úteis, por exemplo, tempo decorrido, tempo do segmento atual, etc.

O programa possui mais de três mil linhas de código que não podem conter um único erro. Um simples ponto-e-vírgula ausente no final de uma linha impede a compilação do programa.

A programação do código levou aproximadamente 11 meses de trabalho e continua em constante atualização. Neste programa, como em qualquer outro, *bugs* pequenos só serão descobertos com a utilização no dia-a-dia. O código fonte completo encontra-se nos apêndices.

3.3 PREPARAÇÃO DE AMOSTRAS

3.3.1 Precursores

Neste trabalho foram utilizados como precursores os seguintes pós: hematita (óxido) e o acetato de manganês tetrahidratado ($Ac_2Mn{\cdot}4H_2O$, sal de ácido carboxílico), os quais estão descritos na Tabela 3.2.

Tabela 3.2 - Precursores utilizados.

Precursor	Fornecedor	Pureza (%)	Fórmula química	Massa molar (g/mol)
Hematita	Carlo Erba	99,9	Fe_2O_3	159,69
$Ac_2Mn{\cdot}4H_2O$	Synth	98,7	$(CH_3COO)_2Mn{\cdot}4H_2O$	245,09

Baseado na massa molar dos precursores, foram calculadas as quantidades em massa de hematita e $Ac_2Mn{\cdot}4H_2O$ a serem utilizadas, de modo a se obter uma razão molar dos cátions Fe:Mn = 2:1. De tal forma que para cada grama de hematita a ser utilizada seria necessário 1,5315 g de $Ac_2Mn{\cdot}4H_2O$.

3.3.2 Secagem por liofilização

Neste trabalho, a mistura de pós de Fe_2O_3 + $Ac_2Mn{\cdot}4H_2O$ foi obtida através de um processo de secagem por liofilização. Foi utilizado um liofilizador (CHRIST ALPHA, modelo 1-2) do Laboratório de Química e Mineralogia do Solo no Departamento de Agronomia da Universidade Estadual de Maringá.

A técnica de secagem por liofilização consiste, primeiramente, da adição dos pós na composição específica, em cerca de 150 ml de água destilada deionizada, seguida de homogeneização em ultra-som. O processo de liofilização (BELLINI, 2001) pode ser descrito utilizando-se um diagrama de fases esquemático de uma mistura aquosa, como aquele que é mostrado na Figura 3.12. As etapas da liofilização são:

Etapa (1)→ (2): a mistura é, então, transferida para um frasco plástico e congelada em nitrogênio líquido. Para se criar em seu interior uma fina casca congelada, faz-se necessário o uso de movimentos circulares do frasco dentro do nitrogênio líquido. O congelamento é conduzido à pressão atmosférica a uma temperatura de cerca de -196 ^{o}C (temperatura que está abaixo do ponto triplo Q da mistura). Após congelado, a temperatura do material não deve aumentar até que ele esteja sob vácuo, ou a fusão do material poderá ocorrer (BELLINI, 2001).

Etapa (2)→(3): o frasco com a mistura congelada é, então, conectado ao liofilizador, o qual é constituído de um sistema de vácuo composto de uma bomba de vácuo mecânica, com sensor e medidor de pressão, e uma armadilha para água. Esta armadilha é, na realidade, um refrigerador, cuja serpentina encontra-se no interior do sistema. A temperatura da armadilha está abaixo de -45 ^{o}C sob vácuo. A função desta armadilha é capturar as moléculas de água que sublimam e impedi-las de chegar até a bomba de vácuo. Se a água chegar até a bomba haverá contaminação do óleo e ele precisará ser substituído. Nesta etapa o material é mantido congelado e a pressão é reduzida abaixo do ponto triplo Q. Imediatamente após a conexão do frasco, a pressão cai lentamente, desde a pressão atmosférica até estabilizar-se em um valor da ordem de 10^{-2} torr (BELLINI, 2001).

Etapa (3)→(4): Esta é a etapa de secagem, onde a água sublima lentamente até que o material seja desidratado. A secagem do pó leva cerca de 18 horas para se completar (BELLINI, 2001).

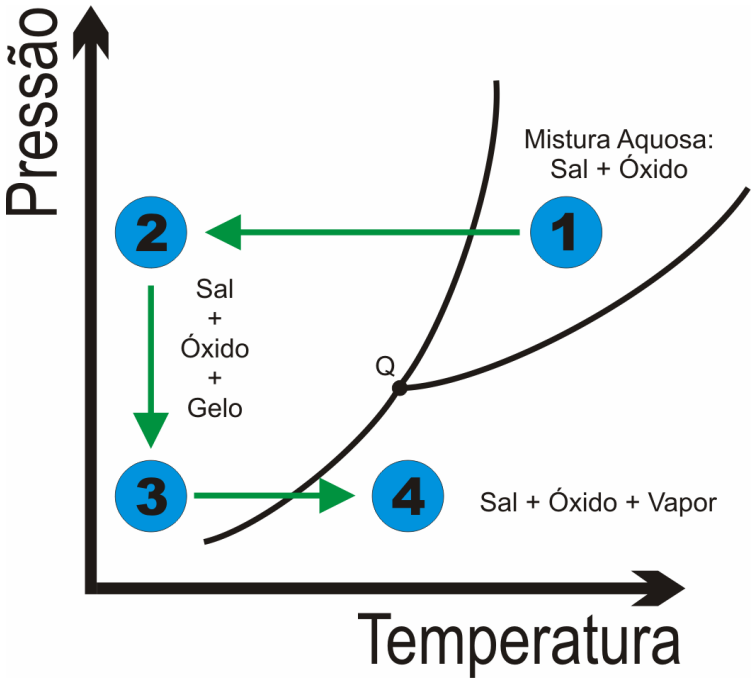

Figura 3.12 - Liofilização em um diagrama de fases esquemático de uma solução aquosa.

Após a liofilização, os pós foram sequencialmente peneirados em peneiras com aberturas de malha de 106 μm e 63 μm, e armazenados em frascos plásticos.

3.3.3 Tratamento térmico

Os pós liofilizados foram tratados termicamente em três diferentes condições:

a) Primeira condição *(amostra TT1)*: foi aquecida com uma taxa de aquecimento de 5 ºC/min até 400 ºC, onde permaneceu por 4 horas, seguido de resfriamento no forno;

b) Segunda condição *(amostra TT2)*: a taxa de aquecimento foi de 5 ºC/min até alcançar 400 ºC, onde permaneceu por apenas 2 horas. Posteriormente, a mesma amostra foi aquecida até 700 ºC a uma taxa de aquecimento de 10 ºC/min, e permaneceu nesta temperatura por 4 horas, seguido de resfriamento no forno;

c) Terceira condição *(amostra TT3)*: a amostra também foi aquecida até 400 ºC a uma taxa de 5 ºC/min onde permaneceu por 2 horas. Posteriormente, a mesma amostra foi aquecida até 1000 ºC a uma taxa de aquecimento de 10 ºC/min, e permaneceu nesta temperatura por 4 horas, seguido de resfriamento no forno. A Figura 3.13 descreve esquematicamente as condições de preparação das amostras.

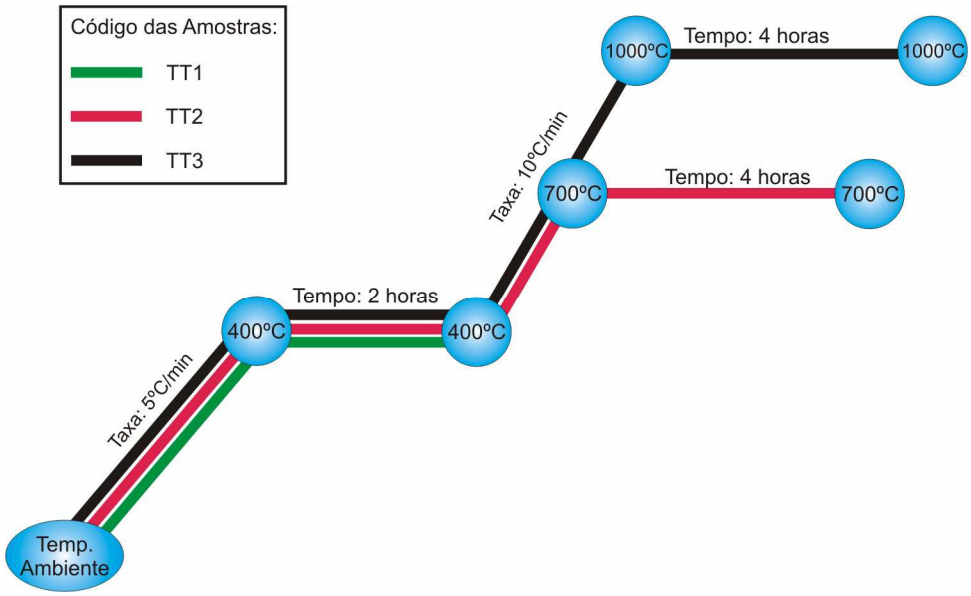

Figura 3.13 - Tratamento térmico dos pós liofilizados.

Após o tratamento térmico, pós foram sequencialmente peneirados da mesma forma que os pós liofilizados, como descrito anteriormente.

3.4 CARACTERIZAÇÃO DAS AMOSTRAS

3.4.1 Difração de raios X

Os pós liofilizados e os tratados termicamente foram caracterizados por difração de raios X (DRX) (SHIMADZU, modelo D6000) à temperatura ambiente. O intervalo de análise dos espectros de difração foi $10º \leq 2\theta \leq 70º$, utilizando uma fonte Cu Kα (λ = 1.5406 Å), 40 kV, 30mA, passo 0,02º e tempo 0,6 s. Todas as medidas foram realizadas na Central Analítica da UEM.

3.4.2 Espectroscopia de absorção atômica

A espectroscopia de absorção atômica foi utilizada para caracterizar os pós liofilizados. Foi utilizado o equipamento de espectroscopia (GBC, modelo 932AA) do Laboratório de Química e Mineralogia do Solo do Departamento de Agronomia da UEM.

3.4.3 Termoeletroresistometria (TER)

Todas as amostras obtidas (TT1, TT2 e TT3) e a amostra liofilizada (sem tratamento térmico) foram caracterizadas por TER, utilizando-se o próprio sistema descrito anteriormente.

Para a caracterização por TER, os pós foram compactados na forma de pastilhas cilíndricas utilizando uma pressão de 243 MPa (como mostrado anteriormente na Figura 3.1b, pág. 24). A massa de pó utilizada em cada pastilha foi de 0,5 g. A taxa de aquecimento utilizada em todas as caracterizações foi de 5 ºC/min. As medidas de TER foram feitas em três diferentes atmosferas: a) ar atmosférico ($P \cong 1$ atm); b) gás ($P < 1$ atm); c) ar atmosférico + gás ($P \cong 1$ atm) .

Nas medidas com gás utilizou-se o metano ultra puro (CH_4). Para todas as medidas realizadas com gás foi utilizado o mesmo volume sob mesma pressão à temperatura ambiente.

4. RESULTADOS E DISCUSSÃO

4.1 O SISTEMA DE TER

O sistema foi testado para temperaturas até 450 ºC. O valor de temperatura lido pelo controlador foi aferido utilizando-se diferentes termopares e um termômetro de mercúrio (neste caso, apenas dentro de sua faixa de operação) e não foram observadas diferenças significativas.

Foram testadas diversas taxas de aquecimento entre 1 e 20 ºC/min. Com taxas de até 10 ºC/min o controlador consegue manter uma diferença muito pequena (2 ou 3 ºC) entre o valor lido e o *setpoint* (temperatura programada). Porém, para taxas entre 10 e 20 ºC/min começa haver uma discrepância entre o valor lido e o *setpoint.* Esta discrepância é diretamente proporcional ao valor da taxa de aquecimento e deixa de existir em torno de 100 ºC. Isto se dá em virtude da inércia térmica do elemento de aquecimento.

Todos os vazamentos da câmara cilíndrica foram solucionados, de tal forma que a bomba de vácuo mecânica conseguiu diminuir a pressão até 2 x 10^{-2} Torr. Este é considerado um bom valor devido às dimensões da câmara e o número de conexões existentes.

4.2 O SOFTWARE DE AQUISIÇÃO

O software elaborado até o presente momento não apresenta falhas. Foram testadas as variáveis temporais, utilizando-se um cronômetro, e mesmo depois de horas de aquisição a diferença de tempo não é maior do que 1 segundo.

O problema da aquisição de valores negativos de resistência elétrica desapareceu após a utilização de resistores em paralelo com tolerância de 1%. Já foram realizados testes de aquisição com mais de 10 mil pontos e nenhum deles precisou ser descartado. Entretanto, a opção de filtragem continua ativa caso seja viável utilizar um resistor com tolerância maior.

Os gráficos plotados automaticamente pelo programa também foram verificados. Os dados foram plotados no software Origin 7.0 (OriginLab Corporation) e não foram encontradas diferenças.

4.3 ESPECTROSCOPIA DE ABSORÇÃO ATÔMICA

A amostra liofilizada foi caracterizada por espectroscopia de absorção atômica e os resultados mostraram que ela é constituída de 31,10% de Fe e 16,20% Mn, o que corresponde à uma razão molar dos cátions Fe: Mn = 1,92, com erro experimental aceitável de 4%.

4.4 DIFRATOMETRIA DE RAIOS X

Todas as amostras foram caracterizadas por difratometria de raios X. As fases encontradas estão listadas na Tabela 4.1. A amostra liofilizada apresentou as fases hematita (Fe_2O_3) e acetato de manganês dihidratado ($Ac_2Mn{\cdot}2H_2O$). Durante o processo de liofilização o $Ac_2Mn{\cdot}4H_2O$ é parcialmente desidratado, o que dá origem à nova fase $Ac_2Mn{\cdot}2H_2O$ já descrita na literatura por BELLINI *et al*. (2007). Para a mostra TT1 foram encontradas as fases da hematita e hausmanita (Mn_3O_4). Na amostra TT2 encontraram-se as fases da hematita e óxido de manganês (Mn_2O_3). Finalmente, para a amostra TT3 foram encontradas as fases hematita e óxido de manganês ferro ($FeMnO_3$). A Figura 4.1 mostra os difratogramas das amostras liofilizada, TT1, TT2 e TT3, respectivamente.

Tabela 4.1 – Dados de difração de raios-x das amostras liofilizadas e tratadas termicamente.

Amostra	Fase	PDF	Sistema	Nome
Liofilizada	α-Fe_2O_3	33-0664	Romboédrico	Hematita
	$Ac_2Mn{\cdot}2H_2O$	-	-	Acetato de manganês dihidratado
TT1	α-Fe_2O_3	33-0664	Romboédrico	Hematita
	Mn_3O_4	24-0734	Tetragonal	Hausmanita
TT2	α-Fe_2O_3	33-0664	Romboédrico	Hematita
	Mn_2O_3	41-1442	Cúbico	Óxido de manganês
TT3	α-Fe_2O_3	33-0664	Romboédrico	Hematita
	$FeMnO_3$	75-0894	Cúbico	Óxido de manganês ferro

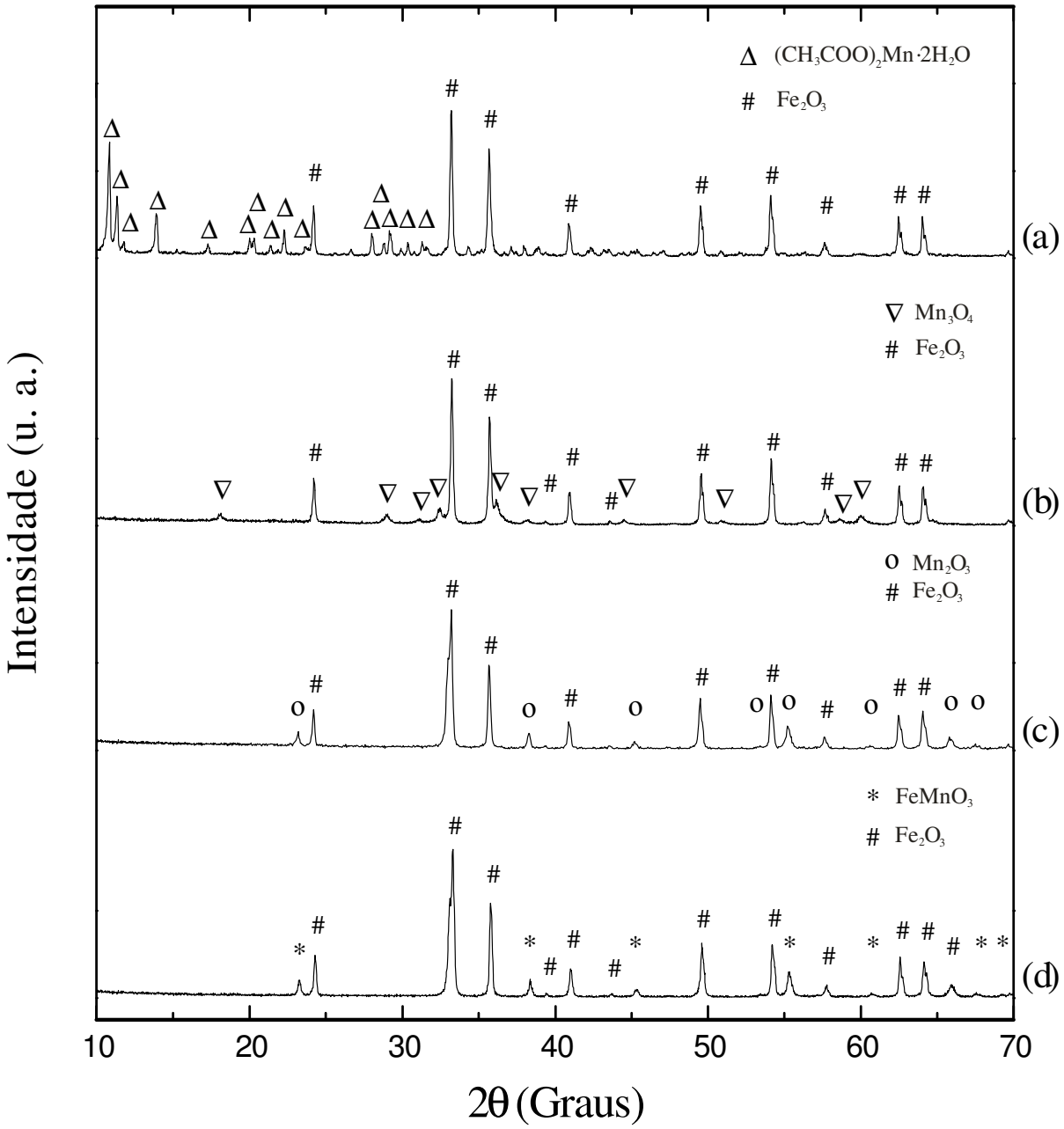

Figura 4.1 – Difratogramas: a) amostra liofilizada; b) amostra TT1; c) amostra TT2; d) amostra TT3.

Medidas de difração de raios X indicaram que a amostra TT3 contém a fase $FeMnO_3$. Dados da literatura apresentados por SEIFU *et. al.* (2000), indicam que esta fase tem comportamento ferrimagnético e que pode ser obtida por moagem utilizando-se como precursores pós de α-Fe_2O_3 e Mn_2O_3. A reação sugerida é dada pela Eq. 4.1.

$$\alpha\text{-}Fe_2O_3 + Mn_2O_3 \rightarrow 2\ FeMnO_3 \qquad (4.1)$$

Como foi mostrado na Tabela 4.1, a amostra TT2 contém somente as fases Fe_2O_3 e Mn_2O_3. O tratamento posterior da amostra TT2 a 1000 ºC resultou no aparecimento da fase $FeMnO_3$ (que neste caso foi obtida por reação do estado sólido em alta temperatura) a qual manteve a mesma estrutura cúbica do Mn_2O_3. A amostra TT3 contém as fases Fe_2O_3 e $FeMnO_3$. Isto indica que átomos de Fe substituíram parcialmente átomos de Mn, o que significa que a reação do estado sólido foi incompleta para formação da fase $FeMnO_3$. Estes resultados estão de acordo com aqueles apresentados por SEIFU *et. al.* (2000).

4.5 CARACTERIZAÇÃO POR TER

Todas as amostras obtidas neste trabalho foram caracterizadas por TER com taxa de aquecimento de 5 ºC/min. Foram realizadas várias medidas para cada amostra que comprovou a repetibilidade das mesmas. A Figura 4.2 mostra o resultado da medida de TER realizada em ar atmosférico, para a amostra liofilizada. Pode-se observar variações nos valores da resistência elétrica, os quais representam eventos térmicos, principalmente relacionados com a desidratação, formação de fases intermediárias, e com a decomposição do $Ac_2Mn{\cdot}2H_2O$.

Na literatura, BELLINI *et al.* (2007) estudaram o curso da decomposição térmica do sistema cerâmico baseado em Fe_2O_3 + $Ac_2Mn{\cdot}2H_2O$ obtido por liofilização, assim como a caracterização dos produtos sólidos após tratamentos térmicos em diferentes temperaturas e atmosferas. Os autores concluíram que existem três eventos térmicos importantes: I) desidratação (25-125 ºC); II) formação de uma fase intermediária (125-250 ºC); III) decomposição da fase intermediária (250-400 ºC). Estes dados comparados com os obtidos neste trabalho por TER, para o mesmo sistema cerâmico indicam que a queda da resistência elétrica observada até 125 ºC (pico em 77 ºC) está relacionada com a desidratação do material. O aumento abrupto de resistência elétrica ocorrendo na faixa de 125-250 ºC (máximo em 195 ºC) está relacionado com a formação da referida fase intermediária (não identificada até o presente momento). A queda abrupta em resistência elétrica ocorrendo na faixa de 250 a 400 ºC (com pico em 325 ºC) está relacionada com a decomposição da referida fase. Durante o aquecimento a pastilha sofre transformações físicas irreversíveis como grande perda de massa devido a desidratação e decomposição térmica que leva a liberação de gases (p.ex, CO_2, CH_4, CO) e produtos voláteis como acetona (CH_3COCH_3).

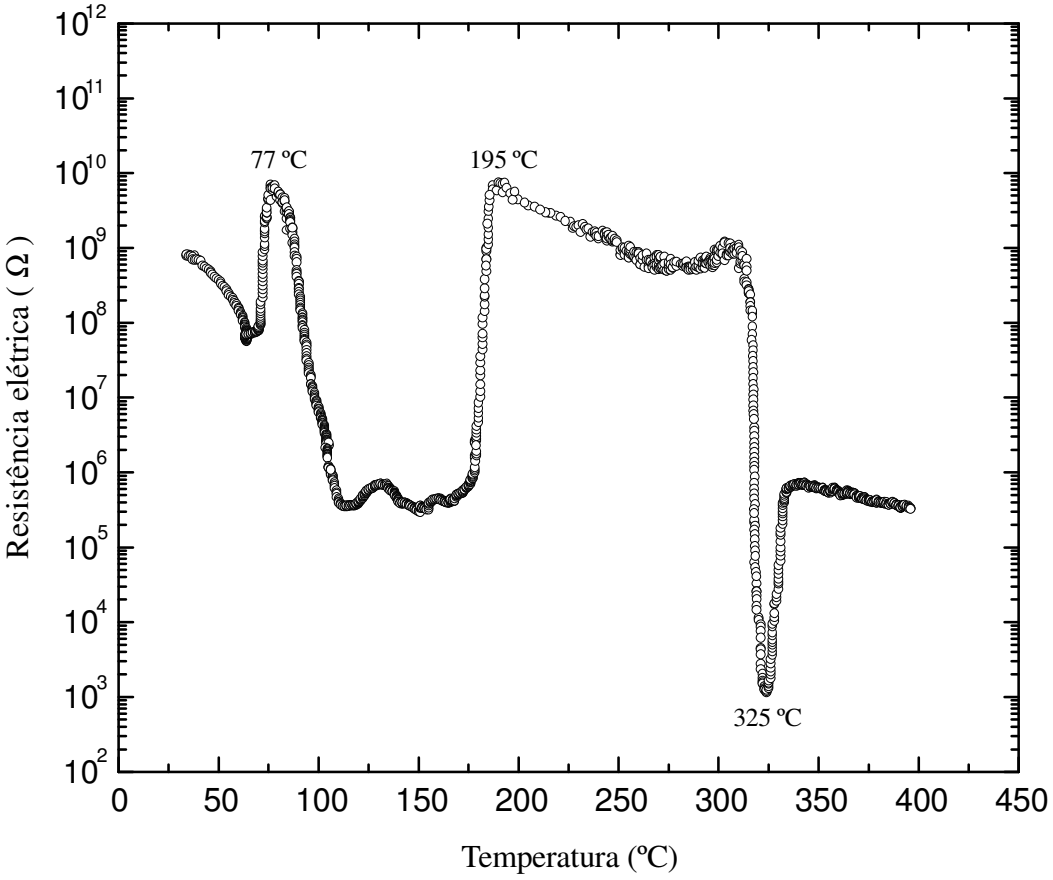

Figura 4.2 – Medida de TER da amostra liofilizada.

TER associada com DSC/TG constitui uma importante ferramenta para o estudo do curso da decomposição térmica de materiais termicamente sensíveis, como por exemplo, acetatos metálicos.

Todas as medidas de TER das amostras TT1, TT2 e TT3 apresentam uma queda no valor da resistência com o acréscimo de temperatura. Isto ocorre porque com o aumento da temperatura mais elétrons passam da banda de valência para a banda de condução do material, sendo esse um comportamento típico de materiais semicondutores.

A Figura 4.3 mostra medidas de TER das amostras TT1, TT2 e TT3, realizadas em ar atmosférico (P ≅ 1 atm), respectivamente. Em escala linear os dados de TER puderam ser bem representados por funções com decaimento exponencial. Em escala mono-log, como são apresentadas as medidas de TER, conseqüentemente as curvas mostram um decaimento linear. Em particular, a amostra TT3 apresentou uma anomalia em torno de 50 ºC a qual se

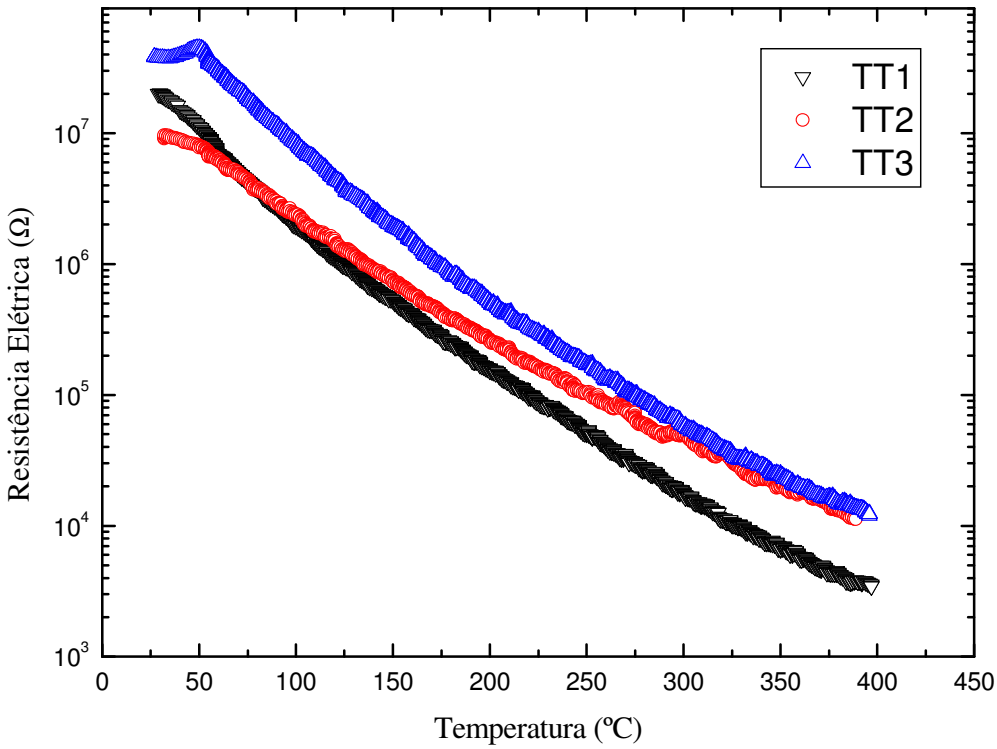

Figura 4.3 – Medidas de TER das amostras TT1, TT2 e TT3 em ar atmosférico (P ≅ 1 atm).

repetiu para várias medidas em diferentes condições, cuja causa não foi identificada até o presente momento.

A Figura 4.4 representa medidas de TER das amostras TT1, TT2 e TT3, realizadas com gás CH_4 (P < 1 atm). O comportamento dessas amostras sob gás CH_4 foi o mesmo que aquele apresentado quando as medidas foram realizadas em ar atmosférico (Figura 4.3). Em particular, apenas a amostra TT1 apresentou uma diferença no aspecto da curva, na faixa de 250-400 ºC, quando comparada com sua medida em ar atmosférico.

Na Figura 4.5 são apresentados os resultados de TER das amostras TT1, TT2 e TT3, realizadas em ar + gás sob pressão atmosférica. As amostras não apresentaram diferença significativa de resistência elétrica quando comparadas com suas medidas realizadas sem gás em ar atmosférico (Figura 4.3).

Apenas em uma condição foi obtida diferença significativa de resistência elétrica. Esta diferença ocorre quando se compara as curvas da amostra TT1 em ar atmosférico e em gás sob baixa pressão. A Figura 4.6 mostra esta comparação, e a Figura 4.7 mostra a sensibilidade desta amostra para a condição mencionada.

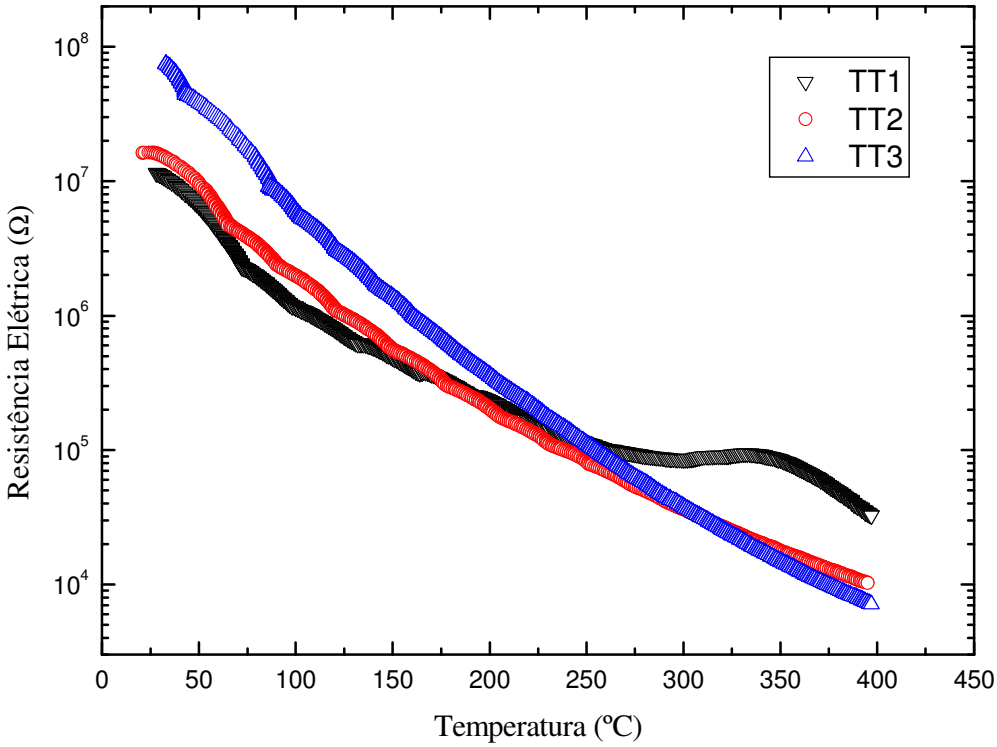

Figura 4.4 – Medidas de TER das amostras TT1, TT2 e TT3 com gás (P < 1 atm).

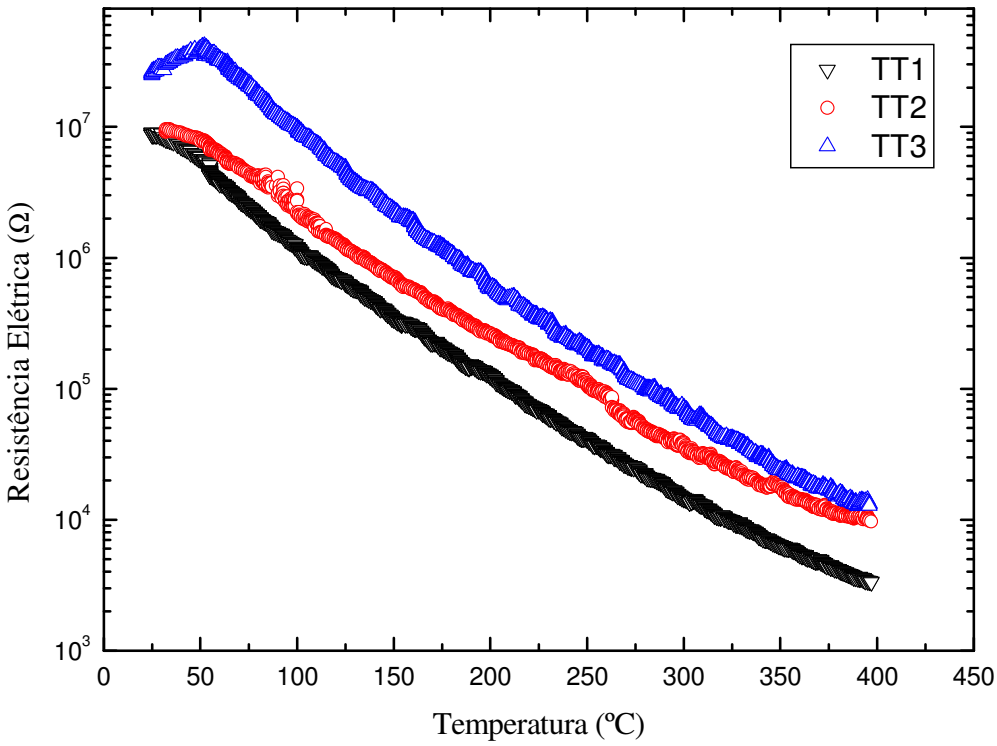

Figura 4.5 – Medidas de TER das amostras TT1, TT2 e TT3 em ar atmosférico com gás (P ≅ 1 atm).

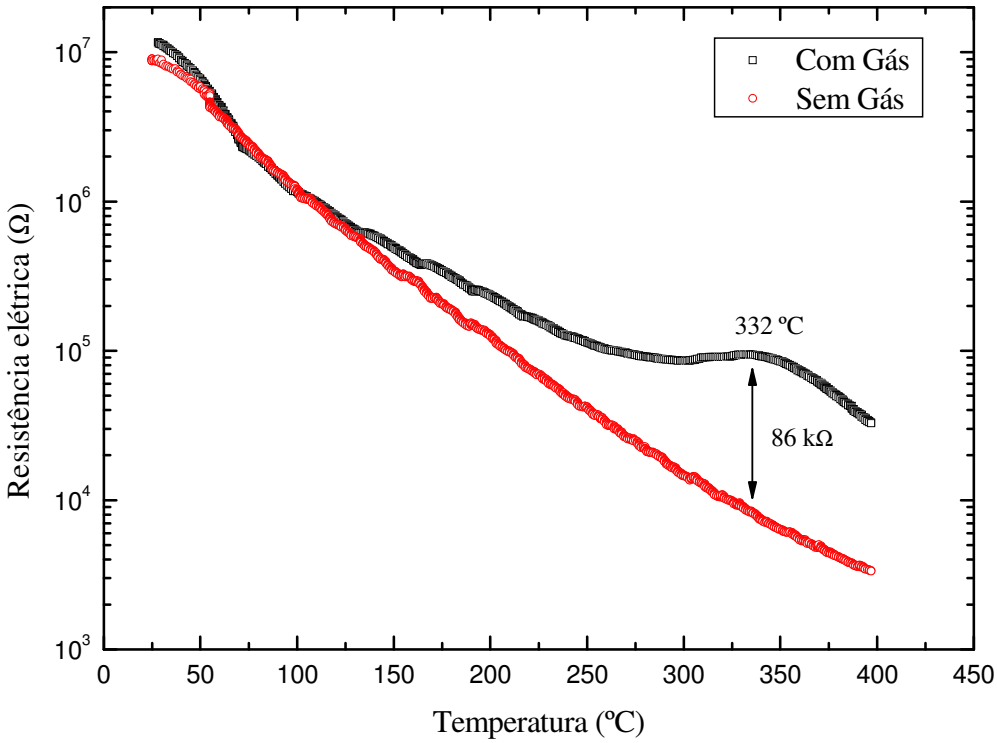

Figura 4.6 – Comparação de medidas de TER da amostra TT1. Em vermelho medida realizada sem gás em ar atmosférico (P ≅ 1 atm); em preto medida realizada com gás sob baixa pressão (P < 1 atm).

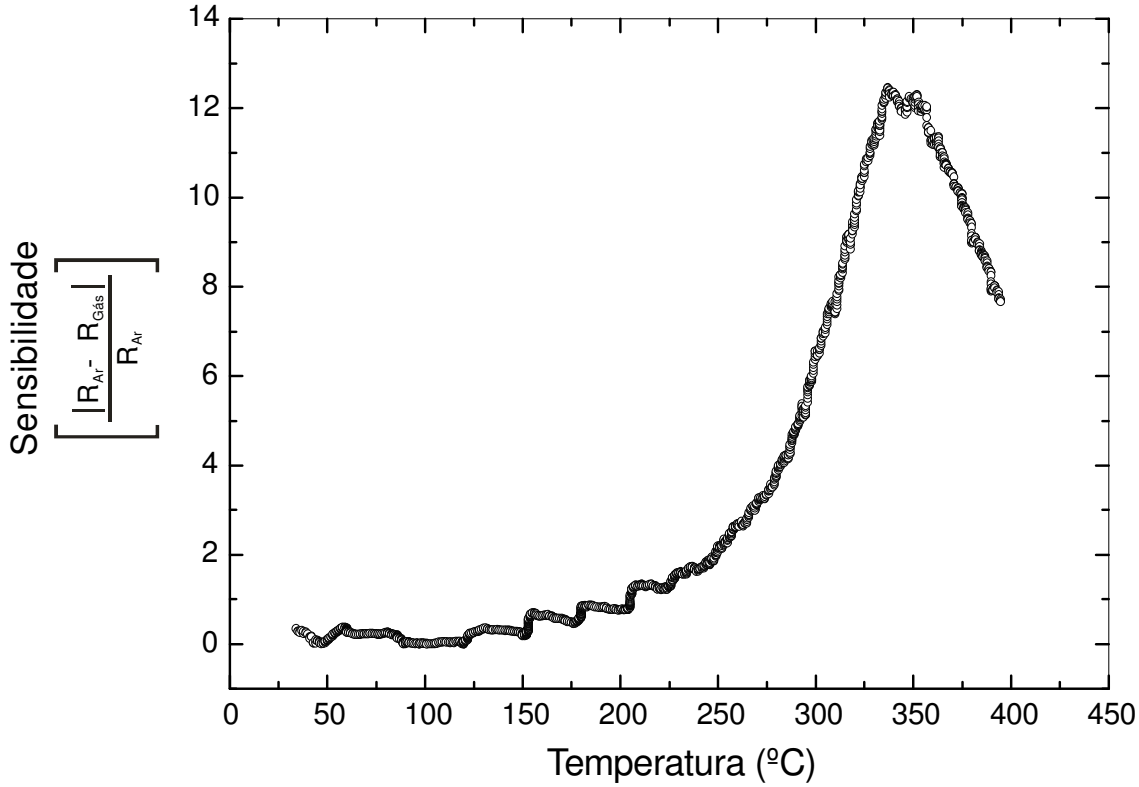

Figura 4.7 - Sensibilidade de amostra TT1.

5 CONCLUSÕES

Com o intuito de testar a sensibilidade de novos sensores de gás do estado sólido, um aparato experimental para medidas de termoeletroresistometria foi planejado, montado e devidamente testado. O sistema permite temperaturas de até 450 ºC com taxas de aquecimento de até 20 ºC/min. O multímetro utilizado nas medidas possui precisão suficiente para captar tanto as menores como as maiores variações de resistência elétrica do material. Além disso, a câmara cilíndrica onde é injetado o gás, não apresenta vazamentos, possibilitando a utilização de gases tóxicos e inflamáveis com relativa segurança.

Para a automatização do sistema, fez-se necessário à elaboração de um programa computacional escrito em pascal, o qual após testes demonstrou atender todas as necessidades exigidas pelo sistema de caracterização. Suas ferramentas adicionais como plotagem de dados em tempo real, programação do controlador via software, e outras aplicações facilitam para o usuário a utilização do sistema. Além disso, essas ferramentas adicionais minimizam a probabilidade de erros.

O método de secagem por liofilização é um método muito eficaz para a obtenção de misturas homogêneas de pós. Foi obtido com sucesso uma razão molar dos cátions de Fe:Mn de aproximadamente 2:1, o que possibilitaria a obtenção da ferrita de manganês via tratamento térmico. Entretanto, o que se observa, após a realização de tratamentos térmicos em ar, são misturas de óxidos, como mostrou a caracterização por raios X.

Dos óxidos obtidos, apenas a amostra TT1 mostrou boa sensibilidade ao gás CH_4, porém sob baixa pressão e na presença exclusiva do gás. Tal condição impossibilita a utilização desta amostra como sensor, afinal de contas um sensor de gás precisa trabalhar em ar atmosférico. Nenhuma das outras misturas de óxidos obtidas (TT2 e TT3) se mostrou eficiente como sensor nas condições testadas. Porém, existe uma infinidade de parâmetros a serem estudados e modificados que podem permitir a criação de um bom sensor do estado sólido, utilizando estes mesmos precursores.

REFERÊNCIAS

ANDRADE, Ricardo L. T.; LINDINO, Cleber A.; SOUSA BULHÕES, Luis O. Sistema com atmosfera controlada para a caracterização de sensors a gases. *Química Nova*, v. 21, n. 3, p. 348-350, 1998.

ARSHAK, K.; GAIDAN, I. NiO/Fe_2O_3 polymer thick films as room temperature gas sensors. *Thin Solid Films*, v. 495, p. 286-291, 2006.

AZEVEDO-RAMOS, Claudia; CARVALHO JR, Oswaldo de; NASI, Robert. *Animal indicators*: a tool to assess biotic integrity after logging tropical forest. Brasília:IPAM, 2003. 69 p.

BELLINI, Jusmar Valentin. *Síntese por liofilização e caracterização de varistores ZnO-CuO-vidro*. 2001. 121 p. Tese (Dotourado)–Departamento de Engenharia de Materiais, Universidade Federal de São Carlos, São Carlos, 2001.

BELLINI, J. V.; MACHADO, R.; MORELLI, M. R.; KIMINAMI, R.H.G.A. Thermal, Structural and morphological characterisation of freeze-dried copper(II) acetate monohydrate and its solid decompositon products. *Materials Research*. v. 5, n. 4, p. 453-457, 2002[a].

BELLINI, J. V.; MORELLI, M. R.; KIMINAMI, R. H.G.A. Electrical properties of polycrystalline ZnO:Cu obtained from freeze-dried ZnO+copper(II) acetate powders. *Journal of Materials Science: Materials in Electronics*. v. 13, p. 485-489, 2002[b].

BELLINI, Jusmar V.; MORELLI, Marcio R.; KIMINAMI, Ruth H.G.A. Physical changes of sintered ceramics obtained from freeze-dried $ZnO+(CH_3COO)_2Cu{\cdot}H_2O$ powders. *Materials Letters*. v. 57, p. 3325-3329, 2003[a].

BELLINI, Jusmar V.; MORELLI, Marcio R.; KIMINAMI, Ruth H.G.A. Ceramic system based on $ZnO{\cdot}CuO$ obtained by freeze-drying. *Materials Letters*. v. 57, p. 3775-3778, 2003[b].

BELLINI, Jusmar Valentin; PINEDA, Edgardo Alfonso Gomez; ROCHA, Raquel de Almeida; PONZONI, Andre Luiz de Lima; PAESANO JR, Andrea. Thermoelectrical and thermal analyses of copper(II) acetate monohydrate ZnO-matrix composite powder obtained by freeze-drying. *Thermochimica Acta*. v. 441, p. 111-115, 2006.

BELLINI, J. V.; MEDEIROS, S. N. de; PONZONI, A. L. L.; LONGEN, F. R.; MELO, M. A. C. de; PAESANO JR, A. Manganese ferrite synthesized from Mn(II) acetate + hematite freeze-dried powders. *Materials Chemistry and Physycs*, 2007 (aceito para publicação).

CANTÚ, MARCO. *Dominando o Delphi 7*, 1ª Edição, São Paulo: Editora Makron Books, 2003, p. 896.

CHOU, Jack. *Hazardous gas monitors*: a pratical guide to selection, operation and applications, New York: McGraw-Hill Book Company, 2000, 258 p.

GOPAL REDDY, C.V.; MANORAMA, S. V.; RAO, V. J. Semiconducting gas sensor for chlorine based on inverse spinel nickel ferrite. *Sensors and Actuators B*, v. 55, p. 90-95, 1999.

JING, Zhihong. Fabrication and gas sensing properties of Ni-doped gamma-Fe_2O_3 by anhydrous solvent method. *Materials Letters*, v. 60, p. 3315-3318, 2006[a].

JING, Zhihong. Ethanol sensing properties of γ-Fe_2O_3 nanopowder doped with Mg by nanoaqueous medium method. *Materials Science and Engineering B*, v 133, p. 213-217, 2006[b].

LIU, Xingqin; XU, Zhengliang; LIU, Yafei; SHEN, Yusheng. A novel high performance ethanol gas sensor based on CdO-Fe_2O_3 semiconducting materials. *Sensors and Actuators B*, v. 52, p. 270-273, 1998.

OSIER, Dan; GROBMAN, Steve; BATSON, Steve. *Aprenda em 14 dias Delphi 3*, 3ª Edição, Rio de Janeiro: Editora Campus, 1998. p. 580.

SATYANARAYANA, L.; MADHUSUDAN REDDY, K.; MANORAMA, Sunkara V. Nanosized spinel $NiFe_2O_4$: A novel material for the detection of liquefied pretroleum gas in air. *Materials Chemistry and Physics*, v. 82, p. 21-26, 2003.

SEIFU, D.; KEBEDE, A.; OLIVER, F. W.; HOFFMAN, E.; HAMMOND, E.; WYNTER, C.; ANING, A.; TAKACS, L.; SIU, I.-L.; WALKER, J. C.; TASSEMA, G.; SEEHRA, M. S. Evidence of ferrimagnetic ordering in $FeMnO_3$ produced by mechanical alloying. *Journal of Magnetism and Magnetic Materials*, v. 212, p. 178-182, 2000.

TAN, O. K.; CAO, W.; HU, Y.; ZHU, W. Nanostructured oxides by high-energy ball milling technique: application as gas sensing materials. *Solid State Ionics*, v. 172, p. 309-316, 2004.

TAO, Shanwen; GAO, Feng; LIU, Xingqin; SØRENSEN, Ole Toft. Preparation and gas-sensing properties of $CuFe_2O_4$ at reduced temperatura. *Materials Science and Engineering B*, v. 77, p. 172-176, 2000.

APÊNDICE A – CÓDIGO FONTE – UNIDADE 1

```
unit Unit1;   // Arquivo principal do programa
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Menus, ExtCtrls, Buttons, jpeg, Registry, Spin, NumEdit, Math,
  Grids;

type TGetTR = class(TThread)
   Private
   Protected
   Procedure Execute; override;
   Procedure GetResistance;
   Procedure GetTemperature;
   Procedure SalvaDados;
   Procedure Separador;
End;

type
  TForm1 = class(TForm)
    Timer1: TTimer;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    SaveDialog1: TSaveDialog;
    GroupBox1: TGroupBox;
    FloatEdit1: TFloatEdit;
    SpinEdit1: TSpinEdit;
    Label3: TLabel;
    CheckBox1: TCheckBox;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    ComboBox1: TComboBox;
    SpeedButton1: TSpeedButton;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    StringGrid1: TStringGrid;
    CheckBox2: TCheckBox;
    procedure Timer1Timer(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);  //Mostra portas COMx.
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
const                          // Constantes do Programa
  LEN_BUFFER = 100;
  BIT0=1;
  BIT1=2;
  BIT2=4;
  BIT3=8;
  BIT4=16;
  BIT5=32;
  BIT6=64;
  BIT7=128;

Var                            // Variáveis do Programa
  Form1                                  : TForm1;
  hCom1,hCom2                            : THANDLE;
  dcb                                    : TDCB;
  CommTimeOuts                           : TCOMMTIMEOUTS;
  BytesEscritosR, BytesEscritosT         : DWORD;
  BytesLidosR   , BytesLidosT            : DWORD;
  BufferRecebeR , BufferRecebeT          : array [0..LEN_BUFFER] of char;
  Resist, Temper, ResistP                : String;
  TempoS                                 : Integer;
  TempoInicial                           : TDateTime;
  Intervalo                                      : Integer;
  Arq                                            : TextFile;
  NroGraf                                : Integer;
  TempoAquisicao                                 : LongInt;
  TempoG1, TempoG2, TempoG3              : LongInt;
  TempoG4, TempoG5                       : LongInt;
  RF, Potencia                           : Real;
  TTP                                    : String;
  Contador, Contador2                            : Integer;
  pd, pu                                 : Integer;

implementation

uses Unit2, Unit3, Unit5, Unit6;
//=================================ABRE A PORTA COM1
Function AbrirCOM1:boolean;
begin
   hCom1 := CreateFile( PChar('COM1'),GENERIC_READ or GENERIC_WRITE, 0, nil,
OPEN_EXISTING,0,0);

   if(hCom1 = INVALID_HANDLE_VALUE) then  //Se houve algum erro ao abrir a porta.
      result := false
   else
      result:= true;
end;
//=================================ABRE A PORTA COM2
Function AbrirCOM2:boolean;
begin
   hCom2 := CreateFile( PChar('COM2'),GENERIC_READ or GENERIC_WRITE, 0, nil,
OPEN_EXISTING,0,0);

   if(hCom2 = INVALID_HANDLE_VALUE) then  //Se houve algum erro ao abrir a porta.
      result := false
   else
      result:= true;
end;
//=================================CONFIGURAÇÃO DA PORTA COM1
Function ConfiguraCOM1:boolean;
```

```
begin
  if not GetCommState(hCom1, dcb) then
    result:= false;
  dcb.BaudRate := CBR_9600;   //define velocidade em bps.
  dcb.ByteSize := 8;                  //define bits de dados.
  dcb.Parity := NOPARITY;      //define paridade
  dcb.StopBits := ONESTOPBIT;  //define stop bit.

  if not SetCommState(hCom1, dcb) then
    result:= false
  else
    result:= true;
end;
//================================CONFIGURAÇÃO DA PORTA COM2
Function ConfiguraCOM2:boolean;
begin
  if not GetCommState(hCom2, dcb) then
    result:= false;

  dcb.BaudRate := CBR_9600;   //define velocidade em bps.
  dcb.ByteSize := 8;          //define bits de dados.
  dcb.Parity := NOPARITY;      //define paridade
  dcb.StopBits := ONESTOPBIT;  //define stop bit.

  if not SetCommState(hCom2, dcb) then
    result:= false
  else
    result:= true;
end;
//================================== DEFINE TIMEOUTs COM1
Function ConfiguraTimeOutsCOM1:boolean;
begin
  if not GetCommTimeouts(hCom1, CommTimeouts) then
    result:= false;
  CommTimeouts.ReadIntervalTimeout := 2;
  CommTimeouts.ReadTotalTimeoutMultiplier := 0;
  CommTimeouts.ReadTotalTimeoutConstant := 2;
  CommTimeouts.WriteTotalTimeoutMultiplier := 5;
  CommTimeouts.WriteTotalTimeoutConstant := 5;

  if not SetCommTimeouts(hCom1, CommTimeouts) then
    result:= false
  else
    result:= true;
end;
//================================== DEFINE TIMEOUTs COM2
Function ConfiguraTimeOutsCOM2:boolean;
begin
  if not GetCommTimeouts(hCom2, CommTimeouts) then
    result:= false;

  CommTimeouts.ReadIntervalTimeout := 2;
  CommTimeouts.ReadTotalTimeoutMultiplier := 0;
  CommTimeouts.ReadTotalTimeoutConstant := 2;
  CommTimeouts.WriteTotalTimeoutMultiplier := 5;
  CommTimeouts.WriteTotalTimeoutConstant := 5;

  if not SetCommTimeouts(hCom2, CommTimeouts) then
    result:= false
  else
```

```
     result:= true;
end;
//================================INICIANDO COMUNICAÇÃO
Procedure TGetTR.Execute;
Var  X    : Integer;
    SR,SL: String;
Begin
    Form1.BitBtn1.Enabled:=false;     // Desabilita o botão da Aquisição
    pd:=0; pu:=0;
//============================== Deixando O Multímetro no Modo Remoto

  SR:='System:Remote' + #13#10;  //concatena CR/LF ao final da String a ser enviada.
  WriteFile( hCom2,PChar(SR)^,15, BytesEscritosR, nil); //Envia string.

//============================== Determinando a Resistencia Fixa (se houver)

   IF (Form1.CheckBox1.Checked=True) Then    // Se estiver medindo resistencia equivalente
   Begin
       IF (Form1.ComboBox1.Text='Ohms') Then Potencia:=0;
       IF (Form1.ComboBox1.Text='KOhms') Then Potencia:=3;
       IF (Form1.ComboBox1.Text='MOhms') Then Potencia:=6;
       RF:=Form1.FloatEdit1.Value*Power(10,Potencia);
   End;
//================================   Looping da Aquisição
    X:=0;
    Form2.Caption:='Gráfico [Adquirindo Segmento 1]';
    while (TempoS<TempoAquisicao) do begin
       GetResistance;
       GetTemperature;
       SalvaDados;
       If (StrToFloat(Resist)>0) Then
       Begin
          Form1.StringGrid1.Cells[0,Contador2]:=Resist;
          Form1.StringGrid1.Cells[1,Contador2]:=Temper;
          Form1.StringGrid1.Cells[2,Contador2]:=IntToStr(TempoS);
          IF (Form1.CheckBox2.Checked=True) Then
          Begin
            Form1.StringGrid1.Row:=Contador2;
          End;
          pu:=pu+1;
          Form1.Label7.Caption:=IntToStr(PU);
       End Else Begin
          pd:=pd+1;
          Form1.Label5.Caption:=IntToStr(PD);
       End;

       contador2:=contador2+1;
//============================== Se estiver dentro do Tempo do Segmento 1
       IF (TempoS<TempoG1) and (StrToFloat(Resist)>0) Then
       Begin
          IF (VTemper1<>VTemper2) Then   // Se for Rampa
          Begin
              Form2.Series1.AddXY(StrToFloat(Temper),StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
          End;
          IF (VTemper1=VTemper2) Then   // Se for Patamar
          Begin
              Form2.Series1.AddXY((TempoS/60) ,StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
          End;
```

```
        End;
//============================== Se estiver dentro do Tempo do Segmento 2
        IF (TempoS>TempoG1) AND (TempoS<(TempoG1+TempoG2)) AND (NroGraf>=2) AND
(StrToFloat(Resist)>0) Then
        Begin
           Form2.Caption:='Gráfico [Adquirindo Segmento 2]';
           IF (VTemper2<>VTemper3) Then   // Se for Rampa
           Begin
              Form2.Series2.AddXY(StrToFloat(Temper),StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
           End;
           IF (VTemper2=VTemper3) Then   // Se for Patamar
           Begin
              Form2.Series2.AddXY(((TempoS-TempoG1)/60) ,StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
           End;
         End;
//============================== Se estiver dentro do Tempo do Segmento 3
        IF (TempoS>(TempoG1+TempoG2)) AND (TempoS<(TempoG1+TempoG2+TempoG3)) AND
(NroGraf>=3) AND (StrToFloat(Resist)>0) Then
        Begin
           Form2.Caption:='Gráfico [Adquirindo Segmento 3]';
           IF (VTemper3<>VTemper4) Then   // Se for Rampa
           Begin
              Form2.Series3.AddXY(StrToFloat(Temper),StrToFloat(Resist));                IF (intervalo<>0) then
sleep(intervalo);
           End;
           IF (VTemper3=VTemper4) Then   // Se for Patamar
           Begin
              Form2.Series3.AddXY(((TempoS-TempoG1-TempoG2)/60) ,StrToFloat(Resist));                IF
(intervalo<>0) then sleep(intervalo);
           End;
        End;
//============================== Se estiver dentro do Tempo do Segmento 4
        IF (TempoS>(TempoG1+TempoG2+TempoG3)) AND
(TempoS<(TempoG1+TempoG2+TempoG3+TempoG4)) AND (NroGraf>=4) AND (StrToFloat(Resist)>0)
Then
        Begin
           Form2.Caption:='Gráfico [Adquirindo Segmento 4]';
           IF (VTemper4<>VTemper5) Then   // Se for Rampa
           Begin
              Form2.Series4.AddXY(StrToFloat(Temper),StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
           End;
           IF (VTemper4=VTemper5) Then   // Se for Patamar
           Begin
              Form2.Series4.AddXY(((TempoS-TempoG1-TempoG2-TempoG3)/60) ,StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
           End;
        End;
//============================== Se estiver dentro do Tempo do Segmento 5
        IF (TempoS>(TempoG1+TempoG2+TempoG3+TempoG4)) AND
(TempoS<(TempoG1+TempoG2+TempoG3+TempoG4+TempoG5)) AND (NroGraf=5) AND
(StrToFloat(Resist)>0) Then
        Begin
           Form2.Caption:='Gráfico [Adquirindo Segmento 5]';
           IF (VTemper5<>VTemper6) Then   // Se for Rampa
           Begin
              Form2.Series5.AddXY(StrToFloat(Temper),StrToFloat(Resist));
              IF (intervalo<>0) then sleep(intervalo);
```

```
          End;
          IF (VTemper5=VTemper6) Then   // Se for Patamar
          Begin
             Form2.Series5.AddXY(((TempoS-TempoG1-TempoG2-TempoG3-TempoG4)/60)
,StrToFloat(Resist));
             IF (intervalo<>0) then sleep(intervalo);
          End;
       End;
    end;   // Fim do Looping
    form1.Timer1.Enabled:=False;   // Desliga o Cronômetro !
//================================ Deixar o multímetro no modo local
    sleep(500);    //espera meio segundo!
    SL:= 'System:Local' + #13#10;  //concatena CR/LF ao final da String a ser enviada.
    WriteFile( hCom2,PChar(SL)^,14, BytesEscritosR, nil); //Envia string.

//================================ Fechando as Portas Seriais

    CloseHandle(hCom1);        // Fecha a porta serial COM1;
    CloseHandle(hCom2);        // Fecha a porta serial COM2.

//================================ Atividades necessárias

    Form1.Edit1.Text:='';                  // Limpa o nome do Arquivo
    Form1.SaveDialog1.FileName:='';        // Limpa o nome do Arquivo
    Form1.SpinEdit1.Enabled:=True;         // Habilita a Seleção de Intervalo temporal !
    Form1.BitBtn3.Enabled:=True;           // Habilita o botão de acesso a rampas e patamares !
    Form1.BitBtn4.Enabled:=True;           // Habilita o botão de seleção de arquivo
    Form1.CheckBox1.Enabled:=True;         // Habilita o checkbox !
    Form1.ComboBox1.Enabled:=True;         // Habilita o ComboBox !
    Form1.FloatEdit1.Enabled:=True;        // Habilita o FloatEdit!
    Form1.SpeedButton1.Enabled:=True;      // Habilita o Botão Sobre o Programa
    Form1.Edit1.Text:='';
    Showmessage('Aquisição concluída !');
End;
//=======================SALVANDO NO ARQUIVO

Procedure TGetTr.SalvaDados;
Begin
  If (StrToFloat(Resist)>0) Then Begin
    AssignFile(Arq,Form1.Edit1.Text);
    Append(Arq);
    WriteLn(Arq, Resist + ' ' + Temper + ' ' + IntToStr(TempoS));
    CloseFile(Arq);
    IF (TempoS>TempoG1)                          AND (Contador=0) Then Separador;
    IF (TempoS>(TempoG1+TempoG2))                AND (Contador=1) Then Separador;
    IF (TempoS>(TempoG1+TempoG2+TempoG3))        AND (Contador=2) Then Separador;
    IF (TempoS>(TempoG1+TempoG2+TempoG3+TempoG4))  AND (Contador=3) Then Separador;
    IF (TempoS>(TempoG1+TempoG2+TempoG3+TempoG4+TempoG5)) AND (Contador=4) Then Separador;
  End;
End;
//================ PEDIR E OBTER O VALOR DE RESISTÊNCIA
Procedure TGetTR.GetResistance;
Var MR, Res    : String;
   RA, RE     : Real;    // RA= Resistencia Amostra, RE = Resistencia Equivalente
Begin
     //ENVIANDO MEASURE:RESISTANCE?
       sleep(800);    //espera meio segundo!
       MR:= 'Measure:Resistance?' + #13#10;  //concatena CR/LF ao final da String a ser enviada.
       WriteFile( hCom2,PChar(MR)^,21, BytesEscritosR, nil); //Envia string.
```

```
    //PEGANDO A RESPOSTA
      sleep(800);    //espera 800 mili segundos!
      BytesLidosR := 0;
      Readfile(hCom2, BufferRecebeR, LEN_BUFFER, BytesLidosR, nil);  //Lê a porta Serial.
        if BytesLidosR > 0 then //Se algum caracter foi lido.
        begin
          Res:=BufferRecebeR;
          Res:=copy(Res, 2,14);                  // para obter no formato 8.19747000E+02
          Res:=copy(Res, 1,1)+','+Copy(Res, 3,12);  // Troca o ponto por vírgula, para reconhecer como número
          IF (Form1.CheckBox1.Checked=True) Then    // Se estiver medindo Resist Equiv.
          Begin
             RE:=StrToFloat(res);
             RA:=(1/RE)-(1/RF);
             RA:=(1/RA);
             Res:=FloatToStr(RA);
             Resist:=Res;
          End Else Begin                                  // Caso Contrário
             Resist:=Res;
          End;
        end;
End;
//======================== PEDIR E OBTER TEMPERATURA
Procedure TGetTR.GetTemperature;
Var  n,n2:integer;
Begin
  WriteFile( hCom1,PChar(#2#3#0#74#0#0#100#47)^,8, BytesEscritosT, nil); //Envia string.
  sleep(400);
  BytesLidosT:= 0;
  Readfile(hCom1, BufferRecebeT, LEN_BUFFER, BytesLidosT, nil);  //Lê a porta Serial.
  if BytesLidosT > 0 then     //Se algum caracter foi lido.
  begin
      n:= ORD(BufferRecebeT[5]);
      n2:=ORD(BufferRecebeT[4]);
      n:=n+(n2*256);
      Temper:=IntToStr(n);
   end;
End;

//======================== SEPARADOR DE SEGMENTOS
Procedure TGetTR.Separador;
Begin
     AssignFile(Arq,Form1.Edit1.Text);
     Append(Arq);
     WriteLn(Arq, '---------------------------- FIM DO SEGMENTO ----------------------------------------------');
     CloseFile(Arq);
     Contador:=Contador+1;
End;

Procedure TForm1.Timer1Timer(Sender: TObject);
Var TempoAtual:TDateTime;
   TempoDecorrido:String;
   H,M,S:Integer;
begin
   TempoAtual:=Time;
   TempoDecorrido:=TimeToStr(TempoAtual-TempoInicial);
   Form1.Label2.Caption:=TempoDecorrido;
   Form2.Label17.Caption:=TempoDecorrido;
   H:=StrToInt(Copy(TempoDecorrido,1,2));
   M:=StrToInt(Copy(TempoDecorrido,4,2));
   S:=StrToInt(Copy(TempoDecorrido,7,2));
```

```
   TempoS:=((H*60)+M)*60+S;
end;

procedure TForm1.CheckBox1Click(Sender: TObject);
begin
   IF (Form1.CheckBox1.Checked=True) Then
   Begin
      Form1.FloatEdit1.Enabled:=True;
      Form1.ComboBox1.Enabled:=True;
   End Else Begin
      Form1.FloatEdit1.Enabled:=False;
      Form1.ComboBox1.Enabled:=False;
   End;
end;

procedure TForm1.BitBtn1Click(Sender: TObject);
Var PCOM1, PCOM2      :Boolean;
   sucesso1,sucesso2 :Boolean;
begin
  TempoInicial:=Time;
  form1.Timer1.Enabled:=true;
  Form1.CheckBox1.Enabled:=False;
  Form1.FloatEdit1.Enabled:=False;
  Form1.ComboBox1.Enabled:=False;
  Form1.SpeedButton1.Enabled:=False;
  SpinEdit1.Enabled:=False;
  Form1.BitBtn3.Enabled:=False;
  Form1.BitBtn4.Enabled:=False;
  Intervalo:=(SpinEdit1.Value-2)*1000;
  TempoS:=0;
  Contador:=0;
  Contador2:=1;        // Define o contador para a primeira linha do StringGrid

  Form1.StringGrid1.RowCount:=(TempoAquisicao DIV Form1.SpinEdit1.Value)+5;

  IF (AbrirCom1=True) Then PCOM1:=True          // Abre COM1
    Else ShowMessage('Erro ao abrir a Porta COM1, confira as configurações');
  IF (AbrirCom2=True) Then PCOM2:=True          // Abre COM2
    Else ShowMessage('Erro ao abrir a Porta COM2, confira as configurações');

  IF (PCOM1=True) And (PCOM2=True) Then      // Se as duas portas foram abertas corretamente
  Begin
    sucesso1:=ConfiguraCOM1();
    sucesso1:=ConfiguraTimeOutsCOM1();        //    Configurando as duas portas!
    sucesso2:=ConfiguraCOM2();
    sucesso2:=ConfiguraTimeOutsCOM2();
    IF (sucesso1=False) Then                  //    Se teve erro ao configurar COM1
    Begin
       CloseHandle(hCom1);                    // Fecha a porta serial COM1;
       CloseHandle(hCom2);                    // Fecha a porta serial COM2.
       ShowMessage('Outro programa já está usando, ou a porta COM1 não existe!');
    End;
    IF (sucesso2=False) Then                  //    Se teve erro ao configurar COM2
    Begin
       CloseHandle(hCom1);                    // Fecha a porta serial COM1;
       CloseHandle(hCom2);                    // Fecha a porta serial COM2.
       ShowMessage('Outro programa já está usando, ou a porta COM2 não existe!');
    End;
    IF (sucesso1=True) AND (sucesso2=True) Then   // Se as duas portas foram configuradas corretamente
         TGetTR.Create(False)                 // Inicia a Aquisição
```

```
  End;
end;

//=========================ABRINDO O GRÁFICO
procedure TForm1.BitBtn2Click(Sender: TObject);
begin
  Form2.ComboBox2.Items.Clear;
  IF (NroGraf=1) Then Form2.ComboBox2.Items.Add('Segmento 1');
  IF (NroGraf=2) Then
  Begin
    Form2.ComboBox2.Items.Add('Segmento 1');
    Form2.ComboBox2.Items.Add('Segmento 2');
  End;
  IF (NroGraf=3) Then
  Begin
    Form2.ComboBox2.Items.Add('Segmento 1');
    Form2.ComboBox2.Items.Add('Segmento 2');
    Form2.ComboBox2.Items.Add('Segmento 3');
  End;
  IF (NroGraf=4) Then
  Begin
    Form2.ComboBox2.Items.Add('Segmento 1');
    Form2.ComboBox2.Items.Add('Segmento 2');
    Form2.ComboBox2.Items.Add('Segmento 3');
    Form2.ComboBox2.Items.Add('Segmento 4');
  End;
  IF (NroGraf=5) Then
  Begin
    Form2.ComboBox2.Items.Add('Segmento 1');
    Form2.ComboBox2.Items.Add('Segmento 2');
    Form2.ComboBox2.Items.Add('Segmento 3');
    Form2.ComboBox2.Items.Add('Segmento 4');
    Form2.ComboBox2.Items.Add('Segmento 5');
  End;
  Form2.ComboBox2.Text:='Segmento 1';
  IF (VTempo1<>0) Then
  Begin
    Form2.SPI.Caption:=IntToStr(VTemper1)+' ºC';
    Form2.SPF.Caption:=IntToStr(VTemper2)+' ºC';
    Form2.Taxa.Caption:=IntToStr((VTemper2-VTemper1) DIV VTempo1)+' ºC/min';
    IF VTemper1<>VTemper2 Then Form2.Tipo.Caption:='Rampa' Else Form2.Tipo.Caption:='Patamar';
    Form2.temposeg.caption:=IntToStr(VTempo1)+' min';
    Form2.Label19.Caption:=TTP+':00';
  End;
  form2.show;           // Exibe a janela do gráfico
end;

procedure TForm1.BitBtn3Click(Sender: TObject);
begin
  form5.showmodal;    // Abre a janela para programação de rampas e patamares
end;

procedure TForm1.BitBtn4Click(Sender: TObject);  // Define o arquivo a ser salvo
begin
  savedialog1.Execute;
  edit1.Text:=savedialog1.FileName;
  IF (Edit1.Text<>'') Then
  Begin
    Rewrite(Arq,Edit1.Text);
    CloseFile(Arq);
```

```
    IF (NroGraf<>0) Then Form1.BitBtn1.Enabled:=True;
  End;
end;

procedure TForm1.ComboBox1Change(Sender: TObject);
begin
   If (Form1.ComboBox1.Text<>'Ohms') AND
      (Form1.ComboBox1.Text<>'KOhms') AND
      (Form1.ComboBox1.Text<>'MOhms')
   Then Form1.ComboBox1.Text:='MOhms';
end;

procedure TForm1.SpeedButton1Click(Sender: TObject);
begin
   Form6.Showmodal;       // Abre a janela sobre o programa
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  Form1.StringGrid1.Cells[0,0]:='Resistência (Ohms)';
  Form1.StringGrid1.Cells[1,0]:='Temperatura (ºC)';
  Form1.StringGrid1.Cells[2,0]:='Tempo (s)';
end;

end.
```

APÊNDICE B – CÓDIGO FONTE – UNIDADE 2

```
unit Unit2;  // Janela do Gráfico

interface

uses           // Bibliotecas necessárias
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, TeeProcs, TeEngine, Chart, Series, Spin,
  ColorGrd, Mask, NumEdit, Buttons;

type
  TForm2 = class(TForm)
    Chart1: TChart;
    Series1: TPointSeries;
    GroupBox1: TGroupBox;
    RadioButton1: TRadioButton;
    RadioButton2: TRadioButton;
    GroupBox2: TGroupBox;
    ComboBox1: TComboBox;
    Label4: TLabel;
    Label5: TLabel;
    SpinEdit1: TSpinEdit;
    ColorBox1: TColorBox;
    Label6: TLabel;
    CheckBox2: TCheckBox;
    FloatEdit1: TFloatEdit;
    FloatEdit2: TFloatEdit;
    FloatEdit3: TFloatEdit;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Button4: TButton;
    GroupBox3: TGroupBox;
    RadioButton3: TRadioButton;
    RadioButton4: TRadioButton;
    CheckBox3: TCheckBox;
    Label7: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    FloatEdit4: TFloatEdit;
    FloatEdit5: TFloatEdit;
    FloatEdit6: TFloatEdit;
    Button3: TButton;
    GroupBox4: TGroupBox;
    ComboBox2: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    SPI: TLabel;
    SPF: TLabel;
    Taxa: TLabel;
    Tipo: TLabel;
    temposeg: TLabel;
    Series2: TPointSeries;
    Series3: TPointSeries;
```

```
    Series4: TPointSeries;
    Series5: TPointSeries;
    Label16: TLabel;
    Label17: TLabel;
    BitBtn1: TBitBtn;
    Label18: TLabel;
    Label19: TLabel;
    BitBtn2: TBitBtn;
    salvargrafico: TSaveDialog;
    procedure RadioButton1Click(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure SpinEdit1Change(Sender: TObject);
    procedure ColorBox1Change(Sender: TObject);
    procedure RadioButton2Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure CheckBox2Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure RadioButton3Click(Sender: TObject);
    procedure RadioButton4Click(Sender: TObject);
    procedure CheckBox3Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure ComboBox2Change(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;
  VT   : Integer;   //Variável de Tempo, para definir o gráfico

implementation

Uses Unit1, Unit5;
{$R *.dfm}

procedure TForm2.RadioButton1Click(Sender: TObject);
begin
   If (RadioButton1.Checked=true) then
      Chart1.LeftAxis.Logarithmic:=false;
   If (RadioButton2.Checked=true) then
      Chart1.LeftAxis.Logarithmic:=true;
end;

procedure TForm2.ComboBox1Change(Sender: TObject);
begin

// caso string inválida..
    IF (Combobox1.text<>'Círculo')   AND (Combobox1.text<>'Quadrado') AND
       (Combobox1.text<>'Triangulo') AND (Combobox1.text<>'Nabla')    AND
       (Combobox1.text<>'Cruz')      AND (Combobox1.text<>'X')        AND
       (Combobox1.text<>'Estrela')   AND (Combobox1.text<>'Losango')  AND
       (Combobox1.text<>'Ponto')
    then begin Combobox1.Text:='Círculo'; series1.Pointer.Style:=psCircle; end;
    IF (Form2.ComboBox2.Text='Segmento 1') Then
```

```
Begin
  IF (Combobox1.text='Círculo')   then series1.Pointer.Style:=psCircle;
  IF (Combobox1.text='Quadrado')  then series1.Pointer.Style:=psRectangle;
  IF (Combobox1.text='Triangulo') then series1.Pointer.Style:=psTriangle;
  IF (Combobox1.text='Nabla')     then series1.Pointer.Style:=psDownTriangle;
  IF (Combobox1.text='Cruz')      then series1.Pointer.Style:=psCross;
  IF (Combobox1.text='X')         then series1.Pointer.Style:=psDiagCross;
  IF (Combobox1.text='Estrela')   then series1.Pointer.Style:=psStar;
  IF (Combobox1.text='Losango')   then series1.Pointer.Style:=psDiamond;
  IF (Combobox1.text='Ponto')     then series1.Pointer.Style:=psSmallDot;
End;
IF (Form2.ComboBox2.Text='Segmento 2') Then
Begin
  IF (Combobox1.text='Círculo')   then series2.Pointer.Style:=psCircle;
  IF (Combobox1.text='Quadrado')  then series2.Pointer.Style:=psRectangle;
  IF (Combobox1.text='Triangulo') then series2.Pointer.Style:=psTriangle;
  IF (Combobox1.text='Nabla')     then series2.Pointer.Style:=psDownTriangle;
  IF (Combobox1.text='Cruz')      then series2.Pointer.Style:=psCross;
  IF (Combobox1.text='X')         then series2.Pointer.Style:=psDiagCross;
  IF (Combobox1.text='Estrela')   then series2.Pointer.Style:=psStar;
  IF (Combobox1.text='Losango')   then series2.Pointer.Style:=psDiamond;
  IF (Combobox1.text='Ponto')     then series2.Pointer.Style:=psSmallDot;
End;
IF (Form2.ComboBox2.Text='Segmento 3') Then
Begin
  IF (Combobox1.text='Círculo')   then series3.Pointer.Style:=psCircle;
  IF (Combobox1.text='Quadrado')  then series3.Pointer.Style:=psRectangle;
  IF (Combobox1.text='Triangulo') then series3.Pointer.Style:=psTriangle;
  IF (Combobox1.text='Nabla')     then series3.Pointer.Style:=psDownTriangle;
  IF (Combobox1.text='Cruz')      then series3.Pointer.Style:=psCross;
  IF (Combobox1.text='X')         then series3.Pointer.Style:=psDiagCross;
  IF (Combobox1.text='Estrela')   then series3.Pointer.Style:=psStar;
  IF (Combobox1.text='Losango')   then series3.Pointer.Style:=psDiamond;
  IF (Combobox1.text='Ponto')     then series3.Pointer.Style:=psSmallDot;
End;
IF (Form2.ComboBox2.Text='Segmento 4') Then
Begin
  IF (Combobox1.text='Círculo')   then series4.Pointer.Style:=psCircle;
  IF (Combobox1.text='Quadrado')  then series4.Pointer.Style:=psRectangle;
  IF (Combobox1.text='Triangulo') then series4.Pointer.Style:=psTriangle;
  IF (Combobox1.text='Nabla')     then series4.Pointer.Style:=psDownTriangle;
  IF (Combobox1.text='Cruz')      then series4.Pointer.Style:=psCross;
  IF (Combobox1.text='X')         then series4.Pointer.Style:=psDiagCross;
  IF (Combobox1.text='Estrela')   then series4.Pointer.Style:=psStar;
  IF (Combobox1.text='Losango')   then series4.Pointer.Style:=psDiamond;
  IF (Combobox1.text='Ponto')     then series4.Pointer.Style:=psSmallDot;
End;

IF (Form2.ComboBox2.Text='Segmento 5') Then
Begin
  IF (Combobox1.text='Círculo')   then series5.Pointer.Style:=psCircle;
  IF (Combobox1.text='Quadrado')  then series5.Pointer.Style:=psRectangle;
  IF (Combobox1.text='Triangulo') then series5.Pointer.Style:=psTriangle;
  IF (Combobox1.text='Nabla')     then series5.Pointer.Style:=psDownTriangle;
  IF (Combobox1.text='Cruz')      then series5.Pointer.Style:=psCross;
  IF (Combobox1.text='X')         then series5.Pointer.Style:=psDiagCross;
  IF (Combobox1.text='Estrela')   then series5.Pointer.Style:=psStar;
  IF (Combobox1.text='Losango')   then series5.Pointer.Style:=psDiamond;
  IF (Combobox1.text='Ponto')     then series5.Pointer.Style:=psSmallDot;
End;
```

```
end;

procedure TForm2.SpinEdit1Change(Sender: TObject);
begin
  IF (Form2.ComboBox2.Text='Segmento 1') Then
  Begin
     series1.Pointer.HorizSize:=form2.SpinEdit1.value;
     series1.Pointer.VertSize :=form2.SpinEdit1.value;
  End;
  IF (Form2.ComboBox2.Text='Segmento 2') Then
  Begin
     series2.Pointer.HorizSize:=form2.SpinEdit1.value;
     series2.Pointer.VertSize :=form2.SpinEdit1.value;
  End;
  IF (Form2.ComboBox2.Text='Segmento 3') Then
  Begin
     series3.Pointer.HorizSize:=form2.SpinEdit1.value;
     series3.Pointer.VertSize :=form2.SpinEdit1.value;
  End;
  IF (Form2.ComboBox2.Text='Segmento 4') Then
  Begin
     series4.Pointer.HorizSize:=form2.SpinEdit1.value;
     series4.Pointer.VertSize :=form2.SpinEdit1.value;
  End;
  IF (Form2.ComboBox2.Text='Segmento 5') Then
  Begin
     series5.Pointer.HorizSize:=form2.SpinEdit1.value;
     series5.Pointer.VertSize :=form2.SpinEdit1.value;
  End;
end;

procedure TForm2.ColorBox1Change(Sender: TObject);
begin
  IF (Form2.ComboBox2.Text='Segmento 1') Then series1.Pointer.Brush.Color:=colorbox1.Selected;
  IF (Form2.ComboBox2.Text='Segmento 2') Then series2.Pointer.Brush.Color:=colorbox1.Selected;
  IF (Form2.ComboBox2.Text='Segmento 3') Then series3.Pointer.Brush.Color:=colorbox1.Selected;
  IF (Form2.ComboBox2.Text='Segmento 4') Then series4.Pointer.Brush.Color:=colorbox1.Selected;
  IF (Form2.ComboBox2.Text='Segmento 5') Then series5.Pointer.Brush.Color:=colorbox1.Selected;
end;

procedure TForm2.RadioButton2Click(Sender: TObject);
begin
   If (RadioButton1.Checked=true) then
      Chart1.LeftAxis.Logarithmic:=false;
   If (RadioButton2.Checked=true) then
      Chart1.LeftAxis.Logarithmic:=true;
end;

procedure TForm2.FormCreate(Sender: TObject);
begin
   series1.Pointer.Brush.Color:=ClBlue; Form2.ColorBox1.Selected:=ClBlue;
   Series2.Pointer.Brush.Color:=clgreen;
   Series3.Pointer.Brush.Color:=clyellow;
   Series4.Pointer.Brush.Color:=ClBlue;
   Series5.Pointer.Brush.Color:=ClWhite;
end;

procedure TForm2.CheckBox2Click(Sender: TObject);
begin
   IF (checkbox2.checked=true) then
```

```
  Begin
    FloatEdit1.Enabled:=false;
    FloatEdit2.Enabled:=false;
    FloatEdit3.Enabled:=false;
    Button4.Enabled:=false;
    chart1.LeftAxis.Automatic:=True;  // escala automatica
    chart1.LeftAxis.Increment:=0;

  End Else Begin
    FloatEdit1.Enabled:=True;
    FloatEdit2.Enabled:=True;
    FloatEdit3.Enabled:=True;
    Button4.Enabled:=True;
    chart1.LeftAxis.Automatic:=False;
    chart1.LeftAxis.AutomaticMaximum:=False;
    chart1.LeftAxis.AutomaticMinimum:=False;
    FloatEdit1.Text:=FloatToStr(chart1.LeftAxis.Maximum);
    FloatEdit2.Text:=FloatToStr(chart1.LeftAxis.Minimum);
    FloatEdit3.Text:=FloatToStr(chart1.LeftAxis.Increment);

  End;


end;

procedure TForm2.Button4Click(Sender: TObject);
begin
 IF (floatEdit1.text>FloatEdit2.text) Then
 Begin
    Chart1.LeftAxis.Maximum:=StrToFloat(FloatEdit1.Text);
    Chart1.LeftAxis.Minimum:=StrToFloat(FloatEdit2.Text);
    Chart1.LeftAxis.Increment:=StrToFloat(FloatEdit3.Text);
 End Else
 Begin
    showmessage('O Limite Superior deve ser Maior que o Inferior!');
    FloatEdit2.Text:='0';
    FloatEdit2.SetFocus;
 End;



end;

procedure TForm2.RadioButton3Click(Sender: TObject);
begin
  If (RadioButton4.Checked=true) then
     Chart1.BottomAxis.Logarithmic:=false;
  If (RadioButton3.Checked=true) then
     Chart1.BottomAxis.Logarithmic:=true;
end;

procedure TForm2.RadioButton4Click(Sender: TObject);
begin
  If (RadioButton4.Checked=true) then
     Chart1.BottomAxis.Logarithmic:=false;
  If (RadioButton3.Checked=true) then
     Chart1.BottomAxis.Logarithmic:=true;
end;

procedure TForm2.CheckBox3Click(Sender: TObject);
```

```
begin
  IF (checkbox3.checked=true) then
  Begin
    FloatEdit4.Enabled:=false;
    FloatEdit5.Enabled:=false;
    FloatEdit6.Enabled:=false;
    Button3.Enabled:=false;
    chart1.BottomAxis.Automatic:=True;  // escala automatica
    chart1.BottomAxis.Increment:=0;

  End Else Begin
    FloatEdit4.Enabled:=True;
    FloatEdit5.Enabled:=True;
    FloatEdit6.Enabled:=True;
    Button3.Enabled:=True;
    chart1.BottomAxis.Automatic:=False;
    chart1.BottomAxis.AutomaticMaximum:=False;
    chart1.BottomAxis.AutomaticMinimum:=False;
    FloatEdit4.Text:=FloatToStr(chart1.BottomAxis.Maximum);
    FloatEdit5.Text:=FloatToStr(chart1.BottomAxis.Minimum);
    FloatEdit6.Text:=FloatToStr(chart1.BottomAxis.Increment);
  End;


end;

procedure TForm2.Button3Click(Sender: TObject);
begin
  IF FloatEdit4.Text>FloatEdit5.text then
  Begin
     Chart1.BottomAxis.Maximum:=StrToFloat(FloatEdit4.Text);
     Chart1.BottomAxis.Minimum:=StrToFloat(FloatEdit5.Text);
     Chart1.BottomAxis.Increment:=StrToFloat(FloatEdit6.Text);
  End else
  Begin
    showmessage('O Limite Superior deve ser Maior que o Inferior!');
    FloatEdit5.Text:='0';
    FloatEdit5.SetFocus;
  End;
end;

Procedure DefineGrafico;
Begin
    IF (Form2.Tipo.Caption='Rampa') Then
    Begin
      Form2.GroupBox2.Caption:='Escala de Temperatura: ';
      Form2.Chart1.BottomAxis.Title.Caption:='Temperatura (ºC)';
      Form2.RadioButton4.Checked:=True;
      Form2.Chart1.BottomAxis.Logarithmic:=False;
      Form2.CheckBox3.Checked:=False;
      Form2.Chart1.BottomAxis.Automatic:=False;
      Form2.Chart1.BottomAxis.Maximum:=500;
      Form2.Chart1.BottomAxis.Minimum:=0;
      Form2.Chart1.BottomAxis.Increment:=25;
      Form2.FloatEdit4.Value:=500;
      Form2.FloatEdit5.Value:=0;
      Form2.FloatEdit6.Value:=25;
    End Else
    Begin
      Form2.GroupBox2.Caption:='Escala de Tempo: ';
```

```
        Form2.Chart1.BottomAxis.Title.Caption:='Tempo (min)';
        Form2.RadioButton4.Checked:=True;
        Form2.Chart1.BottomAxis.Logarithmic:=False;
        Form2.CheckBox3.Checked:=False;
        Form2.Chart1.BottomAxis.Automatic:=False;
        Form2.Chart1.BottomAxis.Maximum:=VT;
        Form2.Chart1.BottomAxis.Minimum:=0;
        IF (VT<=30) Then
        Begin
            Form2.Chart1.BottomAxis.Increment:=2;
            Form2.FloatEdit6.Value:=2;
        End Else Begin
            Form2.Chart1.BottomAxis.Increment:=5;
            Form2.FloatEdit6.Value:=5;
        End;
        Form2.FloatEdit4.Value:=VT;
        Form2.FloatEdit5.Value:=0;
      End;
End;

procedure TForm2.ComboBox2Change(Sender: TObject);
begin
  IF (Form2.ComboBox2.Text<>'Segmento 1') and
     (Form2.ComboBox2.Text<>'Segmento 2') and
     (Form2.ComboBox2.Text<>'Segmento 3') and
     (Form2.ComboBox2.Text<>'Segmento 4') and
     (Form2.ComboBox2.Text<>'Segmento 5') Then
     Form2.ComboBox2.Text:='Segmento 1';

  IF (Form2.ComboBox2.Text='Segmento 1') AND (VTempo1<>0) Then
  Begin
     Form2.SPI.Caption:=IntToStr(VTemper1)+' ºC';
     Form2.SPF.Caption:=IntToStr(VTemper2)+' ºC';
     Form2.Taxa.Caption:=IntToStr((VTemper2-VTemper1) DIV VTempo1)+' ºC/min';
     IF VTemper1<>VTemper2 Then Form2.Tipo.Caption:='Rampa' Else Form2.Tipo.Caption:='Patamar';
     Form2.temposeg.caption:=IntToStr(VTempo1)+' min';
//   Deixando o Somente o Segmento 1 Visível
     Form2.Series1.Pointer.Visible:=True;  Form2.Series2.Pointer.Visible:=False;
Form2.Series3.Pointer.Visible:=False;
     Form2.Series4.Pointer.Visible:=False; Form2.Series5.Pointer.Visible:=False;
//   Carregando o Formato do Ponto
     IF (Form2.Series1.Pointer.Style=psCircle)       Then Form2.Combobox1.text:='Círculo';
     IF (Form2.Series1.Pointer.Style=psRectangle)    Then Form2.Combobox1.text:='Quadrado';
     IF (Form2.Series1.Pointer.Style=psTriangle)     Then Form2.Combobox1.text:='Triangulo';
     IF (Form2.Series1.Pointer.Style=psDownTriangle) Then Form2.Combobox1.text:='Nabla';
     IF (Form2.Series1.Pointer.Style=psCross)        Then Form2.Combobox1.text:='Cruz';
     IF (Form2.Series1.Pointer.Style=psDiagCross)    Then Form2.Combobox1.text:='X';
     IF (Form2.Series1.Pointer.Style=psStar)         Then Form2.Combobox1.text:='Estrela';
     IF (Form2.Series1.Pointer.Style=psDiamond)      Then Form2.Combobox1.text:='Losango';
     IF (Form2.Series1.Pointer.Style=psSmallDot)     Then Form2.Combobox1.text:='Ponto';
//   Carregando o Tamanho do Ponto
     Form2.SpinEdit1.value:=Series1.Pointer.HorizSize;
     Form2.SpinEdit1.value:=Series1.Pointer.VertSize;
//   Carregando a Cor do Ponto
     Form2.Colorbox1.Selected:=series1.Pointer.Brush.Color;
//   Definindo Gráfico em Função do Tempo ou Temperatura
     VT:=VTempo1;
     DefineGrafico;
  End;
  IF (Form2.ComboBox2.Text='Segmento 2') AND (VTempo2<>0) Then
```

```
  Begin
    Form2.SPI.Caption:=IntToStr(VTemper2)+' ºC';
    Form2.SPF.Caption:=IntToStr(VTemper3)+' ºC';
    Form2.Taxa.Caption:=IntToStr((VTemper3-VTemper2) DIV VTempo2)+' ºC/min';
    IF VTemper2<>VTemper3 Then Form2.Tipo.Caption:='Rampa' Else Form2.Tipo.Caption:='Patamar';
    Form2.temposeg.caption:=IntToStr(VTempo2)+' min';
//  Deixando o Somente o Segmento 2 Visível
    Form2.Series1.Pointer.Visible:=False; Form2.Series2.Pointer.Visible:=True;
Form2.Series3.Pointer.Visible:=False;
    Form2.Series4.Pointer.Visible:=False; Form2.Series5.Pointer.Visible:=False;
//  Carregando o Formato do Ponto
    IF (Form2.Series2.Pointer.Style=psCircle)       Then Form2.Combobox1.text:='Círculo';
    IF (Form2.Series2.Pointer.Style=psRectangle)    Then Form2.Combobox1.text:='Quadrado';
    IF (Form2.Series2.Pointer.Style=psTriangle)     Then Form2.Combobox1.text:='Triangulo';
    IF (Form2.Series2.Pointer.Style=psDownTriangle) Then Form2.Combobox1.text:='Nabla';
    IF (Form2.Series2.Pointer.Style=psCross)        Then Form2.Combobox1.text:='Cruz';
    IF (Form2.Series2.Pointer.Style=psDiagCross)    Then Form2.Combobox1.text:='X';
    IF (Form2.Series2.Pointer.Style=psStar)         Then Form2.Combobox1.text:='Estrela';
    IF (Form2.Series2.Pointer.Style=psDiamond)      Then Form2.Combobox1.text:='Losango';
    IF (Form2.Series2.Pointer.Style=psSmallDot)     Then Form2.Combobox1.text:='Ponto';
//  Carregando o Tamanho do Ponto
    Form2.SpinEdit1.value:=Series2.Pointer.HorizSize;
    Form2.SpinEdit1.value:=Series2.Pointer.VertSize;
//  Carregando a Cor do Ponto
    IF (Form2.ColorBox1.Color=-16777216) Then Begin
    Showmessage('sem cor');
    End;
    Form2.Colorbox1.Selected:=series2.Pointer.Brush.Color;
//  Definindo Gráfico em Função do Tempo ou Temperatura
    VT:=VTempo2;
    DefineGrafico;
  End;
  IF (Form2.ComboBox2.Text='Segmento 3') AND (VTempo3<>0) Then
  Begin
    Form2.SPI.Caption:=IntToStr(VTemper3)+' ºC';
    Form2.SPF.Caption:=IntToStr(VTemper4)+' ºC';
    Form2.Taxa.Caption:=IntToStr((VTemper4-VTemper3) DIV VTempo3)+' ºC/min';
    IF VTemper3<>VTemper4 Then Form2.Tipo.Caption:='Rampa' Else Form2.Tipo.Caption:='Patamar';
    Form2.temposeg.caption:=IntToStr(VTempo3)+' min';
//  Deixando o Somente o Segmento 3 Visível
    Form2.Series1.Pointer.Visible:=False; Form2.Series2.Pointer.Visible:=False;
Form2.Series3.Pointer.Visible:=True;
    Form2.Series4.Pointer.Visible:=False; Form2.Series5.Pointer.Visible:=False;
//  Carregando o Formato do Ponto
    IF (Form2.Series3.Pointer.Style=psCircle)       Then Form2.Combobox1.text:='Círculo';
    IF (Form2.Series3.Pointer.Style=psRectangle)    Then Form2.Combobox1.text:='Quadrado';
    IF (Form2.Series3.Pointer.Style=psTriangle)     Then Form2.Combobox1.text:='Triangulo';
    IF (Form2.Series3.Pointer.Style=psDownTriangle) Then Form2.Combobox1.text:='Nabla';
    IF (Form2.Series3.Pointer.Style=psCross)        Then Form2.Combobox1.text:='Cruz';
    IF (Form2.Series3.Pointer.Style=psDiagCross)    Then Form2.Combobox1.text:='X';
    IF (Form2.Series3.Pointer.Style=psStar)         Then Form2.Combobox1.text:='Estrela';
    IF (Form2.Series3.Pointer.Style=psDiamond)      Then Form2.Combobox1.text:='Losango';
    IF (Form2.Series3.Pointer.Style=psSmallDot)     Then Form2.Combobox1.text:='Ponto';
//  Carregando o Tamanho do Ponto
    Form2.SpinEdit1.value:=Series3.Pointer.HorizSize;
    Form2.SpinEdit1.value:=Series3.Pointer.VertSize;
//  Carregando a Cor do Ponto
    Form2.Colorbox1.Selected:=series3.Pointer.Brush.Color;
//  Definindo Gráfico em Função do Tempo ou Temperatura
    VT:=VTempo3;
```

```
      DefineGrafico;
    End;
    IF (Form2.ComboBox2.Text='Segmento 4') AND (VTempo4<>0) Then
    Begin
      Form2.SPI.Caption:=IntToStr(VTemper4)+' ºC';
      Form2.SPF.Caption:=IntToStr(VTemper5)+' ºC';
      Form2.Taxa.Caption:=IntToStr((VTemper5-VTemper4) DIV VTempo4)+' ºC/min';
      IF VTemper4<>VTemper5 Then Form2.Tipo.Caption:='Rampa' Else Form2.Tipo.Caption:='Patamar';
      Form2.temposeg.caption:=IntToStr(VTempo4)+' min';
//    Deixando o Somente o Segmento 4 Visível
      Form2.Series1.Pointer.Visible:=False;  Form2.Series2.Pointer.Visible:=False;
Form2.Series3.Pointer.Visible:=False;
      Form2.Series4.Pointer.Visible:=True;   Form2.Series5.Pointer.Visible:=False;
//    Carregando o Formato do Ponto
      IF (Form2.Series4.Pointer.Style=psCircle)       Then Form2.Combobox1.text:='Círculo';
      IF (Form2.Series4.Pointer.Style=psRectangle)    Then Form2.Combobox1.text:='Quadrado';
      IF (Form2.Series4.Pointer.Style=psTriangle)     Then Form2.Combobox1.text:='Triangulo';
      IF (Form2.Series4.Pointer.Style=psDownTriangle) Then Form2.Combobox1.text:='Nabla';
      IF (Form2.Series4.Pointer.Style=psCross)        Then Form2.Combobox1.text:='Cruz';
      IF (Form2.Series4.Pointer.Style=psDiagCross)    Then Form2.Combobox1.text:='X';
      IF (Form2.Series4.Pointer.Style=psStar)         Then Form2.Combobox1.text:='Estrela';
      IF (Form2.Series4.Pointer.Style=psDiamond)      Then Form2.Combobox1.text:='Losango';
      IF (Form2.Series4.Pointer.Style=psSmallDot)     Then Form2.Combobox1.text:='Ponto';
//    Carregando o Tamanho do Ponto
      Form2.SpinEdit1.value:=Series4.Pointer.HorizSize;
      Form2.SpinEdit1.value:=Series4.Pointer.VertSize;
//    Carregando a Cor do Ponto
      Form2.Colorbox1.Selected:=series4.Pointer.Brush.Color;
//    Definindo Gráfico em Função do Tempo ou Temperatura
      VT:=VTempo4;
      DefineGrafico;
    End;
    IF (Form2.ComboBox2.Text='Segmento 5') AND (VTempo5<>0) Then
    Begin
      Form2.SPI.Caption:=IntToStr(VTemper5)+' ºC';
      Form2.SPF.Caption:=IntToStr(VTemper6)+' ºC';
      Form2.Taxa.Caption:=IntToStr((VTemper6-VTemper5) DIV VTempo5)+' ºC/min';
      IF VTemper5<>VTemper6 Then Form2.Tipo.Caption:='Rampa' Else Form2.Tipo.Caption:='Patamar';
      Form2.temposeg.caption:=IntToStr(VTempo5)+' min';
//    Deixando o Somente o Segmento 5 Visível
      Form2.Series1.Pointer.Visible:=False;  Form2.Series2.Pointer.Visible:=False;
Form2.Series3.Pointer.Visible:=False;
      Form2.Series4.Pointer.Visible:=False;  Form2.Series5.Pointer.Visible:=True;
//    Carregando o Formato do Ponto
      IF (Form2.Series5.Pointer.Style=psCircle)       Then Form2.Combobox1.text:='Círculo';
      IF (Form2.Series5.Pointer.Style=psRectangle)    Then Form2.Combobox1.text:='Quadrado';
      IF (Form2.Series5.Pointer.Style=psTriangle)     Then Form2.Combobox1.text:='Triangulo';
      IF (Form2.Series5.Pointer.Style=psDownTriangle) Then Form2.Combobox1.text:='Nabla';
      IF (Form2.Series5.Pointer.Style=psCross)        Then Form2.Combobox1.text:='Cruz';
      IF (Form2.Series5.Pointer.Style=psDiagCross)    Then Form2.Combobox1.text:='X';
      IF (Form2.Series5.Pointer.Style=psStar)         Then Form2.Combobox1.text:='Estrela';
      IF (Form2.Series5.Pointer.Style=psDiamond)      Then Form2.Combobox1.text:='Losango';
      IF (Form2.Series5.Pointer.Style=psSmallDot)     Then Form2.Combobox1.text:='Ponto';
//    Carregando o Tamanho do Ponto
      Form2.SpinEdit1.value:=Series5.Pointer.HorizSize;
      Form2.SpinEdit1.value:=Series5.Pointer.VertSize;
//    Carregando a Cor do Ponto
      Form2.Colorbox1.Selected:=series5.Pointer.Brush.Color;
//    Definindo Gráfico em Função do Tempo ou Temperatura
      VT:=VTempo5;
```

```
      DefineGrafico;
   End;

end;

procedure TForm2.BitBtn1Click(Sender: TObject);
begin
   Form2.Close;
end;

procedure TForm2.BitBtn2Click(Sender: TObject);
begin
    Form2.salvargrafico.FileName:='';
    Form2.salvargrafico.Execute;
    IF (Form2.SalvarGrafico.FileName<>'') Then
      Chart1.SaveToBitmapFile(Form2.salvargrafico.FileName);
end;

end.
```

APÊNDICE C – CÓDIGO FONTE – UNIDADE 3

```
unit Unit3;   //  Programação do Controlador

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, NumEdit, ExtCtrls, Buttons, ComCtrls;

type LeituraC = class(TThread)
   Private
   Protected
   Procedure Execute; override;
End;

type EscritaC = class(TThread)
   Private
   Protected
   Procedure Execute; override;
End;

type
  TForm3 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    SP1L: TLabel;
    E1L: TLabel;
    T1L: TLabel;
    SP2L: TLabel;
    E2L: TLabel;
    T2L: TLabel;
    SP3L: TLabel;
    E3L: TLabel;
    T3L: TLabel;
    SP4L: TLabel;
    E4L: TLabel;
    T4L: TLabel;
    Label8: TLabel;
    Taxa1L: TLabel;
    Label9: TLabel;
    Taxa2L: TLabel;
    Label13: TLabel;
    Taxa3L: TLabel;
    SP1: TIntEdit;
    E1: TComboBox;
    Label16: TLabel;
    Label17: TLabel;
    Splitter1: TSplitter;
    Label20: TLabel;
```

```
  Label18: TLabel;
  Label19: TLabel;
  Label21: TLabel;
  Label22: TLabel;
  Label23: TLabel;
  Label24: TLabel;
  Label25: TLabel;
  Label26: TLabel;
  Label27: TLabel;
  Label28: TLabel;
  Label29: TLabel;
  Label30: TLabel;
  Taxa1: TLabel;
  Taxa2: TLabel;
  Taxa3: TLabel;
  Taxa4: TLabel;
  Taxa4L: TLabel;
  T1: TIntEdit;
  SP2: TIntEdit;
  T2: TIntEdit;
  SP3: TIntEdit;
  T3: TIntEdit;
  SP4: TIntEdit;
  T4: TIntEdit;
  SP5: TIntEdit;
  T5: TIntEdit;
  SP6: TIntEdit;
  E6L: TLabel;
  Taxa5: TLabel;
  Taxa5L: TLabel;
  E2: TComboBox;
  E3: TComboBox;
  E4: TComboBox;
  E5: TComboBox;
  E6: TComboBox;
  SP5L: TLabel;
  E5L: TLabel;
  T5L: TLabel;
  SP6L: TLabel;
  GroupBox1: TGroupBox;
  ProgressBar1: TProgressBar;
  Label31: TLabel;
  BitBtn1: TBitBtn;
  BitBtn2: TBitBtn;
  BitBtn3: TBitBtn;
  GroupBox2: TGroupBox;
  CheckBox1: TCheckBox;
  procedure BitBtn1Click(Sender: TObject);
  procedure BitBtn2Click(Sender: TObject);
  procedure BitBtn3Click(Sender: TObject);
  procedure CheckBox1Click(Sender: TObject);
  procedure E2Change(Sender: TObject);
  procedure E3Change(Sender: TObject);
  procedure E4Change(Sender: TObject);
  procedure E5Change(Sender: TObject);
  procedure FormActivate(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
```

```
  end;

const
  LEN_BUFFER = 100; //Tamanho dos Buffers de dados.
  BIT0=1;
  BIT1=2;
  BIT2=4;
  BIT3=8;
  BIT4=16;
  BIT5=32;
  BIT6=64;
  BIT7=128;

var
  Form3: TForm3;
  hCom1C                        : THANDLE;
  dcb                           : TDCB;
  CommTimeOutsC                 : TCOMMTIMEOUTS;
  BytesEscritosC                : DWORD;
  BytesLidosC                   : DWORD;
  BufferRecebeC                 : array [0..LEN_BUFFER] of char;

implementation

uses Unit4, Unit5;

{$R *.dfm}

Function AbrirCOM1C:boolean;
begin
   hCom1C := CreateFile( PChar('COM1'),GENERIC_READ or GENERIC_WRITE, 0, nil,
OPEN_EXISTING,0,0);

   if(hCom1C = INVALID_HANDLE_VALUE) then  //Se houve algum erro ao abrir a porta.
      result := false
   else
      result:= true;
end;

Function ConfiguraCOM1C:boolean;
begin
   if not GetCommState(hCom1C, dcb) then
      result:= false;

   dcb.BaudRate := CBR_9600;      //define velocidade em bps.
   dcb.ByteSize := 8;             //define bits de dados.
   dcb.Parity := NOPARITY;        //define paridade
   dcb.StopBits := ONESTOPBIT;  //define stop bit.

   if not SetCommState(hCom1C, dcb) then
      result:= false
   else
      result:= true;
end;

Function ConfiguraTimeOutsCOM1C:boolean;
begin
   if not GetCommTimeouts(hCom1C, CommTimeoutsC) then
      result:= false;
```

```
  CommTimeoutsC.ReadIntervalTimeout := 2;
  CommTimeoutsC.ReadTotalTimeoutMultiplier := 0;
  CommTimeoutsC.ReadTotalTimeoutConstant := 2;
  CommTimeoutsC.WriteTotalTimeoutMultiplier := 5;
  CommTimeoutsC.WriteTotalTimeoutConstant := 5;

  if not SetCommTimeouts(hCom1C, CommTimeoutsC) then
    result:= false
  else
    result:= true;
end;


Procedure EscritaC.Execute;
Var  ValorHexaT, ValorHexa1, ValorHexa2          :String;
     ValorInt1 , ValorInt2,ValorTInt             :Integer;
     Caracter1 , Caracter2                       :Char;
     Comando                                     :String;
Begin
  Form3.BitBtn1.Enabled:=False;
  Form3.BitBtn2.Enabled:=False;
  Form3.BitBtn3.Enabled:=False;
  Form3.CheckBox1.Enabled:=False;
  Form3.ProgressBar1.Position:=0;
//////////  Determinando SetPoint 1
  ValorTInt :=Form3.SP1.Value*10;           // Multiplicando o valor digitado por 10
  ValorHexaT:=IntToHex(ValorTInt,4);      // Transformando em Hexa com 4 casas..
  ValorHexa1:=Copy(ValorHexaT,1,2);      // Separando as 4 casas
  ValorHexa2:=Copy(ValorHexaT,3,2);      // Separando as 4 casas
  ValorInt1 :=StrtoInt('$'+ValorHexa1);    // Transformando a primeira casa em decimal
  ValorInt2 :=StrtoInt('$'+ValorHexa2);    // Transformando a segunda casa em decimal
  Caracter1 :=CHR(ValorInt1);              // Obtendo o caracter do primeiro byte
  Caracter2 :=CHR(ValorInt2);              // Obtendo o caracter do segundo byte
  Comando:=#02#06#182#113+Caracter1+Caracter2+#127#233;
//////////  Enviando SetPoint 1
  WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
//////////  Enviando Continua Ciclo 1 - Obrigatóriamente SIM, opção 1
  WriteFile( hCom1C,PChar(#2#6#182#115#0#1#159#170)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#1) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
//////////  Determinando Tempo 1
  ValorTInt :=Form3.T1.Value;
  ValorHexaT:=IntToHex(ValorTInt,4);              // Transformando em Hexa com 4 casas..
  ValorHexa1:=Copy(ValorHexaT,1,2);              // Separando as 4 casas
```

```
   ValorHexa2:=Copy(ValorHexaT,3,2);                  // Separando as 4 casas
   ValorInt1 :=StrtoInt('$'+ValorHexa1);              // Transformando a primeira casa em decimal
   ValorInt2 :=StrtoInt('$'+ValorHexa2);              // Transformando a segunda casa em decimal
   Caracter1 :=CHR(ValorInt1);                        // Obtendo o caracter do primeiro byte
   Caracter2 :=CHR(ValorInt2);                        // Obtendo o caracter do segundo byte
   Comando:=#02#06#182#117+Caracter1+Caracter2+#190#100;
 //////////  Enviando Tempo 1
   WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
   sleep(250);
   BytesLidosC:= 0;
   Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
   if BytesLidosC > 0 then //Se algum caracter foi lido.
   begin
     IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
       Form4.showmodal;
   end;
   Form3.ProgressBar1.StepBy(4);
 //////////  Enviando D1
   WriteFile( hCom1C,PChar(#02#06#182#119#0#0#31#171)^,8, BytesEscritosC, nil); //Envia string.
   sleep(250);
   BytesLidosC:= 0;
   Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
   if BytesLidosC > 0 then //Se algum caracter foi lido.
   begin
     IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#0) Then
       Form4.showmodal;
   end;
   Form3.ProgressBar1.StepBy(4);
 //////////  Determinando SetPoint 2
   ValorTInt :=Form3.SP2.Value*10;           // Multiplicando o valor digitado por 10
   ValorHexaT:=IntToHex(ValorTInt,4);      // Transformando em Hexa com 4 casas..
   ValorHexa1:=Copy(ValorHexaT,1,2);      // Separando as 4 casas
   ValorHexa2:=Copy(ValorHexaT,3,2);      // Separando as 4 casas
   ValorInt1 :=StrtoInt('$'+ValorHexa1);      // Transformando a primeira casa em decimal
   ValorInt2 :=StrtoInt('$'+ValorHexa2);      // Transformando a segunda casa em decimal
   Caracter1 :=CHR(ValorInt1);                // Obtendo o caracter do primeiro byte
   Caracter2 :=CHR(ValorInt2);               // Obtendo o caracter do segundo byte
   Comando:=#02#06#182#121+Caracter1+Caracter2+#126#214;
 //////////  Enviando SetPoint 2
   WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
   sleep(250);
   BytesLidosC:= 0;
   Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
   if BytesLidosC > 0 then //Se algum caracter foi lido.
   begin
     IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
       Form4.showmodal;
   end;
   Form3.ProgressBar1.StepBy(4);
 //////////  Enviando Continua Ciclo 2
   IF (Form3.E2.Text='SIM') Then WriteFile( hCom1C,PChar(#2#6#182#123#0#1#30#104)^,8, BytesEscritosC,
 nil);
   IF (Form3.E2.Text='NÃO') Then WriteFile( hCom1C,PChar(#2#6#182#123#0#0#30#104)^,8,
 BytesEscritosC, nil);
   sleep(250);
   BytesLidosC:= 0;
   Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);
   if BytesLidosC > 0 then //Se algum caracter foi lido.
   begin
     IF (BufferRecebeC[4]<>#0) Then
```

```
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
////////// Determinando Tempo 2
    ValorTInt :=Form3.T2.Value;
    ValorHexaT:=IntToHex(ValorTInt,4);
    ValorHexa1:=Copy(ValorHexaT,1,2);
    ValorHexa2:=Copy(ValorHexaT,3,2);
    ValorInt1 :=StrtoInt('$'+ValorHexa1);
    ValorInt2 :=StrtoInt('$'+ValorHexa2);
    Caracter1 :=CHR(ValorInt1);
    Caracter2 :=CHR(ValorInt2);
    Comando:=#02#06#182#125+Caracter1+Caracter2+#191#161;
////////// Enviando Tempo 2
    WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
    sleep(250);
    BytesLidosC:= 0;
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin
      IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
////////// Enviando D2
    WriteFile( hCom1C,PChar(#02#06#182#127#0#0#158#105)^,8, BytesEscritosC, nil);
    sleep(250);
    BytesLidosC:= 0;
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin
      IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#0) Then
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
////////// Determinando SetPoint 3
    ValorTInt :=Form3.SP3.Value*10;
    ValorHexaT:=IntToHex(ValorTInt,4);
    ValorHexa1:=Copy(ValorHexaT,1,2);
    ValorHexa2:=Copy(ValorHexaT,3,2);
    ValorInt1 :=StrtoInt('$'+ValorHexa1);
    ValorInt2 :=StrtoInt('$'+ValorHexa2);
    Caracter1 :=CHR(ValorInt1);
    Caracter2 :=CHR(ValorInt2);
    Comando:=#02#06#182#129+Caracter1+Caracter2+#94#31;
////////// Enviando SetPoint 3
    WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
    sleep(250);
    BytesLidosC:= 0;
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin  end;
    Form3.ProgressBar1.StepBy(4);
////////// Enviando Continua Ciclo 3
    IF (Form3.E3.Text='SIM') Then WriteFile( hCom1C,PChar(#2#6#182#131#0#1#159#153)^,8,
BytesEscritosC, nil); //Envia string.
    IF (Form3.E3.Text='NÃO') Then WriteFile( hCom1C,PChar(#2#6#182#131#0#0#159#153)^,8,
BytesEscritosC, nil); //Envia string.
    sleep(250);
    BytesLidosC:= 0;
```

```
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin
      IF (BufferRecebeC[4]<>#0) Then
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
  //////////  Determinando Tempo 3
    ValorTInt :=Form3.T3.Value;
    ValorHexaT:=IntToHex(ValorTInt,4);
    ValorHexa1:=Copy(ValorHexaT,1,2);
    ValorHexa2:=Copy(ValorHexaT,3,2);
    ValorInt1 :=StrtoInt('$'+ValorHexa1);
    ValorInt2 :=StrtoInt('$'+ValorHexa2);
    Caracter1 :=CHR(ValorInt1);
    Caracter2 :=CHR(ValorInt2);
    Comando:=#02#06#182#133+Caracter1+Caracter2+#62#95;
  //////////  Enviando Tempo 3
    WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
    sleep(250);
    BytesLidosC:= 0;
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin
      IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
  //////////  Enviando D3
    WriteFile( hCom1C,PChar(#02#06#182#135#0#0#31#152)^,8, BytesEscritosC, nil); //Envia string.
    sleep(250);
    BytesLidosC:= 0;
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin
      IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#0) Then
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
  //////////  Determinando SetPoint 4
    ValorTInt :=Form3.SP4.Value*10;
    ValorHexaT:=IntToHex(ValorTInt,4);
    ValorHexa1:=Copy(ValorHexaT,1,2);
    ValorHexa2:=Copy(ValorHexaT,3,2);
    ValorInt1 :=StrtoInt('$'+ValorHexa1);
    ValorInt2 :=StrtoInt('$'+ValorHexa2);
    Caracter1 :=CHR(ValorInt1);
    Caracter2 :=CHR(ValorInt2);
    Comando:=#02#06#182#137+Caracter1+Caracter2+#125#247;
  //////////  Enviando SetPoint 4
    WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
    sleep(250);
    BytesLidosC:= 0;
    Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
    if BytesLidosC > 0 then //Se algum caracter foi lido.
    begin
      IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
        Form4.showmodal;
    end;
    Form3.ProgressBar1.StepBy(4);
```

```
//////////  Enviando Continua Ciclo 4
  IF (Form3.E4.Text='SIM') Then WriteFile( hCom1C,PChar(#2#6#182#139#0#1#30#91)^,8, BytesEscritosC,
nil); //Envia string.
  IF (Form3.E4.Text='NÃO') Then WriteFile( hCom1C,PChar(#2#6#182#139#0#0#30#91)^,8, BytesEscritosC,
nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>#0) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
//////////  Determinando Tempo 4
  ValorTInt :=Form3.T4.Value;
  ValorHexaT:=IntToHex(ValorTInt,4);        // Transformando em Hexa com 4 casas..
  ValorHexa1:=Copy(ValorHexaT,1,2);         // Separando as 4 casas
  ValorHexa2:=Copy(ValorHexaT,3,2);         // Separando as 4 casas
  ValorInt1 :=StrtoInt('$'+ValorHexa1);     // Transformando a primeira casa em decimal
  ValorInt2 :=StrtoInt('$'+ValorHexa2);     // Transformando a segunda casa em decimal
  Caracter1 :=CHR(ValorInt1);               // Obtendo o caracter do primeiro byte
  Caracter2 :=CHR(ValorInt2);               // Obtendo o caracter do segundo byte
  Comando:=#02#06#182#141+Caracter1+Caracter2+#63#132;
//////////  Enviando Tempo 4
  WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
//////////  Enviando D4
  WriteFile( hCom1C,PChar(#02#06#182#143#0#0#158#90)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#0) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
//////////  Determinando SetPoint 5
  ValorTInt :=Form3.SP5.Value*10;           // Multiplicando o valor digitado por 10
  ValorHexaT:=IntToHex(ValorTInt,4);        // Transformando em Hexa com 4 casas..
  ValorHexa1:=Copy(ValorHexaT,1,2);         // Separando as 4 casas
  ValorHexa2:=Copy(ValorHexaT,3,2);         // Separando as 4 casas
  ValorInt1 :=StrtoInt('$'+ValorHexa1);     // Transformando a primeira casa em decimal
  ValorInt2 :=StrtoInt('$'+ValorHexa2);     // Transformando a segunda casa em decimal
  Caracter1 :=CHR(ValorInt1);               // Obtendo o caracter do primeiro byte
  Caracter2 :=CHR(ValorInt2);               // Obtendo o caracter do segundo byte
  Comando:=#02#06#182#145+Caracter1+Caracter2+#253#240;
//////////  Enviando SetPoint 5
  WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
```

```
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
////////// Enviando Continua Ciclo 5
  IF (Form3.E5.Text='SIM') Then WriteFile( hCom1C,PChar(#2#6#182#147#0#1#158#92)^,8, BytesEscritosC,
nil); //Envia string.
  IF (Form3.E5.Text='NÃO') Then WriteFile( hCom1C,PChar(#2#6#182#147#0#0#158#92)^,8,
BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>#0) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
////////// Determinando Tempo 5
  ValorTInt :=Form3.T5.Value;
  ValorHexaT:=IntToHex(ValorTInt,4);        // Transformando em Hexa com 4 casas..
  ValorHexa1:=Copy(ValorHexaT,1,2);         // Separando as 4 casas
  ValorHexa2:=Copy(ValorHexaT,3,2);         // Separando as 4 casas
  ValorInt1 :=StrtoInt('$'+ValorHexa1);     // Transformando a primeira casa em decimal
  ValorInt2 :=StrtoInt('$'+ValorHexa2);     // Transformando a segunda casa em decimal
  Caracter1 :=CHR(ValorInt1);               // Obtendo o caracter do primeiro byte
  Caracter2 :=CHR(ValorInt2);               // Obtendo o caracter do segundo byte
  Comando:=#02#06#182#149+Caracter1+Caracter2+#191#146;
////////// Enviando Tempo 5
  WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
////////// Enviando D5
  WriteFile( hCom1C,PChar(#02#06#182#151#0#0#30#93)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#0) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
////////// Determinando SetPoint 6
  ValorTInt :=Form3.SP6.Value*10;           // Multiplicando o valor digitado por 10
  ValorHexaT:=IntToHex(ValorTInt,4);        // Transformando em Hexa com 4 casas..
  ValorHexa1:=Copy(ValorHexaT,1,2);         // Separando as 4 casas
  ValorHexa2:=Copy(ValorHexaT,3,2);         // Separando as 4 casas
  ValorInt1 :=StrtoInt('$'+ValorHexa1);     // Transformando a primeira casa em decimal
  ValorInt2 :=StrtoInt('$'+ValorHexa2);     // Transformando a segunda casa em decimal
```

```
  Caracter1 :=CHR(ValorInt1);                    // Obtendo o caracter do primeiro byte
  Caracter2 :=CHR(ValorInt2);                    // Obtendo o caracter do segundo byte
  Comando:=#02#06#182#153+Caracter1+Caracter2+#126#182;
////////// Enviando SetPoint 6
  WriteFile( hCom1C,PChar(Comando)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>Caracter1) or (BufferRecebeC[5]<>Caracter2) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
////////// Enviando Continua Ciclo 6 - Obrigatóriamente NAO, opção 0
  WriteFile( hCom1C,PChar(#2#6#182#155#0#0#222#94)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[4]<>#0) or (BufferRecebeC[5]<>#0) Then
      Form4.showmodal;
  end;
  Form3.ProgressBar1.StepBy(4);
  Form3.ProgressBar1.Position:=100;
  CloseHandle(HCom1C);                    // Fecha a Porta!
  EscritaC.Create(True);                  // Finaliza a Aquisição
  Form3.BitBtn1.Enabled:=True;
  Form3.BitBtn2.Enabled:=True;
  Form3.BitBtn3.Enabled:=True;
  Form3.CheckBox1.Enabled:=True;
End;


Procedure LeituraC.Execute;
Var SetPoint1, SetPoint2, SetPoint3, SetPoint4, SetPoint5, SetPoint6 :Integer;
   Tempo1, Tempo2, Tempo3, Tempo4 , Tempo5, n                    :Integer;
   Taxa                                                          :Integer;
Begin
  Form3.BitBtn1.Enabled:=False;
  Form3.BitBtn2.Enabled:=False;
  Form3.BitBtn3.Enabled:=False;
  Form3.CheckBox1.Enabled:=False;
  Form3.ProgressBar1.Position:=0;
  ///  LENDO SETPOINT 1
  WriteFile( hCom1C,PChar(#02#03#182#113#0#01#242#106)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    SetPoint1:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    SetPoint1:=SetPoint1+(n*256);
    SetPoint1:=SetPoint1 DIV 10;
    Form3.SP1L.Caption:=IntToStr(SetPoint1)+ ' ºC';
  end else Form3.SP1L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  //  LENDO CONTINUA CICLO 1
```

```
  WriteFile( hCom1C,PChar(#02#03#182#115#0#01#83#170)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[5]=#1) Then Begin Form3.E1L.Caption:='SIM'; Form3.E1L.Font.Color:=ClGreen; End;
    IF (BufferRecebeC[5]=#0) Then Begin Form3.E1L.Caption:='NÃO'; Form3.E1L.Font.Color:=ClRed;   End;
    IF (BufferRecebeC[5]<>#0) AND (BufferRecebeC[5]<>#1) Then Form3.E1L.Caption:='ERRO...';
  end else Form3.E1L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO TEMPO 1
  WriteFile( hCom1C,PChar(#02#03#182#117#0#01#179#171)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    Tempo1:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    Tempo1:=Tempo1+(n*256);
    Form3.T1L.Caption:=IntToStr(Tempo1)+ ' min';
  end else Form3.T1L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO SETPOINT 2
  WriteFile( hCom1C,PChar(#02#03#182#121#0#01#115#168)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    SetPoint2:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    SetPoint2:=SetPoint2+(n*256);
    SetPoint2:=SetPoint2 DIV 10;
    Form3.SP2L.Caption:=IntToStr(SetPoint2)+ ' ºC';
    IF (SetPoint2<>SetPoint1) Then
    Begin
       Taxa:=(SetPoint2-SetPoint1) DIV Tempo1;
       Taxa:=StrToInt( Copy(IntToStr(Taxa),1,5));
       Form3.Taxa1L.Caption:=FloatToStr(Taxa)+' ºC/min';
    End Else Form3.Taxa1L.Caption:='Patamar';
  end else Form3.SP2L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);

//  LENDO CONTINUA CICLO 2
  WriteFile( hCom1C,PChar(#02#03#182#123#0#01#210#104)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[5]=#1) Then Begin Form3.E2L.Caption:='SIM'; Form3.E2L.Font.Color:=ClGreen; End;
    IF (BufferRecebeC[5]=#0) Then Begin Form3.E2L.Caption:='NÃO'; Form3.E2L.Font.Color:=ClRed;   End;
    IF (BufferRecebeC[5]<>#0) AND (BufferRecebeC[5]<>#1) Then Form3.E2L.Caption:='ERRO...';
  end else Form3.E2L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO TEMPO 2
  WriteFile( hCom1C,PChar(#02#03#182#125#0#01#50#105)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
```

```
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    Tempo2:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    Tempo2:=Tempo2+(n*256);
    Form3.T2L.Caption:=IntToStr(Tempo2)+ ' min';
  end else Form3.T2L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO SETPOINT 3
  WriteFile( hCom1C,PChar(#02#03#182#129#0#01#242#89)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    SetPoint3:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    SetPoint3:=SetPoint3+(n*256);
    SetPoint3:=SetPoint3 DIV 10;
    Form3.SP3L.Caption:=IntToStr(SetPoint3)+ ' ºC';
    IF (SetPoint3<>SetPoint2) Then
    Begin
       IF (Tempo2<>0) Then Taxa:=(SetPoint3-SetPoint2) DIV Tempo2;
       Taxa:=StrToInt( Copy(IntToStr(Taxa),1,5));
       IF (Tempo2<>0) Then Form3.Taxa2L.Caption:=IntToStr(Taxa)+' ºC/min' Else
Form3.Taxa2L.Caption:='Taxa';
    End Else IF (Tempo2<>0) Then Form3.Taxa2L.Caption:='Patamar' Else Form3.Taxa2L.Caption:='Taxa';
  end else Form3.SP3L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  //  LENDO CONTINUA CICLO 3
  WriteFile( hCom1C,PChar(#02#03#182#131#0#01#83#153)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[5]=#1) Then Begin Form3.E3L.Caption:='SIM'; Form3.E3L.Font.Color:=ClGreen; End;
    IF (BufferRecebeC[5]=#0) Then Begin Form3.E3L.Caption:='NÃO'; Form3.E3L.Font.Color:=ClRed;   End;
    IF (BufferRecebeC[5]<>#0) AND (BufferRecebeC[5]<>#1) Then Form3.E3L.Caption:='ERRO...';
  end else Form3.E3L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO TEMPO 3
  WriteFile( hCom1C,PChar(#02#03#182#133#0#01#179#152)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    Tempo3:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    Tempo3:=Tempo3+(n*256);
    Form3.T3L.Caption:=IntToStr(Tempo3)+ ' min';
  end else Form3.T3L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO SETPOINT 4
  WriteFile( hCom1C,PChar(#02#03#182#137#0#01#115#155)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
```

```
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  Begin
    SetPoint4:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    SetPoint4:=SetPoint4+(n*256);
    SetPoint4:=SetPoint4 DIV 10;
    Form3.SP4L.Caption:=IntToStr(SetPoint4)+ ' ºC';
    IF (SetPoint4<>SetPoint3) Then
    Begin
       IF (Tempo3<>0) then Taxa:=(SetPoint4-SetPoint3) DIV Tempo3;
       Taxa:=StrToInt( Copy(IntToStr(Taxa),1,5));
       IF (Tempo3<>0) then Form3.Taxa3L.Caption:=IntToStr(Taxa)+' ºC/min' Else
Form3.Taxa3L.Caption:='Taxa';
    End Else IF (Tempo3<>0) Then Form3.Taxa3L.Caption:='Patamar' Else Form3.Taxa3L.Caption:='Taxa';
  end else Form3.SP4L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  //  LENDO CONTINUA CICLO 4
  WriteFile( hCom1C,PChar(#02#03#182#139#0#01#210#91)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[5]=#1) Then Begin Form3.E4L.Caption:='SIM'; Form3.E4L.Font.Color:=ClGreen; End;
    IF (BufferRecebeC[5]=#0) Then Begin Form3.E4L.Caption:='NÃO'; Form3.E4L.Font.Color:=ClRed;   End;
    IF (BufferRecebeC[5]<>#0) AND (BufferRecebeC[5]<>#1) Then Form3.E4L.Caption:='ERRO...';
  end else Form3.E4L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  // LENDO TEMPO 4
  WriteFile( hCom1C,PChar(#02#03#182#141#0#01#50#90)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    Tempo4:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    Tempo4:=Tempo4+(n*256);
    Form3.T4L.Caption:=IntToStr(Tempo4)+ ' min';
  end else Form3.T4L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO SETPOINT 5
  WriteFile( hCom1C,PChar(#02#03#182#145#0#01#243#156)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    SetPoint5:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    SetPoint5:=SetPoint5+(n*256);
    SetPoint5:=SetPoint5 DIV 10;
    Form3.SP5L.Caption:=IntToStr(SetPoint5)+ ' ºC';
    IF (SetPoint5<>SetPoint4) Then
    Begin
       IF (Tempo4<>0) Then Taxa:=(SetPoint5-SetPoint4) DIV Tempo4;
       Taxa:=StrToInt( Copy(IntToStr(Taxa),1,5));
       IF (Tempo4<>0) Then Form3.Taxa4L.Caption:=IntToStr(Taxa)+' ºC/min' Else
Form3.Taxa4L.Caption:='Taxa';
```

```
    End Else IF (Tempo4<>0) Then Form3.Taxa4L.Caption:='Patamar' Else Form3.Taxa4L.Caption:='Taxa';
  end else Form3.SP5L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  //  LENDO CONTINUA CICLO 5
  WriteFile( hCom1C,PChar(#02#03#182#147#0#01#82#92)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[5]=#1) Then Begin Form3.E5L.Caption:='SIM'; Form3.E5L.Font.Color:=ClGreen; End;
    IF (BufferRecebeC[5]=#0) Then Begin Form3.E5L.Caption:='NÃO'; Form3.E5L.Font.Color:=ClRed;   End;
    IF (BufferRecebeC[5]<>#0) AND (BufferRecebeC[5]<>#1) Then Form3.E5L.Caption:='ERRO...';
  end else Form3.E5L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  // LENDO TEMPO 5
  WriteFile( hCom1C,PChar(#02#03#182#149#0#01#178#93)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    Tempo5:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    Tempo5:=Tempo5+(n*256);
    Form3.T5L.Caption:=IntToStr(Tempo5)+ ' min';
  end else Form3.T5L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  ///  LENDO SETPOINT 6
  WriteFile( hCom1C,PChar(#02#03#182#153#0#01#114#94)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    SetPoint6:=ORD(BufferRecebeC[5]);
    n:=ORD(BufferRecebeC[4]);
    SetPoint6:=SetPoint6+(n*256);
    SetPoint6:=SetPoint6 DIV 10;
    Form3.SP6L.Caption:=IntToStr(SetPoint6)+ ' ºC';
    IF (SetPoint6<>SetPoint5) Then
    Begin
       IF (Tempo5<>0) Then Taxa:=(SetPoint6-SetPoint5) DIV Tempo5;
       Taxa:=StrToInt( Copy(IntToStr(Taxa),1,5));
       IF (Tempo5<>0) Then Form3.Taxa5L.Caption:=IntToStr(Taxa)+' ºC/min' Else
Form3.Taxa5L.Caption:='Taxa';
    End Else IF (Tempo5<>0) Then Form3.Taxa5L.Caption:='Patamar' Else Form3.Taxa5L.Caption:='Taxa';
  end else Form3.SP6L.Caption:='Sem Resposta';
  Form3.ProgressBar1.StepBy(5);
  //  LENDO CONTINUA CICLO 6
  WriteFile( hCom1C,PChar(#02#03#182#155#0#01#211#158)^,8, BytesEscritosC, nil); //Envia string.
  sleep(250);
  BytesLidosC:= 0;
  Readfile(hCom1C, BufferRecebeC, LEN_BUFFER, BytesLidosC, nil);  //Lê a porta Serial.
  if BytesLidosC > 0 then //Se algum caracter foi lido.
  begin
    IF (BufferRecebeC[5]=#1) Then Begin Form3.E6L.Caption:='SIM'; Form3.E6L.Font.Color:=ClGreen; End;
    IF (BufferRecebeC[5]=#0) Then Begin Form3.E6L.Caption:='NÃO'; Form3.E6L.Font.Color:=ClRed;   End;
    IF (BufferRecebeC[5]<>#0) AND (BufferRecebeC[5]<>#1) Then Form3.E6L.Caption:='ERRO...';
  end else Form3.E6L.Caption:='Sem Resposta';
```

```
  Form3.ProgressBar1.Position:=100;
  CloseHandle(HCom1C);                  // Fecha a Porta!
  LeituraC.Create(True);               // Finaliza a Aquisição
  Form3.BitBtn1.Enabled:=True;
  Form3.BitBtn2.Enabled:=True;
  Form3.BitBtn3.Enabled:=True;
  Form3.CheckBox1.Enabled:=True;
End;

procedure TForm3.BitBtn1Click(Sender: TObject);
var sucessoC:Boolean;
begin
  IF (AbrirCom1C=True) Then
  Begin
    sucessoC:=ConfiguraCOM1C();
    sucessoC:=ConfiguraTimeOutsCOM1C();        //    Configurando a porta!
    IF (sucessoC=False) Then                   //    Se teve erro ao configurar COM1
    Begin
       CloseHandle(hCom1C);                    // Fecha a porta serial COM1;
       ShowMessage('Outro programa já está usando, ou a porta COM1 não existe!');
    End Else Begin
       LeituraC.Create(False)                  // Inicia a Aquisição
    End;
  End;
end;

procedure TForm3.BitBtn2Click(Sender: TObject);
Var SucessoC:Boolean;
begin
  IF (AbrirCom1C=True) Then
  Begin
    sucessoC:=ConfiguraCOM1C();
    sucessoC:=ConfiguraTimeOutsCOM1C();        //    Configurando a porta!
    IF (sucessoC=False) Then                   //    Se teve erro ao configurar COM1
    Begin
       CloseHandle(hCom1C);                    // Fecha a porta serial COM1;
       ShowMessage('Outro programa já está usando, ou a porta COM1 não existe!');
    End Else Begin
       EscritaC.Create(False)                  // Inicia a Escrita !
    End;
  End;

end;

procedure TForm3.BitBtn3Click(Sender: TObject);
begin
   Form3.Close;
end;

procedure TForm3.CheckBox1Click(Sender: TObject);
begin
   IF (Form3.CheckBox1.Checked=True) Then
   Begin
     Form3.SP1.ReadOnly:=True;
     Form3.SP2.ReadOnly:=True;
     Form3.SP3.ReadOnly:=True;
     Form3.SP4.ReadOnly:=True;
     Form3.SP5.ReadOnly:=True;
     Form3.SP6.ReadOnly:=True;
```

```
      Form3.T1.ReadOnly:=True;
      Form3.T2.ReadOnly:=True;
      Form3.T3.ReadOnly:=True;
      Form3.T4.ReadOnly:=True;
      Form3.T5.ReadOnly:=True;
      Form3.E1.Enabled:=False;
      Form3.E2.Enabled:=False;
      Form3.E3.Enabled:=False;
      Form3.E4.Enabled:=False;
      Form3.E5.Enabled:=False;
      Form3.E6.Enabled:=False;
    End Else Begin
      Showmessage('Tenha certeza dos valores que irá inserir!');
      Form3.SP1.ReadOnly:=False;
      Form3.SP2.ReadOnly:=False;
      Form3.SP3.ReadOnly:=False;
      Form3.SP4.ReadOnly:=False;
      Form3.SP5.ReadOnly:=False;
      Form3.SP6.ReadOnly:=False;
      Form3.T1.ReadOnly:=False;
      Form3.T2.ReadOnly:=False;
      Form3.T3.ReadOnly:=False;
      Form3.T4.ReadOnly:=False;
      Form3.T5.ReadOnly:=False;
      Form3.E1.Enabled:=True;
      Form3.E1.Enabled:=True;
      Form3.E2.Enabled:=True;
      Form3.E3.Enabled:=True;
      Form3.E4.Enabled:=True;
      Form3.E5.Enabled:=True;
      Form3.E6.Enabled:=True;
    End;
end;

procedure TForm3.E2Change(Sender: TObject);
begin
  IF (Form3.E2.Text='NÃO') Then
  Begin
    Form3.T2.Value:=0;
    Form3.Taxa2.Caption:='Taxa';
    Form3.SP3.Value:=0;
    Form3.E3.Text:='NÃO';
    Form3.T3.Value:=0;
    Form3.Taxa3.Caption:='Taxa';
    Form3.SP4.Value:=0;
    Form3.E4.Text:='NÃO';
    Form3.T4.Value:=0;
    Form3.Taxa4.Caption:='Taxa';
    Form3.SP5.Value:=0;
    Form3.E5.Text:='NÃO';
    Form3.T5.Value:=0;
    Form3.Taxa5.Caption:='Taxa';
    Form3.SP6.Value:=0;
  End;
end;

procedure TForm3.E3Change(Sender: TObject);
begin
  IF (Form3.E3.Text='NÃO') Then
  Begin
```

```
    Form3.T3.Value:=0;
    Form3.Taxa3.Caption:='Taxa';
    Form3.SP4.Value:=0;
    Form3.E4.Text:='NÃO';
    Form3.T4.Value:=0;
    Form3.Taxa4.Caption:='Taxa';
    Form3.SP5.Value:=0;
    Form3.E5.Text:='NÃO';
    Form3.T5.Value:=0;
    Form3.Taxa5.Caption:='Taxa';
    Form3.SP6.Value:=0;
  End;
end;

procedure TForm3.E4Change(Sender: TObject);
begin
  IF (Form3.E4.Text='NÃO') Then
  Begin
    Form3.T4.Value:=0;
    Form3.Taxa4.Caption:='Taxa';
    Form3.SP5.Value:=0;
    Form3.E5.Text:='NÃO';
    Form3.T5.Value:=0;
    Form3.Taxa5.Caption:='Taxa';
    Form3.SP6.Value:=0;
  End;

end;

procedure TForm3.E5Change(Sender: TObject);
begin
  IF (Form3.E5.Text='NÃO') Then
  Begin
    Form3.T5.Value:=0;
    Form3.Taxa5.Caption:='Taxa';
    Form3.SP6.Value:=0;
  End;

end;

procedure TForm3.FormActivate(Sender: TObject);
Var SucessoC:Boolean;
begin

  IF (AbrirCom1C=True) Then
  Begin
    sucessoC:=ConfiguraCOM1C();
    sucessoC:=ConfiguraTimeOutsCOM1C();        //    Configurando a porta!
    IF (sucessoC=False) Then                   //    Se teve erro ao configurar COM1
    Begin
       CloseHandle(hCom1C);                    // Fecha a porta serial COM1;
       ShowMessage('Outro programa já está usando, ou a porta COM1 não existe!');
    End Else Begin
       LeituraC.Create(False)                  // Inicia a Aquisição
    End;
  End;

end;

end.
```

APÊNDICE D – CÓDIGO FONTE – UNIDADE 4

```
unit Unit4;    //  Janela de Configuração de Rampas e Patamares

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, TeEngine, Series, ExtCtrls, TeeProcs, Chart, NumEdit,
  Buttons;

type
  TForm5 = class(TForm)
    Chart1: TChart;
    Series1: TLineSeries;
    GroupBox1: TGroupBox;
    SPI: TIntEdit;
    SPF: TIntEdit;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Button1: TButton;
    GroupBox2: TGroupBox;
    RadioButton2: TRadioButton;
    RadioButton1: TRadioButton;
    Label1: TLabel;
    Label10: TLabel;
    Button3: TButton;
    SPP: TIntEdit;
    Label11: TLabel;
    Label12: TLabel;
    GroupBox3: TGroupBox;
    GroupBox4: TGroupBox;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    TempoRampa: TIntEdit;
    Taxa: TIntEdit;
    TempoPatamar: TIntEdit;
    log: TMemo;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    BitBtn4: TBitBtn;
    Label18: TLabel;
    Label19: TLabel;
    BitBtn5: TBitBtn;
    BitBtn6: TBitBtn;
    BitBtn7: TBitBtn;
    BitBtn8: TBitBtn;
```

```
    BitBtn9: TBitBtn;
    procedure RadioButton1Click(Sender: TObject);
    procedure RadioButton2Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure TaxaChange(Sender: TObject);
    procedure TempoRampaChange(Sender: TObject);
    procedure SPIChange(Sender: TObject);
    procedure SPFChange(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure BitBtn6Click(Sender: TObject);
    procedure BitBtn5Click(Sender: TObject);
    procedure BitBtn7Click(Sender: TObject);
    procedure BitBtn8Click(Sender: TObject);
    procedure BitBtn9Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form5      :TForm5;
  TempoTotal :Integer;
  NroSeg     :Integer;
// Variáveis para o processo ABRIR e Para Form que escreve os dados no controlador
  VTemper1, VTemper2  :Integer;
  VTemper3, Vtemper4  :Integer;
  VTemper5, Vtemper6  :Integer;
  VTempo1 , VTempo2   :Integer;
  VTempo3 , VTempo4   :Integer;
  VTempo5 , n1        :Integer;
  VCC1,VCC2,VCC3,VCC4 :Integer;
  VCC5,VCC6           :Integer;
  Verif               :String;



implementation

uses Unit1, Unit2, Unit3;

{$R *.dfm}

procedure TForm5.RadioButton1Click(Sender: TObject);
begin
    form5.RadioButton2.Checked:=False;
// Ativando as opções da rampa...
    Form5.SPI.Enabled:=True;          Form5.SPI.Value:=Form5.SPP.Value;
    Form5.SPF.Enabled:=True;          Form5.SPF.Value:=0;
    Form5.Taxa.Enabled:=True;         Form5.Taxa.Value:=0;
    Form5.TempoRampa.Enabled:=True;   Form5.TempoRampa.Value:=0;
    Form5.Button1.Enabled:=True;

// Desativando as opções do patamar...
```

```
    Form5.SPP.Enabled:=False;
    Form5.TempoPatamar.Enabled:=False;
    Form5.Button3.Enabled:=False;


end;

procedure TForm5.RadioButton2Click(Sender: TObject);
begin
   Form5.RadioButton1.Checked:=False;
// Desativando as opções da rampa...
    Form5.SPI.Enabled:=False;
    Form5.SPF.Enabled:=False;
    Form5.Taxa.Enabled:=False;
    Form5.TempoRampa.Enabled:=False;
    Form5.Button1.Enabled:=False;
// Ativando as opções do patamar...
    Form5.SPP.Enabled:=True;            Form5.SPP.Value:=Form5.SPF.Value;
    Form5.TempoPatamar.Enabled:=True;   Form5.TempoPatamar.Value:=0;
    Form5.Button3.Enabled:=True;


end;

procedure TForm5.FormCreate(Sender: TObject);
begin
   TempoTotal:=0;
   NroSeg:=0;
   Series1.Clear;
end;

Function GetTime:String;
Var Hora, TT: Integer;
    TTs     : String;
Begin
   Hora:=0;
   IF (TempoTotal<60) Then
   Begin
      IF (TempoTotal>=10) Then
          GetTime:='00:'+IntToStr(TempoTotal)
      Else GetTime:='00:0'+IntToStr(TempoTotal);
   End Else Begin
      TT:=TempoTotal;
      While (TT>=60) Do
      Begin
         TT:=TT-60;
         Hora:=Hora+1;
      End;
      TTs:=IntToStr(TT);
      IF (Length(TTs)=1) Then TTs:='0'+TTs;

      IF (Length(IntToStr(Hora))=2) then
          GetTime:=IntToStr(Hora)+':'+TTs
      Else GetTime:='0'+IntToStr(Hora)+':'+TTs
   End;
End;


procedure TForm5.Button1Click(Sender: TObject);
```

```
begin
  IF (NroSeg<5) Then
  Begin
     IF (Form5.TempoRampa.Value=0) OR (Form5.Taxa.Value=0) Then
     Begin
       Showmessage('As propriedades Tempo de Rampa e Taxa não podem ser nulas!!');
     End Else Begin
       Form5.Series1.AddXY(TempoTotal,SPI.Value);
       TempoTotal:=TempoTotal+Form5.TempoRampa.Value;
       Form5.Series1.AddXY(TempoTotal,SPF.Value);
       NroSeg:=NroSeg+1;
       // Logando....
          IF (NroSeg=1) Then Form5.log.Lines.Add('SP'+IntToStr(NroSeg)+'='+IntToStr(SPI.Value)); // Para o
SP Inicial aparecer somente uma vez
          Form5.log.Lines.Add('E'+IntToStr(NroSeg)+'=1');
          Form5.log.Lines.Add('T'+IntToStr(NroSeg)+'='+IntToStr(TempoRampa.Value));
          Form5.log.Lines.Add('SP'+IntToStr(NroSeg+1)+'='+IntToStr(SPF.Value));
       // Fim do Log...
       Label14.Caption:=IntToStr(NroSeg);
       Label16.Caption:=IntToStr(TempoTotal)+' min';
       Label17.Caption:=Gettime;
     End;
  End Else Showmessage('Este programa suporta um máximo de 5 segmentos!');
end;

procedure TForm5.TaxaChange(Sender: TObject);
begin
  IF (Form5.Taxa.Focused=True) AND (Form5.Taxa.Value<>0) Then
     Form5.TempoRampa.Value:=(SPF.Value-SPI.Value) DIV Form5.Taxa.Value;
  IF (Form5.Taxa.Value=0) then Form5.TempoRampa.Value:=0;
  IF (Form5.TempoRampa.Value<0) Then
  Begin
    Showmessage('Para decréscimo de temperatura utilize taxas negativas!');
    Form5.Taxa.Value:=Form5.Taxa.Value*(-1);
  End;
end;

procedure TForm5.TempoRampaChange(Sender: TObject);
begin
  IF (Form5.TempoRampa.Focused=True) AND (Form5.TempoRampa.Value<>0) Then
    Form5.Taxa.Value:=(Form5.SPF.Value-Form5.SPI.Value) DIV Form5.TempoRampa.Value;
  IF (Form5.TempoRampa.Value=0) Then Form5.Taxa.Value:=0;
end;

procedure TForm5.SPIChange(Sender: TObject);
begin
   Form5.Taxa.Value:=0;
   Form5.TempoRampa.Value:=0;
end;

procedure TForm5.SPFChange(Sender: TObject);
begin
   Form5.Taxa.Value:=0;
   Form5.TempoRampa.Value:=0;
end;

procedure TForm5.Button3Click(Sender: TObject);
begin
  IF (NroSeg<5) Then
  Begin
```

```
      IF (Form5.TempoPatamar.Value=0) Then
      Begin
        Showmessage('A propriedade Tempo de Patamar não pode ser nula!!');
      End Else Begin
        Form5.Series1.AddXY(TempoTotal,SPP.Value);
        TempoTotal:=TempoTotal+Form5.TempoPatamar.Value;
        Form5.Series1.AddXY(TempoTotal,SPP.Value);
        NroSeg:=NroSeg+1;
        // Logando....
           IF (NroSeg=1) Then Form5.log.Lines.Add('SP'+IntToStr(NroSeg)+'='+IntToStr(SPP.Value)); // Para o
SP Inicial aparecer somente uma vez
           Form5.log.Lines.Add('E'+IntToStr(NroSeg)+'=1');
           Form5.log.Lines.Add('T'+IntToStr(NroSeg)+'='+IntToStr(TempoPatamar.Value));
           Form5.log.Lines.Add('SP'+IntToStr(NroSeg+1)+'='+IntToStr(SPP.Value));
        // Fim do Log...
        Label14.Caption:=IntToStr(NroSeg);
        Label16.Caption:=IntToStr(TempoTotal)+' min';
        Label17.Caption:=Gettime;
      End;
  End Else Showmessage('Este programa suporta um máximo de 5 segmentos!');
end;

procedure TForm5.BitBtn1Click(Sender: TObject);
Var UltimoSeg  :Integer;
   RetTempo   :String;
   totallinhas:Integer;
begin
   UltimoSeg:=series1.LastValueIndex;
   IF (UltimoSeg<>(-1)) Then
   Begin
     series1.Delete(UltimoSeg);
     UltimoSeg:=UltimoSeg-1;
     series1.Delete(UltimoSeg);
     nroseg:=nroseg-1;
     label14.caption:=IntToStr(NroSeg);
     // Determinando o tempo a ser subtraído e excluindo do log!
        totallinhas:=form5.log.Lines.Capacity;
        Form5.log.Lines.Delete(totallinhas-1);
        RetTempo:=Form5.log.Lines.Strings[totallinhas-2];
        Form5.log.Lines.Delete(totallinhas-2);
        Form5.log.Lines.Delete(totallinhas-3);
        IF (NroSeg=0) Then Form5.log.Lines.Delete(1);
        RetTempo:=Copy(RetTempo,4,5);
     // Retirando o tempo...
        TempoTotal:=TempoTotal - StrToInt(RetTempo);
        Form5.Label16.Caption:=IntToStr(TempoTotal);
        Form5.Label17.Caption:=GetTime;
   End;
end;

procedure TForm5.BitBtn2Click(Sender: TObject);
Var UltimoSeg:Integer;
begin
   UltimoSeg:=series1.LastValueIndex;
   IF (UltimoSeg<>(-1)) Then
   Begin
     series1.Clear;
     NroSeg:=0;
     TempoTotal:=0;
     Label14.caption:='0';
```

```
      Label16.caption:='00 min';
      Label17.Caption:='00:00';
      Log.Clear; Log.Lines.Add('Resistometria');
    End;
end;

procedure TForm5.BitBtn3Click(Sender: TObject);
begin
   If (Form5.BitBtn4.Caption='Concluir Gráfico') Then
   Begin
     Showmessage('O programa ainda não está concluído!');
   End Else Begin
     Form5.SaveDialog1.Execute;
     IF (Form5.SaveDialog1.FileName<>'') Then
     Begin
       Form5.log.Lines.SaveToFile(Form5.SaveDialog1.FileName);
       Form5.Caption:=Form5.SaveDialog1.FileName;
     End;
   End;
end;

procedure TForm5.BitBtn4Click(Sender: TObject);
var Linhas:Integer;
begin
  IF (Form5.BitBtn4.Caption='Concluir Gráfico') Then
  Begin
    IF (NroSeg=0) Then Showmessage('Este gráfico não tem nenhum segmento adicionado!')
    Else
    Begin
      Form5.log.Lines.Add('E'+IntToStr(NroSeg+1)+'=0');
      Form5.log.Lines.Strings[0]:='Termoresistometria['+IntToStr(NroSeg)+']';
      Form5.BitBtn1.Enabled:=False;
      Form5.BitBtn2.Enabled:=False;
      Form5.Button1.Enabled:=False;
      Form5.Button3.Enabled:=False;
      Form5.RadioButton1.Enabled:=False;
      Form5.RadioButton2.Enabled:=False;
      Form5.BitBtn4.Caption:='Editar Gráfico';
      Form5.Label19.Caption:='Sim';
      Form5.Label19.Font.Color:=clgreen;
    End
  End Else Begin
    linhas:=Form5.log.Lines.Capacity;
    Form5.log.Lines.Delete(Linhas-1);
    Form5.log.Lines.Strings[0]:='Termoresistometria';
    Form5.BitBtn1.Enabled:=True;
    Form5.BitBtn2.Enabled:=True;
    Form5.RadioButton1.Enabled:=True;
    Form5.RadioButton2.Enabled:=True;
    IF (Form5.RadioButton1.Checked=True) Then Form5.button1.Enabled:=True;
    IF (Form5.RadioButton2.Checked=True) Then Form5.button3.Enabled:=True;
    Form5.BitBtn4.Caption:='Concluir Gráfico';
    Form5.Label19.Caption:='Não';
    Form5.Label19.Font.Color:=clred;
  End;
end;

procedure TForm5.BitBtn6Click(Sender: TObject);
begin
   IF (Form5.Label19.Caption='Sim') Then
```

```
    Begin
        // Lendo no LOG o tempo correto de cada patamar!
        Verif:='';
        N1:=1;
        VTemper1:=0; VTemper2:=0; VTemper3:=0; VTemper4:=0; VTemper5:=0; VTemper6:=0;
        VTempo1:=0;  VTempo2 :=0; VTempo3 :=0; VTempo4 :=0; VTempo5 :=0;
        VCC1:=1; VCC2:=0; VCC3:=0; VCC4:=0; VCC5:=0; VCC6:=0;
        While (Verif<>'E'+IntToStr(NroSeg+1)+'=0') Do
        Begin
           Verif:=Form5.log.Lines[N1];
           N1:=N1+1;
           IF (N1=2)  Then VTemper1:=StrToInt( Copy(Form5.Log.Lines[1] ,5,4));
           IF (N1=3)  Then VCC1    :=StrToInt( Copy(Form5.Log.Lines[2] ,4,1));
           IF (N1=4)  Then VTempo1 :=StrToInt( Copy(Form5.Log.Lines[3] ,4,4));
           IF (N1=5)  Then VTemper2:=StrToInt( Copy(Form5.Log.Lines[4] ,5,4));
           IF (N1=6)  Then VCC2    :=StrToInt( Copy(Form5.Log.Lines[5] ,4,1));
           IF (N1=7)  Then VTempo2 :=StrToInt( Copy(Form5.Log.Lines[6] ,4,4));
           IF (N1=8)  Then VTemper3:=StrToInt( Copy(Form5.Log.Lines[7] ,5,4));
           IF (N1=9)  Then VCC3    :=StrToInt( Copy(Form5.Log.Lines[8] ,4,1));
           IF (N1=10) Then VTempo3 :=StrToInt( Copy(Form5.Log.Lines[9] ,4,4));
           IF (N1=11) Then VTemper4:=StrToInt( Copy(Form5.Log.Lines[10],5,4));
           IF (N1=12) Then VCC4    :=StrToInt( Copy(Form5.Log.Lines[11],4,1));
           IF (N1=13) Then VTempo4 :=StrToInt( Copy(Form5.Log.Lines[12],4,4));
           IF (N1=14) Then VTemper5:=StrToInt( Copy(Form5.Log.Lines[13],5,4));
           IF (N1=15) Then VCC5    :=StrToInt( Copy(Form5.Log.Lines[14],4,1));
           IF (N1=16) Then VTempo5 :=StrToInt( Copy(Form5.Log.Lines[15],4,4));
           IF (N1=17) Then VTemper6:=StrToInt( Copy(Form5.Log.Lines[16],5,4));
           IF (N1=18) Then VCC6    :=StrToInt( Copy(Form5.Log.Lines[17],4,1));
        End;
      TempoG1:=VTempo1*60;
      TempoG2:=VTempo2*60;
      TempoG3:=VTempo3*60;
      TempoG4:=VTempo4*60;
      TempoG5:=VTempo5*60;
      TempoAquisicao:=(VTempo1+VTempo2+VTempo3+VTempo4+VTempo5)*60;
      NroGraf:=StrToInt(Form5.Label14.Caption);
      Form2.Series1.Clear;
      Form2.Series2.Clear;
      Form2.Series3.Clear;
      Form2.Series4.Clear;
      Form2.Series5.Clear;
      TTP:=Label17.Caption;
      IF (Form1.Edit1.Text<>'') Then Form1.BitBtn1.Enabled:=True;
    End;
    Form5.close;
end;

procedure TForm5.BitBtn5Click(Sender: TObject);
begin
   Form5.OpenDialog1.Execute;
   IF (Form5.OpenDialog1.filename<>'') Then
   Begin
      // Limpando as Variáveis:
      VTemper1:=0; VTemper2:=0; VTemper3:=0; VTemper4:=0; VTemper5:=0; VTemper6:=0;
      VTempo1 :=0; Vtempo2 :=0; VTempo3 :=0; VTempo4 :=0; VTempo5 :=0;
      VCC1    :=0; VCC2    :=0; VCC3    :=0; VCC4    :=0; VCC5    :=0; VCC6    :=0;
      n1:=0;
      // Fim da Limpeza
       Form5.Log.Lines.LoadFromFile(Form5.OpenDialog1.FileName);
       Verif:=Form5.log.Lines[0];
```

```
Verif:=Copy(Verif,1,19);
IF (Verif='Termoresistometria[') Then
Begin
//  Abrindo o arquivo e adicionando no gráfico!
   Form5.Caption:=Form5.OpenDialog1.FileName;
   NroSeg:=StrToInt( Copy(Form5.log.Lines[0],20,1) );
   Form5.Label14.Caption:=IntToStr(NroSeg);
   Form5.Label19.Caption:='Sim';
   Form5.Label19.Font.Color:=clgreen;
   Form5.BitBtn4.Caption:='Editar Gráfico';
   Form5.BitBtn1.Enabled:=False;
   Form5.BitBtn2.Enabled:=False;
   Form5.RadioButton1.Enabled:=False;
   Form5.RadioButton2.Enabled:=False;
   Form5.Button1.Enabled:=False;
   Form5.Button3.Enabled:=False;
// Começando a carregar os dados
   Verif:='';
   N1:=1;
   While (Verif<>'E'+IntToStr(NroSeg+1)+'=0') Do
   Begin
      Verif:=Form5.log.Lines[N1];
      N1:=N1+1;
      IF (N1=2)  Then VTemper1:=StrToInt( Copy(Form5.Log.Lines[1] ,5,4));
      IF (N1=4)  Then VTempo1 :=StrToInt( Copy(Form5.Log.Lines[3] ,4,4));
      IF (N1=5)  Then VTemper2:=StrToInt( Copy(Form5.Log.Lines[4] ,5,4));
      IF (N1=7)  Then VTempo2 :=StrToInt( Copy(Form5.Log.Lines[6] ,4,4));
      IF (N1=8)  Then VTemper3:=StrToInt( Copy(Form5.Log.Lines[7] ,5,4));
      IF (N1=10) Then VTempo3 :=StrToInt( Copy(Form5.Log.Lines[9] ,4,4));
      IF (N1=11) Then VTemper4:=StrToInt( Copy(Form5.Log.Lines[10],5,4));
      IF (N1=13) Then VTempo4 :=StrToInt( Copy(Form5.Log.Lines[12],4,4));
      IF (N1=14) Then VTemper5:=StrToInt( Copy(Form5.Log.Lines[13],5,4));
      IF (N1=16) Then VTempo5 :=StrToInt( Copy(Form5.Log.Lines[15],4,4));
      IF (N1=17) Then VTemper6:=StrToInt( Copy(Form5.Log.Lines[16],5,4));
   End;
// Plotando no Gráfico os Valores Obtidos !
   TempoTotal:=0;
   Form5.Series1.Clear;
   IF (Vtempo1<>0) Then
   Begin
     Form5.Series1.AddXY(TempoTotal,VTemper1);
     TempoTotal:=TempoTotal+VTempo1;
     Form5.Series1.AddXY(TempoTotal,VTemper2);
   End;
   IF (Vtempo2<>0) Then
   Begin
     Form5.Series1.AddXY(TempoTotal,VTemper2);
     TempoTotal:=TempoTotal+VTempo2;
     Form5.Series1.AddXY(TempoTotal,VTemper3);
   End;
   IF (Vtempo3<>0) Then
   Begin
     Form5.Series1.AddXY(TempoTotal,VTemper3);
     TempoTotal:=TempoTotal+VTempo3;
     Form5.Series1.AddXY(TempoTotal,VTemper4);
   End;
   IF (Vtempo4<>0) Then
   Begin
     Form5.Series1.AddXY(TempoTotal,VTemper4);
     TempoTotal:=TempoTotal+VTempo4;
```

```
          Form5.Series1.AddXY(TempoTotal,VTemper5);
        End;
        IF (Vtempo5<>0) Then
        Begin
          Form5.Series1.AddXY(TempoTotal,VTemper5);
          TempoTotal:=TempoTotal+VTempo5;
          Form5.Series1.AddXY(TempoTotal,VTemper6);
        End;
        Label16.Caption:=IntToStr(TempoTotal)+' min';
        Label17.Caption:=Gettime;

      End Else Begin
         Showmessage('Arquivo inválido!');
         Form5.log.Clear;
      End;
  End;
end;

procedure TForm5.BitBtn7Click(Sender: TObject);
Var UltimoSeg : Integer;
begin
   UltimoSeg:=series1.LastValueIndex;
   IF (UltimoSeg<>(-1)) Then
   Begin
     series1.Clear;
     NroSeg:=0;
     TempoTotal:=0;
     Label14.caption:='0';
     Label16.caption:='00 min';
     Label17.Caption:='00:00';
     Label19.Caption:='Não'; Label19.Font.Color:=ClRed;
     Log.Clear; Log.Lines.Add('Resistometria');
     Form5.Caption:='Novo Programa';
     Form5.BitBtn4.Caption:='Concluir Gráfico';
     Form5.RadioButton1.Enabled:=True;
     Form5.RadioButton2.Enabled:=True;
     Form5.RadioButton1.Checked:=True;
     Form5.RadioButton2.Checked:=False;
     Form5.Button1.Enabled:=True;
     Form5.Button3.Enabled:=False;
     Form5.BitBtn1.Enabled:=True;
     Form5.BitBtn2.Enabled:=True;
   End;
end;

procedure TForm5.BitBtn8Click(Sender: TObject);
begin
   If (Form5.BitBtn4.Caption='Concluir Gráfico') Then
   Begin
      Showmessage('O programa ainda não está concluído!');
   End Else
   Begin
     IF (Form5.Caption<>'Programação') AND (Form5.Caption<>'Novo Programa') Then
     Begin
       Form5.log.Lines.SaveToFile(Form5.Caption);
     End Else
     Begin
       Form5.SaveDialog1.Execute;
       IF (Form5.SaveDialog1.FileName<>'') Then
       Begin
```

```
            Form5.log.Lines.SaveToFile(Form5.SaveDialog1.FileName);
            Form5.Caption:=Form5.SaveDialog1.FileName;
          End;
       End;
    End;
end;

procedure TForm5.BitBtn9Click(Sender: TObject);
Var Tx1,Tx2,Tx3,Tx4,Tx5 : Integer;
begin
    Tx1:=0; Tx2:=0; Tx3:=0; Tx4:=0; Tx5:=0;
    IF (Label19.Caption='Não') Then Showmessage('Você deve concluir o gráfico primeiro!')
    Else
    Begin
      // Carregando os dados em variáveis para usar no form seguinte!
        Verif:='';
        N1:=1;
        VTemper1:=0; VTemper2:=0; VTemper3:=0; VTemper4:=0; VTemper5:=0; VTemper6:=0;
        VTempo1:=0;  VTempo2 :=0; VTempo3 :=0; VTempo4 :=0; VTempo5 :=0;
        VCC1:=1; VCC2:=0; VCC3:=0; VCC4:=0; VCC5:=0; VCC6:=0;

        While (Verif<>'E'+IntToStr(NroSeg+1)+'=0') Do
        Begin
          Verif:=Form5.log.Lines[N1];
          N1:=N1+1;
          IF (N1=2)  Then VTemper1:=StrToInt( Copy(Form5.Log.Lines[1] ,5,4));
          IF (N1=3)  Then VCC1    :=StrToInt( Copy(Form5.Log.Lines[2] ,4,1));
          IF (N1=4)  Then VTempo1 :=StrToInt( Copy(Form5.Log.Lines[3] ,4,4));
          IF (N1=5)  Then VTemper2:=StrToInt( Copy(Form5.Log.Lines[4] ,5,4));
          IF (N1=6)  Then VCC2    :=StrToInt( Copy(Form5.Log.Lines[5] ,4,1));
          IF (N1=7)  Then VTempo2 :=StrToInt( Copy(Form5.Log.Lines[6] ,4,4));
          IF (N1=8)  Then VTemper3:=StrToInt( Copy(Form5.Log.Lines[7] ,5,4));
          IF (N1=9)  Then VCC3    :=StrToInt( Copy(Form5.Log.Lines[8] ,4,1));
          IF (N1=10) Then VTempo3 :=StrToInt( Copy(Form5.Log.Lines[9] ,4,4));
          IF (N1=11) Then VTemper4:=StrToInt( Copy(Form5.Log.Lines[10],5,4));
          IF (N1=12) Then VCC4    :=StrToInt( Copy(Form5.Log.Lines[11],4,1));
          IF (N1=13) Then VTempo4 :=StrToInt( Copy(Form5.Log.Lines[12],4,4));
          IF (N1=14) Then VTemper5:=StrToInt( Copy(Form5.Log.Lines[13],5,4));
          IF (N1=15) Then VCC5    :=StrToInt( Copy(Form5.Log.Lines[14],4,1));
          IF (N1=16) Then VTempo5 :=StrToInt( Copy(Form5.Log.Lines[15],4,4));
          IF (N1=17) Then VTemper6:=StrToInt( Copy(Form5.Log.Lines[16],5,4));
          IF (N1=18) Then VCC6    :=StrToInt( Copy(Form5.Log.Lines[17],4,1));
        End;
      // Escrevendo os dados no proximo FORM!
      Form3.SP1.Value:=VTemper1;
      Form3.SP2.Value:=VTemper2;
      Form3.SP3.Value:=VTemper3;
      Form3.SP4.Value:=VTemper4;
      Form3.SP5.Value:=VTemper5;
      Form3.SP6.Value:=VTemper6;
      Form3.T1.Value :=VTempo1;
      Form3.T2.Value :=VTempo2;
      Form3.T3.Value :=VTempo3;
      Form3.T4.Value :=VTempo4;
      Form3.T5.Value :=VTempo5;
      IF (VCC1=1) Then Form3.E1.Text:='SIM' Else Form3.E1.Text:='NÃO';
      IF (VCC2=1) Then Form3.E2.Text:='SIM' Else Form3.E2.Text:='NÃO';
      IF (VCC3=1) Then Form3.E3.Text:='SIM' Else Form3.E3.Text:='NÃO';
      IF (VCC4=1) Then Form3.E4.Text:='SIM' Else Form3.E4.Text:='NÃO';
      IF (VCC5=1) Then Form3.E5.Text:='SIM' Else Form3.E5.Text:='NÃO';
```

```
      IF (VCC6=1) Then Form3.E6.Text:='SIM' Else Form3.E6.Text:='NÃO';
      // Calculando as taxas...
      IF (VTempo1>0) Then Tx1:=(VTemper2-VTemper1) DIV VTempo1;
      IF (VTempo2>0) Then Tx2:=(VTemper3-VTemper2) DIV VTempo2;
      IF (VTempo3>0) Then Tx3:=(VTemper4-VTemper3) DIV VTempo3;
      IF (VTempo4>0) Then Tx4:=(VTemper5-VTemper4) DIV VTempo4;
      IF (VTempo5>0) Then Tx5:=(VTemper6-VTemper5) DIV VTempo5;

      IF (Tx1<>0) Then Form3.Taxa1.Caption:=IntToStr(Tx1)+' ºC/min' Else Form3.Taxa1.Caption:='Patamar';
      IF (Tx2<>0) Then Form3.Taxa2.Caption:=IntToStr(Tx2)+' ºC/min' Else Form3.Taxa2.Caption:='Patamar';
      IF (Tx3<>0) Then Form3.Taxa3.Caption:=IntToStr(Tx3)+' ºC/min' Else Form3.Taxa3.Caption:='Patamar';
      IF (Tx4<>0) Then Form3.Taxa4.Caption:=IntToStr(Tx4)+' ºC/min' Else Form3.Taxa4.Caption:='Patamar';
      IF (Tx5<>0) Then Form3.Taxa5.Caption:=IntToStr(Tx5)+' ºC/min' Else Form3.Taxa5.Caption:='Patamar';
      Form3.Showmodal;
    End;
end;  end.
```

APÊNDICE E – CÓDIGO FONTE – UNIDADE 5

```
unit Unit5;    // Janela sobre o programa

interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, Buttons;

type Sobre = class(TThread)
   Private
   Protected
   Procedure Execute; override;
   Procedure Rolagem;
End;

type
  TForm6 = class(TForm)
    Memo1: TMemo;
    SpeedButton1: TSpeedButton;
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form6: TForm6;
  XX   : Integer;
implementation

Procedure Sobre.Execute;
Begin
    While (XX=0) Do
    Begin
      Rolagem;
      Form6.Memo1.Top:=260;
    End;
End;

Procedure Sobre.Rolagem;
Begin
    While (Form6.Memo1.Top>-500) AND (XX=0) Do
    Begin
       Form6.Memo1.Top:=Form6.Memo1.Top-1;
       Sleep(30);
    End;
End;




procedure TForm6.SpeedButton1Click(Sender: TObject);
begin
    XX:=1;
    Sobre.Create(True);
```

```
   Form6.Close;
end;

procedure TForm6.FormActivate(Sender: TObject);
begin
   Form6.Memo1.Top:=260;
   XX:=0;
   Sobre.Create(False);
end;
end.
```

APÊNDICE F – FOTOGRAFIAS DO SISTEMA