Este exemplar corresponde a redação final da tese defendida por **Valmir Fascio Juliano** e aprovado pela comissão julgadora em 08/07/1991.

Prof. Dr. Marco-Aurelio De Paoli

UNIVERSIDADE ESTADUAL DE CAMPINAS

INSTITUTO DE QUÍMICA

Título:

CONSTRUÇÃO DE EQUIPAMENTO PARA ELETROQUÍMICA E CARACTERIZAÇÃO DE FILMES DE POLIPIRROL

Tese de Mestrado

Valmir Fascio Juliano

Orientador: Marco-Aurelio De Paoli

Campinas - 1991

Dedico este trabalho a quem dedicou a mim grande parte de sua vida:

meus pais: Valdemar e Mercedes;

AGRADECIMENTOS

Agradeço,

Ao CNPq e CAPES pelas bolsas de estudo;

Ao Marco que, além de orientador, foi sempre um grande amigo;

A minha esposa Marcia que, além de ter aturado a minha falta de colaboração em casa durante algum tempo, me deu apoio e auxílio quando foram necessários;

A minha sogra Leonor, ao meu sogro Orlando e a meus pais pela ajuda financeira nos momentos de atraso da bolsa;

Ao grande amigo e sócio Claudimir pelas explicações na área de eletrônica, além de colaborar no desenvolvimento deste trabalho;

Ao pessoal do grupo do Marco que foram "cobaias" durante os testes com os programas desenvolvidos e aos demais do grupo pela ajuda de um modo geral;

A todos que direta ou indiretamente ajudaram na realização deste trabalho;

ÍNDICE

RESUMO

Na última década, a pesquisa em novos materiais poliméricos condutores tem aumentado intensivamente como consequência do seu grande potencial de aplicações tecnológicas, tais como: baterias plásticas recarregáveis, *displays* eletro-ópticos, blindagem para radiação eletromagnética, embalagens anti-estáticas, superfícies condutoras, etc.. A facilidade de preparação destes materiais condutores, em escala de laboratório, tem também contribuido para este interesse. Entretanto sua produção em larga escala ainda depende fortemente da processabilidade do material.

Polipirrol (e outros polímeros heterocíclicos) pode ser preparado por oxidação eletroquímica do monômero, produzindo filmes condutores os quais são dopados simultaneamente com a sua formação. Foi sintetizado desta forma pela primeira vez por Weiss e colaboradores em 1965. Mais tarde, vários pesquisadores começaram a estudar intensivamente o polipirrol sob várias condições. Muitos autores também tem reportado que polipirrol, obtido na presença de surfactantes, tem melhores propriedades mecânicas associadas com a condutividade elétrica e a eletroatividade. De todos estes estudos constatou-se que as propriedades destes filmes dependem do potencial elétrico usado, da densidade de corrente, da concentração dos reagentes, do solvente, dos eletrodos, do formato da cela eletroquímica e principalmente do contra-íon.

Estes métodos eletroquímicos para síntese e análise, quando realizados pela maneira convencional, ou seja, utilizando um potenciostato/galvanostato, um gerador de funções e um registrador XY ou XT, requerem um certo gasto de tempo. Este tempo, obviamente, está dividido entre ficar monitorando as condições nas quais está ocorrendo o ensaio, a coleta de dados do

papel do registrador e o seu tratamento. Estas duas últimas operações são as que consomem mais tempo. Por outro lado, quando realizadas através do auxílio de um computador, este toma o lugar do operador na monitoração das condições do ensaio, bem como torna muito mais rápido e eficaz o tratamento dos dados coletados, os quais ficam na memória do computador ou arquivados em disquete.

O sistema desenvolvido (*hardware* e *software*) tornou possível a aquisição de dados para as técnicas de voltametria cíclica, cronoamperometria, cronopotenciometria e cronocoulometria. Permitiu também medidas da variação da transmitância em função do potencial aplicado em experimentos de eletrocromismo.

O *hardware*, construído a partir de componentes eletrônicos bastante comuns, mostrou-se eficaz até na técnica de cronoampermometria, onde as respostas precisam ser rápidas (poucos milissegundos).

O *software*, englobando uma série de programas, permitiu além da aquisição o tratamento de dados, tornando possível o cálculo dos parâmetros eletroquímicos como potencial redox, corrente de pico, carga anódica e catódica e outros. Estes dados são necessários para a determinação de certas propriedades do material em estudo.

Com o intuito de verificar o desempenho do equipamento, foram feitos ensaios com soluções de $FeCl_3$, $K_3Fe(CN)_6$ e $Fe(C_5H_5)_2$ e com filmes de polipirrol em condições bem conhecidas. Os resultados foram bastante satisfatórios e, mais uma vez, demonstraram a versatilidade do sistema informatizado.

ABSTRACT

In the last decade, the research in new conducting polymeric materials has increased intensively as a result of their great potential in technological applications, such as: rechargeable all-plastic batteries, electro-optical devices, electromagnetic radiation shielding, anti-static packages, conducting surfaces, etc. The facility of preparation of these conducting materials, in laboratory scale, has also contributed for this interest. However, its production in large scale strongly depends on the material processability.

Polypyrrole (and other heterocyclic polymers) can be prepared by electrochemical oxidation of the monomer, producing conducting films, which are doped simultaneously with its formation. It was first sinthetized in this form by Weiss and co-workers in 1965. Later, many researchers started to study polypyrrole intensively under different conditions. Many authors also reported that polypyrrole, obtained in the presence of surfactants, has better mechanical properties, electrical conductivity and electroativity.

From these studies we observe that the film properties depends on several variables: electrical potential, current density, reactants concentration, solvent, electrodes, electrochemical cell shape and principally on the counter ion.

These electrochemical methods for synthesis and analysis, when realized by the conventional manner, i.e., using a potentiostat/galvanostat, a function generator and a XY or XT plotter, require a long time. This time is obviously used to monitorize the conditions which the experiment is occurring, the data acquisition by the plotter and data treatment. These last two operations are the most time consuming. On the other hand, when they are realized with computer assistance, the operator is not

necessary to monitorize the experimental conditions. Also the data treatment becomes very fast and effective. In addition the data can be stored in floppy disks making its treatment a lot easier.

The system developed (hardware and software) enables the data acquisition and treatment for cyclic voltammetry, chronoamperometry, chronopotenciometry and chronocoulometry. It also permits measurement of transmittance variation as a function of applied potential in experiments of electrochromism.

The hardware, constructed with common electronic components, is efficient even for the technic of chronoamperometry, where the response times required are very short (few milliseconds).

The software, comprising a series of programs, permited, beyond acquisition, the treatment of the data, making possible the calculation of electrochemical parameters, such as: redox potential, peak current, anodic and cathodic charges and others. These data are necessary for determination of the properties of the material in study.

In order to verify the performance of the equipment, experiments with solutions of $FeCl_3$, $K_3Fe(CN)_6$ and $Fe(C_5H_5)_2$ and films of polypyrrole dodecylsulfate in well known conditions were performed.

The results were very satisfactory and, once again, demonstrated the versatility of the computerized system.

ABREVIAÇÕES

A/D	Analógico para Digital
AEN	*Address Enable* (Habilitação de Endereços)
AMPOP	Amplificador Operacional
CI	Circuito Integrado
D/A	Digital para Analógico
DMA	*Direct Memory Access* (Acesso Direto a Memória)
EA	Eletrodo Auxiliar
ER	Eletrodo de Referência
ET	Eletrodo de Trabalho
H	Um número seguido de uma letra H indica a base Hexadecimal
IOR	*Input & Output - Read* (Entrada e Saída - Leitura)
IOW	*Input & Output - Write* (Entrada e Saída - Escrita)
LSB	*Least Significant Bit* (Bit menos significativo)
MSB	*Most Significant Bit* (Bit mais significativo)
PPi	Polipirrol
PPi/DS	Polipirrol/Dodecilsulfato
PPi/DBS	Polipirrol/Dodecilbenzenosulfonato
S&H	*Sample and Hold* (Amostragem e Retenção)

SÍMBOLOS E SUAS UNIDADES

A	área do eletrodo de trabalho. (cm^2)
α	coeficiente de transferência de carga.
C_i	concentração da espécie i. ($mol.cm^{-3}$)
C^*	concentração das espécies eletroativas. ($mol.cm^{-3}$)
C_{Ox}	concentração das espécies oxidadas. ($mol.cm^{-3}$)
C_{Rd}	concentração das espécies reduzidas. ($mol.cm^{-3}$)
D_i	coeficiente de difusão da espécie i. ($cm^2.s^{-1}$)
D_{Ox}	coeficiente de difusão da espécie oxidada. ($cm^2.s^{-1}$)
D_{Rd}	coeficiente de difusão da espécie reduzida. ($cm^2.s^{-1}$)
ΔE_p	$E_{pa} - E_{pc}$. (V)
E^o	potencial padrão de reação. (V)
E	potencial do eletrodo. (V)
$E_{1/2}$	potencial de meia onda. (V)
$E_{pa/2}$	potencial de meio pico anódico. (V)
$E_{pc/2}$	potencial de meio pico catódico. (V)
E_{pa}	potencial de pico anódico. (V)
E_{pc}	potencial de pico catódico. (V)
F	constante de Faraday. (96489 J/V)
i	corrente elétrica. (A)
i_p	corrente de pico (anódico ou catódico). (A)
i_{pa}	corrente de pico anódico. (A)
i_{pc}	corrente de pico catódico. (A)

J	fluxo de matéria na interface do eletrodo. (mol.cm^{-2}.s^{-1})
J_i	fluxo de matéria da espécie i. (mol.cm^{-2}.s^{-1})
k^o	constante de transferência de carga. (cm.s^{-1})
ℓ	espessura do filme aderido na superfície do eletrodo de trabalho. (cm)
n	número de elétrons envolvidos no processo redox.
Q_d	carga devida ao processo difusional. (C)
Q_{dc}	carga devida à dupla camada elétrica. (C)
R	constante dos gases. (8,314 J.mol^{-1}.K^{-1})
T	temperatura. (K)
t	tempo. (s)
υ	velocidade de varredura. (V.s^{-1})

CAPÍTULO 1
INTRODUÇÃO

Neste capítulo serão vistos os motivos que levaram um grande número de pesquisadores a se dedicarem ao estudo dos polímeros condutores, as técnicas empregadas nestes estudos e os avanços na área de eletroquímica graças à informática.

1.1. SEMICONDUTORES ORGÂNICOS

Nos últimos anos a utilização de semicondutores orgânicos (incluindo os polímeros condutores) tem aumentado significativamente. Embora estes materiais sejam estudados intensivamente a mais de trinta anos, estamos longe de entendê-los tão bem quanto os semicondutores inorgânicos.

Muito esfôrço tem sido feito e reportado para usar estes materiais como componente ativo em dispositivos elétricos, eletrônicos e eletro-ópticos, tais como baterias recarregáveis, *displays* eletrocrômicos, células solares, transistores de efeito de campo, etc..[1]

Inicialmente os pesquisadores estavam interessados principalmente no efeito fotovoltáico com o objetivo de produzir células solares orgânicas mais baratas. Para tal intento, corantes orgânicos tais como merocianinas, ftalocianinas e diftalocianinas foram os compostos mais estudados devido à sua alta absorção na faixa do visível.[2-4] Células orgânicas fotovoltáicas têm também sido fabricadas com polímeros conjugados como o poliacetileno,[5] poli(N-vinil carbazol),[5] vários derivados do politiofeno,[6] e polipirrol.[7] Mais recentemente, a utilização de semicondutores orgânicos na fabricação de transistores de efeito de campo (FETs) tem sido reportada. Os materiais mais usados foram metaloftalocianinas, poliacetileno, derivados do politiofeno e

α-sexitienil, sendo que este último, um oligômero de tiofeno, tem sido estudado como uma molécula modêlo para seus polímeros homólogos.[5,8-11]

O interesse crescente por polímeros condutores, tendo em vista a sua aplicação em dispositivos eletrônicos, é devido à sua característica peculiar que permite variar reversivelmente a condutividade elétrica, passando de isolante para semicondutor e para condutor, através de dopagem química ou eletroquímica com um aceptor ou doador de elétrons. Estas mudanças na condutividade (várias ordens de grandeza) são acompanhadas de modificações nas suas propriedades eletrônicas, tal como a estrutura das bandas de energia, as quais podem levar à fabricação de um dispositivo eletrônico com características únicas.

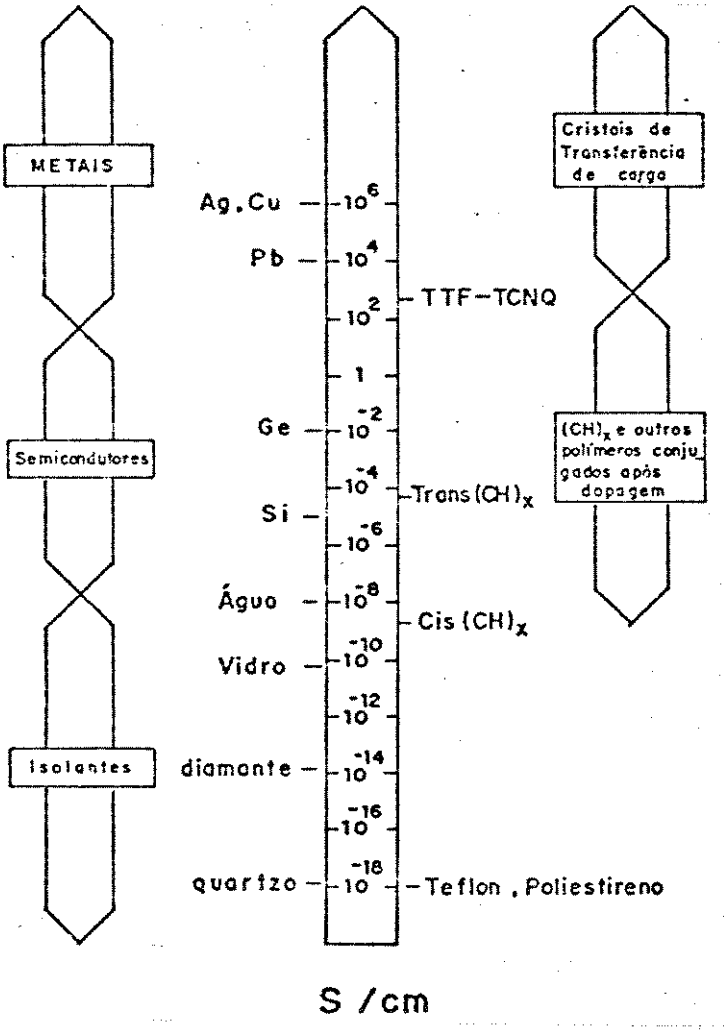


Figura 1.1: Condutividade elétrica de alguns materiais.[12]

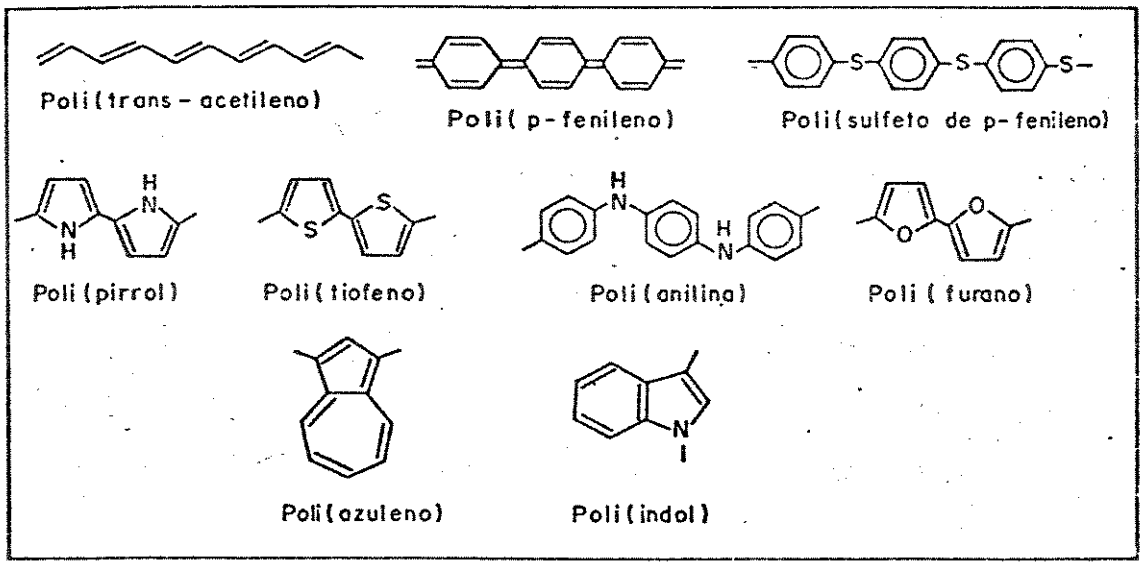


Figura 1.2: Estrutura dos polímeros condutores mais estudados.

O termo dopagem citado anteriormente foi emprestado da química dos semicondutores inorgânicos, onde o dopante é um átomo com deficiência (dopante do tipo *p*) ou excesso de elétrons (dopante do tipo *n*) que é inserido na rede cristalina do semicondutor em pequenas quantidades (ppm). Já para os polímeros condutores o dopante atua como um agente oxidante ou redutor e participa em quantidades estequiométricas.

A figura 1.1 mostra um quadro comparativo entre a condutividade elétrica de diversos materiais e a figura 1.2 mostra a estrutura dos polímeros condutores mais estudados.

A facilidade de preparação destes polímeros condutores, em escala de laboratório, tem também contribuído para o aumento de pesquisas nesta área, entretanto, sua produção em larga escala ainda depende fortemente da processabilidade do material.

1.2. O POLIPIRROL (PPi)

A maioria dos polímeros condutores podem ser preparados pela oxidação eletroquímica do monômero, produzindo filmes condutores aderidos sobre o eletrodo de trabalho, os quais são dopados simultaneamente com a sua formação. Foi desta maneira

que Weiss e colaboradores[13] prepararam pela primeira vez o PPi em 1965. Mais tarde, vários pesquisadores começaram a estudar intensivamente o polipirrol sob várias condições.[14-16] Muitos autores também têm reportado que o PPi, obtido na presença de surfactantes, tem melhores propriedades mecânicas associadas com a condutividade e a eletroatividade.[17-22]

A partir de todos estes estudos constatou-se que as propriedades destes filmes dependem do potencial elétrico utilizado, da densidade de corrente, da concentração dos reagentes, do solvente, dos eletrodos, do formato da cela eletroquímica e principalmente do contra-íon (dopante).

1.2.1. SÍNTESE ELETROQUÍMICA DO PPi

A síntese eletroquímica do PPi é feita pela oxidação anódica do pirrol resultando no crescimento de um filme condutor insolúvel aderido à superfície do anôdo (eletrodo de trabalho). O filme polimérico incorpora o ânion do sal usado como eletrólito que age como íon dopante.

Dentro do método eletroquímico, a polimerização pode ser realizada por três métodos diferentes, dependendo da variável em estudo:

- potenciostático: consiste em aplicar uma diferença de potencial constante entre os eletrodos de trabalho e auxiliar;
- potenciodinâmico: consiste em variar linearmente o potencial de um limite a outro a uma dada velocidade de varredura por um certo número de ciclos;
- galvanostático: consiste em manter uma densidade de corrente (corrente por unidade de área do eletrodo de trabalho) constante através da solução durante um determinado tempo.

As vantagens dos métodos eletroquímicos são:[23]

- a reação é feita à temperatura ambiente;
- a espessura do filme pode ser controlada pela quantidade de carga eletroquímica consumida na reação;
- o filme polimérico é formado diretamente na superfície do eletrodo;

- o polímero é dopado simultaneamente com o ânion do sal do eletrólito e o grau de dopagem (grau de oxidação) é controlado pelo potencial do eletrodo;
- o método permite a obtenção de copolímeros, copolímeros enxertados e compósitos.[29]

A principal vantagem do método eletroquímico é a dopagem simultânea ao crescimento do filme. O eletrólito de suporte, além de permitir a passagem de corrente elétrica pela solução, também age como dopante do polímero.[23,24]

A figura 1.3 mostra o mecanismo sugerido para a eletropolimerização do pirrol. Essencialmente, a formação do polímero resulta do acoplamento entre cátions-radicais. Após uma etapa inicial de oxidação, ocorre uma reação de acoplamento seguida de desprotonação com regenaração do sistema aromático.[24,25]

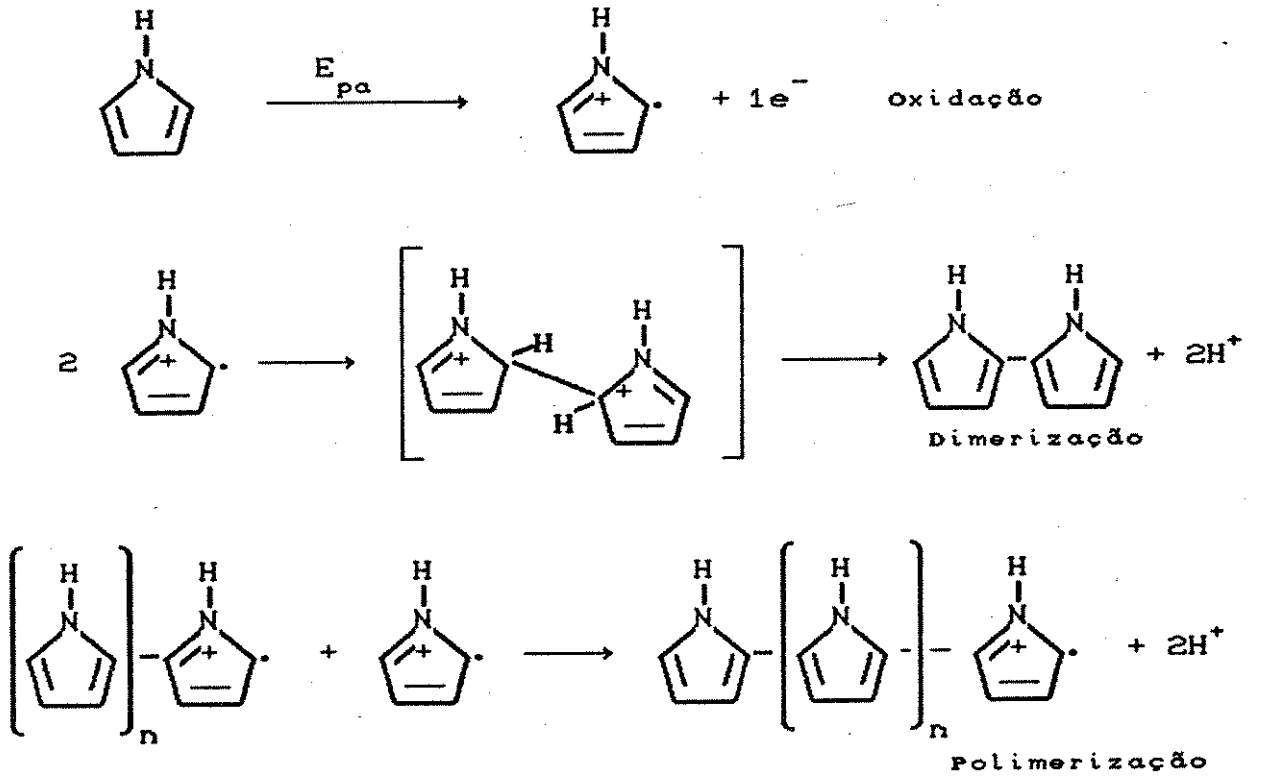


Figura 1.3: Mecanismo da reação de eletropolimerização do pirrol.

1.2.2. PROPRIEDADES ELETROATIVAS DO PPi

As propriedades eletroativas do PPi se devem às características redox do processo de dopagem e desdopagem que envolve a oxidação e redução, respectivamente, do sistema *pi* altamente delocalizado sobre a cadeia polimérica. Esta reação redox é quimicamente reversível e, dependendo do contra-íon, o PPi pode ser ciclado repetidas vezes sem perda de eletroatividade.[24]

O comportamento eletroativo do polímero depende da solução utilizada durante o processo de ciclagem e na preparação do filme. O processo redox do polímero é acompanhado pelo movimento do eletrólito (solvente e ânion) dentro e fora do filme a fim de neutralizar a carga catiônica da cadeia pirrólica[26] e, portanto, depende da difusão do íon através do filme.[25] Como a velocidade de dopagem e desdopagem é limitada pela mobilidade do ânion através do material, a ciclabilidade depende sensivelmente do dopante. Entretanto, este tem influência apenas na cinética do processo redox, não interferindo no potencial de oxidação do polímero.[24]

1.3. A ELETROQUÍMICA

A eletroquímica é uma área especializada e bem desenvolvida da química, com uma série completa de teorias e relações quantitativas. É uma das mais velhas especialidades da físico-química clássica e suas origens remontam da metade do século dezenove. Estas relações quantitativas tornam possível aplicar a eletroquímica para caracterizar espécies químicas e processos em solução.[27]

Durante as últimas décadas a dinâmica e mecanismos de processos de transferência de elétrons tem sido estudados por numerosos grupos através do mundo científico. Isto tem sido feito pela aplicação da teoria de transição de estado aos processos cinéticos eletroquímicos. Como resultado, a cinética dos processos de transferência de elétrons (de um eletrodo sólido para espécies em solução) podem ser caracterizados quantitativamente.[27]

Com o advento dos modernos equipamentos eletrônicos associados ao desenvolvimento de teorias mais pragmáticas tornou-se possível o estudo termodinâmico de reações instáveis, cinéticas de processos de transferência de elétrons em meio aquoso e não aquoso e caracterização dos fenômenos de adsorção (importantes para o entendimento dos processos catalíticos).[27]

As aplicações da eletroquímica não se restringem somente a físico-química e química analítica. Algumas das mais interessantes aplicações tem ocorrido nas áreas de química orgânica, inorgânica e bioquímica. Na área de química orgânica o controle dos processos de oxidação ou redução através da eletroquímica é muito mais preciso que através de reagentes químicos. Na área de química inorgânica a eletroquímica tem sido, especialmente, útil para a determinação de fórmulas de complexos de coordenação e síntese e determinação da estequiometria de transferência de elétrons em compostos organometálicos. Na área de bioquímica, a eletroquímica tem sido utilizada para estudos de reações de oxi-redução por enzimas.[27]

1.3.1. MÉTODOS ELETROQUÍMICOS

Como foi mencionado nos ítens anteriores, a dopagem e desdopagem de polímeros condutores se dão através de um processo redox, conferindo propriedades eletroativas ao material. Devido a isto, os polímeros condutores têm sido caracterizados principalmente por métodos eletroquímicos.

Se for considerado uma reação genérica,

$$Ox + ne^- \rightleftharpoons Rd$$

onde, Ox e Rd correspondem às espécies oxidada e reduzida, respectivamente, o processo mais simples que pode ocorrer na superfície de um eletrodo é uma reação de transferência de carga.

Outro processo que pode ocorrer é o transporte de matéria, que consiste no movimento das espécies do seio da solução até o eletrodo, sob influência de um campo de potencial elétrico.

A transferência de carga na interface do eletrodo é

descrita pela equação de Butler-Volmer, que é a equação básica da cinética eletroquímica.[28]

$$J = \frac{i}{n \times F \times A} = k^{\circ} \times \left\{ C_{ox}(O,t) \times K^{\circ} \times \exp\left[-\alpha \times \frac{n \times F}{R \times T} \times (E - E^{\circ})\right] - C_{Rd}(O,t) \times \exp\left[(1-\alpha) \times \frac{n \times F}{R \times T} \times (E - E^{\circ})\right] \right\}$$

onde:
- k° = constante de transferência de carga
- α = coeficiente de transferência de carga
- E° = potencial padrão de reação
- E = potencial do eletrodo
- J = fluxo de matéria na interface do eletrodo
- F = constante de Faraday
- R = constante dos gases
- T = temperatura
- n = número de elétrons envolvidos na reação
- i = corrente elétrica
- A = área do eletrodo de trabalho
- C_{ox} = concentração das espécies oxidadas
- C_{Rd} = concentração das espécies reduzidas.

A transferência de massa é descrita pelas leis de difusão de Fick:[27,29]

$$J_i = - D_i \frac{\partial C_i}{\partial x} \qquad e \qquad \frac{\partial C_i}{\partial t} = - D_i \frac{\partial C_i}{\partial x^2}$$

No caso de um polímero eletroativo aderido a um eletrodo metálico, considera-se duas interfaces distintas: a interface eletrodo/polímero, onde ocorre a transferência eletrônica entre o metal e o polímero, e a interface polímero/solução de eletrólito onde os contra-íons vêm compensar a carga elétrica e são transferidos do seio da solução para os sítios ativos imobilizados dentro da matriz polimérica. Admite-se, em primeira aproximação, que a transferência eletrônica seja muito rápida (sistema reversível) e que a velocidade da reação redox é cinéticamente limitada pelo transporte de matéria, que pode ser considerado equivalente à um processo de difusão. Classifica-se o

transporte de matéria em 3 modos denominados: transporte por transiente não convectivo, estacionário convectivo e temporal.[30,31]

Para caracterização de filmes poliméricos aderidos a um eletrodo são utilizados, basicamente, quatro métodos: 1) Voltametria cíclica, 2) Cronoamperometria, 3) Cronocoulometria e 4) Cronopotenciometria. Embora existam outros métodos, estes são os mais simples e mais diretos. A tabela 1.1 mostra os métodos, as variáveis controladas e as variáveis mensuráveis.

Método	Variável controlada	Variável mensurável
Voltametria cíclica	E(t)	i
Cronoamperometria	salto de E	i(t)
Cronocoulometria	salto de E	$\int i\,dt$
Cronopotenciometria	salto de i	E(t)

Tabela 1.1. Nomenclatura da Metodologia Eletroquímica.[27]

1.3.1.1. Voltametria Cíclica

Voltametria cíclica é um método onde o sinal de excitação do sistema eletroquímico é um onda triangular. Esta onda é obtida através da variação linear com o tempo do potencial do eletrodo, indo de um potencial inicial, E_i, até um potencial final, E_f, e retornando ao potencial inicial E_i, com uma inclinação que depende da velocidade de varredura. Isto pode ser visto na figura 1.4a.

Como resposta do sistema tem-se a variação da corrente em função do potencial, conforme pode ser visto na figura 1.4b, de onde obtém-se diretamente os seguintes dados:

E_{pa} = potencial de pico anódico
E_{pc} = potencial de pico catódico
i_{pa} = corrente de pico anódico
i_{pc} = corrente de pico catódico
$\Delta E_p = E_{pa} - E_{pc}$

$E_{1/2}$ = potencial de meia onda
$E_{pa/2}$ = potencial de meio pico anódico
$E_{pc/2}$ = potencial de meio pico catódico

O potencial padrão de reação E° é obtido pela intersecção da linha que une os pontos (E_{pa}, i_{pa}) (E_{pc}, i_{pc}) com o eixo de potencial.

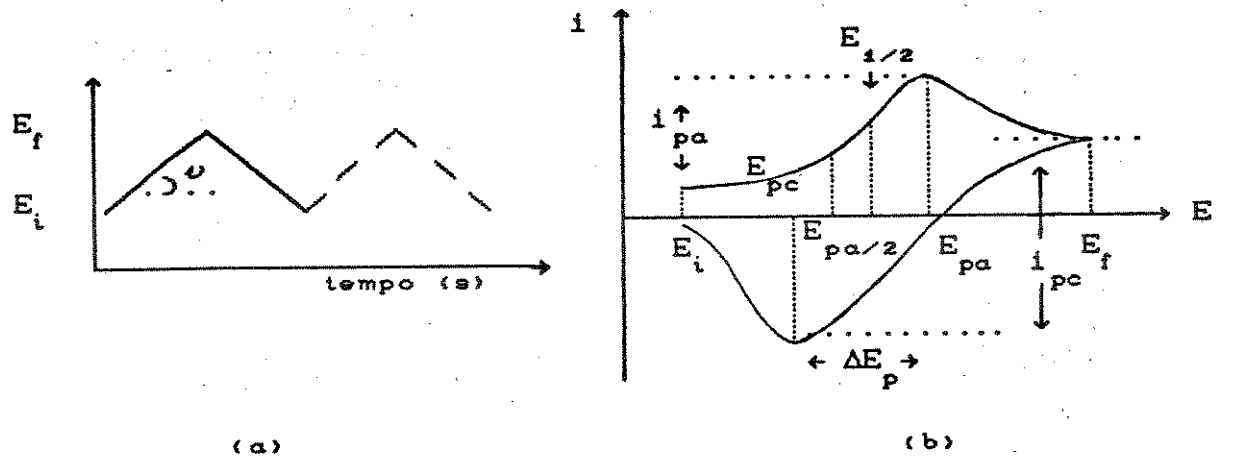


Figura 1.4: Experimento de voltametria cíclica: (a) aspecto do sinal de excitação - varredura de potencial com velocidade υ e (b) aspecto da resposta do sistema eletroquímico à excitação.

O valor de ΔE_p fornece informações a respeito da reversibilidade do sistema. Se $\Delta E_p \geq 90$ mV o sistema é irreversível e se $\Delta E_p < 90$ mV o sistema é reversível.[29,30]

No caso de uma reação reversível, onde a reação é rápida o suficiente para manter as concentrações das formas oxidadas e reduzidas em equilíbrio no eletrodo, as seguintes relações podem ser aplicadas:[27,32]

$$\Delta E_p = |E_{pa} - E_{pc}| = 59,5 / n \quad (\text{mV a } 25^{\circ}C)$$

$$E_{1/2} = E_{pa} - \Delta E_p / 2 = (E_{pa} + E_{pc}) / 2$$

$$|E_p - E_{p/2}| = 2,2 \times (RxT / nxF) = 56,5 / n \quad (\text{mV a } 25^{\circ}C)$$

$$|E_{1/2} - E_{p/2}| = 1,09 \times (RxT / nxF) = 28 / n \quad (\text{mV a } 25^{\circ}C)$$

$$E^{\circ} = E_{1/2} + (RxT / nxF) \times \ln(D_{Ox} / D_{Rd})^{1/2}$$

onde : D_{Ox} = coeficiente de difusão da espécie oxidada

D_{Rd} = coeficiente de difusão da espécie reduzida

$E_{p/2} = E_{pa/2}$ ou $E_{pc/2}$.

Neste método são observados 2 modos de transporte de matéria e a expressão da corrente é obtida a partir da equação de Nernst e das leis de Fick (válida para oxidação e redução).

I - Transporte por transiente não convectivo: a cinética da reação é limitada pela difusão das espécies do seio da solução até a superfície do eletrodo (difusão semi-infinita).

$$i_p = 0,4463 \times (nxF)^{3/2} \times C^* \times A \times (D/RxT)^{1/2} \times \upsilon^{1/2} \Rightarrow i_p/\upsilon^{1/2} = \text{constante}$$

II - Transporte temporal: o transporte de matéria é desprezível, caracterizado por uma reação superficial com difusão apenas na superfície do eletrodo (difusão em camada fina).

$$i_p = [(nxF)^2 \times C^* \times \upsilon \times \ell \times A]/4xRxT \Rightarrow i_p/\upsilon = \text{constante}$$

onde : i_p = corrente de pico (anódico ou catódico)

C^* = C_{Ox} ou C_{Rd}

D = D_{Ox} ou D_{Rd}

υ = velocidade de varredura

ℓ = espessura do filme aderido na superfície do eletrodo de trabalho.

1.3.1.2. Cronoamperometria

Cronoamperometria é um método onde se faz o potencial do eletrodo mudar instantaneamente de um valor inicial E_i para um valor final E_f, monitorando-se o transiente $i-t$, que resulta à medida que o sistema relaxa ao estado estacionário. Em cronoamperometria o transiente $i-t$ é registrado diretamente e

analisado em sua forma. Na figura 1.5 é mostrado a forma do sinal de excitação e a resposta do sistema eletroquímico a este tipo de excitação.

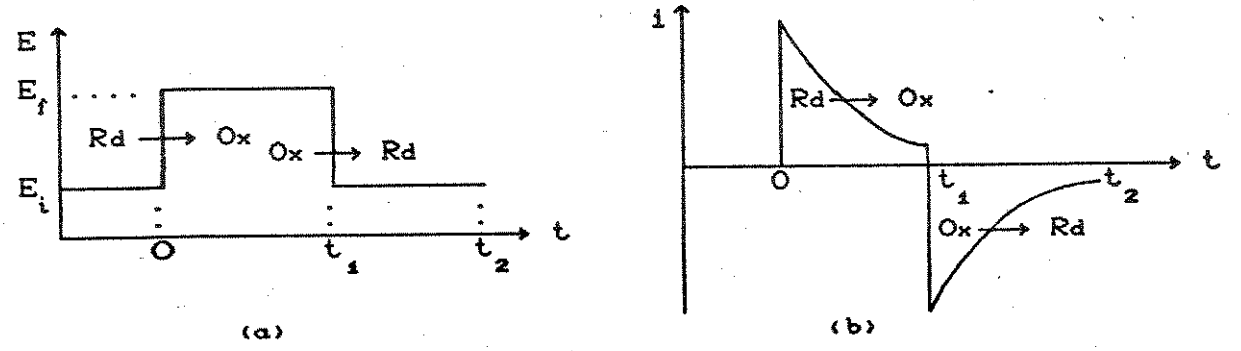


Figura 1.5: Experimento de Cronoamperometria: (a) forma de onda do sinal de excitação e (b) resposta do sistema para este tipo de excitação.

Técnicas de salto de potencial não são muito eficientes para investigação do mecanismo de reações químicas homogêneas na superfície do eletrodo. Por outro lado, se o mecanismo é conhecido, esta técnica oferece um método simples para obter dados quantitativos. O método tem vantagem sobre a voltametria cíclica naqueles estudos que são experimentalmente mais simples.[29]

Nesta técnica deve-se ter o cuidado de escolher os potenciais entre os quais será realizado o salto. O potencial inicial deve ser suficientemente negativo para que nenhuma reação ocorra e que exista em solução e no eletrodo somente as espécies reduzidas. Por outro lado o potencial final deve ser suficientemente positivo para que a reação seja máxima e limitada pela difusão das espécies reduzidas, as quais são consumidas rapidamente.[30] Deste modo a corrente é expressa pela equação de Cottrell:[27,32]

$$i(t) = i_d = (nxFxAxD^{1/2}xC^*)/(\Pi^{1/2}xt^{1/2}) \quad \Rightarrow \quad i = f(t^{-1/2})$$

1.3.1.3. Cronocoulometria

Este método, uma variação da cronoamperometria,

consiste em integrar a corrente (equação de Cottrell) em função do tempo, obtendo como resultado as curvas mostradas na figura 1.6.

A expressão para a carga Q em função do tempo é dada por:[27,32]

$$Q(t) = 2xnxFxAxD^{1/2}xC^{*}xt^{1/2} / \Pi^{1/2}$$

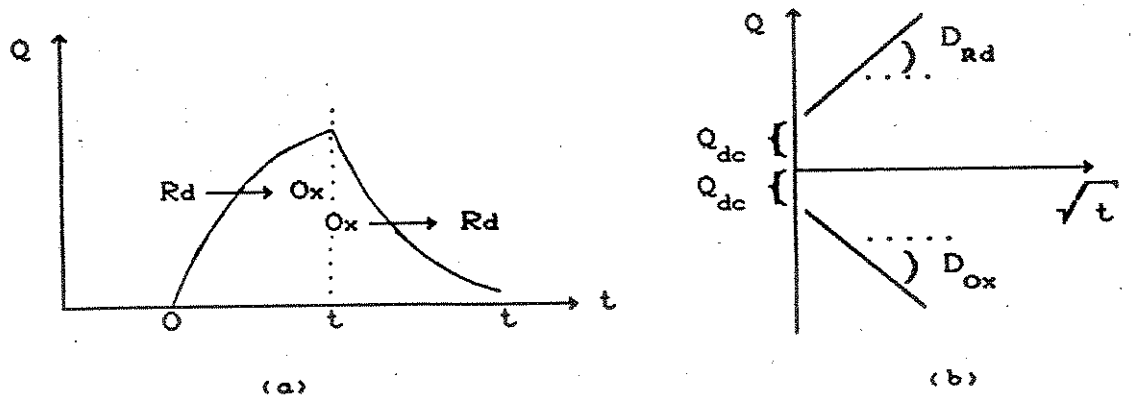


Figura 1.6: (a) Curva obtida após a integração da corrente em função do tempo e (b) procedimento gráfico para determinar os coeficientes de difusão.

De acordo com a expressão da carga, no tempo igual a zero a carga deve ser igual a zero. Entretanto, as vezes, existe um valor que é diferente de zero e, este, é devido à capacitância da dupla camada elétrica, assim, a carga deve ser expressa como

$$Q(t) = Q_d(t) + Q_{dc}(t)$$

onde: Q_d = carga devida ao processo difusional
Q_{dc} = carga devida à dupla camada elétrica.

A dupla camada elétrica é originada na interface entre a solução e a superfície do eletrodo, sendo constituída de excesso de cátions ou ânions e atuando como um capacitor.[30,32,33]

1.3.1.4. Cronopotenciometria

A cronopotenciometria consiste na monitoração do potencial do eletrodo de trabalho em função do tempo para uma dada corrente imposta ao sistema eletroquímico. A figura 1.7 mostra a forma do sinal de excitação e a forma da resposta do sistema. De

modo semelhante à cronoamperometria, a corrente i_a deve ser suficientemente anódica para que todas as espécies reduzidas sejam rapidamente oxidadas e a corrente i_c, por outro lado, deve ser suficientemente catódica para que todas as espécies oxidadas sejam rapidamente reduzidas.[27,30-32]

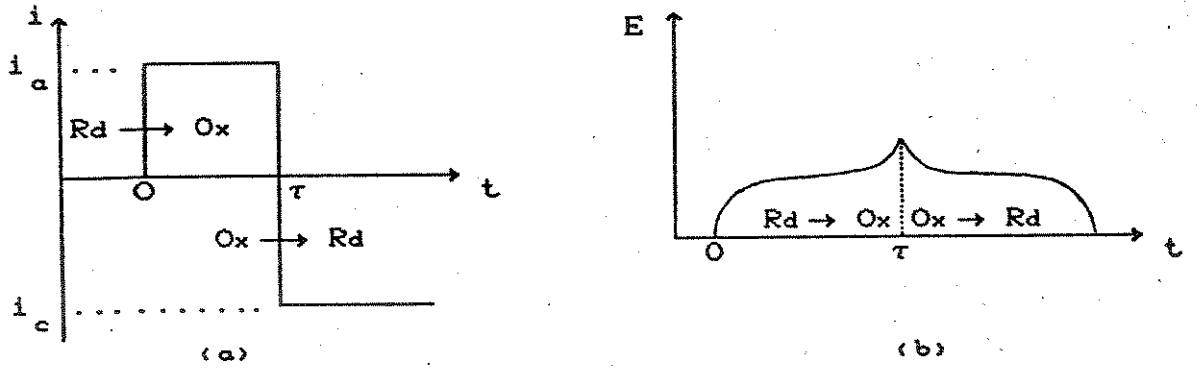


Figura 1.7: Experimento de cronopotenciometria: (a) forma do sinal de excitação e (b) resposta do sistema eletroquímico para a excitação.

O formato da curva do potencial em função do tempo depende da reversibilidade da reação no eletrodo, mas em geral, o potencial do eletrodo se move em direção do valor de potencial característico do par redox e varia com o tempo em função da razão das concentrações das espécies oxidada e reduzida na superfície do eletrodo.[92]

Sand em 1905 deduziu a equação que descreve a dependência do tempo de transição (τ - onde a concentração da espécie oxidada ou reduzida torna-se zero) com a corrente constante em um processo controlado por difusão.[27,92]

$$i \times \tau^{1/2} = \Pi^{1/2} \times n \times F \times A \times D^{1/2} \times C^{*} / 2$$

A partir desta equação e da equação de Nernst, a expressão do potencial em função do tempo é dada por

$$E = E_{\tau/4} + (R \times T / n \times F) \times \ln [(\tau^{1/2} - t^{1/2}) / t^{1/2}]$$

sendo $E_{\tau/4}$ o potencial de quarto de onda (equivalente ao $E_{1/2}$ da

voltametria) dado por:

$$E_{\tau/4} = E^{\circ} - (RxT / nxF) \times \ln (D_{Ox} / D_{Rd})^{1/2}$$

A reversibilidade do sistema eletroquímico é verificada pela linearidade do gráfico E *vs* $\log[(\tau^{1/2} - t^{1/2}) / t^{1/2}]$ com uma inclinação de 0,0595 / n volts.[92]

1.3.2. INSTRUMENTAÇÃO ELETROQUÍMICA

Toda medição elétrica envolve uma sequência de operações e cuidados a serem tomados, os quais têm efeito sobre a variável a ser medida. Por exemplo, para medir a diferença de potencial entre dois pontos deve-se utilizar um voltímetro com alta impedância de entrada, de modo que este não cause perturbações (queda de potencial) no valor a ser medido.

Assim, para medições em sistemas eletroquímicos, onde as diferenças de potencial a serem medidas possuem, na maioria dos casos, valores pequenos da ordem de μV e mV, há necessidade de instrumentos com alta impedância de entrada e com uma grande sensibilidade (ganho).

Os instrumentos modernos destinados a este tipo de medida são baseados nos amplificadores operacionais (AMPOPs - circuitos integrados a base de semicondutores inorgânicos) que possuem impedância de entrada da ordem de 10^{5} - 10^{13} ohms (valor típico $10^{6}\Omega$) e valores de ganho de 10^{4} - 10^{8} (valor típico 10^{5}), além de sua boa estabilidade térmica.[92]

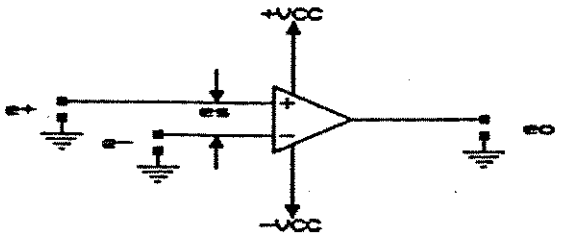


Figura 1.8: Diagrama esquemático de um Amplificador Operacional.

A figura 1.8 mostra um diagrama esquemático do AMPOP, onde e_- é a tensão da entrada inversora em relação ao terra (ponto comum do circuito), e_+ é a tensão da entrada não-inversora em relação ao terra, e_s é a diferença de tensão da entrada inversora em relação à entrada não-inversora e e_o é a tensão de saída (que equivale a e_s invertida e amplificada) em relação ao terra. Assim,

$$e_o = -Axe_s = -Axe_- + Axe_+$$

onde A é o ganho do AMPOP.

Com apenas um AMPOP, alguns resistores e alguns capacitores é possível montar diversos circuitos tais como, seguidores de corrente, seguidores de tensão, somadores, controladores de tensão, controladores de corrente, escalador/inversor, integradores e diferenciadores, cada um com diversas aplicações.[32]

A figura 1.9 mostra alguns dos tipos dos circuitos baseados nos AMPOPs, citados acima, bastante utilizados em instrumentação.

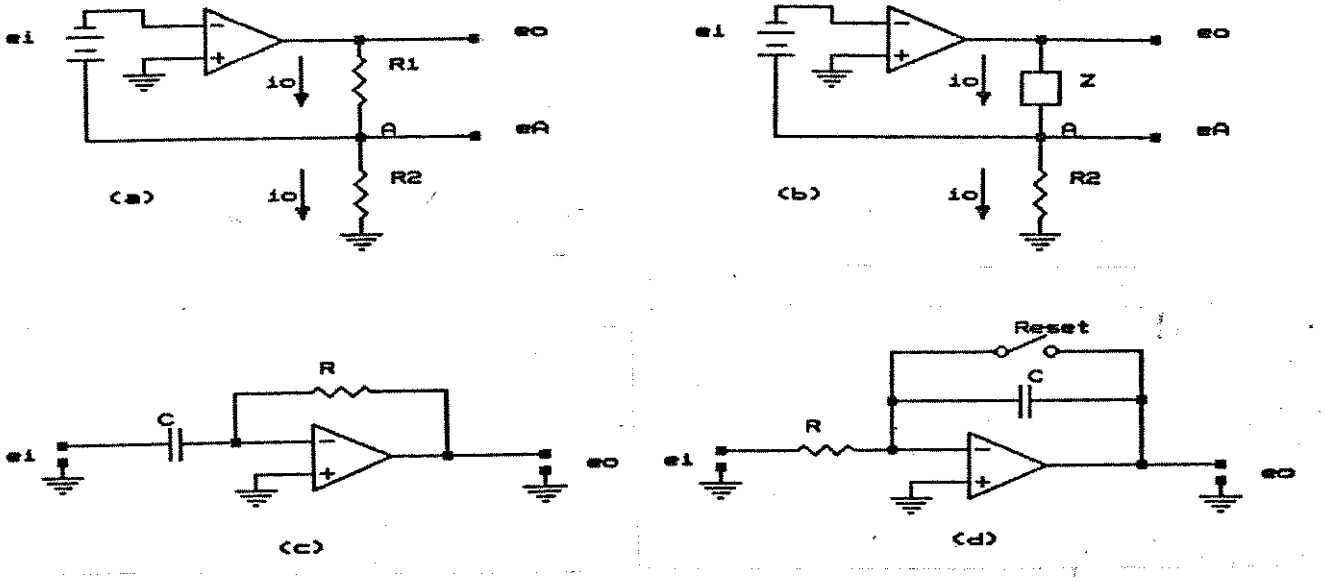


Figura 1.9: Alguns tipos de circuitos baseados nos AMPOPs: (a) controlador de tensão no ponto A; (b) controlador de corrente através de uma carga Z; (c) diferenciador e (d) integrador.

1.3.2.1. POTENCIOSTATOS

Do ponto de vista eletrônico, uma cela eletroquímica pode ser considerada como um sistema de várias impedâncias associadas, como pode ser visto na figura 1.10. Nesta figura Z_a e Z_t representam as impedâncias interfaciais nos eletrodos auxiliar (EA) e trabalho (ET), respectivamente, e a resistência da solução é dividida em duas frações, R_a e R_t, dependendo da posição do eletrodo de referência (ER) dentro da cela.

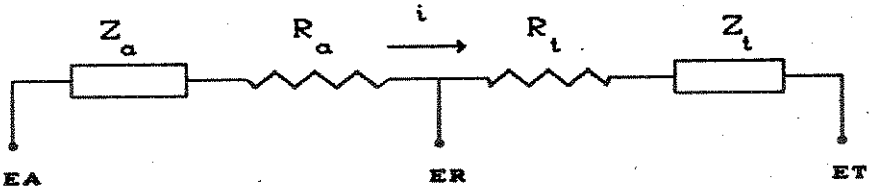


Figura 1.10: Diagrama representativo de uma cela eletroquímica do ponto de vista eletrônico.

Incorporando-se à cela um circuito como mostrado na figura 1.11, obtem-se um simples potenciostato baseado no circuito controlador de tensão da figura 1.9a. Nesta configuração, a tensão no eletrodo de referência é $-e_i$ em relação ao terra e, como o eletrodo de trabalho está aterrado, sua tensão é e_i em relação ao eletrodo de referência.[92]

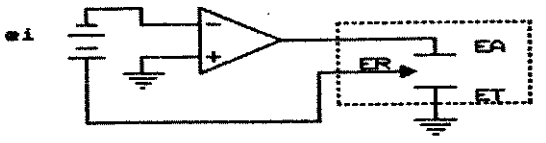


Figura 1.11: Diagrama de um simples potenciostato baseado no circuito controlador de tensão da figura 1.9a.

1.3.2.2. GALVANOSTATOS

Controlar a corrente através de uma cela eletroquímica é mais simples que controlar o potencial em um

eletrodo uma vez que somente dois elementos da cela, os eletrodos de trabalho e auxiliar, são envolvidos no circuito de controle. Como em experimentos galvanostáticos é de interesse monitorar o potencial do eletrodo de trabalho em relação ao eletrodo de referência, isto pode ser feito sem prejudicar o circuito de controle. A figura 1.12 mostra um diagrama de um galvanostato baseado no circuito de controlador de corrente da figura 1.9b.[32]

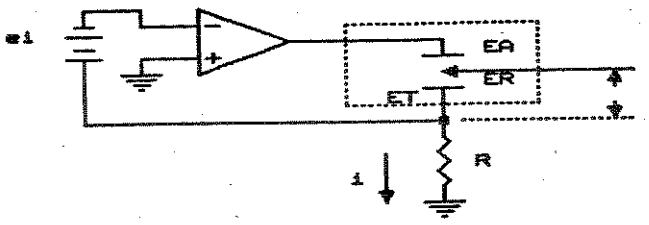


Figura 1.12: Diagrama de um simples galvanostato baseado no circuito controlador de corrente da figura 1.9b.

Com esta configuração, a corrente que atravessa a cela é igual a corrente que atravessa o resistor R ($i = e_i/R$) e é controlada pelo valor da tensão e_i.

1.4. INSTRUMENTAÇÃO DIGITAL

Apesar da natureza analógica da instrumentação eletroquímica, os computadores têm tomado lugar na aquisição e análise de dados eletroquímicos. Eles oferecem maior flexibilidade, versatilidade e sofisticação na execução e controle de experimentos, além de simplificar a operação e antecipar, prevenir ou avisar os erros do operador.

1.4.1. CONVERSÃO DIGITAL PARA ANALÓGICO (D/A)

A conversão D/A é o processo que corresponde a geração de sinais analógicos a partir de dígitos. Através dela é possível utilizar os computadores para gerar um grande número de

formas de onda e entregá-las a um determinado equipamento. Muitas destas formas de onda seriam impossíveis de serem geradas por um instrumento de natureza puramente analógica.[32]

Existem várias configurações possíveis para a construção de um conversor D/A. Um exemplo típico é mostrado na figura 1.13. Neste conversor, tipo escada R-2R invertida, a posição das chaves S estão relacionadas com os dígitos, em binário, ou seja, para um dígito *zero* a chave se encontra na posição 0 e para um dígito *um* a chave se encontra na posição 1. Desta forma a tensão de saída V_o é dada por

$$V_o = (V_r / 2^n) \times (S_{n-1} \times 2^{n-1} + S_{n-2} \times 2^{n-2} + \ldots + S_1 \times 2^1 + S_0 \times 2^0)$$

onde V_r é a tensão de referência e n é o número de bits do conversor. Assim, considerando-se n igual a 4 e V_r igual a 2 V, para um número **1001** em binário, correspondendo a nove em decimal, tem-se

$$V_o = (2 / 2^4) \times (1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0) = 0{,}125 \times 9 = 1{,}125 \text{ V}$$

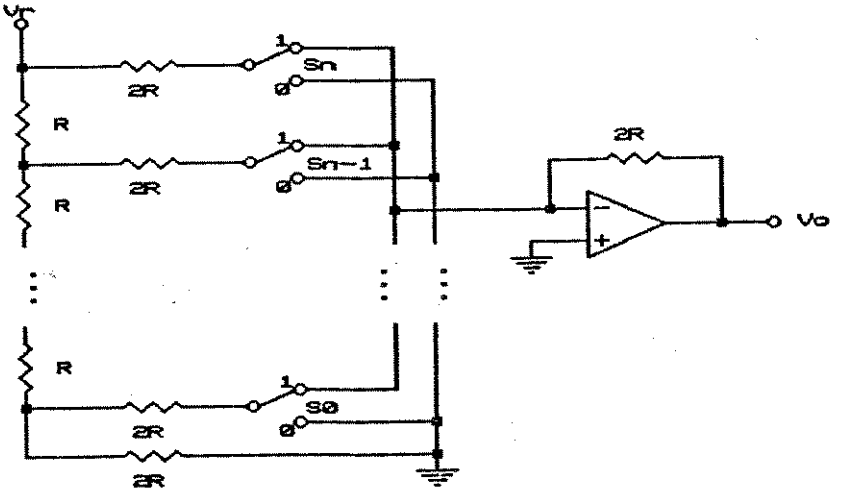


Figura 1.13: Esquema simplificado de um conversor D/A na configuração de escada R-2R invertida.

Existem vários parâmetros que servem para descrever a qualidade e desempenho de um conversor D/A. Estes parâmetros,

geralmente especificados pelos fabricantes, são:

▸ Resolução: a resolução especifica o números de bits (n) que um conversor pode acomodar e, correspondentemente, o número de tensões de saída. O número de possíveis tensões de saída é 2^n e a mínima variação possível da tensão de saída é $2^{-n} \times V_r$.

▸ Linearidade: a linearidade é a variação da tensão de saída para cada incremento numérico na entrada.

▸ Precisão: a precisão de um conversor é uma medida da diferença entre a tensão analógica obtida na saída e aquela esperada idealmente. A falta de linearidade contribui para a imprecisão do conversor.

▸ Tempo de acomodação (*settling time*): o tempo de acomodação corresponde ao intervalo compreendido entre o instante da variação na entrada até o instante em que a tensão de saída se aproxime o suficiente de seu valor final.

▸ Sensibilidade à temperatura: para qualquer entrada digital fixa, a saída analógica variará com a temperatura. Esta sensibilidade à temperatura varia tipicamente em uma faixa entre ± 50 ppm/°C em um conversor de uso geral e ± 1,5 ppm/°C em conversores de alta qualidade.[34]

Os conversores D/A são frequentemente utilizados dentro da estrutura dos conversores analógicos-digital.

1.4.2. CONVERSÃO ANALÓGICO PARA DIGITAL (A/D)

Os sinais digitais, ao menos idealmente, são representados por formas de onda que fazem transições abruptas entre dois valores. Os sinais que podem assumir qualquer valor em uma faixa contínua são chamados de sinais analógicos. Quando sinais analógicos devem ser processados, há frequentemente uma grande vantagem na conversão do sinal para a forma digital, de tal modo que o processamento possa ser feito digitalmente.

Na entrada de tal sistema de processamento, o processo global de conversão de um sinal analógico para uma forma digital envolve uma sequência de quatro processos individuais chamados de amostragem, retenção, quantização e codificação. Os

dois primeiros são feitos em um circuito amostrador-retentor (*Sample & Hold* - S&H) e a quantização e codificação são feitas simultaneamente em um circuito chamado conversor analógico-digital (A/D). Depois que o processamento digital é completado, a reconstituição do sinal analógico, se for desejado, pode ser feita através de um conversor D/A seguido de filtragem.[34]

Não é necessário ter o sinal analógico durante todo o tempo, porém somente nos instantes de amostragem e, assim, nos intervalos entre as amostras deve-se ter tempo para converter cada tensão de amostra para a forma digital. As amostras são tensões analógicas que variam continuamente. A variação permíssivel sob forma digital não é contínua, já que os valores das amostras devem diferir, no mínimo, de um dígito correspondente ao menos significativo dos dígitos usados na representação digital. Assim, o processo de converter as amostras em dígitos envolve uma aproximação. Este processo de aproximação é chamado de quantização.[34]

Da mesma maneira que existem várias configurações para os conversores D/A, existem várias para os conversores A/D. Uma delas, é apresentada na figura 1.14 de um modo simplificado. Neste conversor, tipo comparador, o conversor D/A gera uma rampa que será comparada com a entrada analógica. Quando a tensão gerada pelo conversor D/A se iguala à tensão analógica de entrada, a saída do comparador muda de nível lógico indicando o final da conversão. O valor digital correspondente à conversão é o último número enviado ao conversor D/A.

Para os conversores A/D também existem vários parâmetros que servem para descrever sua qualidade e desempenho. Estes parâmetros, geralmente especificados pelos fabricantes, são:

- Tensão analógica de entrada: esta especificação designa a máxima faixa de tensões analógicas de entrada permitidas. Valores típicos são 0 a 10 V, ±5 V, ±10 V, etc..
- Impedância de entrada: os valores variam de 1kΩ a 1MΩ, dependendo do tipo de conversor A/D. A capacitância de entrada se situa na faixa das dezenas de picofarads.
- Linearidade: a linearidade é a variação do número digital para

cada incremento na tensão de entrada.

▸ Precisão: a precisão de um conversor A/D inclui o erro de quantização, o ruído do sistema digital incluindo o que está presente na tensão de referência, desvios de linearidade, etc.. Em geral, o erro de quantização é especificado como ± 1/2 LSB (metade do bit menos significativo). Valores típicos de precisão são da ordem de 0,02% do fundo de escala para conversores de boa precisão e 0,001% para conversores de altíssima precisão.

▸ Tempo de conversão: os tempos de conversão variam de 50 μs, para unidades de velocidade moderada, a 50 ns para um dispositivo de alta velocidade.

▸ Sensibilidade à temperatura: a precisão do sistema é geralmente dependente da temperatura. Coeficientes de erro de temperatura típicos são da ordem de 20 ppm do fundo de escala por grau Celsius.

É o processo de conversão A/D que torna possível a utilização dos computadores para aquisição e tratamento de dados de experimentos, não só em eletroquímica, mas em qualquer área.

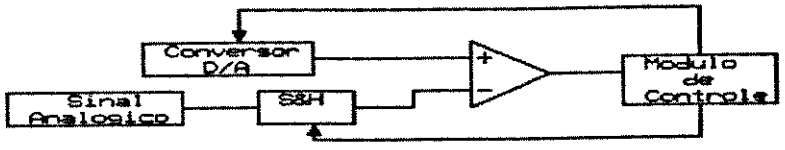


Figura 1.14: Esquema de um conversor A/D do tipo comparador com rampa simples.

CAPÍTULO 2
OBJETIVOS

O objetivo principal deste trabalho foi o desenvolvimento de um sistema informatizado para aquisição e tratamento de dados em experimentos de eletroquímica.

Para tanto, o primeiro objetivo foi o de construir uma interface (*hardware*), contendo conversores A/D e D/A, que pudesse fazer a conexão de um potenciostato/galvanostato a um microcomputador da linha IBM PC/XT.

O segundo, foi o desenvolvimento de programas (*software*) que permitissem a aquisição e tratamento de dados em experimentos de varredura linear de potencial (voltametria cíclica), salto de potencial (cronoamperometria e cronocoulometria) e salto de corrente (cronopotenciometria).

Como objetivo final, a verificação do desempenho do sistema desenvolvido na caracterização de filmes de polipirrol utilizando as técnicas eletroquímicas mencionadas.

CAPÍTULO 3
INTERFACEAMENTO

Neste capítulo serão discutidos os vários aspectos relacionados com o trabalho de interfaceamento de um Potenciostato/Galvanostato com um microcomputador para a criação de um sistema informatizado de aquisição e tratamento de dados para eletroquímica. Os diagramas dos circuitos e o *software* básico serão mostrados e discutidos, enquanto que o *software* de aquisição e tratamento de dados será assunto do próximo capítulo.

3.1. DESCRIÇÃO DO SISTEMA

O sistema é composto por: um microcomputador compatível com o IBM PC/XT com 640 Kbytes de memória RAM, *clock* de 4.77 / 8 MHz, placa gráfica CGA, uma unidade de disco flexível de 5 1/4" e um disco rígido (winchester) de 20 Mbytes; um Potenciostato/Galvanostato marca FAC modelo FAC200A; uma cela eletroquímica de três eletrodos e uma interface contendo conversores D/A e A/D. Para trabalhos envolvendo eletrocromismo inclui-se, também, um espectrofotômetro monofeixe marca Micronal modelo B280.

3.1.1. CELA ELETROQUÍMICA

A figura 3.1 mostra um diagrama esquemático de uma cela eletroquímica convencional. A cela é composta de três eletrodos: EA, ET e ER. Neste tipo de cela o potencial é sempre medido entre ET e ER, ficando a corrente elétrica obrigada a circular entre ET e EA, como será visto mais adiante.

Neste trabalho foi utilizado como EA um fio de platina com diâmetro de 1 mm e 5 cm de comprimento, como ET um fio

de platina com 1 mm de diâmetro recoberto com vidro deixando uma área efetiva de 7.85 x 10^{-3} cm^2, e um eletrodo de calomelano saturado, marca Ingold, como ER.

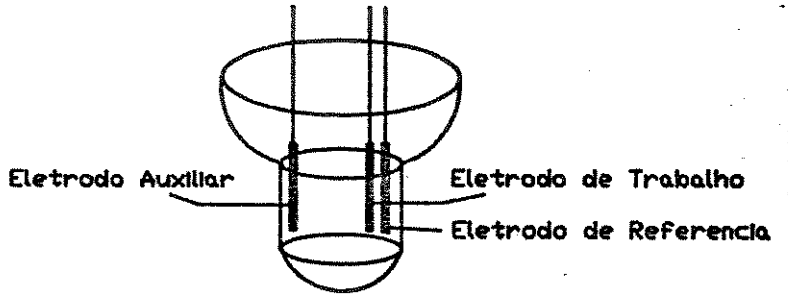
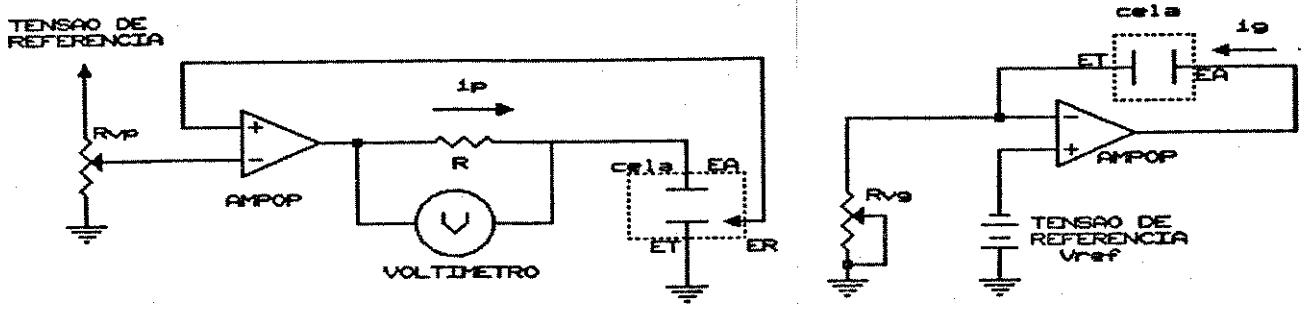


Figura 3.1: Diagrama de uma cela eletroquímica convencional.

3.1.2. POTENCIOSTATO/GALVANOSTATO

Um diagrama esquemático de um potenciostato/galvanostato pode ser visto na figura 3.2. O potenciostato/galvanostato utilizado tem capacidade para fornecer correntes de 1 μA a 1 A e potenciais de 0 a ± 2 V para a cela eletroquímica. Possui 2 saídas analógicas, uma para monitoração da corrente (modo potenciostático) e outra para monitoração do potencial (modo galvanostático), ambas com fundo de escala de ± 10 V.



(a) Potenciostato (b) Galvanostato

Figura 3.2: Diagrama esquemático de um potenciostato/galvanostato.

Uma vez que as entradas do amplificador operacional (AMPOP) têm, virtualmente, o mesmo potencial, a diferença de potencial entre ET e ER é a mesma que E, a qual pode ser controlada externamente através de R_{vp} (figura 3.2a) obtendo-se, então, um controle potenciostático. A corrente através de ER é igual à corrente de polarização da entrada inversora do AMPOP. Um bom AMPOP apresenta corrente de entrada da ordem de 10^{-12}A, de modo que não se observa os problemas de polarização do ER que antigamente atormentavam os eletroquímicos. A corrente que circula entre EA e ET pode ter seu valor facilmente determinado pela medida do potencial sobre a resistência R, conectada ao EA.

Ainda, pelo fato do potencial virtual ser o mesmo nas duas entradas do AMPOP, o controle galvanostático é conseguido pela montagem mostrada na figura 3.2b. A corrente que circula através da resistência variável R_{vg} é igual a V_{ref}/R_{vg}, que é exatamente a mesma corrente i_g que circula através da cela eletroquímica entre EA e ET. Como a cela está associada ao caminho de realimentação do AMPOP, i_g independe da resistência da cela e pode ser controlada externamente através de V_{ref} ou R_{vg} ou ambas, desde que não ultrapasse os limites de operação do AMPOP.

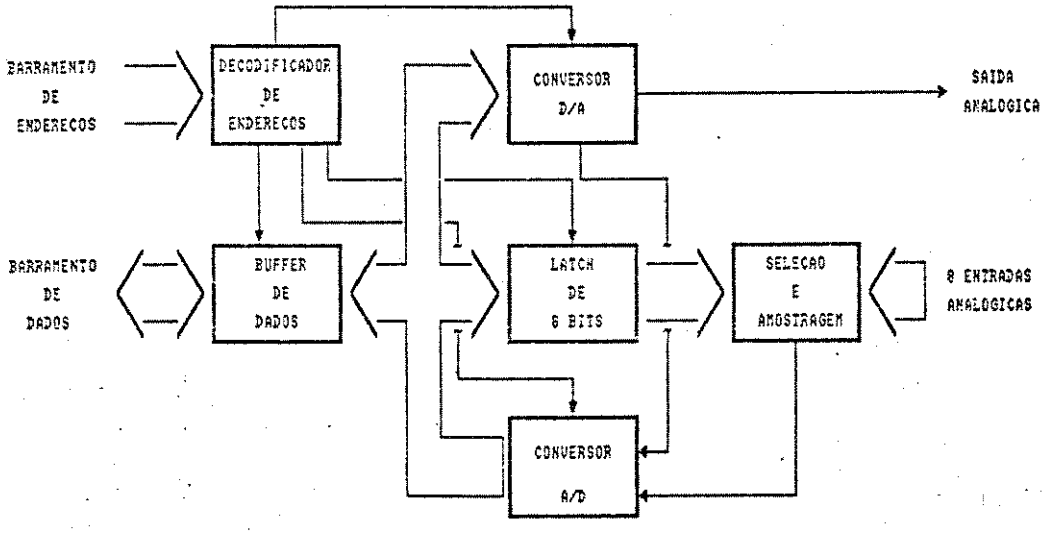


Figura 3.3: Diagrama de blocos da interface.

3.1.3. INTERFACE

A interface foi construída a partir de componentes facilmente encontrados no mercado nacional, com exceção do conversor D/A, que foi adquirido no exterior. Pode ser dividida em 4 partes por questões de didática, pois não existe separação real entre as partes. A figura 3.3 mostra um diagrama de blocos da interface. Os circuitos foram alojados em uma caixa de aluminio, com fonte de alimentação própria (± 5V e ± 12V), e em um cartão (placa) dentro do microcomputador, alimentado pela fonte do mesmo.

3.1.3.1. Entrada e Saída de Dados (E/S)

A figura 3.4 mostra o circuito elétrico do cartão interno ao microcomputador. Este circuito é responsável pela decodificação de endereços e pela proteção do barramento de dados, faz o controle de E/S e se comunica com o resto do circuito através de um cabo flexível de 25 vias (*flat-cable*). A decodificação primária é conseguida através dos CIs 74LS244, 74LS04, 74LS30 e 74LS00 que utilizam as linhas de endereço de A0 a A9 (barramento de endereços), as linhas de IOW e IOR, que indicam se o sinal é de saída ou entrada e a linha de AEN, a qual confirma se o endereço não está sendo utilizado por um processo, interno ao microcomputador, de acesso direto à memória (DMA). A segunda parte da decodificação é conseguida através do CI 74LS138, que manipula os endereços internos da interface. A proteção do barramento de dados é conseguida com o CI 74LS245.

Porta	Função
300H	Byte menos significativo do D/A
301H	Byte mais significativo do D/A
302H	Comparador do A/D
303H	Seletor, Ganho e S&H do A/D
304H	S&H do D/A (sample = 1, hold = 0)

Tabela 3.1: Endereços das Portas de E/S utilizadas pela interface.

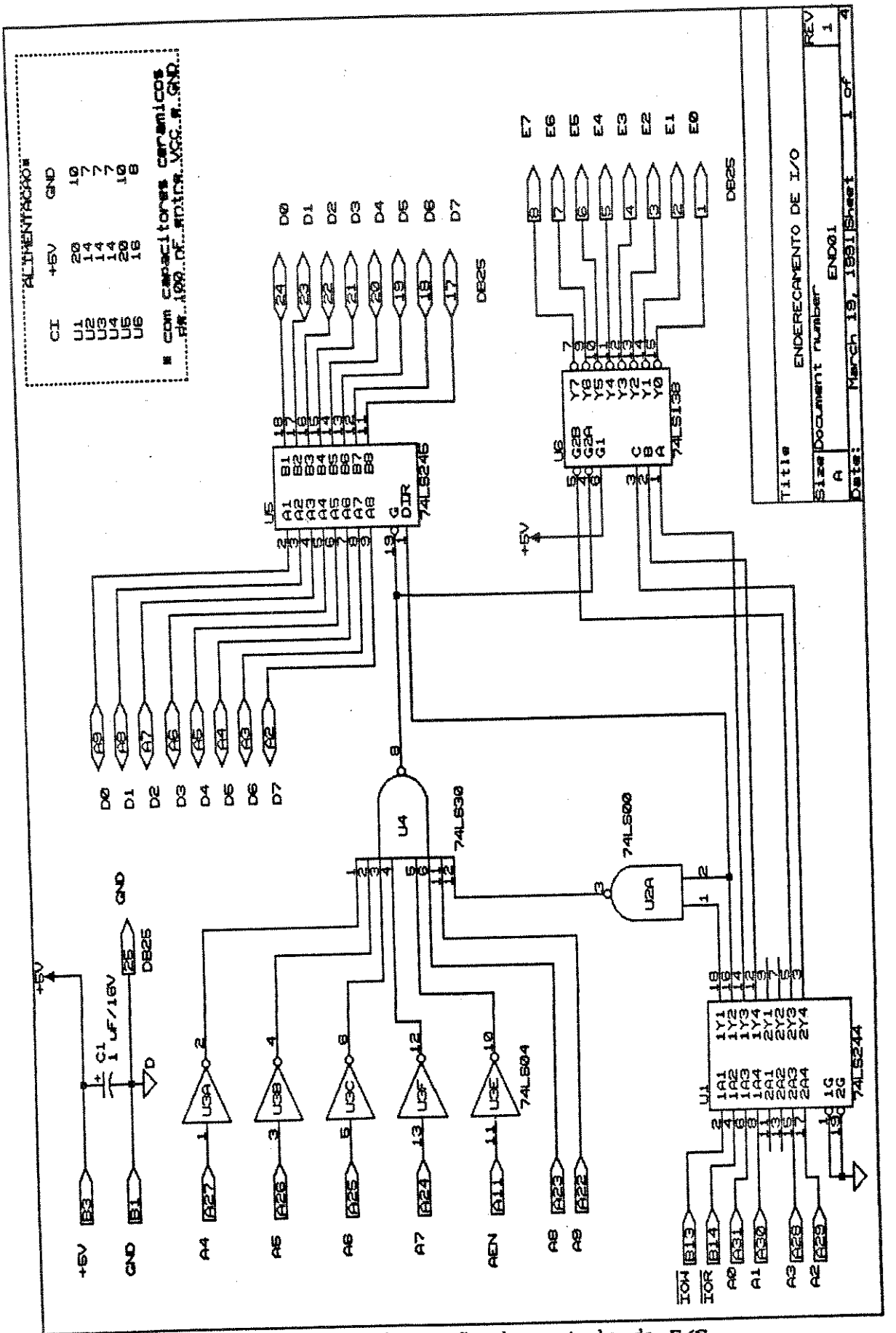


Figura 3.4: Esquema elétrico do cartão de controle de E/S.

As funções da interface são acessadas por instruções específicas, pois foram configuradas como portas de E/S. Na linguagem Pascal, na qual foram desenvolvidos todos os programas, isto é conseguido pelo uso dos vetores Port e PortW. A tabela 3.1 mostra os índices do vetor Port utilizados, os quais são iguais aos endereços de E/S, e suas respectivas funções.

3.1.3.2. Seletor de Entrada, Ganho e S&H

O circuito mostrado na figura 3.5 é responsável pela seleção de um dos oito canais de entrada da interface, pelo ganho (amplificação) dos sinais analógicos provenientes das entradas e pelo S&H do conversor A/D. Possui oito níveis de ganho, o que permite aumentar a faixa dinâmica da leitura. Tanto o canal como o ganho são selecionados por *software*.

Bit	Atuação
0	Seletor
1	Seletor
2	Seletor
3	---
4	Ganho
5	Ganho
6	Ganho
7	S&H do A/D (sample = 1, hold = 0)

Tabela 3.2: Disposição dos bits no byte referente ao endereço 309H.

Valor enviado	Ganho Aproximado
0	1
1	2
2	3
3	5
4	8
5	14
6	28
7	81

Tabela 3.3: Correspondência entre os valores enviados aos bits 4,5 e 6 do byte referente ao endereço 309H e o ganho para o sinal de entrada.

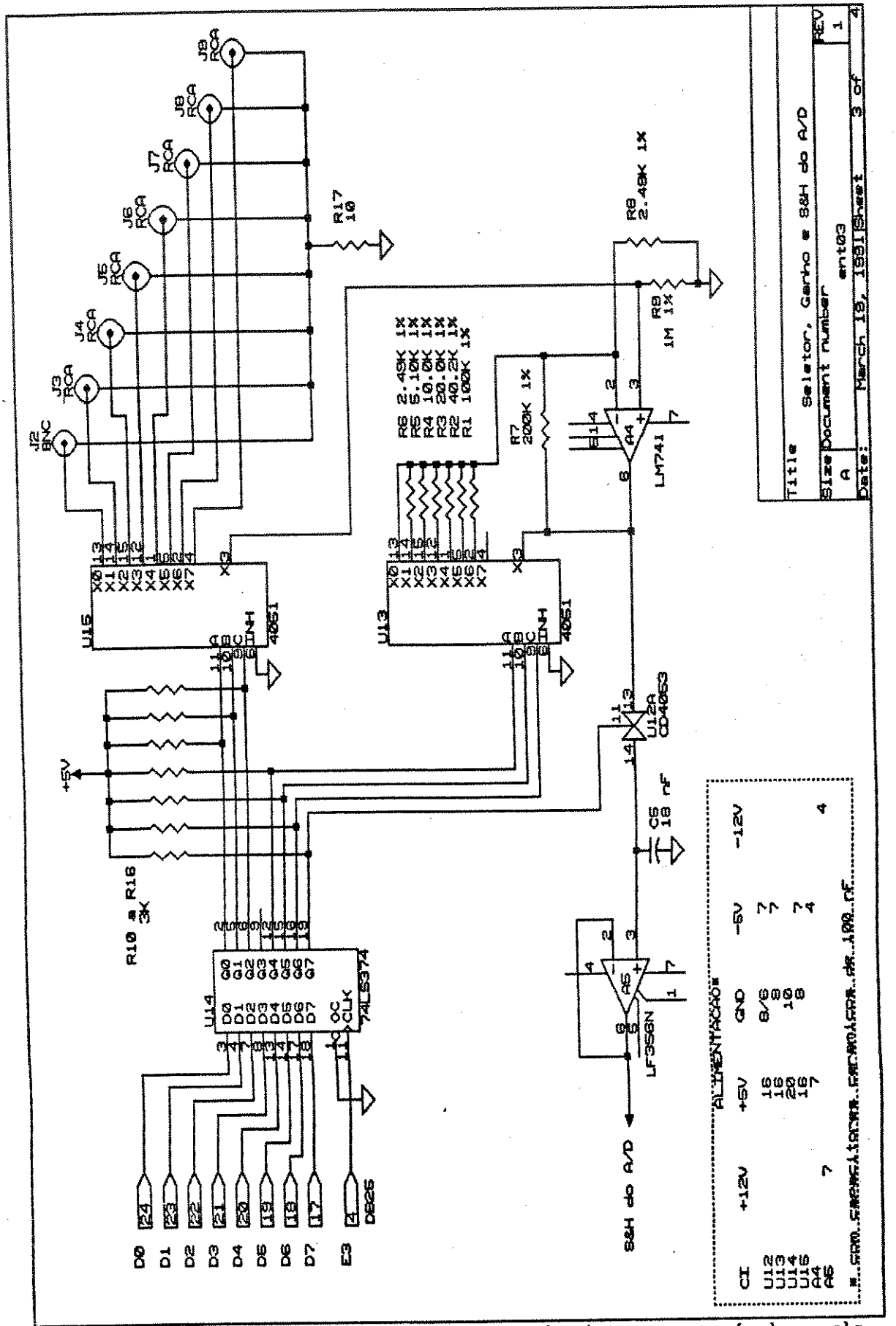


Figura 3.5: Esquema elétrico do circuito responsável pela Seleção de canais, Ganho e S&H do A/D.

O processo de S&H tem como finalidade manter fixo o valor do sinal de entrada durante a conversão. Como a seleção de canal, o ganho e o processo de S&H são feitos em um único endereço, é necessário manter um byte de memória com a imagem da última informação enviada para a porta. Este valor armazenado deve ser manipulado toda vez que se desejar acessar a referida porta, permitindo trabalhar cada bit individualmente.

A tabela 3.2 mostra a disposição dos bits e sua atuação e a tabela 3.3 mostra os valores nominais de ganho.

3.1.3.3. Conversor D/A

A figura 3.6 mostra a parte do circuito da interface correspondente aos conversores D/A e A/D. Segundo o fabricante[95], o conversor D/A (CI AD7541) possui *settling time* típico de 1 μs e linearidade de 0.01%.

Um valor inteiro entre 0 e 4095 (12 bits) é enviado ao registro do D/A (CIs U7 e U8) que o retém para ser convertido em um sinal analógico entre -2 e +2 V, respectivamente. O D/A é atualizado utilizando-se o vetor **PortW**, pois assim dois endereços consecutivos (300H e 301H) são acessados em um intervalo de tempo bem menor, além de simplificar a programação. Como pode ser visto nesta figura, existe um circuito de S&H para o D/A. O S&H é necessário para poder manter fixo o sinal desejado na saída analógica, enquanto o conversor estiver sendo utilizado para fazer a conversão A/D.

As tensões de referência para o D/A foram obtidas a partir do circuito mostrado na figura 3.7. Pelo uso destas tensões, o circuito completo do D/A possui uma faixa de -2000 a 2000 mV com resolução de 1 mV e precisão da ordem de ±3 mV.

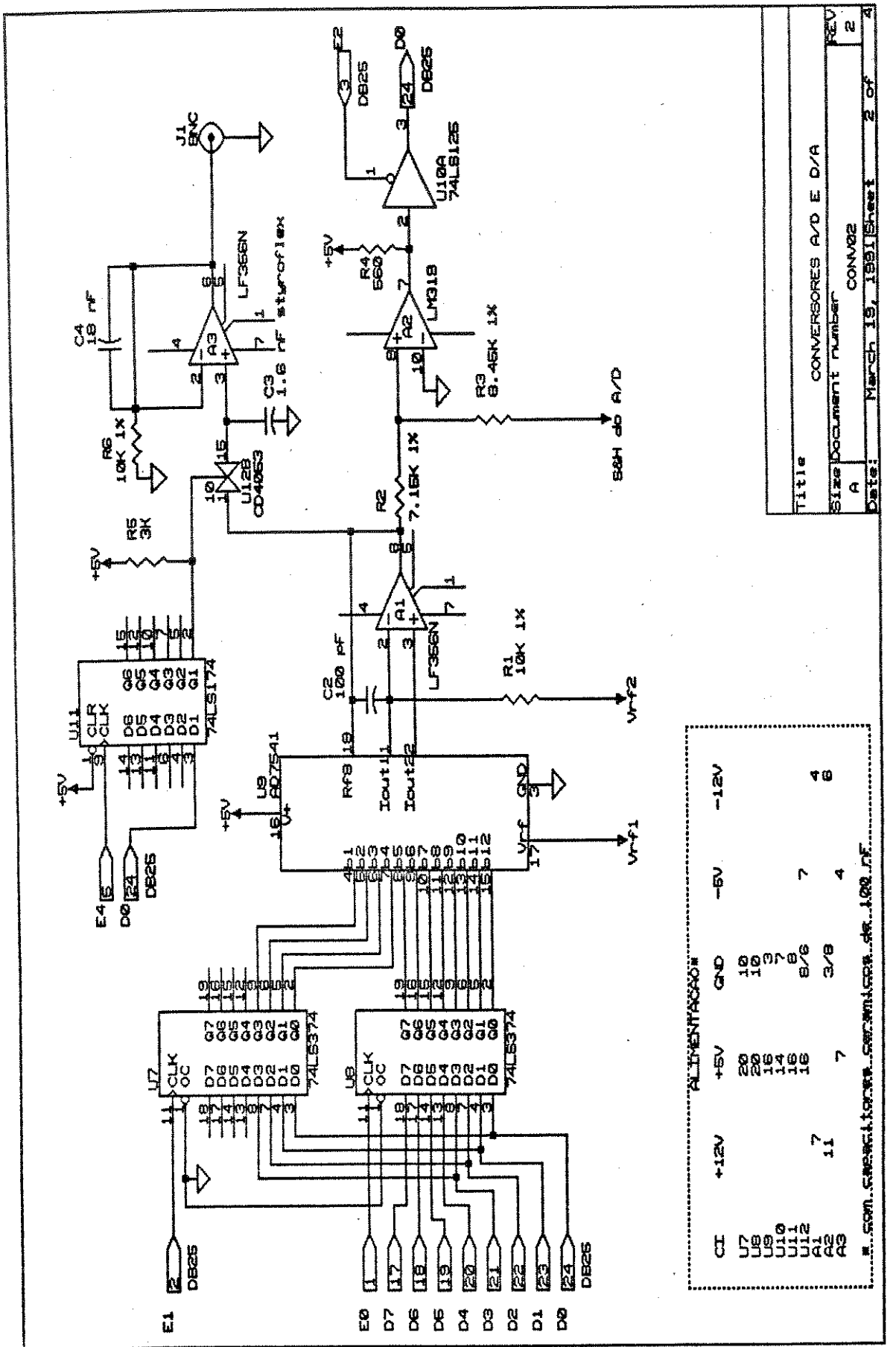


Figura 3.6: Esquema elétrico do circuito do Conversor D/A, A/D e S&H do D/A.

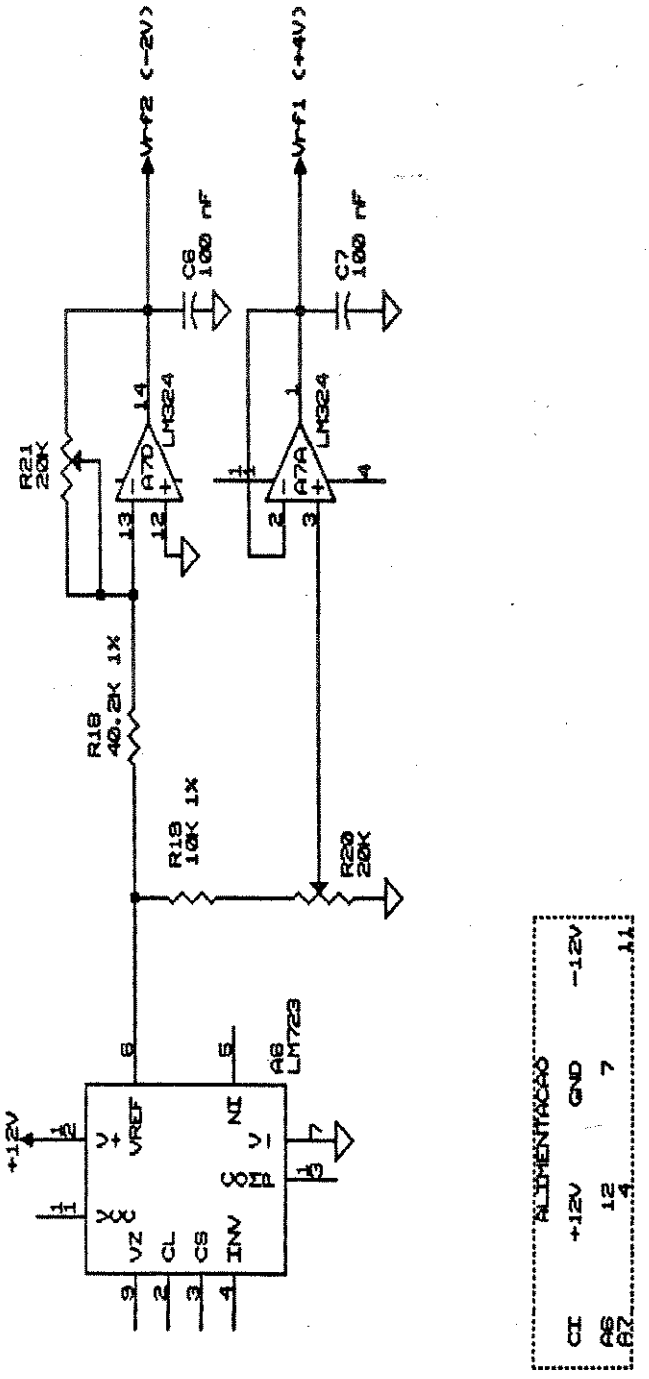


Figura 3.7: Esquema elétrico do circuito responsável pelas tensões de referência do D/A.

3.1.3.4. Conversor A/D

Devido às dificuldades em se adquirir conversores A/D com resolução maior que 8 bits, adotou-se a idéia de utilizar o conversor D/A para fazer a conversão A/D, conforme será descrito a seguir.

Um sinal analógico presente em uma das entradas é convertido em um número inteiro entre 0 e 4095 (12 bits). Tal sinal é retido no circuito de S&H do A/D para ser comparado com valores gerados pelo D/A. Inicialmente é enviado ao registro do D/A o valor 2048. Se a soma do valor gerado pelo D/A e do sinal retido no S&H do A/D é positiva, a saída do comparador A2 (figura 3.6) se encontra em nível lógico alto, indicando que o valor enviado é menor que o sinal de entrada e, portanto, deve ser aceito e armazenado em uma variável. Caso contrário, a saída de A2 se encontra em nível lógico baixo, indicando que o valor não deve ser aceito. Na sequência, o próximo valor enviado é 1024 acrescido do valor armazenado na variável. A conversão é finalizada quando todos os 12 valores (2048, 1024, ... , 2, 1) tiverem sido enviados, sendo que o valor da conversão se encontra armazenado na variável citada.

Este tipo de conversão é conhecido como conversão por aproximações sucessivas ou busca binária.[34]

SINAL DE ENTRADA (mV)	TEMPO DE CONVERSÃO (ms)	
	clock 4.77MHz	*clock* 8MHz
-2000	1.56 ± 0.02	0.93 ± 0.02
-1500	1.55 ± 0.02	0.93 ± 0.02
-1000	1.57 ± 0.02	0.94 ± 0.02
- 500	1.57 ± 0.02	0.94 ± 0.02
0	1.58 ± 0.02	0.94 ± 0.02
+ 500	1.58 ± 0.02	0.95 ± 0.02
+1000	1.58 ± 0.02	0.96 ± 0.02
+1500	1.56 ± 0.02	0.96 ± 0.02
+2000	1.58 ± 0.02	0.95 ± 0.02

Tabela 3.4: Dependência do tempo de conversão A/D com o clock do micromputador.

Como a conversão é feita pelo *software*, o tempo de conversão depende da velocidade de processamento das informações contidas no *software*, ou seja, da linguagem na qual foi desenvolvido, e do *clock* do microcomputador utilizado. Para verificar a dependência com o *clock*, já que todos os programas

foram desenvolvidos em uma mesma linguagem, fêz-se medidas de tempo de conversão para vários valores do sinal de entrada contra os dois valores possíveis de *clock* do computador (4.77 e 8 MHz - que são selecionados através de seu teclado). A tabela 3.4 mostra os valores médios obtidos de uma amostragem de 5000 conversões. Tal amostragem foi necessária para que se pudesse fazer uma medida afetada por um erro pequeno, uma vez que o relógio interno do microcomputador é atualizado somente a cada 55 milissegundos, aproximadamente.[36]

Esta informação a respeito do tempo de conversão é muito útil para se ter idéia da possibilidade de utilizar este sistema em técnicas eletroquímicas tal como cronoamperometria, onde, quanto mais rápida for a resposta, melhores serão os resultados obtidos. Contudo, é importante lembrar que o tempo de conversão é aproximadamente um terço do tempo de aquisição, pois uma aquisição completa envolve a conversão, algumas operações matemáticas e o registro do dado na tela.

O circuito completo do conversor A/D permite cobrir a faixa de potenciais de -2000 a +2000 mV com uma precisão de ±10 mV (9 bits). Já era esperado que a precisão do A/D fosse menor que a do D/A uma vez que este último é o responsável pela verdadeira conversão A/D e as imprecisões vão se acumulando pelos circuitos. Segundo a literatura[27] este valor de precisão é aceitável.

3.1.4. ESPECTROFOTÔMETRO:

Para fazer uso da interface desenvolvida em experimentos envolvendo eletrocromismo, houve a necessidade da utilização de um espectrofotômetro com saída analógica. Como o espectrofotômetro disponível não possuia tal saída, foi montado o circuito mostrado na figura 3.8.

Este circuito foi acoplado após o estágio de amplificação do detetor do espectrofotômetro, permitindo obter na saída um sinal entre 0 e 1V, correspondente a 0 e 100% de transmitância, respectivamente.

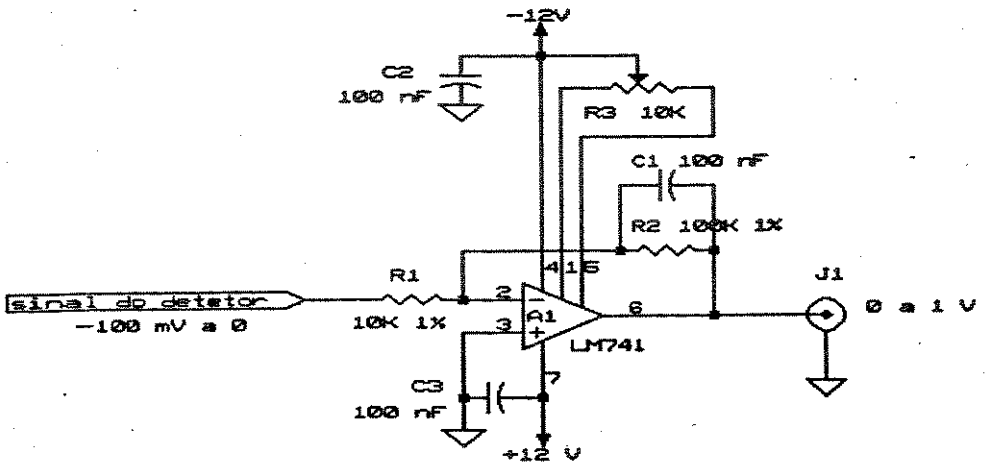


Figura 3.8: Esquema elétrico do circuito desenvolvido para a saída analógica do Espectrofotômetro Micronal B280.

3.2. SOFTWARE BÁSICO

Com o objetivo de verificar o desempenho do equipamento, bem como para a sua calibração foi desenvolvido o programa CALIBRA (vide apêndice 1). Com este programa envia-se valores inteiros entre 0 e 4095 ao conversor D/A e, com o auxílio de um multímetro, com pelo menos 3 $^{1}/_{2}$ dígitos, faz-se a leitura do potencial na saída analógica da interface. Estabelece-se, então, uma relação direta entre os valores inteiros e o potencial. Uma vez calibrado o D/A, o programa é utilizado para fazer a calibração automática do A/D, conectando-se a saída do D/A à entrada do A/D. O programa envia sinais ao D/A e faz a leitura do A/D, estabelecendo-se, assim, outra relação entre o potencial e os valores inteiros do A/D. Estas duas relações estabelecidas são inseridas em todos os programas para aquisição de dados.

3.3. PROBLEMAS ENCONTRADOS E SOLUÇÕES ADOTADAS

Devido ao caráter de protótipo do equipamento, muitos problemas surgiram durante a sua construção e durante o

desenvolvimento do *software*. A seguir serão mostrados os problemas e as medidas adotadas para solucioná-los.

3.3.1. PROBLEMAS COM O *HARDWARE*

Entre o vários problemas enfrentados, os mais relevantes foram a estabilização dos sinais analógicos de entrada e de saída.

Para estabilizar o sinal analógico de saída do conversor D/A foi necessário a inclusão de um capacitor (C4) em paralelo ao caminho de realimentação do AMPOP (A3) na figura 3.6. O valor de C4 é crítico, pois valores maiores que o escolhido comprometem o tempo de acomodação do sinal.

Com respeito a estabilização do sinal de entrada, houve a necessidade de reduzir um pouco a impedância de entrada pela introdução de um resistor (R9) na entrada do AMPOP responsável pela amplificação do sinal (A4) e um resistor (R17) para o terra comum às oito entradas. Houve também a necessidade de aumentar o valor do capacitor de S&H (C5) de 1,6 nF para 18 nF. Neste caso, também, o valor de C5 é crítico, pois o AMPOP do S&H (A5) não suporta capacitâncias elevadas em sua entrada.

3.3.2. PROBLEMAS COM O *SOFTWARE*

Os problemas mais importantes com o *software* foram a velocidade de processamento e "estouro" de variáveis inteiras, sendo que o segundo problema foi originado da solução do primeiro.

Para que fosse possível operar em velocidades de processamento mais rápidas, foi necessário trabalhar com variáveis inteiras. No turbo Pascal, uma variável real ocupa seis bytes de memória e comporta números de $1x10^{-38}$ a $1x10^{38}$, enquanto que uma variável inteira ocupa somente dois bytes e comporta números de -32768 a 32767. A manipulação de uma variável inteira chega a ser vinte vezes mais rápida que a de uma variável real[37], de modo que compensa qualquer trabalho extra de programação para a utilização de variáveis inteiras.

Na aquisição de dados, durante o intervalo entre cada aquisição são feitas várias leituras do mesmo sinal de modo que o sinal resultante é uma média aritmética das leituras. Desta maneira obtém-se um sinal "mais limpo". Como foi necessário utilizar variáveis inteiras, durante a somatória, ocorria o "estouro" das mesmas. Para solucionar o problema somou-se ao valor do sinal um número inteiro positivo e dividiu-se a leitura em duas partes utilizando-se duas funções, **Lo** e **Hi**, existentes no turbo pascal. Assim, o número de leituras pôde ser ampliado, embora tenha sido necessário restringir o máximo em 125 leituras para se obter a média, sendo o excedente ignorado. No final de cada aquisição o valor do sinal é recomposto pela soma das duas partes e pela subtração do número inteiro positivo anteriormente adicionado.

Esta metodologia pode ser vista no programa FAC no *procedure* CalcCont, onde é somado a constante TC (igual a 2300), e nos *procedures* MeioCiclo, Ciclos, VariosPulsos e Pulso, onde aparecem as variáveis ValorL e valorH e a subtração de TC.

Com estas medidas foi possível adquirir um dado numa faixa de tempo de 1 a 3 ms.

CAPÍTULO 4
SOFTWARE

Neste capítulo serão discutidos os aspectos relacionados com o *software* para aquisição e tratamento de dados para as técnicas eletroquímicas de varredura linear de potencial, salto de potencial e salto de corrente.

Os programas foram desenvolvidos em Turbo Pascal versão 3.01A da Borland Inc. e suas listagens encontram-se no apêndice 1. O objetivo de tais programas foi proporcionar ao usuário um melhor aproveitamento das técnicas eletroquímicas.

4.1. SOFTWARE DE AQUISIÇÃO

Foram desenvolvidos dois programas completos para aquisição de dados. Um, denominado FAC, destinado exclusivamente para aquisição de dados para as técnicas eletroquímicas mencionadas acima. Outro, denominado MULTICAN, para uso geral, permite a aquisição de dados de até três canais da interface sequencialmente, possibilitando a utilização da interface em substituição aos registradores XY e XT.

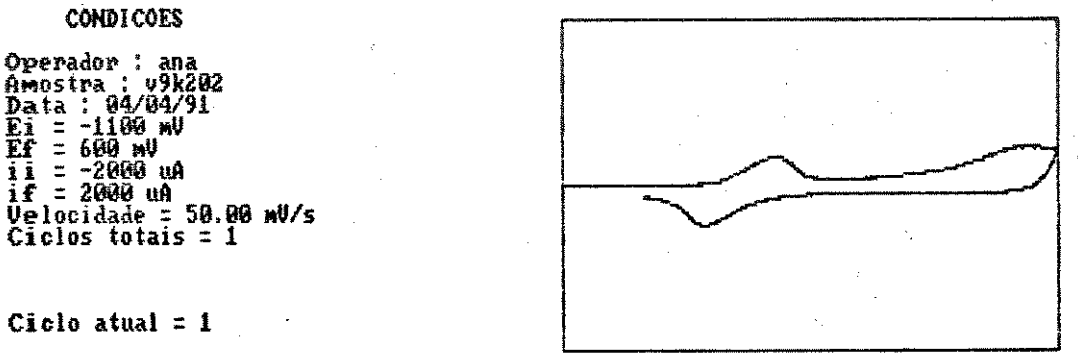
4.1.1. O PROGRAMA FAC

Programa para aquisição, tratamento preliminar de dados e impressão de gráficos. Permite fazer uso de técnicas eletroquímicas de varredura linear de potencial (voltametria cíclica), salto de potencial (cronoamperometria e cronocoulometria) e salto de corrente (cronopotenciometria).

Pelo uso do conversor D/A envia-se sinais ao potenciostato/galvanostato que se incumbe de manter a cela eletroqúimica sob controle. Estes sinais são de dois tipos: onda

triangular (voltametria) e onda quadrada (cronoamperometria, croncoulometria e cronopotenciometria).

a



b

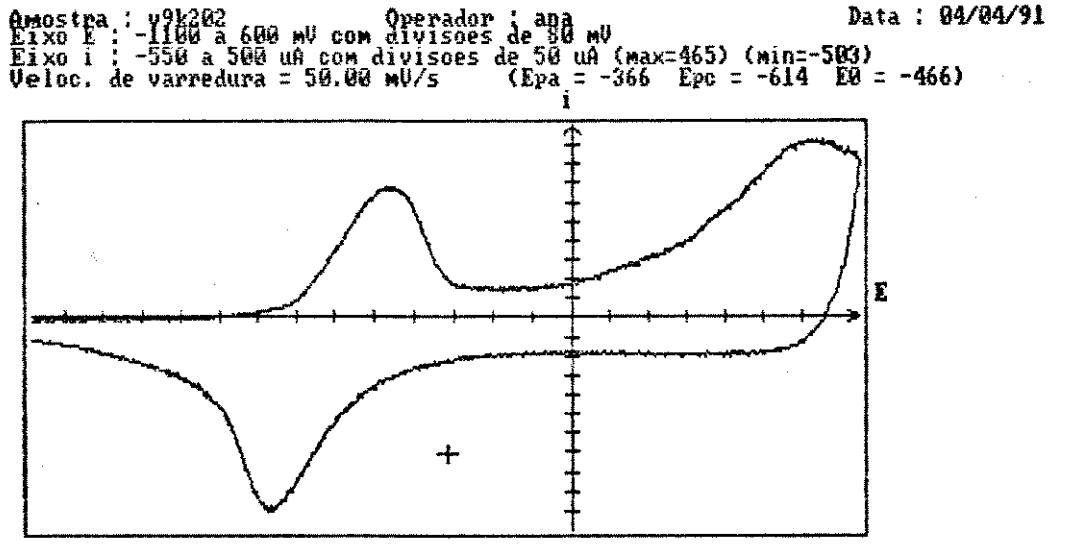


Figura 4.1: Voltamograma cíclico (-1100 a 600 mV a 50 mV/s em solução aquosa de KCl 0,5 M) de um filme de PPi sobre um eletrodo de vidro recoberto com ITO: a) tela do microcomputador durante a aquisição e b) gráfico de i versus V após a manipulação dos dados.

4.1.1.1. Voltametria Cíclica

A varredura linear do potencial é conseguida pelo incremento da tensão na saída do conversor D/A em intervalos de tempo determinados pela velocidade de varredura.

Os limites da varredura podem ser escolhidos dentro da faixa de -2000 a 2000 mV. Entretanto, as velocidades de varredura permitidas dependem dos limites escolhidos, podendo ir de 1 a 4000 mV/s com incrementos unitários.

O sinal proveniente do potenciostato/galvanostato, como resposta da cela eletroquímica, é lido no canal zero da interface através conversor A/D. Este sinal depois de processado é registrado na tela em função do potencial aplicado.

A figura 4.1 mostra um voltamograma cíclico de um filme de PPi sobre um eletrodo de vidro recoberto com óxido de estanho dopado com índio (ITO)[38], sendo que na figura 4.1a é mostrado a tela do microcomputador durante a aquisição e na figura 4.1b o gráfico i *vs* V resultante desta aquisição.

Após a aquisição, parâmetros como E_{pa}, E_{pc} e E^{o} podem ser determinados pela utilização de um cursor (+) que caminha sobre o gráfico, conforme é mostrado na figura 4.1b.

O programa permite também a integração da corrente, possibilitando a obtenção do gráfico da carga em função do potencial, conforme é mostrado na figura 4.2. Esta figura representa a carga anódica do voltamograma da figura 4.1b uma vez que foi integrado nos limites de -1100 a 600 mV

Todos estes parâmetros são prontamente utilizados para a determinação de certas propriedades do material em estudo, como será visto no próximo capítulo.

4.1.1.2. Cronoamperometria

O sinal em forma de onda quadrada é conseguido pela variação brusca (poucos microssegundos) de um potencial inicial à um potencial final, onde em cada um destes potenciais exista somente uma espécie, oxidada ou reduzida, em solução. A resposta

do sistema eletroquímico (variação da corrente) à este estímulo é registrado contra o tempo.

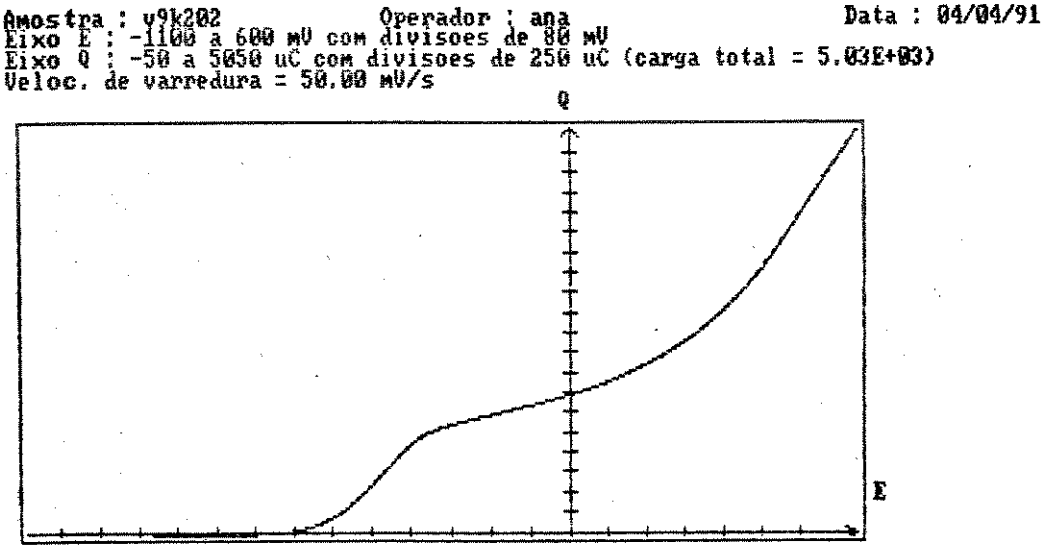


Figura 4.2: Gráfico da carga anódica em função do potencial obtido pela integração da corrente do voltamograma da figura 4.1b nos limites de -1100 a 600 mV.

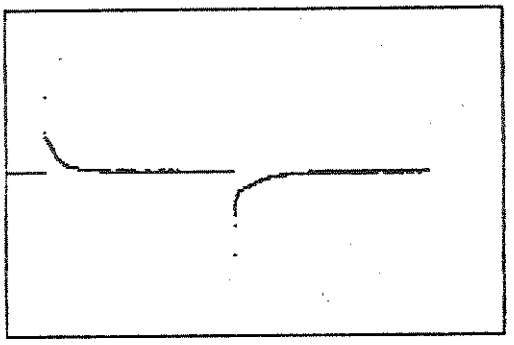
Na figura 4.3 é mostrado um experimento de cronoamperometria com o mesmo filme de PPi utilizado no experimento de voltametria cíclica, mostrado no ítem anterior. O salto de potencial foi realizado de -900 a 100 mV, com uma duração de 5 s, retornando a -900 mV, com uma duração de 7 s. A figura 4.3a mostra a tela do microcomputador durante a aquisição e a figura 4.3b o gráfico de i *vs* t resultante desta aquisição.

No caso da figura 4.3 foi realizado somente um salto de potencial, porém é possível realizar até dez mil saltos consecutivos. A realização de vários saltos consecutivos permite avaliar a estabilidade do material em estudo.

a

CONDICOES

Operador : ana
Amostra : c9k201
Data : 04/04/91
Ei = -900 mV
Ef = 100 mV
ii = -20 mA
if = 20 mA
Tempo antes do pulso = 1 s
Tempo do pulso = 5 s
Tempo apos o pulso = 7 s



b

Amostra : c9k201 Operador : ana Data : 04/04/91
Eixo t : 0 a 13000 ms com divisoes de 650 ms
Eixo i : -12.00 a 12.00 mA com divisoes de 1.20 mA (max=9.33) (min=-10.29)
Tempo de aquisicao = 13 s (E inicial = -900mV) (E final = 100mV)

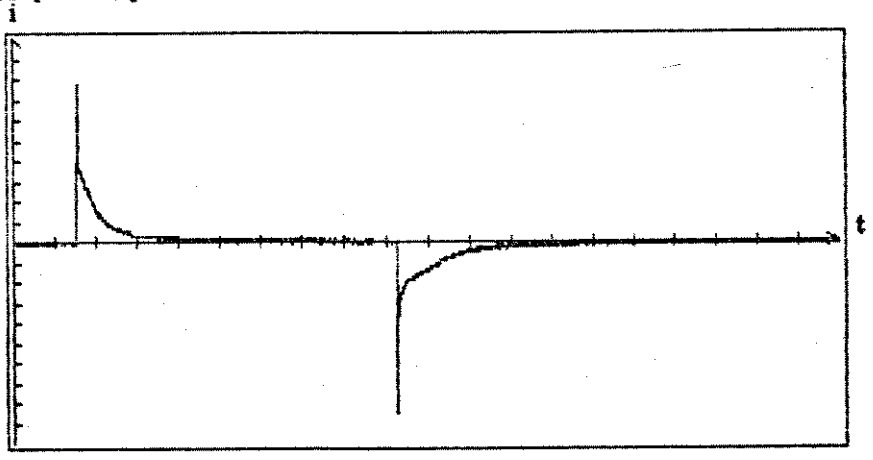


Figura 4.3: Experimento de cronoamperometria com um filme de PPi sobre um eletrodo de vidro recoberto com ITO em solução aquosa de KCl 0,5 M: a) tela do microcomputador durante a aquisição e b) gráfico de i versus t após manipulação dos dados.

4.1.1.3. Cronocoulometria

Após a aquisição do cronoamperograma o programa permite a obtencão da carga pela integração da corrente em função do tempo. Assim o gráfico Q *vs* t pode ser registrado, fornecendo informações a respeito do comportamento do material em estudo.

A figura 4.4 mostra o gráfico da carga em função do tempo obtido pela integração do cronoamperograma mostrado na figura 4.3b.

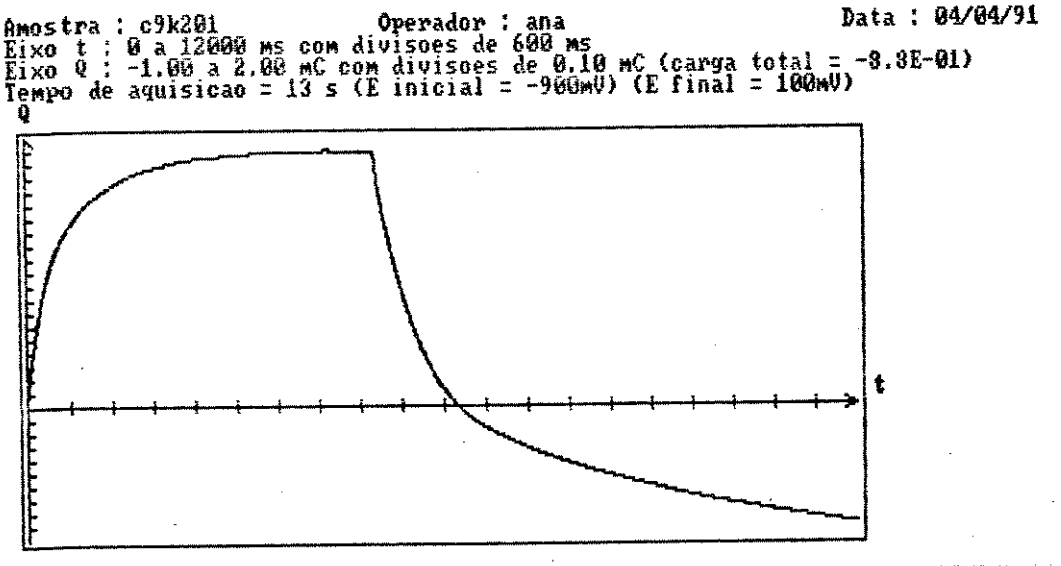


Figura 4.4: Cronocoulograma obtido pela integração do cronoamperograma da figura 4.3b.

4.1.1.4. Cronopotenciometria

Nesta técnica, o sinal de excitação é uma onda quadrada como na técnica de cronoamperometria. Entretanto, difere no fato de que o potenciostato/galvanostato opera no modo galvanostato. Assim, um potencial aplicado em sua entrada é diretamente proporcional à corrente que será aplicada e controlada na cela eletroquímica.

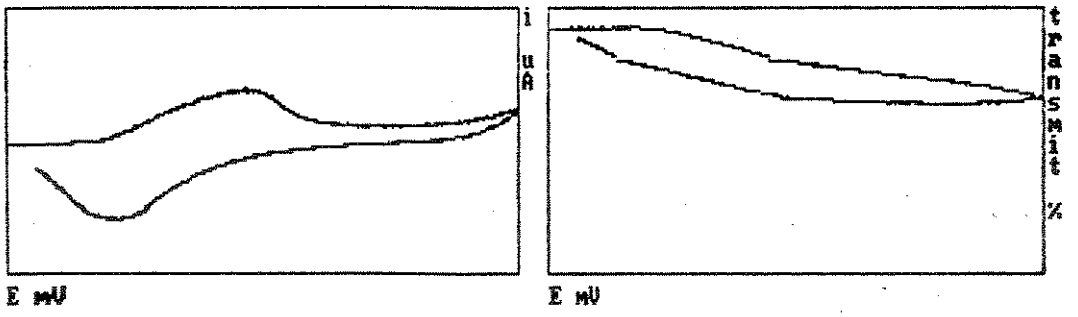
Embora o programa permita a execução de experimentos de cronopotenciometria, esta técnica não foi explorada neste trabalho.

4.1.2. O PROGRAMA MULTICAN

Este programa realiza a leitura de até três canais da interface sequencialmente. Com ele é possível obter várias informações como, por exemplo, corrente e absorbância em função do potencial ou em função do tempo; corrente, absorbância e temperatura em funcão do tempo, etc.. Enfim, o programa pode ser utilizado para medir qualquer grandeza, bastando para isso, que o equipamento, responsável pela medida da grandeza, possua uma saída analógica que possa ser conectada à interface.

As aplicações do programa dependem da necessidade de cada usuário e tudo se passa como se o microcomputador fosse um registrador XY ou XT, onde os dados são registrados. A vantagem está no fato de que não é necessário fazer qualquer ajuste, além de ser muito mais sensível que um registrador convencional.

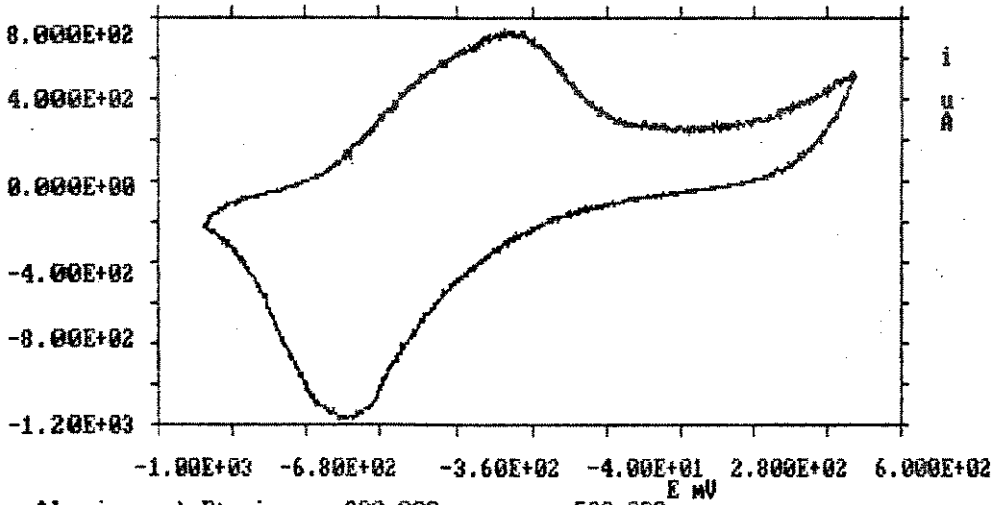
Uma das aplicações mais imediatas do programa foi a aquisição simultânea de dados de corrente e transmitância em um experimento de eletrocromismo com um filme de PPi/DBS sobre um eletrodo de vidro recoberto com ITO[39], conforme é mostrado na figura 4.5. A figura 4.5a mostra a tela do microcomputador durante a aquisição de dados, a figura 4.5b mostra o gráfico da corrente em função do potencial, correspondente ao sinal lido no canal 0 da interface, e a figura 4.5c mostra o gráfico da transmitância em função do potencial, correspondente ao sinal lido no canal 1.



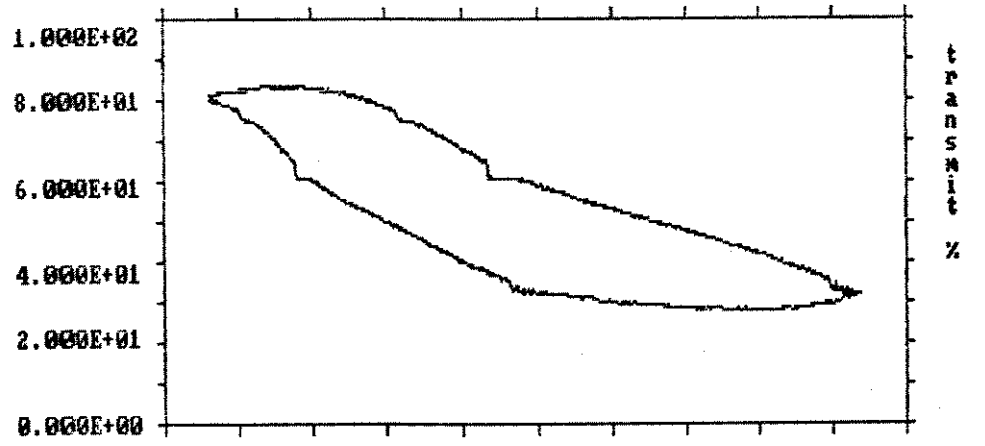
...CONDICOES...

Tempo total de aquisicao -> 5755 s
Intervalo de aquisicao -> 0.360 s
Operador : Rosa
Titulo : elecrvcli7
Data : 07/05/91
Ciclos totais = 10
Ciclo atual = 1

Canal -> 0 ... Amplificacao -> 0
Canal -> 1 ... Amplificacao -> 0
Ei = -900 mV
Ef = 500 mV
ii = -2000 uA
if = 2000 uA
Velocidade = 50 mV/s



Abscissa -> E: min. = -900.000 max. = 500.000
Ordenada -> i: min. = -1178.000 max. = 744.000



Abscissa -> E: min. = -900.000 max. = 500.000
Ordenada -> transmit: min. = 27.800 max. = 86.500

Figura 4.5: Experimento de eletrocromismo com um filme de PPi/DBS sobre um eletrodo de vidro recoberto com ITO: a) tela do microcomputador durante a aquisição de dados (acima), b) gráfico da corrente versus potencial (centro) e c) gráfico da transmitância (em 700 nm) versus potencial (abaixo).

De posse de vários experimentos armazenados em disquete, é possivel fazer a superposição de curvas, o que proporciona ao usuário uma visualização das mudanças ocorridas com o material em estudo em virtude de alguma modificação nas condições dos ensaios.

O programa possui também um menu para tratamento matemático, onde é possível aplicar algumas funções (raiz quadrada, logarítmo, exponencial, etc.) aos dados adquiridos. Estas funções podem ser aplicadas sequencialmente gerando outras funções, ou seja, pode-se aplicar a função exponencial e, em seguida, a raiz quadrada e assim por diante. Desta maneira, caso seja de interesse, é possível obter uma grande variedade de gráficos derivados dos dados originais.

4.2. SOFTWARE DE TRATAMENTO DE DADOS

Foram desenvolvidos dois programas completos. Um, denominado VOLTAM, para tratamento de dados para a técnica de voltametria cíclica e outro, denominado CRONOAM, para tratamento de dados para a técnica de cronoamperometria (incluindo cronocoulometria). A partir destes programas é possível a obtenção dos parâmetros necessários para caracterização das reações eletroquímicas.

4.2.1. O PROGRAMA VOLTAM

Após os dados terem sido adquiridos pelo programa FAC, estes podem ser analisados pelo programa VOLTAM, que permite a determinação da corrente e potencial de pico anódico e catódico, determinação do potencial redox e a determinação da carga anódica e catódica através da integração da curva. Estes parâmetros podem ser vistos na lateral direita da figura 4.6.

Duas linhas, uma vertical e outra horizontal, atuam como marcadoras de coordenadas sobre o gráfico. Através delas é possível determinar linhas-base para os cálculos das correntes de pico e cargas anódica e catódica. A utilização da linha-base

nestes cálculos é muito importante, pois assim é possível descontar o efeito da corrente de fundo (corrente capacitiva) que desloca os valores de máximo e mínimo nas correntes de pico e que afeta o cálculo das cargas.

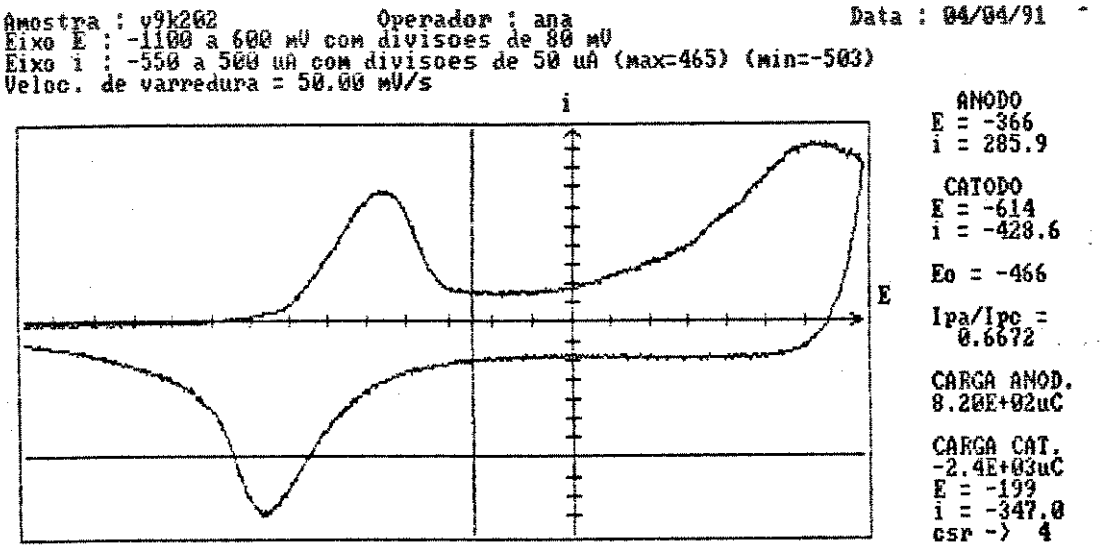


Figura 4.6: Determinação dos parâmetros eletroquímicos através do programa VOLTAM. Os parâmetros podem ser vistos na lateral direita da figura.

O programa, também, permite a superposição de voltamogramas. Isto é muito útil quando se deseja monitorar o comportamento do material em estudo, ou seja, a sua eletroatividade, ou quando se altera alguma condição do ensaio, como por exemplo, velocidade de varredura, concentração do eletrólito, distância entre os eletrodos, etc., como será visto no capítulo 5.

O programa permite, ainda, fazer a relação entre a velocidade de varredura e as correntes de pico anódico e catódico.

Os dados obtidos são gravados em disquete e, posteriormente, podem ser graficados por um outro programa específico para traçado de gráficos. Esta relação fornece informações a respeito da cinética da reação, conforme comentado no item 1.3.1.1. No capítulo 5 este estudo será utilizado para caracterização de filmes de PPi com várias espessuras e para o estudo da influência da velocidade de varredura no aspecto do voltamograma.

4.2.2. O PROGRAMA CRONOAM

O programa CRONOAM foi desenvolvido especificamente para tratamento de dados para a técnica de cronoamperometria e cronocoulometria.

De maneira semelhante ao programa VOLTAM, duas linhas, uma vertical e outra horizontal, são utilizadas como marcadoras de coordenadas para o cálculo das cargas anódica e catódica, conforme pode ser visto na figura 4.7.

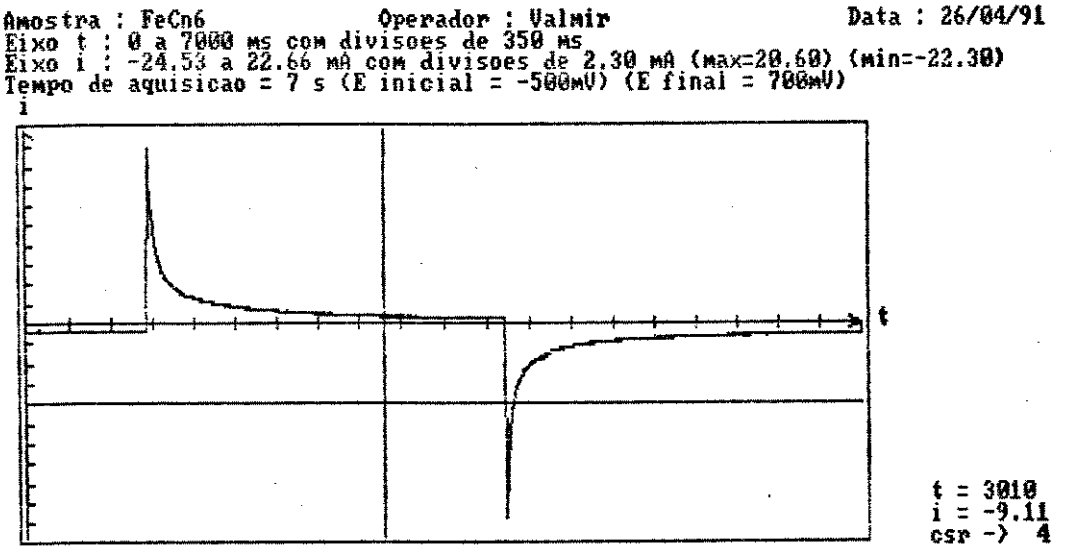


Figura 4.7: Gráfico da corrente em função do tempo obtido de um ensaio de cronoamperometria com uma solução aquosa 0,05 M de K_3FeCN_6.

O programa permite a obtenção de gráficos da carga em função do tempo e em função da raiz quadrada do tempo. A partir deste último pode-se determinar o coeficiente de difusão através da inclinação da reta (ver ítem 1.3.1.3). Na figura 4.8 é mostrado um gráfico da cargas anódica (acima do eixo) e catódica (abaixo do eixo) em função de $t^{1/2}$, obtido a partir da integração da curva da figura 4.7.

O programa também permite a superposição de curvas de corrente e carga em função do tempo. Isto é útil quando o interesse está voltado ao estudo da estabilidade do material.

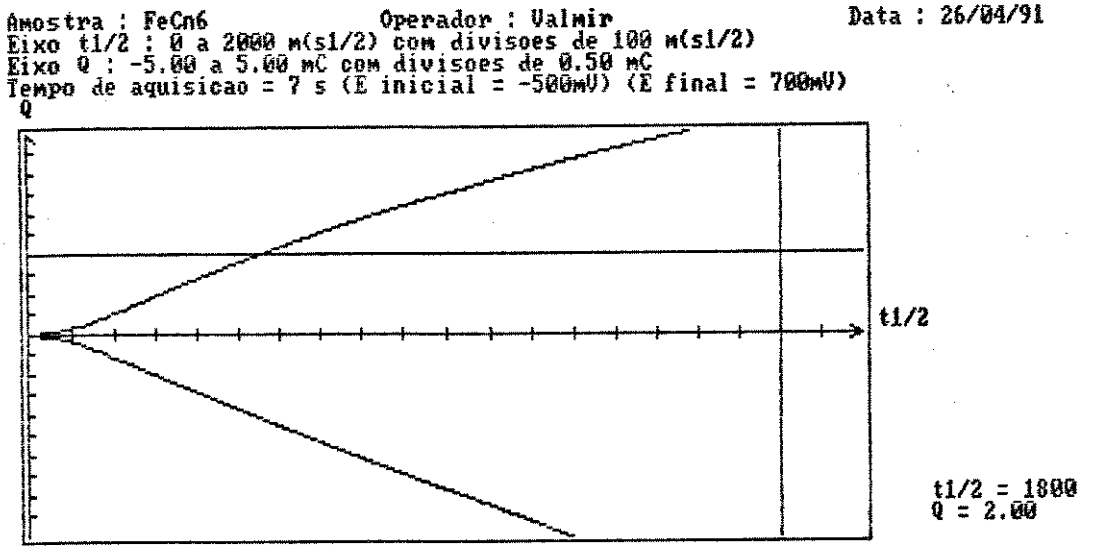


Figura 4.8: Gráfico da carga em função da raiz quadrada do tempo obtido, através do programa **CRONOAM**, a partir da curva da figura 4.7.

CAPÍTULO 5
APLICAÇÃO DO EQUIPAMENTO AO SISTEMA PPi/DS

Neste capítulo serão vistos os aspectos relacionados com a aplicação do equipamento desenvolvido para a caracterização de filmes de PPi/DS.

Com o objetivo de testar os programas bem como para verificar o desempenho do equipamento, foram feitos ensaios com soluções de $FeCl_3$, $K_3Fe(CN)_6$ e $Fe(C_5H_5)_2$. Os dados foram comparados com os valores fornecidos pela literatura e encontram-se tabelados no apêndice 2.

5.1. SÍNTESE DO PPi/DS

O PPi/DS foi sintetizado eletroquimicamente em modo galvanostático com uma corrente de 44 μA por um tempo de 68 s a partir de uma solução aquosa (água bidestilada) 0,05 M de pirrol recém destilado e 0,05 M de dodecilsulfato de sódio (SDS, recristalizado duas vezes). Nestas condições a densidade de corrente é 5,6 mA/cm^2 e a densidade de carga é 0,38 mC/cm^2. Os eletrodos utilizados foram descritos no ítem 3.1.1.

5.2. ELETROATIVIDADE EM FUNÇÃO DO ELETRÓLITO

Preparou-se 3 filmes de PPi/DS conforme descrito no ítem 5.1. Após a preparação de cada filme fez-se a varredura de potencial de -900 a 100 mV com velocidade de 25 mV/s na própria solução de síntese, a fim de se verificar a eletroatividade do filme recém formado.

Não existe problema algum em se fazer a varredura potencial na própria solução de síntese, pois a polimerização do pirrol somente se inicia em potenciais superiores a 600 mV.[22]

Em seguida elevou-se o potencial lenta e manualmente até 200 mV (superior ao potencial de oxidação do PPi) de modo que a corrente tendesse sempre a zero. Este procedimento torna o filme totalmente oxidado, impedindo que o mesmo sofra oxidação pelo oxigênio do ar durante a troca da solução de síntese pela solução de eletrólito. A oxidação pelo oxigênio é irreversível e portanto danificaria o filme.

Na sequência trocou-se a solução original pelas soluções aquosas 0.05 M dos sais KCl, NaCl e LiCl (uma para cada filme) e fez-se voltamogramas de -900 a 100 mV a várias velocidades de varredura. Na figura 5.1 é mostrado a superposição dos voltamogramas para cada eletrólito para a velocidade de varredura de 50 mV/s.

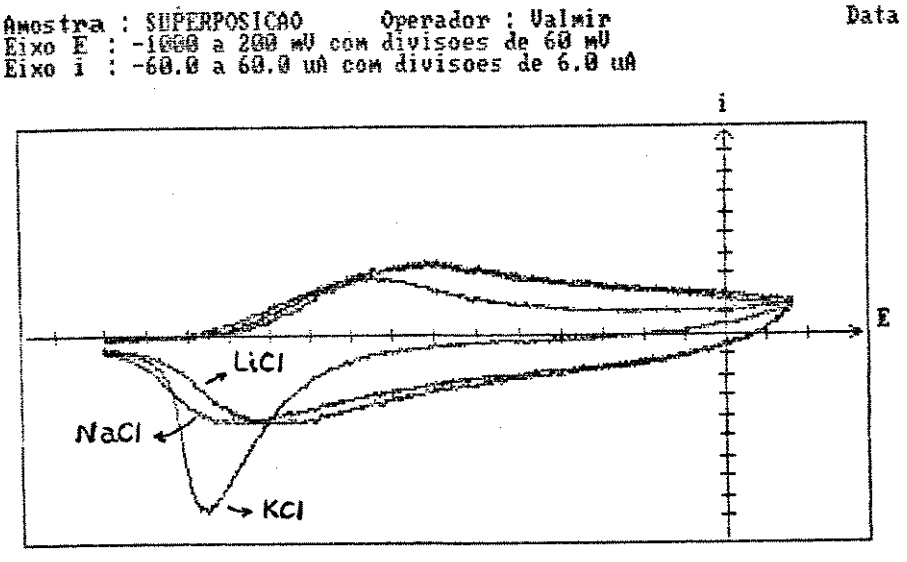
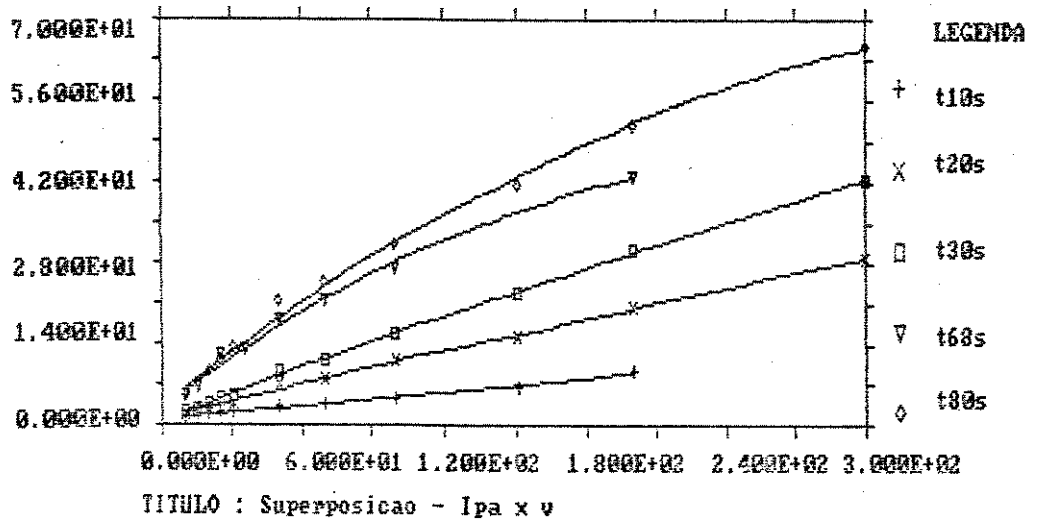


Figura 5.1: Superposição de voltamogramas (-900 a 100 mV a 50 mV/s) obtidos em função do eletrólito (sol. 0.05 M de KCl, NaCl e LiCl).

Como pode ser observado, o comportamento eletroquímico é dependente da natureza do cátion. Explicando-se este comportamento em termos de ΔE_p, uma vez que todos os voltamogramas foram obtidos à mesma velocidade de varredura, a reversibilidade da reação redox diminui de K^+ para Li^+.

Isto é consistente com o caráter eletrofílico (hidrofobicidade) dos cátions, que pode ser visto pelo raio de hidratação: 0,340, 0,276 e 0,232 nm para Li^+, Na^+ e K^+, respectivamente.[40]



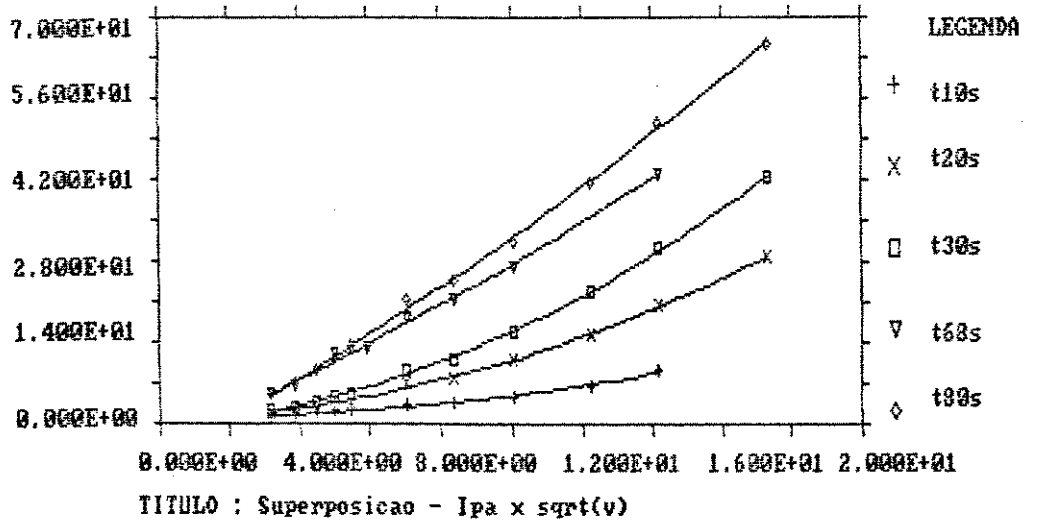


Figura 5.2: Gráficos da corrente de pico em função da velocidade de varredura: a) i_{pa} vs v (acima) e b) i_{pa} vs $\sqrt{v}$ (abaixo).

5.3. ELETROATIVIDADE EM FUNÇÃO DA ESPESSURA

Foram preparados 5 filmes do modo descrito no ítem 5.1, mas variando-se o tempo em cada síntese. Os tempos foram 10, 20, 30, 68 e 80 s, correspondendo a uma densidade de carga de 56, 112, 168, 381 e 448 mC/cm^2 e a filmes com espessuras de 0,17, 0,34, 0,52, 1,17 e 1,38 μm, respectivamente. Com estes filmes fez-se voltamogramas a várias velocidades com a finalidade de se obter dados a respeito da cinética de reação em função da espessura do filme.

A figura 5.2a mostra a superposição dos gráficos de i_{pa} *vs* υ. A partir destas curvas conclui-se que para espessuras maiores que 0,52 μm ocorre a mudança no modo de transporte de matéria (difusão em camada fina para difusão semi-inifinita), evidenciada pela perda de linearidade no gráfico.

A figura 5.2b mostra a superposição dos gráficos de i_{pa} *vs* $\sqrt{\upsilon}$ e, a partir dela, tem-se a confirmação que, para espessuras maiores que 0,52 μm, a cinética passa a ser limitada pelo transporte de matéria do seio da solução até a superfície do eletrodo (difusão semi-infinita), evidenciada pela linearidade das duas curvas correspondentes às espessuras de 1,17 e 1,38 μm.

Isto é consistente com o fato de que ao aumentar a espessura do filme, começa haver uma limitação na difusão dos íons da solução para a superfície do eletrodo, além do fato do filme no estado reduzido não ser condutor, o que dificulta a transferência eletrônica, ocasionando uma maior separação entre os picos.

Quando a reação é cineticamente limitada pela difusão das espécies, a separação entre os picos anódico e catódico aumenta a medida que a difusão se torna menor. Com efeito semelhante, a medida que a transferência eletrônica começa a ser prejudicada, o sistema começa a se tornar irreversível, conforme comentado no ítem 1.3.1. Isto pode ser observado pela superposição dos voltamogramas obtidos para cada espessura a uma velocidade de 50 mV/s, mostrado na figura 5.3, onde o voltamograma do filme mais espesso apresenta um ΔE_p maior que o do filme mais fino. Contudo, neste caso, não ocorre limitação da cinética pela transferência

eletrônica, pois se ocorresse, haveria perda de linearidade no gráfico de i_{pa} *vs* $\sqrt{v}$ para os filmes mais espessos.

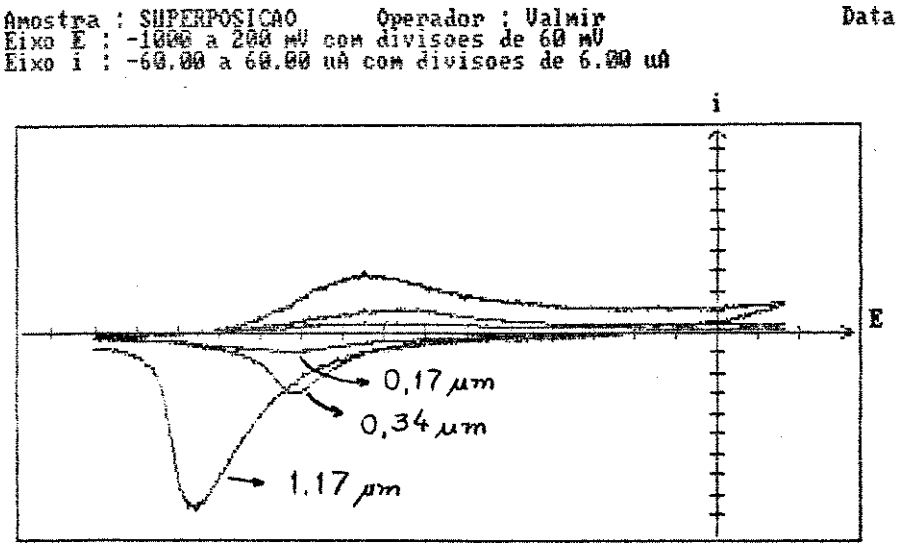


Figura 5.9: Supersposição de voltamogramas (-900 a 100 mV, 50 mV/s em KCl 0,05 M) obtidos de filmes de PPi/DS com espessuras de 0,17, 0,34 e 1,17 µm.

5.4. ELETROATIVIDADE EM FUNÇÃO DA VELOCIDADE DE VARREDURA

Considerando-se um processo controlado apenas pela difusão das espécies, a aplicação de um potencial elétrico impõe

um sentido de movimentação com direções opostas para cátions e ânions. A velocidade com que estes íons se difundem dependem de seu tamanho, mobilidade e carga, em outras palavras, de seu coeficiente de difusão.[33]

Em um experimento de voltametria cíclica a velocidades baixas, os íons têm tempo de se difundirem, entrando em equilíbrio no eletrodo. A medida que a velocidade de varredura aumenta, diminuem as chances do equilíbrio ser alcançado, dando a impressão de se tratar de um sistema irreversível.

Isto pode ser visto na figura 5.4, onde é mostrado a superposição de voltamogramas, obtidos de -900 a 100 mV à várias velocidades de varredura em solução 0,05 M de KCl, do filme de PPi/DS com 0,52 μm de espessura, obtido para o estudo do ítem 5.3.

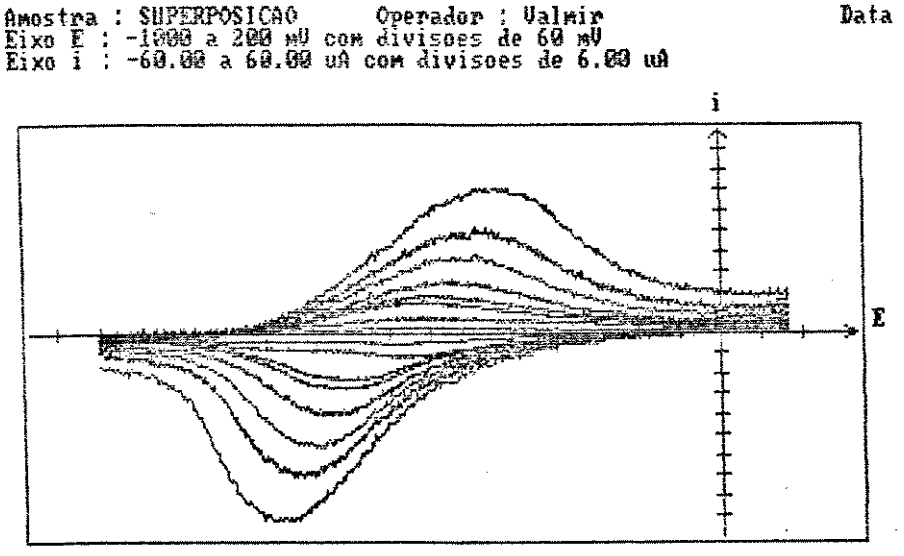


Figura 5.4: Superposição de voltamogramas obtidos de um filme de PPi/DS com espessura de 0,52 μm em solução 0,05M de KCl de -900 a 100 mV a várias velocidades de varredura.

O voltamograma correspondente a velocidade de varredura de 300 mV/s possui um ΔE_p igual a 306 mV, enquanto que para a velocidade de varredura de 25 mV/s o valor de ΔE_p é igual a 32 mV. Portanto, é importante estar ciente que não se constata se um sistema é ou não irreversível pela simples determinação do valor de ΔE_p para uma dada velocidade de varredura. O procedimento correto está em obter o comportamento das correntes de pico anódico e catódico em função da velocidade de varredura.

CAPÍTULO 6
AVALIAÇÃO E CONCLUSÕES

Os resultados obtidos nos ensaios com filmes de PPi, mostrados no capítulo 5, e os ensaios utilizados no capítulo 4 para demonstração dos programas, mostraram que o equipamento desenvolvido teve um bom desempenho, possibilitando a aplicação das técnicas eletroquímicas com bons resultados.

Avaliando-se cada parte do trabalho e, tendo-se em mente o que foi proposto inicialmente, julgamos ter atingido os objetivos e ter dado uma contribuição para a área de instrumentação eletroquímica, através do desenvolvimento de *hardware* e de *software* que permitem a utilização das técnicas eletroquímicas para síntese e caracterização de materiais.

Isto pode ser demonstrado pelo número de trabalhos que já foram e que estão sendo realizados utilizando-se o sistema desenvolvido.[21,22,38,39,41-79]

O sistema desenvolvido tem suas limitações, mas convém ressaltar que mesmo um sistema comercial altamente sofisticado também as possui e, muito dificilmente, é possível fazer algo para contorná-las ou, mais difícil ainda, solucioná-las. A vantagem em se desenvolver um sistema está em poder aperfeiçoá-lo sempre que necessário.

A seguir são enunciados as limitações tanto do *hardware* quanto do *software* e as sugestões para melhorá-las.

6.1. HARDWARE

A principal limitação do *hardware* é o tempo de conversão A/D que se situa na faixa de 1 a 3 ms. Este tempo de conversão impede a utilização do equipamento em alguns experimentos de cronoamperometria onde, para alguns materiais, há

necessidade de tempos de conversão da ordem de microssegundos.

Uma maneira de solucioná-la seria adquirir um conversor A/D de 12 bits com um tempo de conversão de alguns microssegundos, por exemplo, o AD574 da Analog Devices, que custa por volta de US$ 60,00 e não é encontrado no mercado brasileiro.

Apesar desta limitação, o equipamento é prontamente utilizado para as demais técnicas e para a técnica de cronoamperometria com materiais que possuem uma constante de tempo maior.

6.2. SOFTWARE

Com relação ao *software* podemos dizer que a limitação está apenas em empregar mais tempo de programação para torná-lo uma ferramenta mais ampla e mais poderosa.

Deve-se ressaltar que, uma vez que os dados estão digitalizados, a maior ou menor sofisticação no seu tratamento dependerá somente da habilidade do programador.

O conjunto de programas desenvolvidos cobre as necessidades básicas da análise e tratamento dos dados para obtenção dos parâmetros eletroquímicos necessários para a caracterização de espécies eletroativas. Porém, é possível determinar qualquer tipo de parâmetro a partir dos dados de corrente ou tensão obtidos da cela eletroquímica como resposta a uma determinada excitação. Obviamente, tudo se torna mais simples no caso de existir alguma equação que descreva o comportamento dos dados.

No futuro novos programas mais sofisticados poderão ser desenvolvidos para o mesmo equipamento. Outras técnicas também poderão ser empregadas, desde que sejam observadas as limitações do *hardware*. Outra possibilidade consiste na compatibilização dos arquivos de dados com outros *softwares* existentes (tratamento de dados, programas gráficos, etc.).

BIBLIOGRAFIA

1. T. A. Skothein (ed), *Handbook of Conducting Polymers*, Vols 1 and 2, Marcel Decker, New York, 1986.

2. C. W. Tang, *Appl. Phys. Lett.* **48** (1986) 183.

3. G. A. Chamberlain, *J. Appl. Phys.* 53 (1982) 6262.

4. A.-M. Hor, R. O. Loutfy, C.-K. Hsiao, *Appl. Phys. Lett.* **42** (1983) 165.

5. G. Horowitz, *Adv. Mater.* 2 (1990) 287.

6. S. Glenis, G. Horowitz, G. Tourillon, F. Garnier, *Thin Solid Films* **111** (1984) 93.

7. T. Skotheim, O. Inganas, J. Prejza, I. Lundstrom, *Mol. Cryst. Liq. Cryst.* 83 (1982) 329.

8. A. Tsumura, H. Koezuka, T. Ando, *Shynt. Met.* 25 (1988) 11.

9. A. Assadi, C. Svensson, M. Willander, O. Inganas, *Appl. Phys. Lett.* 53 (1988) 195.

10. G. Horowitz, X. Peng, D. Fichou, F. Garnier, *J. Appl. Phys.* **67** (1990) 528.

11. G. Horowitz, D. Fichou, X. Peng, Z. Xu, F. Garnier, *Solid State Commun.* 72 (1989) 381.

12. K. Menke, S. Roth, *Chemie in Unserer Zeit*, 20 (1986) 1.

13. R. MacNeill, D. E. Weiss and D. Willist, *Aust. J. Chem.*, **18** (1965) 477.

14. A. F. Diaz, K. K. Kanazawa and G. P. Gardini, *J. Chem. Soc., Chem. Commun.*, (1979) 635.

15. B. Scrosati, *Prog. Solid State Chem.*, 18 (1988) 1.

16. M.-A. De Paoli e R. K. Menescal, *Quim. Nova*, 9 (1986) 133.

17. M. Satoh, K. Kaneto and K. Yoshino, *Synth. Met.*, **14** (1986) 289.

18. W. Wernet, M. Monkenbusch and G. Wegner, *Makromol. Chem. Rapid Commun.*, 5 (1984) 157.

19. T. Iyoda, A. Othani, T. Shimidzu and K. Honda, *Chem. Lett.*, (1986) 687

20. W. Wernet, M. Monkenbusch and G. Wegner, *Mol. Cryst. Liq. Cryst.*, **118** (1985) 193.

21. R. C. D. Peres, J. M. Pernaut and M.-A. De Paoli, *Synth. Met.*, 28 (1989) C59.

22. J. M. Pernaut, R. C. D. Peres, V. F. Juliano and M.-A. De Paoli, *J. Electroanal. Chem.*, **274** (1989) 225.

23. B. Q. Malhotra, N. Keenar, S. Chandra, *Prog. Polym. Sci*, **12** (1986) 179.

24. T. A. Skothein (ed), *Handbook of Conducting Polymers*, Vol. 1, Marcel Decker, New York, 1986, p. 82.

25. E. M. Genies, G. Bidan, A. F. Diaz, *J. Electroanal. Chem. Interfacial Electrochem.*, **149** (1983) 101.

26. A. F. Diaz, J. I. Castilho, J. A. Logan, W.-Y. Lee, *J. Electroanal. Chem. Interfacial Electrochem.*, 129 (1981) 115.

27. D. T. Sawyer and J. L. Roberts, Jr., *Experimental Electrochemistry for Chemists*, John Wiley & Sons, Inc., New York, 1974, p. 1-6, 222, 329-339, 395-398.

28. J. Heinze, *Angew. Chemie, Int. Ed.*, 23 (1984) 831.

29. D. Pletcher, *Chem. Soc. Rev.*, **4** (1975) 471.

30. J. M. Pernaut, Tese de Doutorado, Grenoble-França (1986).

31. E. Vieil, *Comunicação pessoal* (1989).

32. A. J. Bard, L. R. Faulkner, *Electrochemical Methods - Fundamentals and Applications*, John Wiley & Sons, New York, 1980, p. 136-146, 199-205, 213-231, 249-261, 553-574.

33. W. J. Moore, *Físico-Química*, vol. 2, Ed. Edgard Blücher Ltda, São Paulo, 1976, p. 399, 462.

34. H. Taub, D. Schilling, *Eletrônica Digital*, McGraw-Hill do Brasil Ltda., São Paulo, 1982, p. 437-487.

35. G&E Intersil data-sheet.

36. L. C. Eggebrecht, *Interfacing to the IBM Personal Computer*, 2nd ed., H. W. Sams and Co., 1990.

37. S. K. O'Brien, *Turbo Pascal - The Complete Reference*, Borland-Osborne/McGraw Hill, 1988.

38. Ana Maria Rocco, Tese de Doutorado, IQ-UNICAMP, em andamento.

39. Rosa C. D. Peres, Tese de Doutorado, IQ-UNICAMP, em andamento.

40. F. A. Cotton and G. Wilkinson, *Advanced Inorganic Chemistry*, John Wiley, New York, 1972, p.198.

41. M.-A. De Paoli, R.C.D. Peres, M.W.C. Dezotti, E.L. Tassi, V.F. Juliano, M.A. Rodrigues e J.M. Pernaut, Anais do VI-SEMPOL BRASIL/FRANÇA, Rio de Janeiro (1988).

42. R.C.D. Peres, J.M. Pernaut e M.-A. De Paoli, Anais do III Encontro sobre Materiais na Indústria Eletrônica e de Telecomunicações, Campinas (1988), pp. 457-466.

43. M.-A. De Paoli, R.C.D. Peres e J.M. Pernaut, *J. Braz. Chem. Soc.*, 1 (1990) 50.

44. M.-A. De Paoli, R.C.D. Peres, E.R. Duek, E.L. Tassi e F. Deker, Anais do XVIII Jornadas Chilenas de Química, Santiago (1989), p. 64 a 70.

45. E.A. Rezende Duek e M.-A. De Paoli, Anais do VII Simpósio Brasileiro de Eleltroquímica e Eletroanalítica, Ribeirão Preto (1990) p. 38-43.

46. R.C.D. Peres, L. Miccaroni e M.-A. De Paoli, Anais do VII Simpósio Brasileiro de Eletroquímica e Eletroanalítica, Ribeirão Preto (1990) p. 329-334.

47. M.A. Rodrigues e M.A. De Paoli, Anais do VII Simpósio Brasileiro de Eleltroquímica e Eletroanalítica, Ribeirão Preto (1990) p. 335-340.

48. E. Tassi e M.-A. De Paoli, *J. Chem. Soc., Chem. Commun.* (1990) 155.

49. R.A. Zoppi e M.-A. De Paoli, *J. Electroanal. Chem.*, **290** (1990) 275-282.

50. E.A.R. Duek, M.-A. De Paoli e F. Decker, Anais do IX CBECIMAT, Águas de S. Pedro, 1990, p. 274-277.

51. R.A. Zoppi e M.-A. De Paoli, Anais do IX CBECIMAT, Águas de S. Pedro, 1990, p. 986-990.

52. R.C.D. Peres, L. Miccaroni, M.-A. De Paoli, S. Panero e B. Scrosati, Anais do IX CBECIMAT, Águas de S. Pedro, 1990, p. 664-667.

53. R.C.D. Peres, J.M. Pernaut and M.A. De Paoli, *J. Polym. Sci., Polym. Chem. Ed.*, 29 (1991) 225-231.

54. M.A. Rodrigues e M.-A. De Paoli, *Synth. Met.*, **43** (1991) 2957-2962.

55. E. L. Tassi and M.-A. De Paoli, *POLYMER*, (1991) no prelo.

56. Rosa C. D. Peres e M.-A. De Paoli, *Ciênc. Cult.*, 39 (supl.), (1987) 564

57. R.C. Pereira, M.-A. De Paoli, R.C.D. Peres e G. Oliveira Neto, VI Simpósio de Eletroquímica e Eletroanalítica, São Paulo (1988).

58. R.C.D. Peres e M.-A. De Paoli, 8º Congresso Brasileiro de Engenharia e Ciência dos Materiais, Campinas, 1988, Anais p.345 a 347.

59. R.C.D. Peres e M.-A. De Paoli, Anais do I Simposio Latinoamericano de Polimeros, Porlamar, Venezuela (1988) p. 927 a 931.

60. R.C.D. Peres, J.M. Pernaut e M.-A. De Paoli, *Ciênc. Cult.*, **41** (supl.), (1989) 566

61. E.A.R. Duek e M.-A. De Paoli, VII Simpósio Brasileiro de Eletroquímica e Eletroanalítica, Ribeirão Preto, 1990, Res. p. 38.

62. M.A. Rodrigues e M.-A. De Paoli, VII Simpósio Brasileiro de Eletroquímica e Eletroanalítica, Ribeirão Preto, 1990, Res. p. 335.

63. R.C.D. Peres, L. MIccaroni e M.-A. De Paoli, VII Simpósio Brasileiro de Eletroquímica e Eletroanalítica, Ribeirão Preto, 1990, Res. p. 58

64. M.-A. De Paoli, E.A.R. Duek, E.L. Tassi, R.C.D. Peres, M.A. Rodrigues e M. Martini, 4th Macromolecular Colloquium Porto-Alegre/Freiburg, Gramado (1990), Res. p. 43-45.

65. M.-A. De Paoli, R.A. Zoppi, IX Congresso Iberoamericano de Electroquimica, Tenerife (1990), Res. p. 308.

66. M.-A. De Paoli, E.A.R. Duek e M.A. Rodrigues, IX Congresso Iberoamericano de Electroquimica, Tenerife (1990), Res. p. 307.

67. M.A. Rodrigues e M.-A. De Paoli, International Conference on Science and Technology of Synthetic Metals, Tubingen, RFA, 1990.

68. M.-A. De Paoli, E.R. Duek e M.A. Rodrigues, International Conference on Science and Technology of Synthetic Metals, Tubingen, RFA, 1990.

69. V. Mano, M. I. Felisberti e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. FQ-008.

70. M.A. Rodrigues e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. FQ-009.

71. A.M. Rocco e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. FQ-010.

72. R.A. Zoppi e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. FQ-011.

73. E.R. Duek e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. FQ-012.

74. W.A. Gazotti Jr, R.C.D. Peres e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. IC-09.

75. L. Micaroni e M.-A. De Paoli, 14ª Reunião Anual da SBQ, Caxambu, maio de 1991, Res. IC-23.

76. R. A. Zoppi, Tese de Mestrado, IQ-UNICAMP (1991)

77. M. A. Rodrigues, Tese de Doutorado, IQ-UNICAMP, em andamento.

78. E. R. Duek, Tese de Doutorado, IQ-UNICAMP, em andamento.

79. E. L. Tassi, Tese de Doutorado, IQ-UNICAMP, em andamento.

APÊNDICE 1
LISTAGEM DOS PROGRAMAS

Nesta seção encontram-se listados os programas desenvolvidos neste trabalho:

- Programa CALIBRA;
- Programa FAC;
- Programa MULTICAN;
- Programa VOLTAM;
- Programa CRONOAM.

Embora existam algumas subrotinas comuns aos programas, elas não foram separadas dos mesmos para que não houvesse interrupção ou dúvida na sequência de cada um dos programas.

A1.1. PROGRAMA CALIBRA

```
Program CALIBRA;

{-----------------------------------------------}
{ Este programa faz a calibracao do D/A e do A/D }
{-----------------------------------------------}

Var i,v,canal,ganho,seh,histda,incr : Integer;
    vlr : Real;

Procedure InicializaSeletor;
Begin
  canal := 0;                    { Entrada padrao. }
  ganho := 0;                    { Ganho unitario. }
  seh := 128;                    { Amostra. }
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Procedure Seletor(c : Integer);
Begin
  Port[$303] := c + (ganho shl 4) + seh;
  canal := c;
End;

Procedure Volume(g : Integer);
Begin
  ganho := g shl 4;
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Procedure HoldDA;
Begin
  Port[$304] := 0;
End;

Procedure DAC(valor : Integer);
Begin
  PortW[$300] := valor;
  Port[$304] := 1;        { Sample }
  histda := valor;
End;

Procedure SampleAD;
Begin
  seh := 128;
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Procedure HoldAD;
Begin
  seh := 0;
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Function ADC : Integer;
Var i, ad : Integer;
```



```
Begin
  ad := 0;
  HoldAD;
  HoldDA;
  PortW[$300] := 2048;
  If (Port[$302] and 1) = 1 then ad := ad + 2048;
  PortW[$300] := ad + 1024;
  If (Port[$302] and 1) = 1 then ad := ad + 1024;
  PortW[$300] := ad + 512;
  If (Port[$302] and 1) = 1 then ad := ad + 512;
  PortW[$300] := ad + 256;
  If (Port[$302] and 1) = 1 then ad := ad + 256;
  PortW[$300] := ad + 128;
  If (Port[$302] and 1) = 1 then ad := ad + 128;
  PortW[$300] := ad + 64;
  If (Port[$302] and 1) = 1 then ad := ad + 64;
  PortW[$300] := ad + 32;
  If (Port[$302] and 1) = 1 then ad := ad + 32;
  PortW[$300] := ad + 16;
  If (Port[$302] and 1) = 1 then ad := ad + 16;
  PortW[$300] := ad + 8;
  If (Port[$302] and 1) = 1 then ad := ad + 8;
  PortW[$300] := ad + 4;
  If (Port[$302] and 1) = 1 then ad := ad + 4;
  PortW[$300] := ad + 2;
  If (Port[$302] and 1) = 1 then ad := ad + 2;
  PortW[$300] := ad + 1;
  If (Port[$302] and 1) = 1 then ad := ad + 1;
  DAC(histda);              { Reestabelece o antigo valor para o DA. }
  SampleAD;
  ADC := ad;
End;

BEGIN
  InicializaSeletor;
  Seletor(0); Volume(0);
  ClrScr;
  Writeln;
  incr := 100;
  Writeln('Calibracao do D/A');
  Writeln;
  Write('Qual o incremento nos valores do DA? ');
  ReadLn(incr);
  v := 0;
  Repeat
    DAC(v);
    Writeln('D/A = ',v);
    Repeat Until KeyPressed;
    v := v + incr;
  Until v >= 4095;
  v := 0;
  Writeln;
  Writeln('Calibracao do A/D');
  Writeln;
  Repeat
```

```
    vlr := 0;
    DAC(v);
    Delay(1);
    For i := 1 to 1000 do
      vlr := vlr + ADC;
    vlr := vlr / 1000.0;
    Writeln('DA = ',v,'   A/D = ',vlr:0:0);
    v := v + incr;
    Repeat Until KeyPressed;
  Until v >= 4095;
  Sound(1000); Delay(100); NoSound;
END.
```

A1.2. PROGRAMA FAC

```
Program FAC;

{-------------------------------------------------------------------------}
{                                                                         }
{  Este programa faz voltametria ciclica, cronoamperometria, cronopoten-  }
{ciometria e cronocoulometria utilizando um potenciostato FAC 200 e a     }
{interface desenvolvida por CLL/VFJ que trabalha da seguinte maneira:     }
{                                                                         }
{     * CONVERSOR D/A DE 12 BITS :                                        }
{         Byte menos significativo - Port[$300]                           }
{         4 bits mais significativo - Port[$301]                          }
{                                                                         }
{     * SELETOR E GANHO :                                                 }
{         8 canais - 3 bits menos significativos de Port[$303]            }
{                    BNC 0 - entrada para corrente e tensao               }
{                            +/- 10 ou 2V selecionavel por chave externa  }
{                    RCA 1 a 6 - livres                                   }
{         8 ganhos - bits de 4 a 6 de Port[$303]                          }
{                    valor 0 - ganho 1                                    }
{                    valor 1 - ganho 2                                    }
{                    valor 2 - ganho 3                                    }
{                    valor 3 - ganho 5                                    }
{                    valor 4 - ganho 9                                    }
{                    valor 5 - ganho 17                                   }
{                    valor 6 - ganho 30                                   }
{                    valor 7 - ganho 81                                   }
{                                                                         }
{     * CONVERSOR A/D DE 12 BITS :                                        }
{         Byte menos significativo - Port[$300]                           }
{         4 menos significativos de Port[$301]                            }
{         Comparador em Port[$302]                                        }
{                                                                         }
{-------------------------------------------------------------------------}
{  A constante TA e' o tempo gasto para adquirir um dado do ADC.          }
{  A constante TB e' o tempo gasto para tratar e plotar um ponto do vetor.}
{  TA e TB sao obtidos por regressao linear como descrito no procedure    }
{ COLETA.FAC.                                                             }
{  A constante TC e' o "offset" em mV que permite que os valores do ADC   }
{ sejam positivos. Isto e' necessario para permitir a soma de muitos valo-}
{ res inteiros.                                                           }
{-------------------------------------------------------------------------}

  Const TA = 7.6479625E-04;
        TB = 3.0767831125E-03;  ( correlacao 0.9999973 em 25/09/90 )
        TC = 2300;
        LF = #10;
        CR = #13;
        versao = '2.0';

  Type String80    =  String[80];
       String15    =  String[15];
       String8     =  String[8];
       String2     =  String[2];
       Matrizd     =  ^md;
       md          =  Array[-2020..2020] of Integer;
```



```
       Matriza     =  ^ma;
       ma          =  Array[0..4095] of Integer;
       Vetor       =  ^TipoVet;
       TipoVet     =  Array[0..1600] of Integer;
       Ponteiro1   =  ^TipoVet1;
       TipoVet1    =  Array[0..1600,1..20] of Integer;
       Ponteiro2   =  ^TipoVet2;
       TipoVet2    =  Array[0..32000] of Integer;
       Ponteiro3   =  ^TipoVet3;
       TipoVet3    =  Array[0..4010] of Real;

  Var  MAD         :  Matriza;
       MDA         :  Matrizd;
       D1          :  Ponteiro1;
       D2          :  Ponteiro1;
       D3          :  Ponteiro1;
       Crono       :  Ponteiro2;
       Q           :  Ponteiro3;
       t           :  Ponteiro3;
       mV          :  Vetor;
       h1,m1,s1,
       h2,m2,s2,
       h3,m3,s3,
       histda,
       i,nc,
       imax,imin,
       iini, ifi,
       vmax,vmin,
       vini, vfi,
       corrente,
       intervalo,
       incremento,
       espera,
       primeiro,
       ultimo,passo,
       nq, tpo,
       tpoa,tpod,
       tpotot,
       npulsos,
       linha,coluna :  Integer;
       tecla,
       tecnica      :  Char;
       seh,
       canal,ganho  :  Byte;
       operador,
       amostra      :  String[20];
       data         :  String8;
       tipo,unidade :  String2;
       inii, fii,
       fatortt,
       tempo,velo,
       incr,fEi,
       tempopto     :  Real;
       vunidade     :  Byte absolute unidade;
       FAC200A      :  Boolean;
```

```
        multpot     :  Byte;
        marcaheap   :  ^Char;


procedure Graphics;                                 external 'c:\turbo\GRAPH.BIN';
procedure HiRes;                                    external Graphics[6];
procedure GraphWindow(X1,Y1,X2,Y2: Integer);        external Graphics[18];
procedure Plot(X,Y,Color: Integer);                 external Graphics[21];
procedure Draw(X1,Y1,X2,Y2,Color: Integer);         external Graphics[24];
procedure GetPic(var Buffer; X1,Y1,X2,Y2: Integer); external Graphics[36];
procedure PutPic(var Buffer; X,Y: Integer);         external Graphics[39];
procedure ClearScreen;                              external Graphics[60];

Procedure Borda(x1,y1,x2,y2 : Integer);

Begin
  GraphWindow(0,0,639,199);
  Draw(x1,y1,x2,y1,1); Draw(x2,y1,x2,y2,1);
  Draw(x2,y2,x1,y2,1); Draw(x1,y1,x1,y2,1);
End;

Procedure Janela(x1,y1,x2,y2 : Integer);

Var    i,
       maxx,
       maxy    :  Integer;

Begin
  Window (1,1,80,25);
  GoToXY (x1,y1);
  Window (x1,y1,x2,y2);
  ClrScr; maxx := x2 - x1 + 1; maxy := y2 - y1 + 1;
  GoToXY (1,maxy); Write (#212);
  For i := 2 to maxx - 1 do Write(#205);
  Write (#190);
  GoToXY (1,1); Write (#213);
  For i := 2 to maxx - 1 do Write(#205);
  Write (#184);
  For i := 2 to maxy - 2 do
    begin
      GoToXY (1,i); Write(#179);
      GoToXY (maxx,i); Write(#179);
    end;
  GoToXY (2,2); Window (x1+1,y1+1,x2-1,y2-2);
  linha := y1 ; coluna := x1 + 1;
End;

Procedure TelaInicio;

type  matrizg = Array[0..4099] of Byte;
       Disco = record
                 telag : matrizg;
               end;

Var  arquivo  :  Disco Absolute $B800:0000;
```



```
    buftela  :  file of Disco;

Begin
  ClrScr;
  Assign(buftela,'FAC.TXT');
  Reset(buftela);
  Read(buftela,arquivo);
  Close(buftela);
end;

Procedure Menu;

Begin
  ClrScr; Janela(1,1,80,25);
  GoToXY(8,22);
  Write('PROGRAMA  " FAC " - Versao ',versao,'          (C) CLL/VFJ - 1990');
  Janela(4,3,34,22);
  GoToXY(8,1); Write('MENU INICIAL');
  GoToXY(8,2); Write('============');
  GoToXY(4,3); Write('C - Coletar dados');
  GoToXY(4,5); Write('A - Arquivo em disco');
  GoToXY(4,7); Write('R - Resultados');
  GoToXY(4,9); Write('S - Salvar dados');
  GoToXY(4,11); Write('D - Diretorio');
  GoToXY(4,13); Write('F - Finalizar');
End;

Procedure Date(var dia,mes,ano : Byte);

Type
  RegList  =  Record
    AX, BS, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
  end;

Var   Reg       :   RegList;

Begin
  Reg.Ax := $2A00;
  MSDOS(Reg);
  Reg.CX := Reg.CX - 1900;
  If Reg.CX > 99 then Reg.CX := Reg.CX - 100;
  ano := Reg.CX; mes := Hi(Reg.DX); dia := Lo(Reg.DX);
End;

Procedure ReadIn(c1 : Char; var x : Integer);

type par = Record
             car   : Char;
             atrib : Byte;
           end;

    TipoLinha = Array[1..80] of par;
    TipoTela  = Array[1..25] of TipoLinha;

var numero           : String[20];
```

```
    codigo, spc, c    : Integer;
    tela              : TipoTela absolute $B800:$0000;

Begin
  Str(x, numero);
  Write(numero);
  Repeat Until keypressed;
  Repeat
    spc := length(numero);
    GoToXY(WhereX - spc, WhereY);
    For c := WhereX to WhereX + spc do
      tela[linha + WhereY,coluna + c].car := ' ';
    Read(numero); val(numero, x, codigo);
  Until codigo = 0;
  spc := length(numero); GoToXY(WhereX - spc, WhereY); Write(x);
  If ci = CR then Writeln;
End;

Procedure ReadRe(ci : Char; var x : Real; spc, dec : Integer);

type par = Record
             car   : Char;
             atrib : Byte;
           end;

     TipoLinha = Array[1..80] of par;
     TipoTela  = Array[1..25] of TipoLinha;

var numero      : String[20];
    codigo,c,xi : Integer;
    tela        : TipoTela absolute $B800:$0000;

Begin
  xi := WhereX; Str(x, numero);
  Write(x:spc:dec);
  Repeat Until keypressed;
  Repeat
    GoToXY(xi,WhereY);
    For c := WhereX to WhereX + spc do
      tela[linha + WhereY,coluna + c].car := ' ';
    Read(numero);
    val(numero, x, codigo);
    If codigo <> 0 then spc := length(numero);
  Until codigo = 0;
  GoToXY(xi, WhereY);  Write(x:spc:dec);
  If ci = CR then Writeln;
End;

Procedure InicializaSeletor;
Begin
  canal := 0;                       { Entrada padrao. }
  ganho := 0;                       { Ganho unitario. }
  seh := 128;                       { Amostra. }
  Port[$303] := canal + (ganho shl 4) + seh;
End;
```



```
Procedure Seletor(c : Integer);
Begin
  Port[$303] := c + (ganho shl 4) + seh;
  canal := c;
End;

Procedure Volume(g : Integer);
Begin
  ganho := g shl 4;
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Procedure HoldDA;
Begin
  Port[$304] := 0;
End;

Procedure DAC(valor : Integer);
Begin
  PortW[$300] := MDA^[valor];
  Port[$304] := 1;        { Sample }
  histda := valor;
End;

Procedure SampleAD;
Begin
  seh := 128;
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Procedure HoldAD;
Begin
  seh := 0;
  Port[$303] := canal + (ganho shl 4) + seh;
End;

Procedure ADC(var valorav : Integer);
Var i, ad : Integer;
Begin
  ad := 0;
  HoldAD;
  HoldDA;
  PortW[$300] := 2048;
  If (Port[$302] and 1) = 1 then ad := ad + 2048;
  PortW[$300] := ad + 1024;
  If (Port[$302] and 1) = 1 then ad := ad + 1024;
  PortW[$300] := ad + 512;
  If (Port[$302] and 1) = 1 then ad := ad + 512;
  PortW[$300] := ad + 256;
  If (Port[$302] and 1) = 1 then ad := ad + 256;
  PortW[$300] := ad + 128;
  If (Port[$302] and 1) = 1 then ad := ad + 128;
  PortW[$300] := ad + 64;
  If (Port[$302] and 1) = 1 then ad := ad + 64;
```

```
  PortW[$300] := ad + 32;
  If (Port[$302] and 1) = 1 then ad := ad + 32;
  PortW[$300] := ad + 16;
  If (Port[$302] and 1) = 1 then ad := ad + 16;
  PortW[$300] := ad + 8;
  If (Port[$302] and 1) = 1 then ad := ad + 8;
  PortW[$300] := ad + 4;
  If (Port[$302] and 1) = 1 then ad := ad + 4;
  PortW[$300] := ad + 2;
  If (Port[$302] and 1) = 1 then ad := ad + 2;
  PortW[$300] := ad + 1;
  If (Port[$302] and 1) = 1 then ad := ad + 1;
  DAC(histda);              { Reestabelece o antigo valor para o DA. }
  SampleAD;
  valorav := MAD^[ad];
End;

Procedure CalcCont;

Var   a,b      :  Real;
      c        :  Integer;

Begin
  a := -0.976820; b := 1999.668;       { Converte os valores binarios }
  For c := 0 to 4095 do                { para valores em mV para o DA }
    MDA^[Round(a * c + b)] := c;       { na faixa de -2000 a 2000 mV. }

  a := 1.113952; b := -2246.45 + TC;   { Converte os valores binarios }
  For c := 0 to 4095 do                { para valores em mV para o AD }
    MAD^[c] := Round(a * c + b);       { na faixa de -2000 a 2000 mV. }
End;

Procedure Aviso(mensagem : String80);

Begin
  Janela((39 - length(mensagem) div 2),11,(41 + length(mensagem) div 2),14);
  Write(mensagem);
  Sound(2000); Delay(50); Sound(1000); Delay(50); NoSound;
  Repeat Until KeyPressed;
End;

Overlay Procedure Salva(np : String15; tpotot : Integer);

   Type RegCiclo = Record
                     Dados : TipoVet;
                   end;
        RegCrono = Record
                     Cronos : Ponteiro2;
                   end;
        RegmV    = Record
                     mVs : TipoVet;
                   end;
        RegParametros = Record
                          unidadeP    : String[2];          { CA e V }
                          AmostraP,                         { T }
```



```
                          OperadorP   : String[20];         { T }
                          DataP       : String8;            { T }
                          tecnicaP    : Char;               { T }
                          fatorttP    : Real;               { T }
                          ganhoP      : Byte;               { T }
                          intervaloP,                       { T }
                          tpoP,                             { T }
                          tpoaP,                            { CA e CP }
                          tpodP,                            { CA e CP }
                          tpototP,                          { CA e CP }
                          npulsosP,                         { CA }
                          nCP,                              { V }
                          vmaxP,                            { CP }
                          vminP,                            { CP }
                          viniP,                            { CA e V }
                          vfiP,                             { CA e V }
                          correnteP   : Integer;            { CA e V }
                          veloP,                            { V }
                          incrP       : Real;               { V }
                          primeiroP,                        { V }
                          ultimoP,                          { V }
                          passoP      : Integer;            { V }
                          tempoptoP,                        { CC }
                          tempoP      : Real;               { CA e CP }
                          incrementoP,                      { CC }
                          nqP,                              { CC }
                          iiniP,                            { CP }
                          ifiP  : Integer;                  { CP }
                        end;


var   Parametros    : RegParametros;
      ArqParametros : file of RegParametros;
      nome,drv      : String[10];
      tecla,drive   : Char;
      i, ciclo,
      controle      : Integer;
      ASCII         : Boolean;


procedure SalvaCiclos;

   var UmCiclo      : RegCiclo;
       ArqCiclo     : file of RegCiclo;
       Onda         : RegmV;
       ArqOnda      : file of RegmV;

   begin
     Assign(ArqCiclo,nome + '.DAT');
     {$I-}
     Rewrite(ArqCiclo);
     {$I+}
     If IOResult <> 0 then
       begin
         Aviso('Problemas no disco ou diretorio cheio');
```

```
          controle := 1;
          Close(ArqCiclo);
          EXIT;
        end;
      ciclo := primeiro;
      Repeat
        Case ciclo of
          1..20   : For i := 0 to 1600 do UmCiclo.Dados[i] := D1^[i,ciclo];
          21..40  : For i := 0 to 1600 do UmCiclo.Dados[i] := D2^[i,ciclo-20];
          41..60  : For i := 0 to 1600 do UmCiclo.Dados[i] := D3^[i,ciclo-40];
          end; { case }
        {$I-}
        Write(ArqCiclo,UmCiclo);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Problemas no disco ou disco cheio');
            controle := 1;
            Close(ArqCiclo);
            EXIT;
          end;
        ciclo := ciclo + passo;
      Until ( ciclo > ultimo);
      Close (ArqCiclo);
      Assign(ArqOnda,nome + '.VOL');    { salva a matriz aV }
      {$I-}
      Rewrite(ArqOnda);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas no disco ou diretorio cheio');
          controle := 1;
          Close(ArqOnda);
          EXIT;
        end;
      With Onda do aVs := aV^;
      {$I-}
      Write(ArqOnda,Onda);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas no disco ou disco cheio');
          controle := 1;
          Close(ArqOnda);
          EXIT;
        end;
      Close(ArqOnda);
    end;


procedure SalvaCrono;

  var   UmCrono      : RegCrono;
        ArqCrono     : file of RegCrono;
  var   ArqCronoASCII : file of Integer;
```



```
        i            : Integer;

begin
  If ASCII then
    begin
      Assign(ArqCronoASCII,nome + '.DAT');
      {$I-}
      Rewrite(ArqCronoASCII);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas no disco ou diretorio cheio');
          controle := 1;
          Close(ArqCronoASCII);
          EXIT;
        end;
      {$I-}
      For i := 0 to tpotot do Write(ArqCronoASCII,Crono^[i]);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas no disco ou disco cheio');
          controle := 1;
          Close(ArqCronoASCII);
          EXIT;
        end;
      Close(ArqCronoASCII);
    end
  else
    begin
      Assign(ArqCrono,nome + '.DAT');
      {$I-}
      Rewrite(ArqCrono);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas no disco ou diretorio cheio');
          controle := 1;
          Close(ArqCrono);
          EXIT;
        end;
      With UmCrono do Cronos := Crono;
      {$I-}
      Write(ArqCrono,UmCrono);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas no disco ou disco cheio');
          controle := 1;
          Close(ArqCrono);
          EXIT;
        end;
      Close(ArqCrono);
    end;
end;
```

```
procedure SalvaParametros;

  begin
    Assign(ArqParametros,nome + '.PAR');
    {$I-}
    Rewrite(ArqParametros);
    {$I+}
    If IOResult <> 0 then
      begin
        Aviso('Problemas no disco ou diretorio cheio');
        controle := 1;
        Close(ArqParametros);
        EXIT;
      end;
    With Parametros do
      begin
        unidadeP    := unidade;
        AmostraP    := Amostra;
        OperadorP   := Operador;
        DataP       := Data;
        tecnicaP    := tecnica;
        fatorttP    := fatortt;
        ganhoP      := ganho;
        intervaloP  := intervalo;
        tpoP        := tpo;
        tpoaP       := tpoa;
        tpodP       := tpod;
        tpototP     := tpotot;
        npulsosP    := npulsos;
        ncP         := nc;
        vmaxP       := vmax;
        vminP       := vmin;
        viniP       := vini;
        vfiP        := vfi;
        correnteP   := corrente;
        veloP       := velo;
        incrP       := incr;
        primeiroP   := primeiro;
        ultimoP     := ultimo;
        passoP      := passo;
        tempoptoP   := tempopto;
        tempoP      := tempo;
        incrementoP := incremento;
        nqP         := nq;
        iiniP       := iini;
        ifiP        := ifi;
      end;  { With }
    {$I-}
    Write(ArqParametros,Parametros);
    {$I+}
    If IOResult <> 0 then
      begin
        Aviso('Problemas no disco ou disco cheio');
```



```
        controle := 1;
        Close(ArqParametros);
        EXIT;
      end;
    Close(ArqParametros);
  end;


Begin
  Controle := 0; ASCII := TRUE;
  GetDir(0,drv);
  If drv[1] = 'A' then drive := 'B'
  else drive := 'A';
  If np <> ' ' then nome := drive + ':p' + np;
  If np = ' ' then
    Repeat
      Janela(5,4,60,20);
      ClrScr;  ASCII := TRUE;
      WriteLn('SALVA DADOS EM DISCO',CR,LF);
      Write(' <ESC> - Abandona  '); Read(Kbd,tecla);
      If tecla = #27 then EXIT;
      GoToXY(1,WhereY);
      WriteLn('1) Salva em ASCII      2) Salva em binario   ');
      Repeat
        Read(Kbd,tecla); tecla := UpCase(tecla);
      Until tecla in ['1','2'];
      If tecla = '1' then ASCII := TRUE
      else ASCII := FALSE;
      WriteLn('Qual o nome do arquivo? (maximo 8 caracteres)'); ReadLn(nome);
      If nome[2] <> ':' then nome := drive + ':' + nome;
      If Pos('.',nome) > 0 then
        nome := Copy(nome,1,Pos('.',nome)-1);
      For i := 1 to Length(nome) do nome[i] := UpCase(nome[i]);
      WriteLn('Arquivo de dados: ',nome + '.DAT',LF);
      If tecnica = 'V' then
        WriteLn('                  ',nome + '.VOL',LF);
      WriteLn('Arquivo de parametros: ',nome + '.PAR',LF);
      If tecnica = 'V' then
        begin
          primeiro := 1; ultimo := nc; passo := 1;
          Write('Primeiro ciclo = '); ReadLn(CR,primeiro);
          Write('Ultimo ciclo = '); ReadLn(CR,ultimo);
          If (ultimo > nc) then ultimo := nc;
          If (ultimo < primeiro) then ultimo := primeiro;
          Write('Incremento = '); ReadLn(CR,passo);
          If (passo <= 0) then passo := 1;
        end;
      WriteLn('Confirma? (S/N)'); Read(kbd,tecla);
      tecla := UpCase(tecla);
    Until (tecla = 'S');
  SalvaParametros;
  If controle = 1 then EXIT;
  Case tecnica of
     'V'  : SalvaCiclos;
     'A'  : SalvaCrono;
```

```
        'P'  : SalvaCrono;
      end;
    End;


Overlay Procedure Disco;


  Type RegCiclo = Record
                    Dados : TipoVet;
                  end;
       RegCrono = Record
                    Cronod : Ponteiro2;
                  end;
       RegmV    = Record
                    mVd : TipoVet;
                  end;
       RegParametros = Record
                         unidadeP    : String[2];          { CA e V }
                         AmostraP,                         { T }
                         OperadorP   : String[20];         { T }
                         DataP       : String8;            { T }
                         tecnicaP    : Char;               { T }
                         fatorttP    : Real;               { T }
                         ganhoP      : Byte;               { T }
                         intervaloP;                       { T }
                         tpoP,                             { T }
                         tpoaP,                          { CA e CP }
                         tpodP,                          { CA e CP }
                         tpototP,                        { CA e CP }
                         npulsosP,                         { CA }
                         ncP,                              { V }
                         vmaxP,                            { CP }
                         vminP,                            { CP }
                         viniP,                          { CA e V }
                         vfiP,                           { CA e V }
                         correnteP   : Integer;          { CA e V }
                         veloP,                            { V }
                         incrP       : Real;               { V }
                         primeiroP,                        { V }
                         ultimoP,                          { V }
                         passoP      : Integer;            { V }
                         tempoptoP,                        { CC }
                         tempoP      : Real;             { CA e CP }
                         incrementoP,                      { CC }
                         nqP,                              { CC }
                         iiniP,                            { CP }
                         ifiP  : Integer;                  { CP }
                       end;


  var    Parametros    : RegParametros;
         ArqParametros : file of RegParametros;
         Onda          : RegmV;
         ArqOnda       : file of RegmV;
```



```
         nome,drv      : String[10];
         tecla,drive   : Char;
         ASCII         : Boolean;
         i, ciclo,
         controle      : Integer;


procedure LeCiclos;

    var UmCiclo       : RegCiclo;
        ArqCiclo      : file of RegCiclo;

    begin
      Assign(ArqCiclo,nome + '.DAT');
      {$I-}
      Reset(ArqCiclo);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Arquivo de dados nao encontrado');
          controle := 1;
          Close(ArqCiclo);
          EXIT;
        end;
      ciclo := primeiro;
      Repeat
        {$I-}
        Read(ArqCiclo,UmCiclo);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Problemas na leitura');
            controle := 1;
            Close(ArqCiclo);
            EXIT;
          end;
        Case ciclo of
          1..20   : For i := 0 to 1600 do D1^[i,ciclo] := UmCiclo.Dados[i];
          21..40  : For i := 0 to 1600 do D2^[i,ciclo-20] := UmCiclo.Dados[i];
          41..60  : For i := 0 to 1600 do D3^[i,ciclo-40] := UmCiclo.Dados[i];
          end; { case }
        ciclo := ciclo + passo;
      Until ( ciclo > ultimo);
      Close (ArqCiclo);
      Assign(ArqOnda,nome + '.VOL');     { recupera a matriz mV }
      {$I-}
      Reset(ArqOnda);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Arquivo de dados nao encontrado');
          controle := 1;
          Close(ArqOnda);
          EXIT;
        end;
```

```
      {$I-}
      Read(ArqOnda,Onda);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas na leitura');
          controle := 1;
          Close(ArqOnda);
          EXIT;
        end;
      With Onda do mV^ := mVd;
      Close(ArqOnda);
    end;


procedure LeCrono;

  var  UmCrono       : RegCrono;
       ArqCrono      : file of RegCrono;
       ArqCronoASCII : file of Integer;
       i             : Integer;

  begin
    If ASCII then
      begin
        Assign(ArqCronoASCII,nome + '.DAT');
        {$I-}
        Reset(ArqCronoASCII);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Arquivo de dados nao encontrado');
            controle := 1;
            Close(ArqCronoASCII);
            EXIT;
          end;
        {$I-}
        For i := 0 to tpotot do Read(ArqCronoASCII,Crono^[i]);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Problemas na leitura');
            controle := 1;
            Close(ArqCronoASCII);
            EXIT;
          end;
        Close(ArqCronoASCII);
      end
    else
      begin
        Assign(ArqCrono,nome + '.DAT');
        {$I-}
        Reset(ArqCrono);
        {$I+}
        If IOResult <> 0 then
```



```
          begin
            Aviso('Arquivo de dados nao encontrado');
            controle := 1;
            Close(ArqCrono);
            EXIT;
          end;
        {$I-}
        Read(ArqCrono,UmCrono);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Problemas na leitura');
            controle := 1;
            Close(ArqCrono);
            EXIT;
          end;
        With UmCrono do Crono := Cronod;
        Close(ArqCrono);
      end;
  end;


procedure LeParametros;

  begin
    Assign(ArqParametros,nome + '.PAR');
    {$I-}
    Reset(ArqParametros);
    {$I+}
    If IOResult <> 0 then
      begin
        Aviso('Arquivo de parametros nao encontrado');
        controle := 1;
        Close(ArqParametros);
        EXIT;
      end;
    {$I-}
    Read(ArqParametros,Parametros);
    {$I+}
    If IOResult <> 0 then
      begin
        Aviso('Problemas na leitura');
        controle := 1;
        Close(ArqParametros);
        EXIT;
      end;
    With Parametros do
      begin
        unidade    := unidadeP;
        Amostra    := AmostraP;
        Operador   := OperadorP;
        Data       := DataP;
        tecnica    := tecnicaP;
        fatortt    := fatorttP;
        ganho      := ganhoP;
```

```
        intervalo  := intervaloP;
        tpo        := tpoP;
        tpoa       := tpoaP;
        tpod       := tpodP;
        tpotot     := tpototP;
        npulsos    := npulsosP;
        nc         := ncP;
        vmax       := vmaxP;
        vmin       := vminP;
        vini       := viniP;
        vfi        := vfiP;
        corrente   := correnteP;
        velo       := veloP;
        incr       := incrP;
        primeiro   := primeiroP;
        ultimo     := ultimoP;
        passo      := passoP;
        tempopto   := tempoptoP;
        tempo      := tempoP;
        incremento := incrementoP;
        nq         := nqP;
        iini       := iiniP;
        ifi        := ifiP;
      end;  { With }
    Close(ArqParametros);
  end;


Begin
  controle := 0;  ASCII := TRUE;
  janela(5,4,60,20);
  Repeat
    ClrScr;
    WriteLn('RECOLHE DADOS GRAVADOS EM DISCO',CR,LF);
    Write(' (ESC) - Abandona  '); Read(Kbd,tecla);
    If tecla = #27 then EXIT;
    GoToXY(1,WhereY);
    WriteLn('1) Recolhe em ASCII      2) Recolhe em binario   ');
    Repeat
      Read(Kbd,tecla); tecla := UpCase(tecla);
    Until tecla in ['1','2'];
    If tecla = '1' then ASCII := TRUE
    else ASCII := FALSE;
    GetDir(0,drv);
    Case drv[1] of
      'A'         : drive := 'B';
      'B','C','D' : drive := 'A';
    end; { case }
    WriteLn('Qual o nome do arquivo? (maximo 8 caracteres)'); Readln(nome);
    If nome[2] <> ':' then nome := drive + ':' + nome;
    If Pos('.',nome) > 0 then
      nome := Copy(nome,1,Pos('.',nome)-1);
    For i := 1 to Length(nome) do nome[i] := UpCase(nome[i]);
    WriteLn('Arquivo de dados: ',nome + '.DAT',LF);
    WriteLn('                  ',nome + '.VOL  (para Voltametria)',LF);
```



```
    WriteLn('Arquivo de parametros: ',nome + '.PAR',LF);
    WriteLn('Confirma? (S/N)'); Read(kbd,tecla);
  Until (UpCase(tecla) = 'S');
  LeParametros;
  If controle = 1 then EXIT;
  ClrScr;
  WriteLn('Amostra: ',Amostra,CR,LF,'Operador: ',Operador,LF);
  Write('Tipo de arquivo: ');
  Case tecnica of
    'V' : begin
          WriteLn('Voltametria');
          WriteLn('velocidade: ',velo:5:2,' mV/s');
          WriteLn('Escala de corrente: ',corrente,' ',unidade);
          WriteLn('numero de ciclos: ',nc);
          If nc > 1 then
            begin
              WriteLn('primeiro ciclo:   ',primeiro);
              WriteLn('ultimo ciclo:     ',ultimo);
              WriteLn('incremento:       ',passo);
            end;
          LeCiclos;
        end;
    'A' : begin
          WriteLn('Crono-Amperometria');
          WriteLn('Escala de corrente: ',corrente,' ',unidade);
          WriteLn('Tempo antes do pulso = ',(tpoa*fatortt):8:0);
          WriteLn('Tempo do pulso = ',(tpo*fatortt):8:0);
          WriteLn('Tempo apos o pulso = ',(tpod*fatortt):8:0);
          LeCrono;
        end;
    'P' : begin
          WriteLn('Crono-Potenciometria');
          WriteLn('Escala de corrente: ',corrente,' ',unidade);
          WriteLn('Tempo antes do pulso = ',(tpoa*fatortt):8:0);
          WriteLn('Tempo do pulso = ',(tpo*fatortt):8:0);
          WriteLn('Tempo apos o pulso = ',(tpod*fatortt):8:0);
          LeCrono;
        end;
    end;
  If controle = 1 then EXIT;
  WriteLn(CR,LF,'...Aperte uma tecla...');
  Repeat Until KeyPressed;
End;

Overlay Procedure Coleta;


  Var   tempodepulso,
        tempoant,
        tempopost,
        fatsel,
        fx, fy      :  Real;
        ad,partida  :  Integer;
        tecla       :  Char;
```

```
procedure Informacoes;

  var   d,m,a    :   String[2];
        dia,mes,
        ano      :   Byte;

  begin
    Write(' Operador: ');
    ReadLn(operador);
    Write(' Amostra: ');
    ReadLn(amostra);
    Date(dia,mes,ano);
    Str(dia:2,d); Str(mes:2,m); Str(ano:2,a);
    If d[1] = ' ' then d[1] := '0';
    If m[1] = ' ' then m[1] := '0';
    If a[1] = ' ' then a[1] := '0';
    data := d + '/' + m + '/' + a;
  end;

function x(vx : Integer) : Integer;

  begin
    x := Round(fx*(vx - partida));
  end;

function y(vy : Integer) : Integer;

  begin
    y := Round(fy*(2000.0 - vy));
  end;

procedure Voltametria;

  var   i,
        k, j,
        valorH,
        valorL          :   Integer;
        tecla,
        confirma        :   Char;
        xmV             :   Array[0..1600] of Integer;
        vm              :   Real;



  procedure MeioCiclo;

{  A variavel fatortt e' o tempo gasto para preencher cada posicao do vetor.
   A variavel tpo e' o numero de posicoes preenchidas no vetor.
   A variavel intervalo e' o nro. de dados lidos do ADC para preencher uma
 posicao do vetor.
   A variavel espera e' excedente de 125 de intervalo. Isto e' necessario para
 nao ocorrer "overflow" na variavel valorL.
   As constantes TA e TB foram obtidas por regressao linear com:
       y = ax + b
```



```
onde y e' igual ao tempo cronometrado da voltametria divido por tpo;
     x e' a soma de intervalo mais espera;
     a e' igual a TA e
     b e' igual a TB.                                                    }
    begin
      tpo := 1600;
      fatortt := tempo / tpo;
      If fatortt * velo < 1.0 then   { Menor incremento de 1 mV. }
        begin
          fatortt :=  1.0 / velo;
          tpo := Round(tempo / fatortt);
        end;
      If fatortt < TA + TB then
        begin
          fatortt := TA + TB;
          tpo := Round(tempo / fatortt);
        end;
      intervalo := Round((fatortt - TB)/TA);
      If intervalo > 125 then
        begin
          espera := intervalo - 125;
          intervalo := 125;
        end
        else espera := 0;
      incr := (vfi-vini)/tpo;              { incremento em mV }
      For i := 0 to tpo do
        mV^[i] := vini + Round(i * incr);
      Write('.Espere estabilizar e aperte uma tecla.');
      Repeat Until KeyPressed;
      Window(1,1,80,25);
      HiRes; Borda(329,4,631,126); GraphWindow(330,5,630,125);
      fx := 300.0 / (vfi - vini);  fy := 120.0 / (2 * 2000.0);
      For i := 0 to tpo do xmV[i] := x(mV^[i]);
      GoToXY(5,1); WriteLn('CONDICOES',LF,CR);
      WriteLn('Operador : ',operador,LF,CR,'Amostra : ',amostra);
      WriteLn('Data : ',data);
      WriteLn('Ei = ',vini,' mV',LF,CR,'Ef = ',vfi,' mV');
      WriteLn('ii = ',-corrente,' ',unidade);
      WriteLn('if = ',corrente,' ',unidade);
      WriteLn('Velocidade = ',velo:0:2,' mV/s');
      ad := 0;
      For i := 0 to tpo do
        begin
          DAC(mV^[i]);
          For k := 1 to espera do
            begin
              ADC(ad);
              valorH := valorH + Hi(ad);
              valorL := valorL + Lo(ad);
            end;
          valorH := 0; valorL := 0;
          For k := 1 to intervalo do
            begin
              ADC(ad);
              valorH := valorH + Hi(ad);
```

```
            valorL := valorL + Lo(ad);
          end;
        D1^[i,1] := Round((valorH*256.0 + valorL)/(intervalo)) - TC;
        Plot(xmV[i],y(D1^[i,1]),1);
      end;
      nc := 1;  ultimo := nc;
      Sound(1760); Delay(200); NoSound; DAC(0);
    end;


  procedure Ciclos;
  var j : Integer;

{  Observar comentarios sobre as variaveis no procedure MeioCiclo. }

    begin
      tpo := 1600;
      tempo := tempo*2.0;                    {ida e volta}
      fatortt := tempo / tpo;
      If fatortt * velo < 1.0 then   { Menor incremento de 1 mV. }
        begin
          fatortt :=  1.0 / velo;
          tpo := Round(tempo / fatortt);
        end;
      If fatortt < TA + TB then
        begin
          fatortt := TA + TB;
          tpo := Round(tempo / fatortt);
        end;
      intervalo := Round((fatortt - TB)/TA);
      If intervalo > 125 then
        begin
          espera := intervalo - 125;
          intervalo := 125;
        end
        else espera := 0;
      j := tpo div 2;
      incr := (1.0 * vfi-vini) / j;      { incremento em mV }
      For i := 0 to j do                 { ida }
        mV^[i] := vini + Round(i * incr);
      For i := j + 1 to tpo do           { volta }
        mV^[i] := vfi - Round((i - j) * incr);
      ad := 0;
      Write('.Espere estabilizar e aperte uma tecla.');
      Repeat Until KeyPressed;
      Window(1,1,80,25);
      HiRes; Borda(329,4,631,126); GraphWindow(330,5,630,125);
      fx := 300.0 / (vfi - vini);  fy := 120.0 / (2 * 2000.0);
      For i := 0 to tpo do xmV[i] := x(mV^[i]);
      GoToXY(5,1); WriteLn('CONDICOES',LF,CR);
      WriteLn('Operador : ',operador,LF,CR,'Amostra : ',amostra);
      WriteLn('Data : ',data);
      WriteLn('Ei = ',vini,' mV',LF,CR,'Ef = ',vfi,' mV');
      WriteLn('ii = ',-corrente,' ',unidade);
      WriteLn('if = ',corrente,' ',unidade);
```



```
      WriteLn('Velocidade = ',velo:0:2,' mV/s');
      WriteLn('Ciclos totais = ',nc);
      GoToXY(1,15); Write('Ciclo atual = ');
      j := 0;
      Repeat
        j := succ(j);
        GoToXY(15,15); Write(j);
        Case j of
          1..20 :  For i := 0 to tpo do
                     begin
                       DAC(mV^[i]);
                       For k := 1 to espera do
                         begin
                           ADC(ad);
                           valorH := valorH + Hi(ad);
                           valorL := valorL + Lo(ad);
                         end;
                       valorH := 0; valorL := 0;
                       For k := 1 to intervalo do
                         begin
                           ADC(ad);
                           valorH := valorH + Hi(ad);
                           valorL := valorL + Lo(ad);
                         end;
                       D1^[i,j] := Round((valorH*256.0 + valorL)
                                   /(intervalo)) - TC;
                       Plot(xmV[i],y(D1^[i,j]),1);
                     end;
          21..40 :  For i := 0 to tpo do
                     begin
                       DAC(mV^[i]);
                       For k := 1 to espera do
                         begin
                           ADC(ad);
                           valorH := valorH + Hi(ad);
                           valorL := valorL + Lo(ad);
                         end;
                       valorH := 0; valorL := 0;
                       For k := 1 to intervalo do
                         begin
                           ADC(ad);
                           valorH := valorH + Hi(ad);
                           valorL := valorL + Lo(ad);
                         end;
                       D2^[i,j] := Round((valorH*256.0 + valorL)
                                   /(intervalo)) - TC;
                       Plot(xmV[i],y(D2^[i,j]),1);
                     end;
          41..60 :  For i := 0 to tpo do
                     begin
                       DAC(mV^[i]);
                       For k := 1 to espera do
                         begin
                           ADC(ad);
                           valorH := valorH + Hi(ad);
```

```
                    valorL := valorL + Lo(ad);
                  end;
                valorH := 0; valorL := 0;
                For k := 1 to intervalo do
                  begin
                    ADC(ad);
                    valorH := valorH + Hi(ad);
                    valorL := valorL + Lo(ad);
                  end;
                D3^[i,j] := Round((valorH*256.0 + valorL)
                          /(intervalo)) - TC;
                Plot(xeV[i],y(D3^[i,j]),1);
              end;
        end;  { case }
    Until (KeyPressed or (j = nc));
    Sound(1760); Delay(200); NoSound;
    DAC(0); nc := j;  ultimo := nc;
  end;


begin
  Janela(36,3,76,22);
  Repeat
    ClrScr; DAC(0);
    GoToXY(10,1);
    Write('VOLTAMETRIA CICLICA');
    GoToXY(10,2);
    Write('========== =======');
    GoToXY(10,4); Write('<ESC> - Abandona');
    Read(Kbd,tecla); If tecla = #27 then EXIT;
    GoToXY(1,4); ClrEol;
    Informacoes;
    canal := 0;  ganho := 0;
    Write(' E  Inicial (mV) = '); ReadIn(CR,vini);
    Write(' E  Final   (mV) = '); ReadIn(CR,vfi);
    Write(' Escala de i = '); ReadIn(#8,corrente);
    GoToXY(23,WhereY); Write('Unidade = ',unidade);
    Repeat
      GoToXY(33,WhereY); Read(unidade);
      If (unidade[1] <> 'm') and (unidade[1] <> 'u') then Write(#7);
    Until (unidade[1] = 'm') or (unidade[1] = 'u');
    vunidade := 2; WriteLn;
    fEi := corrente/2000.0;            { fator de conversao E p/ i }
    va := abs(vini - vfi) / (1600.0 * (MAXINT * TA + TB));
    if va < 0.01 then va := 0.01;
    Repeat
      GoToXY(1,WhereY);
      Write(' Veloc. (min.=',va:0:2,' (mV/s) = '); ReadRe(#8,velo,8,2);
      If velo < va then
        begin
          velo := 0;
          Write(#7);
        end;
    Until velo >= va;
    WriteLn;
```



```
    WriteLn(' (M) - 1/2 ciclo');
    WriteLn(' (C) - 1 ou + ciclos ');
    Repeat
      Read(Kbd,tecla);
      tecla := UpCase(tecla);
    Until (tecla = 'M') or (tecla = 'C');
    If tecla = 'C' then
      begin
        Repeat
          GoToXY(1,WhereY);
          Write(' Nro. de ciclos = '); ReadIn(#8,nc);
          If (nc < 0) or (nc > 60) then
            begin
              nc := 0;
              Write(#7);
            end;
        Until ((nc > 0) and (nc <= 60));
        WriteLn;
      end;
    DAC(vini);
    WriteLn('   Confirma ? (S/N) ');
    Repeat
      Read(kbd,confirma); confirma := UpCase(confirma);
    Until confirma in ['N','S'];
  Until confirma = 'S';
  tempo := abs(vini-vfi)/velo;      { tempo em segundos p/ delta V }
  Seletor(canal); Volume(ganho); partida := vini;
  Case tecla of
    'M'  :  MeioCiclo;
    'C'  :  Ciclos;
  end; { case }
end;


procedure VariosPulsos(valorDAi,valorDAm,valorDAf : Integer);

  var  i,k,j,np,
       f0a1,
       f1a2, f2a3,
       fator,
       valorH,
       valorL      :  Integer;
       tBhorasIs   :  Real;
       st          : String[5];

  begin
    tBhorasIs := 8.0 * 3600.0 * 1;
    fator := 1 + trunc(tempo / tBhorasIs);      { fator p/ cada 8 horas }
    fatortt := (tBhorasIs * fator) / 32001.0; { 32000 ptos. p/ cada 8 hs }
    tpo := Round(tempodepulso / fatortt);
    tpoa := Round(tempoant / fatortt);     { converte tempo nro. de dados }
    tpod := Round(tempopost / fatortt);
    tpotot := tpo + tpoa + tpod;       { nro. de dados total da aquisicao }
    intervalo := Round((fatortt - TB)/TA);
    If intervalo <= 0 then intervalo := 1;
```

```
If intervalo > 125 then
  begin
    espera := intervalo - 125;  { limita em 125 leituras p/ a media }
    intervalo := 125;
  end
  else espera := 0;
Seletor(canal);
Volume(ganho);
f0a1 := tpoa + 1;
f1a2 := tpo + tpoa;
f2a3 := tpo + tpoa + 1;
ad := 0;
Write('.Espere estabilizar e aperte uma tecla.');
Repeat Until KeyPressed;
Window(1,1,80,25);
HiRes; Borda(329,4,631,126); GraphWindow(330,5,630,125);
fx := 300.0 / tpotot;  fy := 120.0 / (2 * 2000.0);
GoToXY(5,1); WriteLn('CONDICOES',LF,CR);
WriteLn('Operador : ',operador,LF,CR,'Amostra : ',amostra);
WriteLn('Data : ',data);
If tecnica = 'A' then
  begin
    WriteLn('Ei = ',vini,' mV',LF,CR,'Ef = ',vfi,' mV');
    WriteLn('ii = ',-corrente,' ',unidade);
    WriteLn('if = ',corrente,' ',unidade);
  end;
If tecnica = 'P' then
  begin
    WriteLn('ii = ',inii:0:2,unidade,LF,CR,'if = ',fii:0:2,unidade);
    WriteLn('Ei = ',-2000,' mV');
    WriteLn('Ef = ',+2000,' mV');
  end;
WriteLn('Tempo antes do pulso = ',tempoant:0:0,' s');
WriteLn('Periodo do pulso = ',tempodepulso+tempopost:0:0,' s');
WriteLn('Nro. de pulsos = ',npulsos);
Write('Tempo de Aquisicao = ');
WriteLn((npulsos * (tempodepulso + tempopost) + tempoant):0:0,' s');
j := -1; np := 0;
For i := 0 to tpoa do
  begin
    DAC(valorDAi);
    For k := 1 to espera do
      begin
        ADC(ad);
        valorH := valorH + Hi(ad);
        valorL := valorL + Lo(ad);
      end;
    valorH := 0; valorL := 0;
    For k := 1 to intervalo do
      begin
        ADC(ad);
        valorH := valorH + Hi(ad);
        valorL := valorL + Lo(ad);
      end;
    j := j + 1;
```



```
    Crono^[j] := Round((valorH*256.0 + valorL)/intervalo) - TC;
    Plot(x(j),y(Crono^[j]),1);
  end;
Repeat
  np := np + 1;
  GoToXY(1,WhereY); Write('Pulso Atual = ',np);
  For i := f0a1 to f1a2 do
    begin
      DAC(valorDAm);
      For k := 1 to espera do
        begin
          ADC(ad);
          valorH := valorH + Hi(ad);
          valorL := valorL + Lo(ad);
        end;
      valorH := 0; valorL := 0;
      For k := 1 to intervalo do
        begin
          ADC(ad);
          valorH := valorH + Hi(ad);
          valorL := valorL + Lo(ad);
        end;
      j := j + 1;
      Crono^[j] := Round((valorH*256.0 + valorL)/intervalo) - TC;
      Plot(x(j),y(Crono^[j]),1);
    end;
  For i := f2a3 to tpotot do
    begin
      DAC(valorDAf);
      For k := 1 to espera do
        begin
          ADC(ad);
          valorH := valorH + Hi(ad);
          valorL := valorL + Lo(ad);
        end;
      valorH := 0; valorL := 0;
      For k := 1 to intervalo do
        begin
          ADC(ad);
          valorH := valorH + Hi(ad);
          valorL := valorL + Lo(ad);
        end;
      j := j + 1;
      Crono^[j] := Round((valorH*256.0 + valorL)/intervalo) - TC;
      Plot(x(j),y(Crono^[j]),1);
    end;
  If (j >= 32000) or (np = npulsos) or KeyPressed then
    begin
      Str(np,st);
      Salva(st,j);
      j := -1;
      partida := 0;
    end;
ClearScreen;
If j > -1 then partida := j;
```

```
    If np = 1 then fx := 300.0 / (tpo + tpod);
    Until (np = npulsos) or KeyPressed;
    npulsos := np;
    DAC(valorDAi);
    Sound(1760); Delay(200); NoSound;
  end;

procedure Pulso(valorDAi,valorDAm,valorDAf : Integer);

  var  i,k,f0a1,
       f1a2, f2a3,
       fator,
       valorH,
       valorL       :   Integer;
       tsel         :   Real;

  begin
    tsel := fatsel + 1.0;                 { valores nos procedures CA e CP }
    fator := 1 + trunc(tempo / tsel);     { fator p/ cada fatsel }
    fatortt := (tsel * fator) / 32001.0; { 32000 ptos. p/ cada fatsel }
    tpo := Round(tempodepulso / fatortt);
    tpoa := Round(tempoant / fatortt);     { converte tempo nro. de dados }
    tpod := Round(tempopost / fatortt);
    tpotot := tpo + tpoa + tpod;       { nro. de dados total da aquisicao }
    intervalo := Round((fatortt - TB)/TA);
    If intervalo <= 0 then intervalo := 1;
    If intervalo > 125 then
        begin
          espera := intervalo - 125;  { limita em 125 leituras p/ a media }
          intervalo := 125;
        end
        else espera := 0;
    Seletor(canal);
    Volume(ganho);
    f0a1 := tpoa + 1;
    f1a2 := tpo + tpoa;
    f2a3 := tpo + tpoa + 1;
    ad := 0;
    Write('.Espere estabilizar e aperte uma tecla.');
    Repeat Until KeyPressed;
    Window(1,1,80,25);
    HiRes; Borda(329,4,631,126); GraphWindow(330,5,630,125);
    fx := 300.0 / tpotot;  fy := 120.0 / (2 * 2000.0);
    GoToXY(5,1); WriteLn('CONDICOES',LF,CR);
    WriteLn('Operador : ',operador,LF,CR,'Amostra : ',amostra);
    WriteLn('Data : ',data);
    If tecnica = 'A' then
      begin
        WriteLn('Ei = ',vini,' mV',LF,CR,'Ef = ',vfi,' mV');
        WriteLn('ii = ',-corrente,' ',unidade);
        WriteLn('if = ',corrente,' ',unidade);
      end;
    If tecnica = 'P' then
      begin
        WriteLn('ii = ',inii:0:2,unidade,LF,CR,'if = ',fii:0:2,unidade);
```



```
        WriteLn('Ei = ',-2000,' mV');
        WriteLn('Ef = ',+2000,' mV');
      end;
    WriteLn('Tempo antes do pulso = ',tempoant:0:0,' s');
    WriteLn('Tempo do pulso = ',tempodepulso:0:0,' s');
    WriteLn('Tempo apos o pulso = ',tempopost:0:0,' s');
    For i := 0 to tpoa do
      begin
        DAC(valorDAi);
        For k := 1 to espera do
          begin
            ADC(ad);
            valorH := valorH + Hi(ad);
            valorL := valorL + Lo(ad);
          end;
        valorH := 0; valorL := 0;
        For k := 1 to intervalo do
          begin
            ADC(ad);
            valorH := valorH + Hi(ad);
            valorL := valorL + Lo(ad);
          end;
        Crono^[i] := Round((valorH*256.0 + valorL)/intervalo) - TC;
        Plot(x(i),y(Crono^[i]),1);
      end;
    For i := f0a1 to f1a2 do
      begin
        DAC(valorDAm);
        For k := 1 to espera do
          begin
            ADC(ad);
            valorH := valorH + Hi(ad);
            valorL := valorL + Lo(ad);
          end;
        valorH := 0; valorL := 0;
        For k := 1 to intervalo do
          begin
            ADC(ad);
            valorH := valorH + Hi(ad);
            valorL := valorL + Lo(ad);
          end;
        Crono^[i] := Round((valorH*256.0 + valorL)/intervalo) - TC;
        Plot(x(i),y(Crono^[i]),1);
      end;
    For i := f2a3 to tpotot do
      begin
        DAC(valorDAf);
        For k := 1 to espera do
          begin
            ADC(ad);
            valorH := valorH + Hi(ad);
            valorL := valorL + Lo(ad);
          end;
        valorH := 0; valorL := 0;
        For k := 1 to intervalo do
```

```
          begin
            ADC(ad);
            valorH := valorH + Hi(ad);
            valorL := valorL + Lo(ad);
          end;
        Crono^[i] := Round((valorH*256.0 + valorL)/intervalo) - TC;
        Plot(x(i),y(Crono^[i]),1);
      end;
    DAC(valorDAi);
    Sound(1760); Delay(200); NoSound;
  end;


procedure Amperometria;

  var confirma        : Char;

  begin
    Janela(36,3,76,22);
    Repeat
      ClrScr;  DAC(0);
      GoToXY(10,1);
      Write('CRONO-AMPEROMETRIA');
      GoToXY(10,2);
      Write('==================');
      GoToXY(10,4); Write('<ESC> - Abandona');
      Read(Kbd,confirma); If confirma = #27 then EXIT;
      GoToXY(1,4); ClrEol;
      Informacoes;
      canal := 0;  ganho := 0; npulsos := 1;
      Write(' E  Inicial (mV) = '); ReadIn(CR,vini);
      Write(' E  Final (mV)   = '); ReadIn(CR,vfi);
      Write(' Escala de i = '); ReadIn(#8,corrente);
      GoToXY(23,WhereY); Write('Unidade = ',unidade);
      Repeat
        GoToXY(33,WhereY); Read(unidade);
        If (unidade[1] <> 'm') and (unidade[1] <> 'u') then Write(#7);
      Until (unidade[1] = 'm') or (unidade[1] = 'u');
      vunidade := 2; WriteLn;
      fEi := corrente/2000.0;           ( fator de conversao E p/ i )
      WriteLn(' Tempo antes do pulso :');
      Write(' h = ',h1,'     min = ',m1,'    seg = ',s1);
      GoToXY(6,WhereY); ReadIn(#8,h1);
      GoToXY(18,WhereY); ReadIn(#8,m1);
      GoToXY(29,WhereY);ReadIn(CR,s1);
      tempoant := h1*3600.0 + m1*60.0 + s1;
      WriteLn(' Tempo do pulso :');
      Write(' h = ',h2,'     min = ',m2,'    seg = ',s2);
      GoToXY(6,WhereY); ReadIn(#8,h2);
      GoToXY(18,WhereY); ReadIn(#8,m2);
      GoToXY(29,WhereY);ReadIn(CR,s2);
      tempodepulso := h2*3600.0 + m2*60.0 + s2;
      WriteLn(' Tempo apos o pulso :');
      Write(' h = ',h3,'     min = ',m3,'    seg = ',s3);
      GoToXY(6,WhereY); ReadIn(#8,h3);
```



```
      GoToXY(18,WhereY); ReadIn(#8,m3);
      GoToXY(29,WhereY);ReadIn(CR,s3);
      tempopost := h3*3600.0 + m3*60.0 + s3;
      Write(' Nro. de pulsos = '); ReadIn(CR,npulsos);
      If npulsos < 1 then npulsos := 1;
      DAC(vini);
      WriteLn('      Confirma ? (S/N) ');
      Repeat
        Read(kbd,confirma); confirma := UpCase(confirma);
      Until confirma in ['N','S'];
    Until confirma = 'S';
    tempo := tempodepulso + tempoant + tempopost;
    If tempo = 0.0 then tempo := 1.0;
    partida := 0;
    If npulsos = 1 then
      begin
        If (tempo <= 2.0 * 60.0) then fatsel := 2.0 * 60.0;      ( 2 min )
        If (tempo > 2.0 * 60.0) and (tempo <= 10.0 * 60.0) then
          fatsel := 10.0 * 60;                                   ( 10 min )
        If (tempo > 10.0 * 60.0) and (tempo <= 1.0 * 3600.0) then
          fatsel := 1.0 * 3600.0;                                ( 1 h )
        If (tempo > 1.0 * 3600.0) and (tempo <= 2.0 * 3600.0) then
          fatsel := 2.0 * 3600.0;                                ( 2 h )
        If (tempo > 2.0 * 3600.0) then
          fatsel := 8.0 * 3600.0;                                ( 8 h )
        Pulso(vini,vfi,vini);
      end
    else VariosPulsos(vini,vfi,vini);
  end;


procedure Potenciometria;

  var confirma        : Char;

  begin
    Janela(36,3,76,22);
    Repeat
      ClrScr;  DAC(0);
      GoToXY(10,1);
      Write('CRONO-POTENCIOMETRIA');
      GoToXY(10,2);
      Write('====================');
      GoToXY(10,4); Write('<ESC> - Abandona');
      Read(Kbd,confirma); If confirma = #27 then EXIT;
      GoToXY(1,4); ClrEol;
      Informacoes;
      canal := 0;  ganho := 0;
      Write(' Escala de i = '); ReadIn(#8,corrente);
      GoToXY(23,WhereY); Write('Unidade = ',unidade);
      Repeat
        GoToXY(33,WhereY); Read(unidade);
        If (unidade[1] <> 'm') and (unidade[1] <> 'u') then Write(#7);
      Until (unidade[1] = 'm') or (unidade[1] = 'u');
      vunidade := 2 ; WriteLn;
```

```
      Write(' i  Inicial = '); ReadRe(CR,inii,8,2);
      Write(' i  Final   = '); ReadRe(CR,fii,8,2);
      fEi := corrente/2000.0;            { fator de conversao E p/ i }
      iini := Round(inii/fEi);
      ifi := Round(fii/fEi);
      WriteLn(' Tempo antes do pulso :');
      Write(' h = ',h1,'     min = ',m1,'    seg = ',s1);
      GoToXY(6,WhereY); ReadIn(#8,h1);
      GoToXY(18,WhereY); ReadIn(#8,m1);
      GoToXY(29,WhereY);ReadIn(CR,s1);
      tempoant := h1*3600.0 + m1*60.0 + s1;
      WriteLn(' Tempo do pulso :');
      Write(' h = ',h2,'     min = ',m2,'    seg = ',s2);
      GoToXY(6,WhereY); ReadIn(#8,h2);
      GoToXY(18,WhereY); ReadIn(#8,m2);
      GoToXY(29,WhereY);ReadIn(CR,s2);
      tempodepulso := h2*3600.0 + m2*60.0 + s2;
      WriteLn(' Tempo apos o pulso :');
      Write(' h = ',h3,'     min = ',m3,'    seg = ',s3);
      GoToXY(6,WhereY); ReadIn(#8,h3);
      GoToXY(18,WhereY); ReadIn(#8,m3);
      GoToXY(29,WhereY);ReadIn(CR,s3);
      tempopost := h3*3600.0 + m3*60.0 + s3;
      WriteLn('      Confirma? (S/N) ');
      Repeat
        Read(kbd,confirma); confirma := UpCase(confirma);
      Until confirma in ['N','S'];
    Until confirma = 'S';
    tempo := tempodepulso + tempoant + tempopost;
    If tempo = 0.0 then tempo := 1.0;
    partida := 0;
    If (tempo <= 2.0 * 60.0) then fatsel := 2.0 * 60.0;      { 2 min }
    If (tempo > 2.0 * 60.0) and (tempo <= 10.0 * 60.0) then
      fatsel := 10.0 * 60;                                   { 10 min }
    If (tempo > 10.0 * 60.0) and (tempo <= 1.0 * 3600.0) then
      fatsel := 1.0 * 3600.0;                                { 1 h }
    If (tempo > 1.0 * 3600.0) and (tempo <= 2.0 * 3600.0) then
      fatsel := 2.0 * 3600.0;                                { 2 h }
    If (tempo > 2.0 * 3600.0) then
      fatsel := 8.0 * 3600.0;                                { 8 h }
    Pulso(0,iini,ifi);
    For i := 0 to tpotot do
      Crono^[i] := multpot * Crono^[i];
  end;


Begin
  Repeat
    Janela(26,6,56,20);
    GoToXY(8,1);
    Write('COLETA DE DADOS');
    GoToXY(8,2);
    Write('====== == =====');
    GoToXY(4,4);
    WriteLn('Modo :');
```



```
    GoToXY(1,6);
    WriteLn(' (V) - Voltametria Ciclica');
    WriteLn(' (A) - Crono-Amperometria');
    WriteLn(' (P) - Crono-Potenciometria');
    GoToXY(4,12);
    Write('<ESC> - Menu Inicial');
    Read(Kbd,tecla);
    tecla := UpCase(tecla);
    If (tecla <> #27) then tecnica := tecla;
    Case tecla of
      'V' : Voltametria;
      'A' : Amperometria;
      'P' : Potenciometria;
    end;  { case }
  Until (tecla = #27) or (tecla = 'V') or (tecla = 'A') or (tecla = 'P');
End;

Overlay Procedure Resultados;

  Var  cs                : Array[1..18] of Integer;
       unid              : String[2];
       tecla, opcao      : Char;
       pilha,
       xmin, ymin,
       xmax, ymax,
       iniciox,fimx,
       inicioy,fimy,
       passax,
       passay,
       divx, divy,
       posx,posy,
       x,y,i,j,
       i1,i2,e1,e2,
       nciclo, mult,
       controlaIntegral,
       controlaVoltam,
       controlaAmpero,
       controlaPotenc    :  Integer;
       inicioqx,fimqx,
       inicioqy,fimqy,
       Qmax, Qmin,
       epa,epc,
       ipa,ipc,
       fx, fy, td , m    :  Real;


  function px(x : Integer) : Integer;

    begin
      px := Round(fx*(x - xmin));
    end;


  function py(y : Integer) : Integer;
```

```
  begin
    py := Round(fy*(ymax - y));
  end;


procedure Linha(x1,y1,x2,y2 : Integer);

  begin
    Draw(px(x1),py(y1),px(x2),py(y2),1);
  end;


procedure Condicoes;

  begin
    GoToXY(posx,posy);
    If ((tecnica = 'V') or (tecnica = 'A')) and (tipo <> 'Q') then
      begin
        If (corrente = 2000) or (corrente = 1000) then
          begin
            Write(' (max=',(imax*fEi):0:0,')');
            WriteLn(' (min=',(imin*fEi):0:0,')');
          end;
        If corrente = 200 then
          begin
            Write(' (max=',(imax*fEi):0:1,')');
            WriteLn(' (min=',(imin*fEi):0:1,')');
          end;
        If corrente = 20 then
          begin
            Write(' (max=',(imax*fEi):0:2,')');
            WriteLn(' (min=',(imin*fEi):0:2,')');
          end;
        If corrente = 2 then
          begin
            Write(' (max=',(imax*fEi):0:3,')');
            WriteLn(' (min=',(imin*fEi):0:3,')');
          end;
        If tecnica = 'V' then
          begin
            Write('Veloc. de varredura = ',velo:5:2,' mV/s');
            If m <> 0 then
              begin
                GoToXY(38,4); ClrEol; GoToXY(38,4);
                Write('(Epa = ',Round(epa));
                Write('  Epc = ',Round(epc));
                Write('  E0 = ',Round(epa - ipa/m),')');
              end;
          end
        else
          begin
            Write('Tempo de aquisicao = ');
            Write((((tpo+tpod)*npulsos + tpoa)*fatortt):0:0,' s');
            Write(' (E inicial = ',vini,'mV)');
            Write(' (E final = ',vfi,'mV)');
```



```
          end;
      end;
    If tipo = 'Q' then
      begin
        If tecnica = 'A' then
          begin
            WriteLn(' (carga total = ',(Q^[nq]*fEi):8,')');
            Write('Tempo de aquisicao = ');
            Write((((tpo+tpod)*npulsos + tpoa)*fatortt):0:0,' s');
            Write(' (E inicial = ',vini,'mV) (E final = ',vfi,'mV)');
          end;
        If tecnica = 'V' then
          begin
            WriteLn(' (carga total = ',(Q^[nq]*fEi):8,')');
            Write('Veloc. de varredura = ',velo:5:2,' mV/s');
          end;
      end;
    If tecnica = 'P' then
      begin
        WriteLn(' (max=',vmax,') (min=',vmin,')');
        Write('Tempo de aquisicao = ', tempo:0:0,' s');
        If (corrente = 2000) or (corrente = 1000) then
          begin
            Write(' (i inicial =',(iini*fEi):0:0,unidade,')');
            Write(' (i final =',(ifi*fEi):0:0,unidade,')');
          end;
        If corrente = 200 then
          begin
            Write(' (i inicial = ',(iini*fEi):0:1,unidade,')');
            Write(' (i final = ',(ifi*fEi):0:1,unidade,')');
          end;
        If corrente = 20 then
          begin
            Write(' (i inicial = ',(iini*fEi):0:2,unidade,')');
            Write(' (i final = ',(ifi*fEi):0:2,unidade,')');
          end;
        If corrente = 2 then
          begin
            Write(' (i inicial = ',(iini*fEi):0:3,unidade,')');
            Write(' (i final = ',(ifi*fEi):0:3,unidade,')');
          end;
      end;
  end;


procedure Cursor;

  var     ct        :  Array [1..18] of Integer;
          poscx,
          poscy,
          incr      :  Integer;
          mov       :  Char;


  const   ESC = #27;
```

```
procedure novapos;

  begin
    GetPic(ct,poscx - 6,poscy - 3,poscx + 6,poscy + 3);
  end;

procedure antpos;

  begin
    PutPic(ct,poscx - 6,poscy + 3);
  end;

procedure novocs;

  begin
    PutPic(cs,poscx - 6,poscy + 3);
  end;


begin
  incr := 1;
  poscx := 300;
  poscy := 100;
  novapos;
  novocs;
  epa := 0.0; epc := 0.0;  m := 0.0;
  Repeat
    Read(kbd,mov);
    If (mov = ESC) and KeyPressed then
      Read(kbd,mov);
    Case mov of
      #59  :  begin
                epa := xmin + poscx / fx;
                ipa := ymax - (poscy + 3) / fy;
                Sound(2000); Delay(50); Sound(1000); Delay(50); NoSound;
              end;
      #60  :  begin
                epc := xmin + poscx / fx;
                ipc := ymax - (poscy - 3) / fy;
                Sound(3000); Delay(50); Sound(2000); Delay(50); NoSound;
              end;
      #61  :  begin
                If (epa <> 0.0) and (epc <> 0.0) then
                  begin
                    m := (ipa - ipc)/(epa - epc);
                    Sound(1760); Delay(50); Sound(880);
                    Delay(50); NoSound;
                    Condicoes;
                  end;
              end;
      #65  :  antpos;
      #66  :  begin
                antpos;
```



```
                novocs;
              end;
      #67  :  incr := 1;
      #68  :  incr := 5;
      #75  :  begin
                antpos;
                poscx := poscx - incr;
                If poscx < 6 then
                  begin
                    Sound(880); Delay(100); NoSound; poscx := 6;
                  end;
                novapos;
                novocs;
              end;
      #77  :  begin
                antpos;
                poscx := poscx + incr;
                If poscx > 496 then
                  begin
                    Sound(880); Delay(100); NoSound; poscx := 496;
                  end;
                novapos;
                novocs;
              end;
      #72  :  begin
                antpos;
                poscy := poscy - incr;
                If poscy < 3 then
                  begin
                    Sound(880); Delay(100); NoSound; poscy := 3;
                  end;
                novapos;
                novocs;
              end;
      #80  :  begin
                antpos;
                poscy := poscy + incr;
                If poscy > 147 then
                  begin
                    Sound(880); Delay(100); NoSound; poscy := 147;
                  end;
                novapos;
                novocs;
              end;
      #92  :  incr := 10;
      #93  :  incr := 20;
      end; { case }
    Until (mov = ESC);
  end;

procedure EixoVolt(xc,xu,yc,yu : String2);

  procedure EixoY;

    var  vy  :  Real;
```

```
    begin
      If (tecnica = 'V') or (tecnica = 'A') then
        begin
          If (corrente = 2000) or (corrente = 1000) then
            begin
              vy := ymin * fEi;
              Write('Eixo ',yc,' : ',vy:0:0);
              vy := ymax * fEi;
              Write(' a ',vy:0:0,' ',yu);
              vy := divy * fEi;
              Write(' com divisoes de ',vy:0:0,' ',yu);
            end;
          If corrente = 200 then
            begin
              vy := ymin * fEi;
              Write('Eixo ',yc,' : ',vy:0:1);
              vy := ymax * fEi;
              Write(' a ',vy:0:1,' ',yu);
              vy := divy * fEi;
              Write(' com divisoes de ',vy:0:1,' ',yu);
            end;
          If corrente = 20 then
            begin
              vy := ymin * fEi;
              Write('Eixo ',yc,' : ',vy:0:2);
              vy := ymax * fEi;
              Write(' a ',vy:0:2,' ',yu);
              vy := divy * fEi;
              Write(' com divisoes de ',vy:0:2,' ',yu);
            end;
          If corrente = 2 then
            begin
              vy := ymin * fEi;
              Write('Eixo ',yc,' : ',vy:0:3);
              vy := ymax * fEi;
              Write(' a ',vy:0:3,' ',yu);
              vy := divy * fEi;
              Write(' com divisoes de ',vy:0:3,' ',yu);
            end;
        end
      else
          begin
            Write('Eixo ',yc,' : ',ymin);
            Write(' a ',ymax,' ',yu);
            Write(' com divisoes de ',divy,' ',yu);
          end;
    end;

begin
  tipo := yc;
  Window(1,1,80,25);
  HiRes; ClearScreen;
  Borda(6,43,514,197);
  GraphWindow(10,45,510,195);
```



```
  Draw(294,100,306,100,1);
  Draw(300,97,300,103,1);
  GetPic(cs,294,97,306,103);
  ClearScreen;
  GoToXY(1,1);
  Write('Amostra : ',amostra);
  GoToXY(29,1);
  Write('Operador : ',operador);
  GoToXY(64,1);
  WriteLn('Data : ',data);
  divx := ((xmax - xmin) div 200) * 10;  { calcula as divisoes }
  If divx = 0 then divx := 1;
  divy := ((ymax - ymin) div 200) * 10;
  If divy = 0 then divy := 1;
  Write('Eixo ',xc,' : ',xmin,' a ',xmax);
  WriteLn(' ',xu,' com divisoes de ',divx,' ',xu);
  EixoY;
  posx := WhereX; posy := WhereY;
  Condicoes;
  passay := 0;
  If xmin > 0 then passay := xmin;
  If xmax < 0 then passay := xmax;
  passax := 0;
  If ymin > 0 then passax := ymin;
  If ymax < 0 then passax := ymax;
  Linha(xmin,passax,xmax,passax);                    { Eixo X }
  Draw(px(xmax),py(passax),px(xmax)-7,py(passax)-2,1);
  Draw(px(xmax),py(passax),px(xmax)-7,py(passax)+2,1);
  Linha(passay,ymin,passay,ymax);                    { Eixo Y }
  Draw(px(passay),py(ymax),px(passay)-5,py(ymax)+3,1);
  Draw(px(passay),py(ymax),px(passay)+5,py(ymax)+3,1);
  x := Round((px(passay)+10)/639*79)+1;              { escreve yc }
  GoToXY(x,5);Write(yc);
  y := Round((py(passax)+45)/199*23)+1;              { escreve xc }
  GoToXY(66,y);Write(xc);
  x := passay;                    { desenha as divisoes do eixo x }
  While ( x < (xmax - divx/2) ) do
    begin
      Draw(px(x),py(passax)+2,px(x),py(passax)-2,1);
      x := x + divx;
    end;
  x := passay;
  While ( x > xmin ) do
    begin
      Draw(px(x),py(passax)+2,px(x),py(passax)-2,1);
      x := x - divx;
    end;
  y := passax;                    { desenha as divisoes do eixo y }
  While ( y > ymin ) do
    begin
      Draw(px(passay)-4,py(y),px(passay)+4,py(y),1);
      y := y - divy;
    end;
  y := passax;
  While ( y < ymax - divy/2 ) do
```

```
      begin
        Draw(px(passay)-4,py(y),px(passay)+4,py(y),1);
        y := y + divy;
      end;
  end;


procedure Limites;

  begin
    If xmin > xmax then
      begin
        pilha := xmin;
        xmin := xmax;
        xmax := pilha;
    end;
    If ymin > ymax then
      begin
        pilha := ymin;
        ymin := ymax;
        ymax := pilha;
      end;
    fx := 500.0/(xmax - xmin);
    fy := 150.0/(ymax - ymin);
  end;


procedure Escalas(xc,yc : Char);

  var  iny, fiy  :  Real;

  begin
    GoToXY(1,6);
    WriteLn('Escalas :');
    WriteLn;
    Write(' Eixo ',xc,' - Inicio = '); ReadIn(CR,iniciox);
    Write('          Fim    = '); ReadIn(CR,fimx);
    If (tecnica = 'A') or (tecnica = 'V') then
      begin
        iny := inicioy*fEi; fiy := fimy*fEi;
        Write(' Eixo ',yc,' - Inicio = '); ReadRe(CR,iny,8,2);
        inicioy := Round(iny/fEi);
        Write('          Fim    = '); ReadRe(CR,fiy,8,2);
        fimy := Round(fiy/fEi);
      end
    else
      begin
        Write(' Eixo ',yc,' - Inicio = '); ReadIn(CR,inicioy);
        Write('          Fim    = '); ReadIn(CR,fimy);
      end;
  end;


procedure EscalasQ;
```



```
  var  iny, fiy  :  Real;

  begin
    GoToXY(1,7);
    WriteLn('Escalas :');
    WriteLn;
    If tecnica = 'A' then
      begin
        Write(' Eixo t - Inicio = '); ReadRe(CR,inicioqx,8,2);
        Write('          Fim    = '); ReadRe(CR,fimqx,8,2);
      end
    else
      begin
        Write(' Eixo E - Inicio = '); ReadIn(CR,iniciox);
        Write('          Fim    = '); ReadIn(CR,fimx);
      end;
    iny := inicioqy*fEi;  fiy := fimqy*fEi;
    Write(' Eixo Q - Inicio = '); ReadRe(CR,iny,8,2);
    inicioqy := iny/fEi;
    Write('          Fim    = '); ReadRe(CR,fiy,8,2);
    fimqy := fiy/fEi;
  end;


procedure IxE;

  var  pri, ult, inc  :  Integer;

  begin
    Janela(46,5,76,19);
    WriteLn;
    epa := 0.0; epc := 0.0;  m := 0.0;
    pri := primeiro; ult := ultimo; inc := passo;
    Write(' Nro. do 1o. ciclo = '); ReadIn(CR,pri);
    Write(' Nro. do ultimo ciclo = '); ReadIn(CR,ult);
    If ult > ultimo then ult := ultimo;
    If ult < primeiro then ult := primeiro;
    Write(' Incremento = '); ReadIn(CR,inc);
    If inc <= 0 then inc := 1;
    If controlaVoltam = 0 then
      begin
        imax := -2000;  imin := 2000;
        j := pri;
        Repeat
          Case j of
            1..20    :  For i := 0 to tpo do
                          begin
                            If imax < D1^[i,j] then imax := D1^[i,j];
                            If imin > D1^[i,j] then imin := D1^[i,j];
                          end;
            21..40   :  For i := 0 to tpo do
                          begin
                            If imax < D2^[i,j] then imax := D2^[i,j];
                            If imin > D2^[i,j] then imin := D2^[i,j];
                          end;
```

```
          41..60   :  For i := 0 to tpo do
                      begin
                        If imax < D3^[i,j-40] then imax := D3^[i,j-40];
                        If imin > D3^[i,j-40] then imin := D3^[i,j-40];
                      end;
        end; { case }
       j := j + inc;
      Until (j > ult);
      controlaVoltaa := 1;
    end;
  iniciox := vini; fimx := vfi;
  inicioy := imin - abs(imin div 10);
  fimy := imax + abs(imax div 10);
  If fimy = inicioy then fimy := fimy + 10;
  Escalas('E','i');
  xmin := iniciox; xmax := fimx; ymin := inicioy; ymax := fimy;
  Window(1,1,80,25);
  Limites;
  EixoVolt('E','mV','i',unidade);
  j := pri;
  Repeat
    Case j of
      1..20   :  For i := 1 to tpo do
                   Linha(mV^[i-1], D1^[i-1,j], mV^[i], D1^[i,j]);
      21..40  :  For i := 1 to tpo do
                   Linha(mV^[i-1], D2^[i-1,j-20], mV^[i], D2^[i,j-20]);
      41..60  :  For i := 1 to tpo do
                   Linha(mV^[i-1], D3^[i-1,j-40], mV^[i], D3^[i,j-40]);
    end;  { case }
    j := j + inc;
  Until (j > ult) or KeyPressed;
  Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
  Cursor;
  TextMode;
end;


procedure Ext;

  begin
    If controlaPotenc = 0 then
      begin
        vmax := Crono^[0];  vmin := Crono^[0];
        For i := 1 to tpotot do
          begin
            If vmax < Crono^[i] then vmax := Crono^[i];
            If vmin > Crono^[i] then vmin := Crono^[i];
          end;
        controlaPotenc := 1;
      end;
    iniciox := 0; fimx := round(tpotot * td);
    If (vmin > 0) then inicioy := 0
      else inicioy := vmin - abs(vmin div 10);
    If (vmax < 0) then fimy := 0
      else fimy := vmax + abs(vmax div 10);
```



```
    If fimy = inicioy then fimy := fimy + 10;
    Escalas('t','E');
    xmin := iniciox; xmax := fimx; ymin := inicioy; ymax := fimy;
    If (abs(xmax - xmin) < 330) and (xmax < 330) then
      begin
        If (abs(xmax - xmin) < 33) and (xmax < 33) then
          begin
            mult := 1000;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end
        else
          begin
            mult := 100;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end;
      end
    else mult := 1;
    Window(1,1,80,25);
    Limites;
    If mult = 1000 then EixoVolt('t','ms','V','mV');
    If mult = 100 then EixoVolt('t','cs','V','mV');
    If mult = 1 then EixoVolt('t','s','V','mV');
    i := Round(iniciox / td) + 1;
    Repeat
      Linha(Round((i-1) * td * mult), Crono^[i-1],
            Round(i * td * mult), Crono^[i]);
      i := i + 1;
    Until (i > Round(fimx / td)) or KeyPressed;
    Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
    Repeat Until KeyPressed;
    TextMode;
  end;


procedure Ixt;

  begin
    If controlaAmpero = 0 then
      begin
        imax := Crono^[0];  imin := Crono^[0];
        For i := 1 to tpotot do
          begin
            If imax < Crono^[i] then imax := Crono^[i];
            If imin > Crono^[i] then imin := Crono^[i];
          end;
        controlaAmpero :=1;
      end;
    iniciox := 0; fimx := round(tpotot * td);
    If (imin > 0) then inicioy := 0
      else inicioy := imin - abs(imin div 10);
    If (imax < 0) then fimy := 0
      else fimy := imax + abs(imax div 10);
    If fimy = inicioy then fimy := fimy + 10;
```

```
    Escalas('t','i');
    xmin := iniciox; xmax := fimx; ymin := inicioy; ymax := fimy;
    If (abs(xmax - xmin) < 330) and (xmax < 330) then
      begin
        If (abs(xmax - xmin) < 33) and (xmax < 33) then
          begin
            mult := 1000;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end
        else
          begin
            mult := 100;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end;
      end
    else mult := 1;
    Window(1,1,80,25);
    Limites;
    If mult = 1000 then EixoVolt('t','ms','i',unidade);
    If mult = 100 then EixoVolt('t','cs','i',unidade);
    If mult = 1 then EixoVolt('t','s','i',unidade);
    i := Round(inicio / td) + 1;
    Repeat
      Linha(Round((i-1) * td * mult), Crono^[i-1],
            Round(i * td * mult), Crono^[i]);
      i := i + 1;
    Until (i > Round(fimx / td)) or KeyPressed;
    Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
    Repeat Until KeyPressed;
    TextMode;
  end;


procedure Integral;

  var  i, j   :  Integer;

  begin
    Janela(46,5,76,19);
    If tecnica = 'A' then
      begin
        incremento := 1 + trunc((tpotot - tpoa)/4001.0);
        i := tpoa + 1 + incremento;
        j := 2;
        tempopto := incremento * td;  { tempo correspondente a um ponto }
        Qmax := 0; Qmin := 0;
        Q^[0] := 0;
        Q^[1] := Crono^[tpoa + 1] * td;
        t^[0] := 0;
        t^[1] := td;
        Repeat
          Q^[j] := Q^[j-1] + Crono^[i] * tempopto;{ em sub-unidades de C }
          t^[j] := t^[j-1] + tempopto;
```



```
          If Qmax <= Q^[j] then Qmax := Q^[j];
          If Qmin >= Q^[j] then Qmin := Q^[j];
          i := i + incremento;
          j := j + 1;
        Until ( i > tpotot );
        nq := j - 1;
      end;
    If tecnica = 'V' then
      begin
        e1 := vini;  e2 := vfi;  nciclo := 1;
        GoToXY(10,1); Write('INTEGRACAO');
        GoToXY(10,2); Write('==========='); WriteLn;
        WriteLn(' Limites :');
        Write('    E Inicial = '); ReadIn(CR,e1);
        Write('    E  Final  = '); ReadIn(CR,e2);
        Write('    nro. do ciclo = '); ReadIn(CR,nciclo);
        Qmax := 0; Qmin := 0;
        If (vini < vfi) then
          If (e1 < e2) then
            begin
              i1 := Round((e1 - vini) / incr);
              i2 := Round((e2 - vini) / incr);
            end
          else
            begin
              i1 := Round(((vfi - e1) / incr) + tpo div 2) + 1;
              i2 := Round(((vfi - e2) / incr) + tpo div 2) + 1;
            end
        else
          If (e1 > e2) then
            begin
              i1 := Round((e1 - vini) / incr);
              i2 := Round((e2 - vini) / incr);
            end
          else
            begin
              i1 := Round(((vfi - e1) / incr) + tpo div 2) + 1;
              i2 := Round(((vfi - e2) / incr) + tpo div 2) + 1;
            end;
        Q^[0] := 0;  j := 1;
        Case nciclo of
          1..20  :  For i := i1 + 1 to i2 do
                      begin
                        Q^[j] := abs(mV^[i]-mV^[i-1]) *
                                 ((D1^[i,nciclo]+D1^[i-1,nciclo])/2);
                        Q^[j] :=  Q^[j] / velo + Q^[j-1];
                        If Qmax <= Q^[j] then Qmax := Q^[j];
                        If Qmin >= Q^[j] then Qmin := Q^[j];
                        j := j + 1;
                      end;
          21..40 :  For i := i1 + 1 to i2 do
                      begin
                        Q^[j] := abs(mV^[i]-mV^[i-1]) *
                                 ((D2^[i,nciclo-20]+D2^[i-1,nciclo-20])/2);
                        Q^[j] :=  Q^[j] / velo + Q^[j-1];
```

```
                If Qmax <= Q^[j] then Qmax := Q^[j];
                If Qmin >= Q^[j] then Qmin := Q^[j];
                j := j + 1;
              end;
        41..60 :  For i := i1 + 1 to i2 do
              begin
                Q^[j] := abs(mV^[i]-mV^[i-1]) *
                       ((D3^[i,nciclo-40]+D3^[i-1,nciclo-40]))/2);
                Q^[j] :=  Q^[j] / velo + Q^[j-1];
                If Qmax <= Q^[j] then Qmax := Q^[j];
                If Qmin >= Q^[j] then Qmin := Q^[j];
                j := j + 1;
              end;
      end;
      nq := j - 1;
      ClrScr;
    end;
  GoToXY(10,1); Write('RESULTADOS');
  GoToXY(10,2); Write('==========='); WriteLn; WriteLn;
  If UpCase(unidade[1]) = 'M' then
    begin
      unid := 'mC';
      WriteLn(' Carga maxima : ',(Qmax*fEi): 8,' mC');
      WriteLn(' Carga minima : ',(Qmin*fEi): 8,' mC');
    end;
  If UpCase(unidade[1]) = 'U' then
    begin
      unid := 'uC';
      WriteLn(' Carga maxima : ',(Qmax*fEi): 8,' uC');
      WriteLn(' Carga minima : ',(Qmin*fEi): 8,' uC');
    end;
  Sound(2440); Delay(50); Sound(1220); Delay(50); NoSound;
  controlaIntegral := 1;
end;


procedure Qxt;

  begin
    If (abs(Qmax) > 30000.0) or (abs(Qmin) > 30000.0) then
      begin
        For i := 0 to nq do Q^[i] := Q^[i] / 1000.0;
        Qmax := Qmax / 1000.0;
        Qmin := Qmin / 1000.0;
        If unid = 'mC' then unid := 'C';
        If unid = 'uC' then unid := 'mC';
        If (abs(Qmax) > 30000.0) or (abs(Qmin) > 30000.0) then
          begin
            For i := 0 to nq do Q^[i] := Q^[i] / 1000.0;
            Qmax := Qmax / 1000.0;
            Qmin := Qmin / 1000.0;
            If unid = 'C' then unid := 'kC';
            If unid = 'mC' then unid := 'C';
          end;
      end;
```



```
    inicioqx := 0; fimqx := t^[nq];
    inicioqy := Qmin; fimqy := Qmax;
    If fimqy = inicioqy then fimqy := fimqy + 1.0;
    EscalasQ;
    xmin := Round(inicioqx); xmax := Round(fimqx);
    ymin := Round(inicioqy); ymax := Round(fimqy);
    If (abs(xmax - xmin) < 330) and (xmax < 330) then
      begin
        If (abs(xmax - xmin) < 33) and (xmax < 33) then
          begin
            mult := 1000;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end
        else
          begin
            mult := 100;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end;
      end
    else mult := 1;
    Window(1,1,80,25);
    Limites;
    If mult = 1000 then EixoVolt('t','ms','Q',unid);
    If mult = 100 then EixoVolt('t','cs','Q',unid);
    If mult = 1 then EixoVolt('t','s','Q',unid);
    i := Round(inicioqx / tempopto) + 1;
    Repeat
      Linha(Round(t^[i-1] * mult),Round(Q^[i-1]),
            Round(t^[i] * mult), Round(Q^[i]));
    i := i + 1;
    Until (i > Round(fimqx / tempopto)) or KeyPressed;
    Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
    Repeat Until KeyPressed;
    TextMode;
  end;


procedure QxE;

  begin
    If (abs(Qmax) > 30000.0) or (abs(Qmin) > 30000.0) then
      begin
        For i := 0 to nq do Q^[i] := Q^[i] / 1000.0;
        Qmax := Qmax / 1000.0;
        Qmin := Qmin / 1000.0;
        If unid = 'mC' then unid := 'C';
        If unid = 'uC' then unid := 'mC';
        If (abs(Qmax) > 30000.0) or (abs(Qmin) > 30000.0) then
          begin
            For i := 0 to nq do Q^[i] := Q^[i] / 1000.0;
            Qmax := Qmax / 1000.0;
            Qmin := Qmin / 1000.0;
            If unid = 'C' then unid := 'kC';
```

```
          If unid = 'mC' then unid := 'C';
        end;
      end;
    iniciox := e1; fimx := e2;
    inicioqy := Qmin; fimqy := Qmax;
    If fimqy = inicioqy then fimqy := fimqy + 1.0;
    EscalasQ;
    xmin := iniciox; xmax := fimx;
    ymin := Round(inicioqy); ymax := Round(fimqy);
    Window(1,1,80,25);
    Limites;
    EixoVolt('E','mV','Q',unid);
    j := 1; i := il + 1;
    Repeat
      Linha(mV^[i-1],Round(Q^[j-1]),mV^[i],Round(Q^[j]));
      j := j + 1;
      i := i + 1;
    Until (i > i2) or KeyPressed;
    Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
    Repeat Until KeyPressed;
    TextMode;
  end;


Begin
  td := fatortt;                    ( tempo de aquisicao por dado em s )
  controlaIntegral := 0; controlaAmpero := 0;
  controlaVoltam := 0; controlaPotenc := 0;
  Repeat
    Menu;
    Janela(26,4,56,20);
    GoToXY(5,1);
    Write('MANIPULACAO DOS DADOS');
    GoToXY(5,2);
    Write('=========== === =====');
    GoToXY(1,4);
    Write('Grafico(s)  :  ');
    If tecnica = 'V' then
      begin
        WriteLn('(I) - i x E');
        WriteLn('              (Q) - Q x E');
      end;
    If tecnica = 'P' then WriteLn('(P) - E x t');
    If tecnica = 'A' then
      begin
        WriteLn('(A) - i x t');
        WriteLn('              (Q) - Q x t');
      end;
    GoToXY(4,14);
    Write('<ESC> - Menu Inicial');
    Read(Kbd,tecla);
    tecla := UpCase(tecla);
    Case tecla of
      'I'  :  If tecnica = 'V' then ixE;
      'A'  :  If tecnica = 'A' then ixt;
```



```
      'P'  :  If tecnica = 'P' then Ext;
      'Q'  :  begin
                If controlaIntegral = 0 then Integral;
                If tecnica = 'A' then Qxt;
                If tecnica = 'V' then QxE;
              end;
    end;   { case }
  Until (tecla = #27);
End;

Overlay Procedure Diretorio;

  Const  TextSize = 200;

  Type
    String80      =   String[80];
    String15      =   String[15];
    String5       =   String[5];
    String3       =   String[3];
    LineSize      =   String[15];
    TextArrayType =   Array[1..TextSize] of LineSize;

  Var
    FileSpec    :  String80;
    Path        :  String80;
    Name,drv    :  String15;
    AttrList    :  String5;
    DayName     :  String3;
    TextArray   :  TextArrayType;
    FileBytes,
    totbytes    :  Real;
    Code,j,k    :  Integer;
    Attr, Yr,
    Mon, Day,
    Hr, Min,
    Sec, DayNum,
    ndl         :  Byte;
    tecla,drive,
    opcao       :  Char;


  procedure FileInfo(FileSpec : String80; Var FileBytes : Real;
                     Var Yr, Mon, Day, Hr, Min, Sec : Byte;
                     Var AttrList : String5);

    Type
      RegList  =  Record
                    AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                  end;
      FileData  =  Record
                     Reserved  :  Array[1..21] of Byte;
                     Attr      :  Byte;
                     Time, Date, SizeLo, SizeHi  :  Integer;
                   end;
```

```
    var  Reg  :  RegList;
         DTA  :  Array[1..43] of Byte;
         FD   :  FileData Absolute DTA;

    begin
      Reg.DX := ofs(DTA);
      Reg.DS := seg(DTA);
      Reg.AX := $1A00;
      MSDOS(Reg);
      FileSpec := FileSpec + Chr(0);
      Reg.DX := ofs(FileSpec[1]);
      Reg.DS := seg(FileSpec[1]);
      Reg.CX := $16;
      Reg.AX := $4E00;
      MSDOS(Reg);
      If Lo(Reg.AX) <> 0 then
        begin
          FileBytes := -1.0;
          EXIT
        end;
      FileBytes := Lo(FD.SizeLo) + 256.0 * Hi(FD.SizeLo) +
                   65536.0 * Lo(FD.SizeHi) + 16777216.0 * Hi(FD.SizeHi);
      Yr := (Hi(FD.Date) shr 1) + 80;
      If Yr > 99 then
        Yr := Yr - 100;
      Mon := Lo(FD.Date) shr 5 + (Hi(FD.Date) and $01 shl 3);
      Day := Lo(FD.Date) and $1F;
      Hr := Hi(FD.Time) shr 3;
      Min := Lo(FD.Time) shr 5 + (Hi(FD.Time) and $07 shl 3);
      Sec := (Lo(FD.Time) and $1F) shl 1;
      AttrList := '     ';
      If FD.Attr and $20 = $20 then
        AttrList[1] := 'A';
      If FD.Attr and $10 = $10 then
        AttrList[2] := 'D';
      If FD.Attr and $02 = $02 then
        AttrList[3] := 'H';
      If FD.Attr and $01 = $01 then
        AttrList[4] := 'R';
      If FD.Attr and $04 = $04 then
        AttrList[5] := 'S';
    end;


procedure NextFile(FileSpec : String80; Attr : Byte;
                   var Code : Integer; Var Name : String15);

  Type
    RegList  =  Record
                  AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                end;

  var  j      :  Integer;
       Reg    :  RegList;
       DTA    :  Array[1..43] of Byte;
```



```
begin
  If (Code < 0) or (Code > 1) then
    begin
      Code := -1;
      EXIT;
    end;
  If Code = 0 then
    begin
      Reg.DX := ofs(DTA);
      Reg.DS := seg(DTA);
      Reg.AX := $1A00;
      MSDOS(Reg);
      FileSpec := FileSpec + Chr(0);
      Reg.DX := ofs(FileSpec[1]);
      Reg.DS := seg(FileSpec[1]);
      Reg.CX := Attr;
      Reg.AX := $4E00;
      MSDOS(Reg);
      If Lo(Reg.AX) <> 0 then
        begin
          Name := '';
          Code := 2;
          EXIT;
        end;
      If DTA[22] and $10 <> 0 then
        Name := '[D]'
      else
        Name := '';
      j := 31;
      While DTA[j] <> 0 do
        begin
          Name := Name + Chr(DTA[j]);
          j := j + 1;
        end;
      Code := 1;
    end
  else
    begin
      Reg.DX := ofs(DTA);
      Reg.DS := seg(DTA);
      Reg.AX := $4F00;
      MSDOS(Reg);
      If Lo(Reg.AX) = 0 then
        begin
          If DTA[22] and $10 <> 0 then
            Name := '[D]'
          else
            Name := '';
          j := 31;
          While DTA[j] <> 0 do
            begin
              Name := Name + Chr(DTA[j]);
              j := j + 1;
            end;
```

```
          Code := 1;
        end
      else
        begin
          Name := '';
          Code := 2;
        end;
    end;
end;

procedure FileDel(FileSpec : String80; Var Code : Integer);

  Type
    RegList = Record
                AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
              End;

  var Reg : RegList;

  begin
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.AX := $4100;
    MSDOS(Reg);
    Code := Lo(Reg.AX);
    If (Reg.Flags and $01) = 0 then
      Code := 0;
  end;


procedure SortHT(Var TextArray : TextArrayType;
                 LineCount, Position, Width : Integer);

  Label
    H1, H2, H3, H4, H5, H6, H7, H8;

  var  i, j,
       l, r     :   Integer;
       ThisLine :   LineSize;

  begin
    If LineCount <= 1 then EXIT;
    H1: L := LineCount div 2 + 1;
        R := LineCount;
    H2: If L > 1 then
          begin
            L := L - 1;
            ThisLine := TextArray[L];
          end
        else
          begin
            ThisLine := TextArray[R];
            TextArray[R] := TextArray[1];
            R := R - 1;
```



```
            If R = 1 then
              begin
                TextArray[1] := ThisLine;
                EXIT;
              end;
          end;
    H3: j := L;
    H4: i := j;
        j := j + j;
        If j > R then goto H7;
        If j = R then goto H6;
    H5: If Copy(TextArray[j], Position, Width) <
           Copy(TextArray[j + 1], Position, Width) then j := j+1;
    H6: TextArray[i] := TextArray[j];
        goto H4;
    H7: j := i;
        i := j div 2;
    H8: If (Copy(ThisLine, Position, Width) <=
             Copy(TextArray[i], Position, Width)) or (j = L) then
          begin
            TextArray[j] := ThisLine;
            goto H2;
          end
        else
          begin
            TextArray[j] := TextArray[i];
            goto H7;
          end;
  end;


function Tab(TabCol : Integer) : Char;

  begin
    If TabCol > 79 then
      TabCol := 79
    else
      If TabCol < 1 then
        TabCol := 1;
    GoToXY(TabCol + 1,WhereY);
    Tab := #8;
  end;


Begin
  Janela(5,4,55,20);
  Path := 'A:';
  Attr := $16;
  Code := 0;
  j := 0;
  opcao := 'M';
  Janela(17,6,43,18);
  GoToXY(11,1);
  Write('OPCOES');
  GoToXY(11,2);
```

```
  Write('======');
  GoToXY(2,4);
  Write('(M) - Mostra Arquivos');
  GoToXY(2,6);
  Write('(D) - Deleta Arquivos');
  GoToXY(2,8);
  Write('<ESC> - Abandona   ');
  Repeat
    Read(Kbd,opcao);
    opcao := UpCase(opcao);
  Until (opcao = 'M') or (opcao = 'D') or (opcao = #27);
  If opcao = #27 then EXIT;
  GetDir(0,drv);
    Case drv[1] of
      'A'     : drive := 'B';
      'B','C' : drive := 'A';
    end; ( case )
  Window(6,5,54,18);  ClrScr;
  GoToXY(1,1);
  If opcao = 'D' then
    Write('DRIVE:ARQUIVOS =      ')
  else
    Write('DRIVE:ARQUIVOS = ',drive,':*.DAT');
  Repeat
    GoToXY(18,1); Read(Kbd,tecla); tecla := UpCase(tecla);
  Until tecla in ['A','B','C',CR,#32];
  If (tecla <> CR) and (tecla <> #32) then
    begin
      Write('       ');
      GoToXY(18,1);
      drive := tecla; Write(drive);
      ReadLn(FileSpec);
      FileSpec := drive + FileSpec;
    end
  else
    begin
      If opcao = 'D' then
        FileSpec := drive + ':   '
      else
        FileSpec := drive + ':*.DAT';
      WriteLn;
    end;
  If Copy(FileSpec,1,2) = 'A:' then Path := 'A:';
  If Copy(FileSpec,1,2) = 'B:' then Path := 'B:';
  If Copy(FileSpec,1,2) = 'C:' then Path := 'C:';
  If Copy(FileSpec,1,2) = 'D:' then Path := 'D:';
  If (Length(FileSpec) = 2) then
    If (FileSpec[2] = ':') then
      begin
        If opcao = 'D' then
          FileSpec := '.:   '
        else
          FileSpec := '.:*.DAT';
      end;
  If FileSpec[2] <> ':' then FileSpec := '..' + FileSpec;
```



```
  FileSpec := Path + copy(FileSpec,3,80);
  Repeat
    NextFile(FileSpec, Attr, Code, Name);
    If Code = 1 then
      begin
        j := j + 1;
        TextArray[j] := Name;
      end;
  Until (Code <> 1);
  WriteLn(j,' Arquivos encontrados');
  SortHT(TextArray, j, 1, 15);
  If opcao = 'D' then
    begin
      ndl := 0;
      For k := 1 to j do
        begin
          FileSpec := Path + TextArray[k];
          FileDel(FileSpec, Code);
          If Code <> 0 then
            begin
              WriteLn(FileSpec,' nao deletado');
              ndl := ndl + 1;
            end;
        end;
      WriteLn(j - ndl,' arquivos deletados');
    end
  else
    begin
      totbytes := 0;
      For k := 1 to j do
        begin
          FileSpec := TextArray[k];
          If FileSpec[1] = 'I' then
            FileSpec := Copy(FileSpec, 4, 12);
          FileSpec := Path + FileSpec;
          FileInfo(FileSpec,FileBytes, Yr,Mon,Day, Hr,Min,Sec, AttrList);
          Write(TextArray[k], Tab(14), FileBytes:8:0, Tab(24));
          If FileBytes < 0.0 then
            WriteLn('... Arquivo nao encontrado ...')
          else
            begin
              Write(Day:2,'/',Mon:2,'/',Yr, Tab(34));
              Write(Hr:2,':');
              If Min < 10 then WriteLn('0',Min) else WriteLn(Min:2);
              totbytes := totbytes + FileBytes;
            end;
          If ((k/12) = (k div 12)) and (k < j) then
            begin
              WriteLn('  ... Aperte uma tecla ...');
              Read(Kbd,tecla);
            end;
        end;
      WriteLn('       ',totbytes:8:0,' bytes ocupados');
    end;
  Sound(1760); Delay(50);
```

```
    Sound(880); Delay(50);
    Sound(440); Delay(50);
    NoSound;
    Write('       ... [Enter] - volta ao MENU ...');
    Read(tecla);
  End;

BEGIN
  Mark(marcaheap);
  D1 := nil;
  New(MAD);
  New(MDA);
  New(D1);
  New(D2);
  New(D3);
  New(Crono);
  New(Q);
  New(t);
  New(mV);
  Window(1,1,80,25); ClrScr;
  vini := -2000; vfi := 2000; nc := 1; velo := 100.0;  { Valores  }
  passo := 1;  primeiro := 1; unidade := 'mA';         { Default  }
  inii := 0.0; fii := 0.0; corrente := 200; ;          {  para    }
  h1 := 0;  m1 := 0;  s1 := 0;                         {  todas   }
  h2 := 0;  m2 := 0;  s2 := 1;                         {    as    }
  h3 := 0;  m3 := 0;  s3 := 0; npulsos := 1;           { tecnicas }
  TelaInicio;
  GoToXY(34,1); Write(versao);
  GoToXY(1,25); ClrEol; TextColor(31); GoToXY(25,25);
  Write('... APERTE UMA TECLA ...'); Read(Kbd,tecla);
  ClrScr; GoToXY(13,3);
  WriteLn('PRESSIONE  <CONTROL> <ALT> (+) (do bloco numerico)');
  TextColor(15);
  GoToXY(10,10); Write('Escolha o tipo de potenciostato :');
  GoToXY(10,12); Write('1) FAC200A ou compativel (saida em +/- 10 V)');
  GoToXY(10,13); Write('2) FAC200 ou compativel (saida em +/- 2 V)');
  Repeat
    Read(Kbd,tecla); tecla := UpCase(tecla);
  Until tecla in ['1','2'];
  TextColor(31);  Sound(1000); Delay(200); NoSound;
  GoToXY(10,15);
  If tecla = '1' then
    begin
      FAC200A := TRUE; multpot := 5;
      WriteLn('Posicione a chave de entrada em +/- 10 V');
    end
  else
    begin
      FAC200A := FALSE; multpot := 1;
      WriteLn('Posicione a chave de entrada em +/- 2 V');
    end;
  GoToXY(22,19); TextColor(15); Write('Aperte uma tecla para prosseguir');
  Read(Kbd,tecla); GoToXY(22,19); ClrEol;
  GoToXY(28,19); Write('... A G U A R D E ...');
  CalcCont;
```



```
  DAC(0);                  ( Coloca em 0 volts a saida do D/A. )
  Sound(880); Delay(200); Sound(1760); Delay(200); NoSound;
  InicializaSeletor;
  Repeat
    Menu;
    Read(Kbd,tecla);
    tecla := UpCase(tecla);
    Case tecla of
      'C' : Coleta;
      'A' : begin
              Disco; fEi := corrente/2000.0;
            end;
      'R' : Resultados;
      'D' : Diretorio;
      'S' : begin
              Salva(' ',tpotot);
              passo := 1; primeiro := 1;
            end;
      'F' : begin
              Janela(16,10,46,15);
              GoToXY(2,2); Write('Confirma saida?  (S/N)  ');
              Read(Kbd,tecla);
              tecla := UpCase(tecla);
              If tecla = 'S' then tecla := 's';
            end;
    end;  ( case )
  Until (tecla = 's');
  Window(1,1,80,25); ClrScr;
  Release(marcaheap);
END.
```

A13. PROGRAMA MULTICAN

```
Program MULTICAN;

Const LF    = #10;
      CR    = #13;
      ESC   = #27;
      NUMAX = 10000;
      esq   = 85;
      dir   = 535;
      cima  = 5;
      baixo = 155;

Type String80     =  String[80];
     String15     =  String[15];
     String8      =  String[8];
     String2      =  String[2];
     PonteiroI    =  ^TipoVetI;
     TipoVetI     =  Array[0..NUMAX] of Integer;
     PonteiroR    =  ^TipoVetR;
     TipoVetR     =  Array[0..NUMAX] of Real;
     PonteiroV    =  ^TipoVetV;
     TipoVetV     =  Array[0..1600,1..20] of Integer;
     Matriza      =  Array[0..4095] of Integer;
     Matrizd      =  Array[-2020..2020] of Integer;
     Vetor        =  Array[0..1600] of Integer;

Var  dia, mes, ano,
     hora, min, segu,
     centseg, seh, can, gan    : Byte;
     unidade, ut               : String2;
     vunidade                  : Byte Absolute unidade;
     unidx, unidy              : Array[0..6] of String2;
     titx, tity                : Array[0..6] of String8;
     canal, ganho              : Array[1..3] of Integer;
     fundoesc, vlrmax          : Array[0..6] of Integer;
     fx, fy                    : Array[0..6] of Real;
     ncanais, velo, i, j,
     ndados, aquis, vini, vfi,
     nc, esp, histda,
     primeiro, ultimo, passo,
     linha, coluna, npulsos    : Integer;
     tai, tbi,
     TEMPOAQUIS1,   {Tempo de aquisicao total de 1 dado em seg modo YX.}
     ta2, tb2,
     TEMPOAQUIS2,   {Tempo de aquisicao total de 1 dado em seg modo YT.}
     ta3, tb3,
     TEMPOAQUIS3,   {Tempo de aquisicao total de 1 dado em seg modo multiplo.}
     ta4, tb4,
     TEMPOAQUIS4,   {Tempo de aquisicao total de 1 dado em seg modo Y x V.}
     xmax, xmin, ymax, ymin,
     minx, maxx, miny, maxy,
     tpotot, intaquis          : Real;
     opcao, modo, tecla, drive : Char;
     erro, fazaquis            : Boolean;
     operador, titulo, nome    : String15;
     data, horario, drv        : String8;
```



```
     D1, D2, D3                : PonteiroI;
     DT, fcx, fcy              : PonteiroR;
     DV1, DV2                  : PonteiroV;
     MAD                       : Matriza;
     MDA                       : Matrizd;
     mV                        : Vetor;
     marcaheap                 : ^Char;

procedure Graphics;                                    external 'd:\pascal\v301\GRAPH.BIN';
procedure HiRes;                                       external Graphics[6];
procedure GraphWindow(X1,Y1,X2,Y2: Integer);           external Graphics[18];
procedure Plot(X,Y,Color: Integer);                    external Graphics[21];
procedure Draw(X1,Y1,X2,Y2,Color: Integer);            external Graphics[24];
procedure GetPic(var Buffer; X1,Y1,X2,Y2: Integer);    external Graphics[36];
procedure PutPic(var Buffer; X,Y: Integer);            external Graphics[39];
procedure ClearScreen;                                 external Graphics[60];


Procedure TelaInicio(nome : String8);

type  matrizg = Array[0..4099] of Byte;
      Disco = record
                telag : matrizg;
              end;

Var  arquivo  :  Disco Absolute $B800:0000;
     buftela  :  file of Disco;

Begin
  ClrScr;
  Assign(buftela,nome + '.TXT');
  Reset(buftela);
  Read(buftela,arquivo);
  Close(buftela);
end;

Procedure LeNome(var n : String15; c : Char);

var i : Integer;

Begin
  Read(n);
  If (n[2] = ':') and not(n[1] in ['A'..'E','a'..'e']) then
    n[1] := drive;
  If n[2] <> ':' then n := drive + ':' + n;
  If Pos('.',n) > 0 then
  n := Copy(n,1,Pos('.',n)-1);
  For i := 1 to Length(n) do n[i] := UpCase(n[i]);
  If c = CR then WriteLn;
End;

Function Confirma(mens : String80) : Boolean;

Var resposta : Char;
```

```
Begin
  Confirma := FALSE;
  Write(mens);
  Repeat
    Read(Kbd,resposta); resposta := UpCase(resposta);
  Until resposta in ['N','S'];
  If resposta = 'S' then Confirma := TRUE;
End;

Procedure Janela(xi,yi,x2,y2 : Integer);

Var    i, maxx, maxy    :  Integer;

Begin
  Window (1,1,80,25); TextColor(3);
  GoToXY (xi,yi);
  Window (xi,yi,x2,y2);
  ClrScr; maxx := x2 - xi + 1; maxy := y2 - yi + 1;
  GoToXY (1,maxy); Write (#212);
  For i := 2 to maxx - 1 do Write(#205);
  Write (#190);
  GoToXY (1,1); Write (#213);
  For i := 2 to maxx - 1 do Write(#205);
  Write (#184);
  For i := 2 to maxy - 2 do
    begin
      GoToXY (1,i); Write(#179);
      GoToXY (maxx,i); Write(#179);
    end;
  GoToXY (2,2); Window (xi+1,yi+1,x2-1,y2-2);
  linha := yi ; coluna := xi + 1; TextColor(14);
End;


Procedure WriteM(s : String80; carac : Char);

Var i   : Integer;

Begin
  For i := 1 to Length(s) do
    begin
      If s[i] in ['A'..'Z','0'..'9'] then
        begin
          TextColor(15); Write(s[i]); TextColor(14);
        end
      else Write(s[i]);
    end;
  If carac = CR then WriteLn;
End;

Procedure WriteV(x,y : Integer; mensagem : String15);

Var i  :  Byte;

Begin
```



```
  For i := 1 to Length(mensagem) do
    begin
      GoToXY(x,y);
      Write(mensagem[i]);
      y := y + 1;
    end;
End;

Procedure Time(var h,m,s,cs : Byte);

Type
  RegList  =  Record
                AX, BS, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
              end;

Var   Reg       :   RegList;

Begin
  Reg.Ax := $2C00;
  MSDOS(Reg);
  h := Hi(Reg.CX); m := Lo(Reg.CX); s := Hi(Reg.DX); cs := Lo(Reg.DX);
End;

Procedure Date(var di,me,an : Byte);

Type
  RegList  =  Record
                AX, BS, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
              end;

Var   Reg       :   RegList;

Begin
  Reg.Ax := $2A00;
  MSDOS(Reg);
  Reg.CX := Reg.CX - 1900;
  If Reg.CX ) 99 then Reg.CX := Reg.CX - 100;
  an := Reg.CX; me := Hi(Reg.DX); di := Lo(Reg.DX);
End;

Procedure ReadIn(ci : Char; var x : Integer);

type par = Record
             car   : Char;
             atrib : Byte;
           end;

    TipoLinha = Array[1..80] of par;
    TipoTela  = Array[1..25] of TipoLinha;

var numero           : String[20];
    codigo, spc, c   : Integer;
    tela             : TipoTela absolute $8800:$0000;

Begin
```

```
    Str(x, numero);
    Write(numero);
    Repeat Until keypressed;
    Repeat
      spc := length(numero);
      GoToXY(WhereX - spc, WhereY);
      For c := WhereX to WhereX + spc do
        tela[linha + WhereY,coluna + c].car := ' ';
      Read(numero); val(numero, x, codigo);
    Until codigo = 0;
    spc := length(numero); GoToXY(WhereX - spc, WhereY); Write(x);
    If ci = CR then WriteLn;
  End;

  Procedure ReadRe(ci : Char; var x : Real; spc, dec : Integer);

  type par = Record
               car   : Char;
               atrib : Byte;
             end;

      TipoLinha = Array[1..80] of par;
      TipoTela  = Array[1..25] of TipoLinha;

  var numero          : String[20];
      codigo,c,xi,esp : Integer;
      tela            : TipoTela absolute $B800:$0000;

  Begin
    xi := WhereX; Str(x, numero); esp := 10;
    Write(x:spc:dec);
    Repeat Until keypressed;
    Repeat
      GoToXY(xi,WhereY);
      For c := WhereX to WhereX + esp do
        tela[linha + WhereY,coluna + c].car := ' ';
      Read(numero);
      val(numero, x, codigo);
      If codigo <> 0 then esp := length(numero);
    Until codigo = 0;
    GoToXY(xi, WhereY);  Write(x:spc:dec);
    If ci = CR then WriteLn;
  End;

  Procedure Seletor;

  Begin
    Port[$303] := can + (gan shl 4) + seh;
  End;

  Procedure Atualiza(c,g : Byte);

  Begin
    can := c;  gan := g;
    Seletor;
  End;

  Procedure HoldDA;

  Begin
    Port[$304] := 0;
  End;

  Procedure DAC(valor : Integer);

  Begin
    PortW[$300] := MDA[valor];
    Port[$304] := 1;        { Sample }
    histda := valor;
  End;

  Procedure SampleAD;

  Begin
    seh := 128;
    Port[$303] := can + (gan shl 4) + seh;
  End;

  Procedure HoldAD;

  Begin
    seh := 0;
    Port[$303] := can + (gan shl 4) + seh;
  End;

  Function ADC : Integer;

  Var  ad : Integer;

  Begin
    ad := 0;
    HoldAD;
    HoldDA;
    PortW[$300] := 2048;
    If (Port[$302] and 1) = 1 then ad := ad + 2048;
    PortW[$300] := ad + 1024;
    If (Port[$302] and 1) = 1 then ad := ad + 1024;
    PortW[$300] := ad + 512;
    If (Port[$302] and 1) = 1 then ad := ad + 512;
    PortW[$300] := ad + 256;
    If (Port[$302] and 1) = 1 then ad := ad + 256;
    PortW[$300] := ad + 128;
    If (Port[$302] and 1) = 1 then ad := ad + 128;
    PortW[$300] := ad + 64;
    If (Port[$302] and 1) = 1 then ad := ad + 64;
    PortW[$300] := ad + 32;
    If (Port[$302] and 1) = 1 then ad := ad + 32;
    PortW[$300] := ad + 16;
    If (Port[$302] and 1) = 1 then ad := ad + 16;
    PortW[$300] := ad + 8;
```

```
  If (Port[$302] and i) = i then ad := ad + 8;
  PortW[$300] := ad + 4;
  If (Port[$302] and i) = i then ad := ad + 4;
  PortW[$300] := ad + 2;
  If (Port[$302] and i) = i then ad := ad + 2;
  PortW[$300] := ad + 1;
  If (Port[$302] and i) = i then ad := ad + 1;
  DAC(histda);              { Reestabelece o antigo valor para o DA. }
  SampleAD;
  ADC := MAD[ad];                      { valor ja convertido em mV. }
End;

Procedure CalcCont;

Var   a,b      : Real;
      c        : Integer;

Begin
  GoToXY(74,25); Write('10');
  a := -0.976820; b := 1999.668;       { Converte os valores binarios }
  For c := 0 to 4095 do                { para valores em mV para o DA }
    MDAC[Round(a * c + b)] := c;       { na faixa de -2000 a 2000 mV. }

  GoToXY(74,25); Write(' 9');
  a := 1.113952; b := -2246.45;        { Converte os valores binarios }
  For c := 0 to 4095 do                { para valores em mV para o AD }
    MAD[c] := Round(a * c + b);        { na faixa de -2000 a 2000 mV. }
End;

Procedure Inicializa;

var i : Integer;

Begin
  canal[1] := 1; canal[2] := 2; canal[3] := 3;     { Entradas padroes. }
  ganho[1] := 0; ganho[2] := 0; ganho[3] := 0;     { Ganhos unitarios. }
  For i := 0 to 6 do
    begin
      fundoesc[i] := 2000;
      vlrmax[i] := 2000;
      fx[i] := 300.0 / (2 * fundoesc[i]);
      fy[i] := 98.0 / (2 * fundoesc[i]);
      titx[i] := ' ';
      tity[i] := ' ';
      unidx[i] := 'mV';
      unidy[i] := 'mV';
    end;
  seh := 128;                                      { Amostragem. }
  Atualiza(canal[1],ganho[1]);                     { Seta todos os canais }
  Atualiza(canal[2],ganho[2]);                     { padroes   e   ganhos }
  Atualiza(canal[3],ganho[3]);                     { unitarios.           }
  intaquis := 0.1; tpotot := 1.0; vini := -2000; vfi := 2000;
  velo := 100; primeiro := 1; passo := 1; ultimo := 1; nc := 1;
  unidade := 'uA'; ut := 's'; ncanais := 3;
  operador := '??????????????'; titulo := '??????????????';
```



```
End;

Procedure Aviso(mensagem : String80);

Begin
  Janela((39 - length(mensagem) div 2),11,(42 + length(mensagem) div 2),14);
  Write(mensagem);
  Sound(2000); Delay(50); Sound(1000); Delay(50); NoSound;
  Repeat Until KeyPressed;
End;

Procedure Borda(x1,y1,x2,y2 : Integer);

Begin
  Window(1,1,80,25);
  GraphWindow(0,0,639,199);
  Draw(x1,y1,x2,y1,1); Draw(x2,y1,x2,y2,1);
  Draw(x2,y2,x1,y2,1); Draw(x1,y1,x1,y2,1);
End;

Procedure PosicionaJanela(x1,y1,x2,y2 : Integer; y : Real);

Begin
  maxy := y;
  GraphWindow(0,0,639,199); GraphWindow(x1,y1,x2,y2);
End;

Function x(vx : Real; fundo : Integer) : Integer;

Begin
  x := Round(fx[fundo] * (vx - minx));
End;

Function y(vy : Real; fundo : Integer) : Integer;

Begin
  y := Round(fy[fundo] * (maxy - vy));
End;

Procedure Tempo(var t : Real);

var h,mi,s,cs : Byte;

Begin
  Time(h,mi,s,cs);
  If h < hora then h := h + 24;
  t := cs / 100.0 + s + mi * 60.0 + h * 3600.0;
End;

Procedure Espera;

var  i          : Integer;
     valor      : Real;

Begin
```

```
  For i := 1 to esp do valor := ADC;
End;

Procedure Le(var valormedio : Integer);

var  k           : Integer;
     valorconv   : Real;

Begin
  valorconv := 0.0;
  For k := 1 to aquis do valorconv := valorconv + ADC;
  valormedio := Round(valorconv / aquis);
  Espera;
End;


Procedure DeterminaTempo;   { calcula tempo de aquisicao de um dado,    }
                            { tornando o programa independente do micro}
Var i      : Integer;
    t,
    ti,tf  : Real;

Begin
  Time(hora,min,segu,centseg);
  fx[1] := 300.0 / (2 * fundoesc[1]);  fy[1] := 98.0 / (2 * fundoesc[1]);
  minx := -2000;  maxy := 2000; esp := 0;
  GraphWindow(0,0,639,199);
  GoToXY(74,25); Write(' 8');
  aquis := 5000;
  Tempo(ti);
  For i := 1 to 1 do
    begin
      Atualiza(canal[1],ganho[1]);  Le(D1^[i]);
      Atualiza(canal[2],ganho[2]);  Le(D2^[i]);
      Plot(x(D2^[i],canal[2]),y(D1^[i],canal[1]),0);
    end;
  Tempo(tf);
  t := tf - ti;
  GoToXY(74,25); Write(' 7');
  aquis := 1;
  Tempo(ti);
  For i := 1 to 5000 do
    begin
      Atualiza(canal[1],ganho[1]);  Le(D1^[i]);
      Atualiza(canal[2],ganho[2]);  Le(D2^[i]);
      Plot(x(D2^[i],canal[2]),y(D1^[i],canal[1]),0);
    end;
  Tempo(tf);
  tb1 := ((tf - ti) - t) / 4999.0;
  ta1 := (t - tb1) / 10000.0;
  TEMPOAQUIS1 := ta1 + tb1;
  GoToXY(74,25); Write(' 6');
  aquis := 5000;
  Tempo(ti);
  For i := 1 to 1 do
```



```
    begin
      Le(D1^[i]); Plot(x(i,canal[1]),y(D1^[i],canal[1]),0);
    end;
  Tempo(tf);
  t := tf - ti;
  GoToXY(74,25); Write(' 5');
  aquis := 1;
  Tempo(ti);
  For i := 1 to 5000 do
    begin
      Le(D1^[i]); Plot(x(i,canal[1]),y(D1^[i],canal[1]),0);
    end;
  Tempo(tf);
  tb2 := ((tf - ti) - t) / 4999.0;
  ta2 := (t - tb2) / 5000.0;
  TEMPOAQUIS2 := ta2 + tb2;
  GoToXY(74,25); Write(' 4');
  aquis := 5000;
  Tempo(ti);
  For i := 1 to 1 do
    begin
      Atualiza(canal[1],ganho[1]);  Le(D1^[i]);
      Atualiza(canal[2],ganho[2]);  Le(D2^[i]);
      Atualiza(canal[3],ganho[3]);  Le(D3^[i]);
      PosicionaJanela(1,1,300,98,1); Plot(x(i,canal[1]),y(D1^[i],canal[1]),0);
      PosicionaJanela(321,1,620,98,1); Plot(x(i,canal[2]),y(D2^[i],canal[2]),0);
      PosicionaJanela(1,101,300,198,1); Plot(x(i,canal[3]),y(D3^[i],canal[3]),0);
    end;
  Tempo(tf);
  t := tf - ti;
  GoToXY(74,25); Write(' 3');
  aquis := 1;
  Tempo(ti);
  For i := 1 to 5000 do
    begin
      Atualiza(canal[1],ganho[1]);  Le(D1^[i]);
      Atualiza(canal[2],ganho[2]);  Le(D2^[i]);
      Atualiza(canal[3],ganho[3]);  Le(D3^[i]);
      PosicionaJanela(1,1,300,98,1); Plot(x(i,canal[1]),y(D1^[i],canal[1]),0);
      PosicionaJanela(321,1,620,98,1); Plot(x(i,canal[2]),y(D2^[i],canal[2]),0);
      PosicionaJanela(1,101,300,198,1); Plot(x(i,canal[3]),y(D3^[i],canal[3]),0);
    end;
  Tempo(tf);
  tb3 := ((tf - ti) - t) / 4999.0;
  ta3 := (t - tb3) / 15000.0;
  TEMPOAQUIS3 := ta3 + tb3;
  GoToXY(74,25); Write(' 2');
  aquis := 5000;
  Tempo(ti);
  For i := 1 to 1 do
    begin
      DAC(mV[i]);
      Atualiza(canal[1],ganho[1]);  Le(DV1^[i,1]);
      Atualiza(canal[2],ganho[2]);  Le(DV2^[i,1]);
      PosicionaJanela(1,1,300,98,1);
```

```
      Plot(x(mV[i],canal[1]),y(DV1^[i,1],canal[1]),1);
      PosicionaJanela(321,1,620,98,1);
      Plot(x(mV[i],canal[2]),y(DV2^[i,1],canal[2]),1);
    end;
  Tempo(tf);
  t := tf - ti;
  GoToXY(74,25); Write(' 1');
  aquis := 1;
  Tempo(ti);
  For i := 1 to 5000 do
    begin
      DAC(mV[i]);
      Atualiza(canal[1],ganho[1]);  Le(DV1^[i,1]);
      Atualiza(canal[2],ganho[2]);  Le(DV2^[i,1]);
      PosicionaJanela(1,1,300,98,1);
      Plot(x(mV[i],canal[1]),y(DV1^[i,1],canal[1]),1);
      PosicionaJanela(321,1,620,98,1);
      Plot(x(mV[i],canal[2]),y(DV2^[i,1],canal[2]),1);
    end;
  Tempo(tf);
  tb4 := ((tf - ti) - t) / 4999.0;
  ta4 := (t - tb4) / 10000.0;
  TEMPOAQUIS4 := ta4 + tb4;
End;

Overlay Procedure Aquisicao;

var  vm, i, ndpul, ndappul   : Integer;
     tecla                   : Char;
     ti, tf, tpopul, tpoappul : Real;
     tipomult                : Boolean;


procedure ModoYX;

var i : Integer;

Begin
  HiRes; Borda(0,0,301,99); WriteV(39,1,tity[canal[1]] + ' ' + unidy[canal[1]]);
  GoToXY(1,14); Write(titx[canal[2]],' ',unidx[canal[2]]);
  GoToXY(10,16); Write('...CONDICOES...');
  GoToXY(2,18);
  If tpotot < 100.0 then
    WriteLn('Tempo total de aquisicao -) ',tpotot:0:3,' s')
  else WriteLn('Tempo total de aquisicao -) ',tpotot:0:0,' s');
  WriteLn(' Intervalo de aquisicao -) ',intaquis:0:3,' s');
  WriteLn(' Canal -) ',canal[1],' ... Amplificacao -) ',ganho[1]);
  WriteLn(' Canal -) ',canal[2],' ... Amplificacao -) ',ganho[2]);
  Date(dia,mes,ano); Time(hora,min,segu,centseg);
  i := -1;  PosicionaJanela(1,1,300,98,fundoesc[canal[1]]);
  Sound(3000); Delay(100); NoSound;
  Tempo(ti);
  Repeat
    i := i + 1;
    Atualiza(canal[1],ganho[1]);  Le(D1^[i]);
```



```
    Atualiza(canal[2],ganho[2]);  Le(D2^[i]);
    Plot(x(D2^[i],canal[2]),y(D1^[i],canal[1]),1);
  Until (i )= ndados) or (i = NUMAX) or KeyPressed;
  Tempo(tf);
  ndados := i;  Sound(1760); Delay(200); NoSound;
  tpotot := tf - ti;
  WriteLn(' Tempo total real de aquisicao -) ',tpotot:0:3,' s');
  GoToXY(33,25); Write('... AGUARDE ...');
  intaquis := tpotot / ndados;
  For i := 0 to ndados do DT^[i] := i * intaquis;
  GoToXY(24,25); Write('... Tecle (ENTER) para continuar ...');
  Repeat
    Read(Kbd,tecla);
  Until tecla = CR;
End;

procedure ModoYT;

var i : Integer;

Begin
  HiRes; Borda(0,0,301,99); WriteV(39,1,tity[canal[1]] + ' ' + unidy[canal[1]]);
  GoToXY(1,14); Write(titx[canal[1]],' ',unidx[canal[1]]);
  GoToXY(10,16); Write('...CONDICOES...');
  GoToXY(2,18);
  If tpotot < 100.0 then
    WriteLn('Tempo total de aquisicao -) ',tpotot:0:3,' s')
  else WriteLn('Tempo total de aquisicao -) ',tpotot:0:0,' s');
  WriteLn(' Intervalo de aquisicao -) ',intaquis:0:3,' s');
  WriteLn(' Canal -) ',canal[1],' ... Amplificacao -) ',ganho[1]);
  i := -1;
  Atualiza(canal[1],ganho[1]); PosicionaJanela(1,1,300,98,fundoesc[canal[1]]);
  Date(dia,mes,ano); Time(hora,min,segu,centseg);
  Sound(3000); Delay(100); NoSound;
  Tempo(ti);
  Repeat
    i := i + 1;  Le(D1^[i]);
    Plot(x(i,canal[1]),y(D1^[i],canal[1]),1);
  Until (i )= ndados) or (i = NUMAX) or KeyPressed;
  Tempo(tf);
  ndados := i;  Sound(1760); Delay(200); NoSound;
  tpotot := tf - ti;
  WriteLn(' Tempo total real de aquisicao -) ',tpotot:0:3,' s');
  GoToXY(33,25); Write('... AGUARDE ...');
  intaquis := tpotot / ndados;
  For i := 0 to ndados do DT^[i] := i * intaquis;
  GoToXY(24,25); Write('... Tecle (ENTER) para continuar ...');
  Repeat
    Read(Kbd,tecla);
  Until tecla = CR;
End;

procedure ModoMultiplo;

var i,j,np : Integer;
```

```
    c1,c2,c3,p : Byte;

procedure LePlota;

Begin
  Atualiza(canal[1],ganho[1]);  Le(D1^[i]);
  Atualiza(canal[2],ganho[2]);  Le(D2^[i]);
  Atualiza(canal[3],ganho[3]);  Le(D3^[i]);
  PosicionaJanela(1,1,300,98,fundoesc[canal[1]]);
  Plot(x[i,canal[1]],y(D1^[i,canal[1]),c1);
  PosicionaJanela(321,1,620,98,fundoesc[canal[2]]);
  Plot(x[i,canal[2]],y(D2^[i,canal[2]),c2);
  PosicionaJanela(1,101,300,198,fundoesc[canal[3]]);
  Plot(x[i,canal[3]],y(D3^[i,canal[3]),c3);
End;

Begin
  HiRes; p := 1;
  Borda(0,0,301,99); WriteV(39,1,tity[canal[1]] + ' ' + unidy[canal[1]]);
  If ncanais = 3 then
    begin
      p := 42;
      Borda(319,0,621,99); WriteV(79,1,tity[canal[2]] + ' ' + unidy[canal[2]]);
      Borda(0,100,301,199); WriteV(39,14,tity[canal[3]] + ' ' + unidy[canal[3]]);
    end;
  If ncanais = 2 then
    begin
      Borda(319,0,621,99); WriteV(79,1,tity[canal[2]] + ' ' + unidy[canal[2]]);
    end;
  c1 := 1; c2 := 1; c3 := 1;
  If ncanais (= 2 then c3 := 0;
  If ncanais = 1 then c2 := 0;
  GoToXY(p + 9,14); Write('...CONDICOES...');
  GoToXY(p,16);
  If tpotot ( 100.0 then
    Write('Tempo total de aquisicao -) ',tpotot:0:3,' s')
  else Write('Tempo total de aquisicao -) ',tpotot:0:0,' s');
  GoToXY(p,17); Write('Intervalo de aquisicao -) ',intaquis:0:3,' s');
  If tipomult then
    begin
      GoToXY(p,18); Write('Numero de pulsos -) ',npulsos);
      GoToXY(p,19); Write('E inicial -) ',vini,' mV');
      GoToXY(p,20); Write('E final -) ',vfi,' mV');
    end;
  For i := 1 to ncanais do
    begin
      GoToXY(p,20+i);
      Write('Canal -) ',canal[i],' ... Amplificacao -) ',ganho[i]);
    end;
  WriteLn;
  i := -1;
  Date(dia,mes,ano); Time(hora,min,segu,centseg);
  Sound(3000); Delay(100); NoSound;
  np := 0;
  If tipomult then
```



```
    begin
      Tempo(ti);
      Repeat
        np := np + 1;
        j := 0;
        Repeat
          j := j + 1;
          i := i + 1;
          DAC(vini);
          LePlota;
        Until (j )= ndpul) or (i = NUMAX) or KeyPressed;
        j := 0;
        Repeat
          j := j + 1;
          i := i + 1;
          DAC(vfi);
          LePlota;
        Until (j )= ndappul) or (i = NUMAX) or KeyPressed;
      Until (np )= npulsos) or (i )= ndados) or (i = NUMAX) or KeyPressed;
      Tempo(tf);
    end
  else
    begin
      Tempo(ti);
      Repeat
        i := i + 1;
        LePlota;
      Until (i )= ndados) or (i = NUMAX) or KeyPressed;
      Tempo(tf);
    end;
  ndados := i;  Sound(1760); Delay(200); NoSound;
  tpotot := tf - ti;
  GoToXY(p,WhereY); WriteLn('Tpo. tot. real de aquis. -) ',tpotot:0:3,' s');
  GoToXY(p + 5,25); Write('... AGUARDE ...');
  intaquis := tpotot / ndados;
  For i := 0 to ndados do DT^[i] := i * intaquis;
  GoToXY(p,25); Write('... Tecle (ENTER) para continuar ...');
  Repeat
    Read(Kbd,tecla);
  Until tecla = CR;
End;


procedure MeioCiclo;

var incr : Real;
    i    : Integer;

Begin
  incr := 1.0 * (vfi-vini)/ndados;          ( incremento em mV )
  For i := 0 to ndados do mV[i] := vini + Round(i * incr);
  Write('.Espere estabilizar e aperte uma tecla.');
  Repeat Until KeyPressed;
  Window(1,1,80,25);
  HiRes; Borda(0,0,301,99); WriteV(39,1,tity[canal[i]] + ' ' + unidy[canal[i]]);
```

```
  Borda(319,0,621,99);
  WriteV(79,1,tity[canal[2]] + ' ' + unidy[canal[2]]);
  GoToXY(1,14); Write(titx[canal[1]],' ',unidx[canal[1]]);
  GoToXY(41,14); Write(titx[canal[2]],' ',unidx[canal[2]]);
  GoToXY(31,16); Write('...CONDICOES...');
  GoToXY(2,18);
  If tpotot < 100.0 then
    WriteLn('Tempo total de aquisicao -) ',tpotot:0:3,' s')
  else WriteLn('Tempo total de aquisicao -) ',tpotot:0:0,' s');
  WriteLn(' Intervalo de aquisicao -) ',intaquis:0:3,' s');
  WriteLn(' Operador : ',operador,LF,CR,' Titulo : ',titulo);
  Write(' Data : ',data);
  For i := 1 to ncanais do
    begin
      GoToXY(46,17+i);
      Write('Canal -) ',canal[i],' ... Amplificacao -) ',ganho[i]);
    end;
  GoToXY(46,20); Write('Ei = ',vini,' mV');
  GoToXY(46,21); Write('Ef = ',vfi,' mV');
  GoToXY(46,22); Write('ii = ',-vlrmax[canal[1]],' ',unidade);
  GoToXY(46,23); Write('if = ',vlrmax[canal[1]],' ',unidade);
  GoToXY(46,24); Write('Velocidade = ',velo,' mV/s');
  Date(dia,mes,ano); Time(hora,min,segu,centseg);
  Sound(3000); Delay(100); NoSound;
  i := -1;
  Tempo(ti);
  Repeat
    i := i + 1;
    DAC(mV[i]);
    Atualiza(canal[1],ganho[1]);  Le(DV1^[i,1]);
    Atualiza(canal[2],ganho[2]);  Le(DV2^[i,1]);
    PosicionaJanela(1,1,300,98,fundoesc[canal[1]]);
    Plot(x(mV[i],canal[1]),y(DV1^[i,1],canal[1]),1);
    PosicionaJanela(321,1,620,98,fundoesc[canal[2]]);
    Plot(x(mV[i],canal[1]),y(DV2^[i,1],canal[2]),1);
  Until KeyPressed or (i >= ndados);
  Tempo(tf); nc := 1; ndados := i; ultimo := nc;
  Sound(1760); Delay(200); NoSound; DAC(0);
  tpotot := tf - ti;
  GoToXY(1,23); WriteLn(' Tempo total real');
  Write(' de aquisicao -) ',tpotot:0:3,' s');
  GoToXY(23,25); Write('... Tecle (ENTER) para continuar ...');
  Repeat
    Read(Kbd,tecla);
  Until tecla = CR;
End;


procedure Ciclos;

var i, j  : Integer;
    incr  : Real;

Begin
  j := ndados div 2;
```



```
  incr := 1.0 * (vfi-vini) / j;              ( incremento em mV )
  For i := 0 to j do                         ( ida )
    mV[i] := vini + Round(i * incr);
  For i := j + 1 to ndados do                ( volta )
    mV[i] := vfi - Round((i - j) * incr);
  Write('.Espere estabilizar e aperte uma tecla.');
  Repeat Until KeyPressed;
  Window(1,1,80,25);
  HiRes; Borda(0.0,301,99); WriteV(39,1,tity[canal[1]] + ' ' + unidy[canal[1]]);
  Borda(319,0,621,99);
  WriteV(79,1,tity[canal[2]] + ' ' + unidy[canal[2]]);
  GoToXY(1,14); Write(titx[canal[1]],' ',unidx[canal[1]]);
  GoToXY(41,14); Write(titx[canal[2]],' ',unidx[canal[2]]);
  GoToXY(31,16); Write('...CONDICOES...');
  GoToXY(2,17);
  If nc * tpotot < 100.0 then
    WriteLn('Tempo total de aquisicao -) ',nc * tpotot:0:3,' s')
  else WriteLn('Tempo total de aquisicao -) ',nc * tpotot:0:0,' s');
  WriteLn(' Intervalo de aquisicao -) ',intaquis:0:3,' s');
  WriteLn(' Operador : ',operador,LF,CR,' Titulo : ',titulo);
  WriteLn(' Data : ',data);
  WriteLn(' Ciclos totais = ',nc);
  Write(' Ciclo atual = ');
  For i := 1 to ncanais do
    begin
      GoToXY(46,16+i);
      Write('Canal -) ',canal[i],' ... Amplificacao -) ',ganho[i]);
    end;
  GoToXY(46,19); Write('Ei = ',vini,' mV');
  GoToXY(46,20); Write('Ef = ',vfi,' mV');
  GoToXY(46,21); Write('ii = ',-vlrmax[canal[1]],' ',unidade);
  GoToXY(46,22); Write('if = ',vlrmax[canal[1]],' ',unidade);
  GoToXY(46,23); Write('Velocidade = ',velo,' mV/s');
  j := 0; Tempo(ti);
  Repeat
    j := succ(j);
    GoToXY(16,23); Write(j);
    i := -1;
    Repeat
      i := i + 1;
      DAC(mV[i]);
      Atualiza(canal[1],ganho[1]);  Le(DV1^[i,j]);
      Atualiza(canal[2],ganho[2]);  Le(DV2^[i,j]);
      PosicionaJanela(1,1,300,98,fundoesc[canal[1]]);
      Plot(x(mV[i],canal[1]),y(DV1^[i,j],canal[1]),1);
      PosicionaJanela(321,1,620,98,fundoesc[canal[2]]);
      Plot(x(mV[i],canal[2]),y(DV2^[i,j],canal[2]),1);
    Until KeyPressed or (i >= ndados);
  Until KeyPressed or (j = nc);
  Tempo(tf); DAC(0); nc := j; ndados := i; ultimo := nc;
  Sound(1760); Delay(200); NoSound;
  tpotot := tf - ti;
  GoToXY(1,24); WriteLn(' Tempo total real de aquisicao -) ',tpotot:0:3,' s');
  GoToXY(23,25); Write('... Tecle (ENTER) para continuar ...');
  Repeat
```

```
    Read(Kbd,tecla);
  Until tecla = CR;
End;


procedure Informacoes;

var   d,m,a,h,mi,s    :  String[2];
      tamoperador     :  Byte Absolute operador;
      tamtitulo       :  Byte Absolute titulo;
      tamanho         :  Byte;

Begin
  Janela(38,4,77,11);
  Date(dia,mes,ano);
  Str(dia:2,d); Str(mes:2,m); Str(ano:2,a);
  If d[1] = ' ' then d[1] := '0';
  If m[1] = ' ' then m[1] := '0';
  If a[1] = ' ' then a[1] := '0';
  data := d + '/' + m + '/' + a;
  Time(hora,min,segu,centseg);
  Str(hora:2,h); Str(min:2,mi); Str(segu:2,s);
  If h[1] = ' ' then h[1] := '0';
  If mi[1] = ' ' then mi[1] := '0';
  If s[1] = ' ' then s[1] := '0';
  horario := h + ':' + mi + ':' + s;
  WriteM('         ... INFORMACOES ...',CR);
  WriteLn(' Data: ',data,'   Hora: ',horario);
  tamanho := tamoperador;
  Write(' Operador -) (',operador,')  '); ReadLn(operador);
  If tamoperador = 0 then tamoperador := tamanho;
  tamanho := tamtitulo;
  Write(' Titulo -) (',titulo,')  '); Read(titulo);
  If tamtitulo = 0 then tamtitulo := tamanho;
End;

procedure LeValores(n : String15; v : Integer);

Begin
  Write(' ',n,' ',v,': Canal -) '); ReadLn(CR,canal[v]);
  Write('            Amplificacao -) '); ReadLn(CR,ganho[v]);
  Write('            Fundo de Escala (mV) -) ');
  ReadLn(CR,fundoesc[canal[v]]);
  Write('            Corresponde a -) ');
  ReadLn(CR,vlrmax[canal[v]]);
  If n = 'Ordenada' then
    begin
      Write('            Titulo -) '); ReadLn(tity[canal[v]]);
      Write('            Unidade -) '); ReadLn(unidy[canal[v]]);
    end
  else
    begin
      Write('            Titulo -) '); ReadLn(titx[canal[v]]);
      Write('            Unidade -) '); ReadLn(unidx[canal[v]]);
    end;
```



```
End;

Begin
  Informacoes;
  tpopul := 1.0; tpoappul := 1.0; npulsos := 1;
  Janela(38,11,77,18);
  GoToXY(9,1); WriteM('... MENU DE SELECAO ...',CR);
  WriteLn(' Modo:  (1) Modo YX');
  WriteLn('        (2) Modo YT');
  WriteLn('        (3) Modo Multiplo');
  Write('        (4) Modo YV  ');
  Repeat
    Read(Kbd,modo);
  Until modo in ['1'..'4'];
  Case modo of
    '1' : begin
            Janela(10,4,60,22);
            Repeat
              ClrScr; GoToXY(18,1); WriteM('... MODO YX ...',CR);
              WriteM(' (ESC) - abandona ',#8);
              Read(Kbd,tecla);
              If tecla = ESC then EXIT;
              GoToXY(1,WhereY);
              LeValores('Ordenada',1);
              LeValores('Abscissa',2);
              Write(' Tempo total de aquisicao (s) -) ');
              ReadRe(CR,tpotot,0,3);
              Repeat
                Repeat
                  GoToXY(1,WhereY);
                  Write(' Intervalo de aquisicao (s) -) ');
                  ReadRe(' ',intaquis,0,3);
                Until intaquis ) 0;
              Until (intaquis ( tpotot) and (intaquis )= 2.0 * TEMPOAQUIS1)
                    and (tpotot / intaquis (= NUMAX);
              ndados := Round(tpotot / intaquis);
              WriteLn;
            Until Confirma(' Condicoes corretas? (S/N) ');
            fx[canal[2]] := 300.0 / (2 * fundoesc[canal[2]]);
            fy[canal[1]] := 98.0 / (2 * fundoesc[canal[1]]);
            minx := -fundoesc[canal[2]];
            aquis := Round(((intaquis - tbi) / tai) / 2.0 - 0.5);
            If aquis (= 0 then aquis := 1;
            If aquis ) 1000 then
              begin
                esp := aquis - 1000;
                aquis := 1000;
              end
            else esp := 0;
            ModoYX;
          end;
    '2' : begin
            Janela(10,7,60,20);
            Repeat
              ClrScr; GoToXY(18,1); WriteM('... MODO YT ...',CR);
```

```
        WriteM(' (ESC) - abandona ',#8);
        Read(Kbd,tecla);
        If tecla = ESC then EXIT;
        GoToXY(1,WhereY);
        LeValores('Ordenada',1);
        WriteLn(' Abscissa Tempo:');
        Write(' Tempo total de aquisicao (s) -) ');
        ReadRe(CR,tpotot,0,3);
        Repeat
          Repeat
            GoToXY(1,WhereY);
            Write(' Intervalo de aquisicao (s) -) ');
            ReadRe(' ',intaquis,0,3);
          Until intaquis ) 0;
        Until (intaquis ( tpotot) and (intaquis )= TEMPOAQUIS2)
              and (tpotot / intaquis (= NUMAX);
        ndados := Round(tpotot / intaquis);
        WriteLn;
      Until Confirma(' Condicoes corretas? (S/N) ');
      fx[canal[1]] := 300.0 / ndados;
      fy[canal[1]] := 98.0 / (2 * fundoesc[canal[1]]);
      titx[canal[1]] := 'Tempo'; unidx[canal[1]] := 's';
      minx := 0;
      aquis := Round((intaquis - tb2) / ta2 - 0.5);
      If aquis (= 0 then aquis := 1;
      If aquis ) 1000 then
        begin
          esp := aquis - 1000;
          aquis := 1000;
        end
      else esp := 0;
      ModoYT;
    end;
'3' : begin
      Janela(10,2,60,24);
      Repeat
        ClrScr; GoToXY(15,1); WriteM('... MODO MULTIPLO ...',CR);
        WriteM(' (ESC) - abandona ',#8);
        Read(Kbd,tecla);
        If tecla = ESC then EXIT;
        GoToXY(1,WhereY);
        tipomult := Confirma(' Deseja pulso de Potencial? (S/N) ');
        WriteLn;
        If tipomult then
          begin
            canal[1] := 0; fundoesc[canal[1]] := 2000;
            Repeat
              GoToXY(1,WhereY);
              Write(' Numero de pulsos (max.=10000) -) ');
              ReadIn(' ',npulsos);
            Until (npulsos )= 1) and (npulsos (= 10000);
            WriteLn;
            Write(' E  Inicial (mV) -) '); ReadIn(CR,vini);
            Write(' E  Final   (mV) -) '); ReadIn(CR,vfi);
            Write(' Escala de i = '); ReadIn(#8,vlrmax[canal[1]]);
```



```
            GoToXY(23,WhereY); Write('Unidade = ',unidade);
            Repeat
              GoToXY(33,WhereY); Read(unidade);
              If (unidade[1] () 'm') and (unidade[1] () 'u')
                then Write(#7);
            Until (unidade[1] = 'm') or (unidade[1] = 'u');
            vunidade := 2; WriteLn;
            Write(' Tempo de pulso (s) -) '); ReadRe(CR,tpopul,0,3);
            Write(' Tempo apos pulso (s) -) '); ReadRe(CR,tpoappul,0,3);
            tpotot := npulsos * (tpopul + tpoappul);
            tity[canal[1]] := 'i'; unidy[canal[1]] := unidade;
            WriteLn(' Canal para Corrente -) 0');
            WriteLn(' Amplificacao -) 0');
            WriteLn(' Fundo de Escala (mV) -) ',fundoesc[canal[1]]);
          end;
        Write(' Nro. de canais (max.=3) -) '); ReadIn(CR,ncanais);
        If (ncanais (= 0) or (ncanais ) 3) then ncanais := 3;
        If tipomult then
          For i := 2 to ncanais do
            LeValores('Ordenada',i)
        else
          For i := 1 to ncanais do
            LeValores('Ordenada',i);
        WriteLn(' Abscissa Tempo:');
        If tipomult then
          WriteLn(' Tempo total de aquisicao (s) -) ',tpotot:0:3)
        else
          begin
            Write(' Tempo total de aquisicao (s) -) ');
            ReadRe(CR,tpotot,0,3);
          end;
        Repeat
          Repeat
            GoToXY(1,WhereY);
            Write(' Intervalo de aquisicao (s) -) ');
            ReadRe(' ',intaquis,0,3);
          Until intaquis ) 0;
        Until (intaquis ( tpotot) and (intaquis )= 3.0 * TEMPOAQUIS3)
              and (tpotot / intaquis (= NUMAX);
        ndpul := Round(tpopul / intaquis);
        ndappul := Round(tpoappul / intaquis);
        ndados := Round(tpotot / intaquis);
        WriteLn;
      Until Confirma(' Condicoes corretas? (S/N) ');
      For i := 1 to ncanais do
        begin
          fy[canal[i]] := 98.0 / (2 * fundoesc[canal[i]]);
          fx[canal[i]] := 300.0 / ndados;
          titx[canal[i]] := 'Tempo'; unidx[canal[i]] := 's';
        end;
      minx := 0;
      aquis := Round(((intaquis - tb3) / ta3) / 3.0 - 0.5);
      If aquis (= 0 then aquis := 1;
      If aquis ) 1000 then
        begin
```

```
          esp := aquis - 1000;
          aquis := 1000;
        end
      else esp := 0;
      If tipomult then
        begin
          WriteLn;
          DAC(0);
          Write('.Espere estabilizar e aperte uma tecla.');
          Repeat
            DAC(0);
          Until KeyPressed;
        end;
      ModoMultiplo;
    end;
'4' : begin
      Janela(10,2,60,24);
      ncanais := 2; canal[1] := 0; fundoesc[canal[1]] := 2000;
      ndados := 1600;
      Repeat
        ClrScr; GoToXY(18,1); WriteM('... MODO YV ...',CR);
        WriteM(' (ESC) - abandona ',#8);
        Read(Kbd,tecla);
        If tecla = ESC then EXIT;
        DAC(0);
        GoToXY(1,2);
        Write(' E  Inicial (mV) -) '); ReadIn(CR,vini);
        Write(' E  Final   (mV) -) '); ReadIn(CR,vfi);
        Write(' Escala de i = '); ReadIn(#8,vlrmax[canal[1]]);
        GoToXY(23,WhereY); Write('Unidade = ',unidade);
        Repeat
          GoToXY(33,WhereY); Read(unidade);
          If (unidade[1] <> 'm') and (unidade[1] <> 'u') then Write(#7);
        Until (unidade[1] = 'm') or (unidade[1] = 'u');
        vunidade := 2; WriteLn;
        WriteLn(' Nro. de canais (max.=2) -) ',ncanais);
        WriteLn(' Canal para Corrente -) 0');
        WriteLn(' Amplificacao -) 0');
        WriteLn(' Fundo de Escala (mV) -) ',fundoesc[canal[1]]);
        titx[canal[1]] := 'E'; unidx[canal[1]] := 'mV';
        tity[canal[1]] := 'i'; unidy[canal[1]] := unidade;
        titx[canal[2]] := 'E'; unidx[canal[2]] := 'mV';
        LeValores('Ordenada',2);
        vm := Round(abs(vini - vfi) / (100 * TEMPOAQUIS4));
        If vm < 1 then vm := 1;
        Repeat
          GoToXY(1,WhereY);
          Write(' Veloc. (max.=',vm,' (mV/s)) = ');
          ReadIn(#8,velo);
        Until velo <= vm;
        tpotot:= 1.0 * abs(vini - vfi) / velo;
        WriteLn;
        WriteLn(' (M) - 1/2 ciclo');
        WriteLn(' (C) - 1 ate 20 ciclos ');
        Repeat
```



```
          Read(Kbd,tecla);
          tecla := UpCase(tecla);
        Until (tecla = 'M') or (tecla = 'C');
        If tecla = 'C' then
          begin
            Repeat
              GoToXY(1,WhereY);
              Write(' Nro. de ciclos = '); ReadIn(#8,nc);
              If (nc < 0) or (nc > 20) then
                begin
                  nc := 0;
                  Write(#7);
                end;
            Until ((nc > 0) and (nc <= 20));
            tpotot:= 2.0 * abs(vini-vfi) / velo;
            WriteLn;
          end;
        DAC(vini);
      Until Confirma(' Condicoes corretas? (S/N) ');
      WriteLn;
      intaquis := tpotot / ndados;
      If intaquis < (2.0 * TEMPOAQUIS4) then
        begin
          intaquis := 2.0 * TEMPOAQUIS4;
          ndados := Round(tpotot/intaquis);
        end;
      fx[canal[1]] := 300.0 / (vfi - vini);
      fx[canal[2]] := 300.0 / (vfi - vini);
      fy[canal[1]] := 98.0 / (2 * fundoesc[canal[1]]);
      fy[canal[2]] := 98.0 / (2 * fundoesc[canal[2]]);
      minx := vini;
      aquis := Round(((intaquis - tb4) / ta4) / 2.0 - 0.5);
      If aquis <= 0 then aquis := 1;
      If aquis > 1000 then
        begin
          esp := aquis - 1000;
          aquis := 1000;
        end
      else esp := 0;
      Case tecla of
        'M'   :  MeioCiclo;
        'C'   :  Ciclos;
      end; { case }
    end;
  end; { case }
End;

Overlay Procedure Salva;

Type RegCiclo = Record
                  DadosC : Vetor;
                end;
RegParametros = Record
                  TituloP,
                  OperadorP      : String15;
```

```
                DataP         : String8;
                unidadeP      : String2;
                modoP         : Char;
                ndadosP,
                ncanaisP,
                ncP,
                viniP,
                vfiP,
                veloP,
                primeiroP,
                ultimoP,
                passoP        : Integer;
                canalP,
                ganhoP        : Array[1..3] of Integer;
                fundoescP,
                vlrmaxP       : Array[0..6] of Integer;
                unidxP, unidyP : Array[0..6] of String2;
                titxP, titYP  : Array[0..6] of String8;
                tpototP       : Real;
              end;
      RegmV = Record
                mVs : Vetor;
              end;


var tecla     : Char;
    i, ciclo  : Integer;


procedure SalvaDados;

var DadosI : File of Integer;
    DadosR : File of Real;

begin
{$I-}
  erro := FALSE;
  Assign(DadosI,nome + '.MDA');
  Assign(DadosR,nome + '.MDT');
  ReWrite(DadosI);
  If IOResult <> 0 then
    begin
      Aviso('Problemas no disco ou disco cheio');
      erro := TRUE;
      Close(DadosI);
      EXIT;
    end;
  ReWrite(DadosR);
  If IOResult <> 0 then
    begin
      Aviso('Problemas no disco ou disco cheio');
      erro := TRUE;
      Close(DadosR);
      EXIT;
    end;
```



```
Case modo of
  '1' : For i := 0 to ndados do
          begin
            Write(DadosI,D1^[i],D2^[i]);
            If IOResult <> 0 then
              begin
                Aviso('Problemas no disco ou disco cheio');
                erro := TRUE;
                Close(DadosI);
                EXIT;
              end;
            Write(DadosR,DT^[i]);
            If IOResult <> 0 then
              begin
                Aviso('Problemas no disco ou disco cheio');
                erro := TRUE;
                Close(DadosR);
                EXIT;
              end;
          end;
  '2' : For i := 0 to ndados do
          begin
            Write(DadosI,D1^[i]);
            If IOResult <> 0 then
              begin
                Aviso('Problemas no disco ou disco cheio');
                erro := TRUE;
                Close(DadosI);
                EXIT;
              end;
            Write(DadosR,DT^[i]);
            If IOResult <> 0 then
              begin
                Aviso('Problemas no disco ou disco cheio');
                erro := TRUE;
                Close(DadosR);
                EXIT;
              end;
          end;
  '3' : For i := 0 to ndados do
          begin
            Write(DadosI,D1^[i],D2^[i],D3^[i]);
            If IOResult <> 0 then
              begin
                Aviso('Problemas no disco ou disco cheio');
                erro := TRUE;
                Close(DadosI);
                EXIT;
              end;
            Write(DadosR,DT^[i]);
            If IOResult <> 0 then
              begin
                Aviso('Problemas no disco ou disco cheio');
                erro := TRUE;
                Close(DadosR);
```

```
                EXIT;
              end;
            end;
      end; { case }
      Close(DadosI);
      Close(DadosR);
    {$I+}
    end;

    procedure SalvaDadosC;

    var UmDado   : RegCiclo;
        ArqDados : file of RegCiclo;

    begin
    {$I-}
      erro := FALSE;
      Assign(ArqDados,nome + '.MDA');
      ReWrite(ArqDados);
      If IOResult () 0 then
        begin
          Aviso('Problemas no disco ou disco cheio');
          erro := TRUE;
          Close(ArqDados);
          EXIT;
        end;
      ciclo := primeiro;
      Repeat
        For i := 0 to ndados do
          UmDado.DadosC[i] := DV2^[i,ciclo];
        Write(ArqDados,UmDado);
        If IOResult () 0 then
          begin
            Aviso('Problemas na gravacao ou disco cheio');
            erro := TRUE;
            Close(ArqDados);
            EXIT;
          end;
        ciclo := ciclo + passo;
      Until ( ciclo ) ultimo);
      Close(ArqDados);
    {$I+}
    end;

    procedure SalvaCiclos;

    var UmCiclo   : RegCiclo;
        ArqCiclo  : file of RegCiclo;
        Onda      : RegmV;
        ArqOnda   : file of RegmV;

    begin
    {$I-}
      erro := FALSE;
      Assign(ArqCiclo,nome + '.MCI');
```



```
      Rewrite(ArqCiclo);
      If IOResult () 0 then
        begin
          Aviso('Problemas no disco ou disco cheio');
          erro := TRUE;
          Close(ArqCiclo);
          EXIT;
        end;
      ciclo := primeiro;
      Repeat
        For i := 0 to ndados do
          UmCiclo.DadosC[i] := DV1^[i,ciclo];
        Write(ArqCiclo,UmCiclo);
        If IOResult () 0 then
          begin
            Aviso('Problemas na gravacao ou disco cheio');
            erro := TRUE;
            Close(ArqCiclo);
            EXIT;
          end;
        ciclo := ciclo + passo;
      Until ( ciclo ) ultimo);
      Close (ArqCiclo);
      Assign(ArqOnda,nome + '.MVO');
      ReWrite(ArqOnda);
      If IOResult () 0 then
        begin
          Aviso('Problemas no disco ou disco cheio');
          erro := TRUE;
          Close(ArqOnda);
          EXIT;
        end;
      With Onda do mVs := mV;
      Write(ArqOnda,Onda);
      If IOResult () 0 then
        begin
          Aviso('Problemas na gravacao ou disco cheio');
          erro := TRUE;
          Close(ArqOnda);
          EXIT;
        end;
      Close(ArqOnda);
    {$I+}
    end;

    procedure SalvaParametros;

    var Parametros    : RegParametros;
        ArqParametros : file of RegParametros;

    begin
    {$I-}
      erro := FALSE;
      Assign(ArqParametros,nome + '.PAR');
      Rewrite(ArqParametros);
```

```
  If IOResult () 0 then
    begin
      Aviso('Problemas no disco ou disco cheio');
      erro := TRUE;
      Close(ArqParametros);
      EXIT;
    end;
  With Parametros do
    begin
      TituloP     := Titulo;
      OperadorP   := Operador;
      DataP       := Data;
      unidadeP    := unidade;
      modoP       := modo;
      ndadosP     := ndados;
      ncanaisP    := ncanais;
      ncP         := nc;
      viniP       := vini;
      vfiP        := vfi;
      veloP       := velo;
      primeiroP   := primeiro;
      ultimoP     := ultimo;
      passoP      := passo;
      For i := 1 to 3 do
        begin
          canalP[i] := canal[i];
          ganhoP[i] := ganho[i];
        end;
      For i := 0 to 6 do
        begin
          fundoescP[i] := fundoesc[i];
          vlrmaxP[i] := vlrmax[i];
          titxP[i] := titx[i];
          unidxP[i] := unidx[i];
          tityP[i] := tity[i];
          unidyP[i] := unidy[i];
        end;
      tpototP := tpotot;
    end;  ( with )
  Write(ArqParametros,Parametros);
  If IOResult () 0 then
    begin
      Aviso('Problemas na gravacao ou disco cheio');
      erro := TRUE;
      Close(ArqParametros);
      EXIT;
    end;
  Close(ArqParametros);
($I+)
end;


Begin
  SalvaParametros;
  If erro then EXIT;
```



```
  Case modo of
    '1'..'3' : SalvaDados;
    '4'      : begin
                 SalvaCiclos;
                 If erro then EXIT;
                 SalvaDadosC;
               end;
  end; ( case )
  Sound(1760); Delay(200); NoSound;
End;


Overlay Procedure Recolhe(s : Char);

Type RegCiclo = Record
                  DadosC : Vetor;
                end;
RegParametros = Record
                  TituloP,
                  OperadorP      : String15;
                  DataP          : String8;
                  unidadeP       : String2;
                  modoP          : Char;
                  ndadosP,
                  ncanaisP,
                  ncP,
                  viniP,
                  vfiP,
                  veloP,
                  primeiroP,
                  ultimoP,
                  passoP         : Integer;
                  canalP,
                  ganhoP         : Array[1..3] of Integer;
                  fundoescP,
                  vlrmaxP        : Array[0..6] of Integer;
                  unidxP, unidyP : Array[0..6] of String2;
                  titxP, tityP   : Array[0..6] of String8;
                  tpototP        : Real;
                end;
        RegmV = Record
                  mVs : Vetor;
                end;

var tecla        : Char;
    i, ciclo     : Integer;

procedure LeDados;

var DadosI : File of Integer;
    DadosR : File of Real;

begin
($I-)
  erro := FALSE;
```

```
Assign(DadosI,nome + '.MDA');
Assign(DadosR,nome + '.MDT');
Reset(DadosI);
If IOResult () 0 then
  begin
    Aviso('Arquivo nao encontrado ou problemas no disco');
    erro := TRUE;
    Close(DadosI);
    EXIT;
  end;
Reset(DadosR);
If IOResult () 0 then
  begin
    Aviso('Arquivo nao encontrado ou problemas no disco');
    erro := TRUE;
    Close(DadosR);
    EXIT;
  end;
Case modo of
  '1' : For i := 0 to ndados do
        begin
          Read(DadosI,D1^[i],D2^[i]);
          If IOResult () 0 then
            begin
              Aviso('Problemas na leitura dos dados');
              erro := TRUE;
              Close(DadosI);
              EXIT;
            end;
          Read(DadosR,DT^[i]);
          If IOResult () 0 then
            begin
              Aviso('Problemas na leitura dos dados');
              erro := TRUE;
              Close(DadosR);
              EXIT;
            end;
        end;
  '2' : For i := 0 to ndados do
        begin
          Read(DadosI,D1^[i]);
          If IOResult () 0 then
            begin
              Aviso('Problemas na leitura dos dados');
              erro := TRUE;
              Close(DadosI);
              EXIT;
            end;
          Read(DadosR,DT^[i]);
          If IOResult () 0 then
            begin
              Aviso('Problemas na leitura dos dados');
              erro := TRUE;
              Close(DadosR);
              EXIT;
```



```
            end;
        end;
  '3' : For i := 0 to ndados do
        begin
          Read(DadosI,D1^[i],D2^[i],D3^[i]);
          If IOResult () 0 then
            begin
              Aviso('Problemas na leitura dos dados');
              erro := TRUE;
              Close(DadosI);
              EXIT;
            end;
          Read(DadosR,DT^[i]);
          If IOResult () 0 then
            begin
              Aviso('Problemas na leitura dos dados');
              erro := TRUE;
              Close(DadosR);
              EXIT;
            end;
        end;
  end; { case }
  Close(DadosI);
  Close(DadosR);
{$I+}
end;

procedure LeDadosC;

var UmDado    : RegCiclo;
    ArqDados  : file of RegCiclo;

begin
{$I-}
  erro := FALSE;
  Assign(ArqDados,nome + '.MDA');
  Reset(ArqDados);
  If IOResult () 0 then
    begin
      Aviso('Arquivo de dados nao encontrado ou problemas no disco');
      erro := TRUE;
      Close(ArqDados);
      EXIT;
    end;
  ciclo := primeiro;
  Repeat
    Read(ArqDados,UmDado);
    If IOResult () 0 then
      begin
        Aviso('Problemas na leitura dos dados');
        erro := TRUE;
        Close(ArqDados);
        EXIT;
      end;
    For i := 0 to ndados do
```

```
      DV2^[i,ciclo] := UmDado.DadosCCi];
    ciclo := ciclo + passo;
  Until ( ciclo ) ultimo);
  Close (ArqDados);
(*I+)
end;

procedure LeCiclos;

var UmCiclo   : RegCiclo;
    ArqCiclo  : file of RegCiclo;
    Onda      : RegmV;
    ArqOnda   : file of RegmV;

begin
(*I-)
  erro := FALSE;
  Assign(ArqCiclo,nome + '.NCI');
  Reset(ArqCiclo);
  If IOResult () 0 then
    begin
      Aviso('Arquivo de dados nao encontrado ou problemas no disco');
      erro := TRUE;
      Close(ArqCiclo);
      EXIT;
    end;
  ciclo := primeiro;
  Repeat
    Read(ArqCiclo,UmCiclo);
    If IOResult () 0 then
      begin
        Aviso('Problemas na leitura dos dados');
        erro := TRUE;
        Close(ArqCiclo);
        EXIT;
      end;
    For i := 0 to ndados do
      DV1^[i,ciclo] := UmCiclo.DadosCCi];
    ciclo := ciclo + passo;
  Until ( ciclo ) ultimo);
  Close (ArqCiclo);
  Assign(ArqOnda,nome + '.HV0');
  Reset(ArqOnda);
  If IOResult () 0 then
    begin
      Aviso('Arquivo nao encontrado ou problemas no disco');
      erro := TRUE;
      Close(ArqOnda);
      EXIT;
    end;
  Read(ArqOnda,Onda);
  If IOResult () 0 then
    begin
      Aviso('Problemas na leitura dos dados');
      erro := TRUE;
```



```
      Close(ArqOnda);
      EXIT;
    end;
  With Onda do mV := mVs;
  Close(ArqOnda);
(*I+)
end;


procedure LeParametros;

var Parametros    : RegParametros;
    ArqParametros : file of RegParametros;

begin
(*I-)
  erro := FALSE;
  Assign(ArqParametros,nome + '.PAR');
  Reset(ArqParametros);
  If IOResult () 0 then
    begin
      Aviso('Arquivo de parametros nao encontrado ou problemas no disco');
      erro := TRUE;
      Close(ArqParametros);
      EXIT;
    end;
  Read(ArqParametros,Parametros);
  If IOResult () 0 then
    begin
      Aviso('Problemas na leitura do arquivo de parametros');
      erro := TRUE;
      Close(ArqParametros);
      EXIT;
    end;
  With Parametros do
    begin
      Titulo     := TituloP;
      Operador   := OperadorP;
      Data       := DataP;
      unidade    := unidadeP;
      modo       := modoP;
      ndados     := ndadosP;
      ncanais    := ncanaisP;
      nc         := ncP;
      vini       := viniP;
      vfi        := vfiP;
      velo       := veloP;
      primeiro   := primeiroP;
      ultimo     := ultimoP;
      passo      := passoP;
      For i := 1 to 3 do
        begin
          canal[i] := canalP[i];
          ganho[i] := ganhoP[i];
        end;
```

```
    For i := 0 to 6 do
      begin
        fundoescC[i] := fundoescP[i];
        vlrmaxC[i] := vlrmaxP[i];
        titxC[i] := titxP[i];
        unidxC[i] := unidxP[i];
        tityC[i] := tityP[i];
        unidyC[i] := unidyP[i];
      end;
    tpotot := tpototP;
  end;  { With }
 Close(ArqParametros);
{$I+}
end;


Begin
 LeParametros;
 If erro then EXIT;
 If s <> 'S' then
   begin
     ClrScr;
     WriteLn('Titulo: ',titulo,CR,LF,'Operador: ',Operador);
   end;
 Case modo of
   '1' : begin
         If s <> 'S' then
           begin
             WriteLn('Tecnica de Y x X');
             WriteLn('Tempo de Aquisicao: ',tpotot:0:3,' s');
             WriteLn('Intervalo de Aquisicao: ',intaquis:0:3,' s');
             WriteLn('Nro. de dados coletados: ',ndados);
           end;
         LeDados;
       end;
   '2' : begin
         If s <> 'S' then
           begin
             WriteLn('Tecnica de Y x T - simples');
             WriteLn('Tempo de Aquisicao: ',tpotot:0:3,' s');
             WriteLn('Intervalo de Aquisicao: ',intaquis:0:3,' s');
             WriteLn('Nro. de dados coletados: ',ndados);
           end;
         LeDados;
       end;
   '3' : begin
         If s <> 'S' then
           begin
             WriteLn('Tecnica de Y x T - multiplo');
             WriteLn('Num. de canais: ',ncanais);
             WriteLn('Tempo de Aquisicao: ',tpotot:0:3,' s');
             WriteLn('Intervalo de Aquisicao: ',intaquis:0:3,' s');
             WriteLn('Nro. de dados coletados: ',ndados);
           end;
         LeDados;
```



```
       end;
   '4' : begin
         If s <> 'S' then
           begin
             WriteLn('Tecnica de Y x V');
             WriteLn('velocidade: ',velo,' mV/s');
             WriteLn('Escala de corrente: ',vlrmaxC[canalC[1]],' ',unidade);
             WriteLn('Nro. de dados coletados/ciclo: ',ndados);
             WriteLn('numero de ciclos: ',nc);
             If nc > 1 then
               begin
                 WriteLn('primeiro ciclo: ',primeiro);
                 WriteLn('ultimo ciclo: ',ultimo);
                 WriteLn('incremento: ',passo);
               end;
           end;
         LeCiclos;
         If erro then EXIT;
         LeDadosC;
       end;
 end; { case }
 If erro then EXIT;
 If s <> 'S' then
   begin
     Sound(1760); Delay(200); NoSound;
     Write('...Aperte uma tecla...');
     Repeat Until KeyPressed;
   end;
End;

Overlay Procedure Diretorio;

  Const  TextSize = 200;

  Type
    String80     =   String[80];
    String15     =   String[15];
    String5      =   String[5];
    String3      =   String[3];
    LineSize     =   String[15];
    TextArrayType =  Array[1..TextSize] of LineSize;

  Var
    FileSpec    :  String80;
    Path        :  String80;
    Name,drv    :  String15;
    AttrList    :  String5;
    DayName     :  String3;
    TextArray   :  TextArrayType;
    FileBytes,
    totbytes    :  Real;
    Code,j,k    :  Integer;
    Attr, Yr,
    Mon, Day,
    Hr, Min,
```

```
  Sec, DayNum,
  ndl         :  Byte;
  tecla,drive,
  opcao       :  Char;


procedure FileInfo(FileSpec : String80; Var FileBytes : Real;
                   Var Yr, Mon, Day, Hr, Min, Sec : Byte;
                   Var AttrList : String5);

  Type
    RegList  =  Record
                  AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                end;
    FileData  =  Record
                   Reserved  :  Array[1..21] of Byte;
                   Attr      :  Byte;
                   Time, Date, SizeLo, SizeHi  :  Integer;
                 end;

  var  Reg  :  RegList;
       DTA  :  Array[1..43] of Byte;
       FD   :  FileData Absolute DTA;

  begin
    Reg.DX := ofs(DTA);
    Reg.DS := seg(DTA);
    Reg.AX := $1A00;
    MSDOS(Reg);
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.CX := $16;
    Reg.AX := $4E00;
    MSDOS(Reg);
    If Lo(Reg.AX) <> 0 then
      begin
        FileBytes := -1.0;
        EXIT
      end;
    FileBytes := Lo(FD.SizeLo) + 256.0 * Hi(FD.SizeLo) +
                 65536.0 * Lo(FD.SizeHi) + 16777216.0 * Hi(FD.SizeHi);
    Yr := (Hi(FD.Date) shr 1) + 80;
    If Yr > 99 then
      Yr := Yr - 100;
    Mon := Lo(FD.Date) shr 5 + (Hi(FD.Date) and $01 shl 3);
    Day := Lo(FD.Date) and $1F;
    Hr := Hi(FD.Time) shr 3;
    Min := Lo(FD.Time) shr 5 + (Hi(FD.Time) and $07 shl 3);
    Sec := (Lo(FD.Time) and $1F) shl 1;
    AttrList := '     ';
    If FD.Attr and $20 = $20 then
      AttrList[1] := 'A';
    If FD.Attr and $10 = $10 then
      AttrList[2] := 'D';
```



```
    If FD.Attr and $02 = $02 then
      AttrList[3] := 'H';
    If FD.Attr and $01 = $01 then
      AttrList[4] := 'R';
    If FD.Attr and $04 = $04 then
      AttrList[5] := 'S';
  end;


procedure NextFile(FileSpec : String80; Attr : Byte;
                   var Code : Integer; Var Name : String15);

  Type
    RegList  =  Record
                  AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                end;

  var  j       :  Integer;
       Reg     :  RegList;
       DTA     :  Array[1..43] of Byte;

  begin
    If (Code < 0) or (Code > 1) then
      begin
        Code := -1;
        EXIT;
      end;
    If Code = 0 then
      begin
        Reg.DX := ofs(DTA);
        Reg.DS := seg(DTA);
        Reg.AX := $1A00;
        MSDOS(Reg);
        FileSpec := FileSpec + Chr(0);
        Reg.DX := ofs(FileSpec[1]);
        Reg.DS := seg(FileSpec[1]);
        Reg.CX := Attr;
        Reg.AX := $4E00;
        MSDOS(Reg);
        If Lo(Reg.AX) <> 0 then
          begin
            Name := '';
            Code := 2;
            EXIT;
          end;
        If DTA[22] and $10 <> 0 then
          Name := '[D]'
        else
          Name := '';
        j := 31;
        While DTA[j] <> 0 do
          begin
            Name := Name + Chr(DTA[j]);
            j := j + 1;
          end;
```

```
        Code := 1;
      end
    else
      begin
        Reg.DX := ofs(DTA);
        Reg.DS := seg(DTA);
        Reg.AX := $4F00;
        MSDOS(Reg);
        If Lo(Reg.AX) = 0 then
          begin
            If DTA[22] and $10 <> 0 then
              Name := '[D]'
            else
              Name := '';
            j := 31;
            While DTA[j] <> 0 do
              begin
                Name := Name + Chr(DTA[j]);
                j := j + 1;
              end;
            Code := 1;
          end
        else
          begin
            Name := '';
            Code := 2;
          end;
      end;
  end;

procedure FileDel(FileSpec : String80; Var Code : Integer);

  Type
    RegList = Record
                AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
              End;

  var Reg : RegList;

  begin
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.AX := $4100;
    MSDOS(Reg);
    Code := Lo(Reg.AX);
    If (Reg.Flags and $01) = 0 then
      Code := 0;
  end;


procedure SortHT(Var TextArray : TextArrayType;
                 LineCount, Position, Width : Integer);

  Label
```



```
    H1, H2, H3, H4, H5, H6, H7, H8;

  var  i, j,
       l, r     :  Integer;
       ThisLine :  LineSize;

  begin
    If LineCount <= 1 then EXIT;
    H1: L := LineCount div 2 + 1;
        R := LineCount;
    H2: If L > 1 then
          begin
            L := L - 1;
            ThisLine := TextArray[L];
          end
        else
          begin
            ThisLine := TextArray[R];
            TextArray[R] := TextArray[1];
            R := R - 1;
            If R = 1 then
              begin
                TextArray[1] := ThisLine;
                EXIT;
              end;
          end;
    H3: j := L;
    H4: i := j;
        j := j + j;
        If j > R then goto H7;
        If j = R then goto H6;
    H5: If Copy(TextArray[j], Position, Width) <
           Copy(TextArray[j + 1], Position, Width) then j := j+1;
    H6: TextArray[i] := TextArray[j];
        goto H4;
    H7: j := i;
        i := j div 2;
    H8: If (Copy(ThisLine, Position, Width) <=
            Copy(TextArray[i], Position, Width)) or (j = L) then
          begin
            TextArray[j] := ThisLine;
            goto H2;
          end
        else
          begin
            TextArray[j] := TextArray[i];
            goto H7;
          end;
  end;


function Tab(TabCol : Integer) : Char;

  begin
    If TabCol > 79 then
```

```
      TabCol := 79
    else
      If TabCol < 1 then
        TabCol := 1;
    GoToXY(TabCol + 1,WhereY);
    Tab := #8;
  end;


Begin
  Janela(5,4,55,20);
  Path := 'A:';
  Attr := $16;
  Code := 0;
  j := 0;
  opcao := 'M';
  Janela(17,6,43,18);
  GoToXY(11,1);
  Write('OPCOES');
  GoToXY(11,2);
  Write('======');
  GoToXY(2,4);
  Write('(M) - Mostra Arquivos');
  GoToXY(2,6);
  Write('(D) - Deleta Arquivos');
  GoToXY(2,8);
  Write('(ESC) - Abandona   ');
  Repeat
    Read(Kbd,opcao);
    opcao := UpCase(opcao);
  Until (opcao = 'M') or (opcao = 'D') or (opcao = #27);
  If opcao = #27 then EXIT;
  GetDir(0,drv);
    Case drv[1] of
      'A'     : drive := 'B';
      'B','C' : drive := 'A';
    end; { case }
  Window(6,5,54,18);  ClrScr;
  GoToXY(1,1);
  If opcao = 'D' then
    Write('DRIVE:ARQUIVOS =      ')
  else
    Write('DRIVE:ARQUIVOS = ',drive,':*.MDA');
  Repeat
    GoToXY(18,1); Read(Kbd,tecla); tecla := UpCase(tecla);
  Until tecla in ['A','B','C',CR,#32];
  If (tecla <> CR) and (tecla <> #32) then
    begin
      Write('       ');
      GoToXY(18,1);
      drive := tecla; Write(drive);
      ReadLn(FileSpec);
      FileSpec := drive + FileSpec;
    end
  else
```



```
    begin
      If opcao = 'D' then
        FileSpec := drive + ':    '
      else
        FileSpec := drive + ':*.MDA';
      WriteLn;
    end;
  If Copy(FileSpec,1,2) = 'A:' then Path := 'A:';
  If Copy(FileSpec,1,2) = 'B:' then Path := 'B:';
  If Copy(FileSpec,1,2) = 'C:' then Path := 'C:';
  If Copy(FileSpec,1,2) = 'D:' then Path := 'D:';
  If (Length(FileSpec) = 2) then
    If (FileSpec[2] = ':') then
      begin
        If opcao = 'D' then
          FileSpec := '.:    '
        else
          FileSpec := '.:*.MDA';
      end;
  If FileSpec[2] <> ':' then FileSpec := '..' + FileSpec;
  FileSpec := Path + copy(FileSpec,3,80);
  Repeat
    NextFile(FileSpec, Attr, Code, Name);
    If Code = 1 then
      begin
        j := j + 1;
        TextArray[j] := Name;
      end;
  Until (Code <> 1);
  WriteLn(j,' Arquivos encontrados');
  SortHT(TextArray, j, 1, 15);
  If opcao = 'D' then
    begin
      ndl := 0;
      For k := 1 to j do
        begin
          FileSpec := Path + TextArray[k];
          FileDel(FileSpec, Code);
          If Code <> 0 then
            begin
              WriteLn(FileSpec,' nao deletado');
              ndl := ndl + 1;
            end;
        end;
      WriteLn(j - ndl,' arquivos deletados');
    end
  else
    begin
      totbytes := 0;
      For k := 1 to j do
        begin
          FileSpec := TextArray[k];
          If FileSpec[1] = 'C' then
            FileSpec := Copy(FileSpec, 4, 12);
          FileSpec := Path + FileSpec;
```

```
            FileInfo(FileSpec,FileBytes, Yr,Mon,Day, Hr,Min,Sec, AttrList);
            Write(TextArray[k], Tab(14), FileBytes:8:0, Tab(24));
            If FileBytes < 0.0 then
              WriteLn('... Arquivo nao encontrado ...')
            else
              begin
                Write(Day:2,'/',Mon:2,'/',Yr, Tab(34));
                Write(Hr:2,':');
                If Min < 10 then WriteLn('0',Min) else WriteLn(Min:2);
                totbytes := totbytes + FileBytes;
              end;
            If ((k/12) = (k div 12)) and (k < j) then
              begin
                WriteLn('  ... Aperte uma tecla ...');
                Read(Kbd,tecla);
              end;
          end;
        WriteLn('       ',totbytes:8:0,' bytes ocupados');
      end;
    Sound(1760); Delay(50);
    Sound(880); Delay(50);
    Sound(440); Delay(50);
    NoSound;
    Write('       ... [Enter] - volta ao MENU ...');
    Read(tecla);
  End;

Procedure Quadro(inicioy,fimy,iniciox,fimx : Real);

var   step, k      :  Real;
      pos, cont    :  Integer;

Begin
  Window(1,1,80,25);
  HiRes; Borda(esq,cima,dir,baixo);
  pos := cima; k := fimy; cont := cima;
  step := (fimy - inicioy) / 10.0;
  Repeat
    Draw(esq,pos,esq-4,pos,1); Draw(dir,pos,dir+4,pos,1);
    If (pos=cont) then
      begin
        GoToXY(1,Round(1.5+18.5*(pos-cima)/(baixo-cima)));
        If (pos=baixo) then
          k := inicioy;
        Write(k:9);
        cont := cont + 2 * (baixo - cima) div 10;
      end;
    pos := pos + (baixo - cima) div 10; k := k - step;
  Until pos > baixo;
  k := iniciox; cont := 0;
  For pos := 0 to 10 do
    begin
      Draw(esq+(dir-esq)*pos div 10,cima-4,esq+(dir-esq)*pos div 10,cima,1);
      Draw(esq+(dir-esq)*pos div 10,baixo,esq+(dir-esq)*pos div 10,baixo+4,1);
      If (pos=cont) then
```



```
        begin
          GoToXY((esq+(dir-esq)*pos div 10) div 8, 22);
          Write(k:9); cont := cont+2;
        end;
      k := k+(fimx-iniciox)/10;
    end;
  GraphWindow(esq,cima,dir,baixo);
End;

Overlay Procedure Grafica(s : Char; can, nrc : Integer);

var i, j, cg : Integer;
    fatconv  : Real;

procedure LeMinMax(canalx,canaly : Integer; eixo,unx,uny : String2);

var xmi,xma,ymi,yma,vlrconvx,vlrconvy : Real;

begin
  WriteLn;
  vlrconvy := vlrmax[canaly] / fundoesc[canaly];
  If modo = 'i' then vlrconvx := vlrmax[canalx] / fundoesc[canalx]
  else vlrconvx := 1.0;
  xmi := minx * vlrconvx;
  xma := maxx * vlrconvx;
  ymi := miny * vlrconvy;
  yma := maxy * vlrconvy;
  xmin := xmi; xmax := xma; ymin := ymi; ymax := yma;
  Write(' Escala eixo ',eixo,': minimo (',unx,') -) '); ReadRe(CR,xmi,0,3);
  Write('               maximo (',unx,') -) '); ReadRe(CR,xma,0,3);
  Write(' Escala eixo Y: minimo (',uny,') -) '); ReadRe(CR,ymi,0,3);
  Write('               maximo (',uny,') -) '); ReadRe(CR,yma,0,3);
  If yma = ymi then
    begin
      ymi := ymi - ymi / 10.0;
      yma := yma + yma / 10.0;
    end;
  If xma = xmi then
    begin
      xmi := xmi - xmi / 10.0;
      xma := xma + xma / 10.0;
    end;
  minx := xmi / vlrconvx;
  maxx := xma / vlrconvx;
  miny := ymi / vlrconvy;
  maxy := yma / vlrconvy;
  fx[1] := 1.0 * (dir - esq) / (maxx - minx);
  fy[1] := 1.0 * (baixo - cima) / (maxy - miny);
  Quadro(miny*vlrconvy, maxy*vlrconvy, minx*vlrconvx, maxx*vlrconvx);
  GoToXY(3,24);
  Write('Abscissa -) ',titx[canalx],': min. = ',
        xmin:0:3,'   max. = ',xmax:0:3);
  GoToXY(3,25);
  Write('Ordenada -) ',tity[canaly],': min. = ',
        ymin:0:3,'   max. = ',ymax:0:3);
```

```
end;

procedure MinMax1;

begin
  GoToXY(1,3); ClrEol;
  minx := D2^[0]; maxx := D2^[0];
  miny := D1^[0]; maxy := D1^[0];
  For i := 1 to ndados do
    begin
      If minx ) D2^[i] then minx := D2^[i];
      If maxx ( D2^[i] then maxx := D2^[i];
      If miny ) D1^[i] then miny := D1^[i];
      If maxy ( D1^[i] then maxy := D1^[i];
    end;
  LeMinMax(canal[2],canal[1],'X',unidx[canal[2]],unidy[canal[1]]);
end;

procedure MinMax2;

begin
  GoToXY(1,3); ClrEol;
  minx := 0; maxx := DT^[ndados];
  miny := D1^[0]; maxy := D1^[0];
  For i := 1 to ndados do
    begin
      If miny ) D1^[i] then miny := D1^[i];
      If maxy ( D1^[i] then maxy := D1^[i];
    end;
  LeMinMax(canal[1],canal[1],'T',unidx[canal[1]],unidy[canal[1]]);
end;

procedure MinMax3;

begin
  Repeat
    GoToXY(1,3); ClrEol;
    Write(' Canal a ser graficado (');
    For i := 1 to ncanais do Write(canal[i],',');
    Write(#8,') -) '); ReadIn(CR,cg);
  Until cg in [canal[1],canal[2],canal[3]];
  minx := 0; maxx := DT^[ndados];
  If cg = canal[1] then
    begin
      miny := D1^[0]; maxy := D1^[0];
      For i := 1 to ndados do
        begin
          If miny ) D1^[i] then miny := D1^[i];
          If maxy ( D1^[i] then maxy := D1^[i];
        end;
    end;
  If cg = canal[2] then
    begin
      miny := D2^[0]; maxy := D2^[0];
      For i := 1 to ndados do
```



```
        begin
          If miny ) D2^[i] then miny := D2^[i];
          If maxy ( D2^[i] then maxy := D2^[i];
        end;
    end;
  If cg = canal[3] then
    begin
      miny := D3^[0]; maxy := D3^[0];
      For i := 1 to ndados do
        begin
          If miny ) D3^[i] then miny := D3^[i];
          If maxy ( D3^[i] then maxy := D3^[i];
        end;
    end;
  LeMinMax(cg,cg,'T',unidx[cg],unidy[cg])
end;

procedure MinMax4;

begin
  Repeat
    GoToXY(1,3); ClrEol;
    Write(' Canal a ser graficado (',canal[1],' ou ',canal[2],') ');
    Write('-) '); ReadIn(CR,cg);
  Until cg in [canal[1],canal[2]];
  Write(' Primeiro ciclo -) '); ReadIn(CR,primeiro);
  Write(' Ultimo     "   -) '); ReadIn(CR,ultimo);
  Write(' Passo          -) '); ReadIn(CR,passo);
  minx := vini; maxx := vfi;
  Case cg of
    0    : begin
             miny := DV1^[0,1]; maxy := DV1^[0,1];
             j := primeiro;
             Repeat
               For i := 1 to ndados do
                 begin
                   If miny ) DV1^[i,j] then miny := DV1^[i,j];
                   If maxy ( DV1^[i,j] then maxy := DV1^[i,j];
                 end;
               j := j + passo;
             Until j ) ultimo;
           end;
    1..6 : begin
             miny := DV2^[0,1]; maxy := DV2^[0,1];
             j := primeiro;
             Repeat
               For i := 1 to ndados do
                 begin
                   If miny ) DV2^[i,j] then miny := DV2^[i,j];
                   If maxy ( DV2^[i,j] then maxy := DV2^[i,j];
                 end;
               j := j + passo;
             Until j ) ultimo;
           end;
  end; ( case )
```

```
  LeMinMax(canal[i],cg,'V',unidx[cg],unidy[cg]);
end;

Begin
  If s <> 'S' then
    begin
      cg := canal[i];
      Janela(10,4,60,21);
      GoToXY(15,1); WriteM('... MENU GRAFICO ...',CR);
      GoToXY(1,3); WriteM(' (ESC) abandona ',#8);
      Read(Kbd,tecla);
      If tecla = ESC then EXIT;
    end;
  Case modo of
    '1' : begin
            If s <> 'S' then MinMax1;
            WriteV(71,3,tity[canal[i]] + ' ' + unidy[canal[i]]);
            GoToXY(50,23); Write(titx[canal[2]] + ' ' + unidx[canal[2]]);
            i := 0;
            Repeat
              i := i + 1;
              Draw(x(D2^[i-1],1),y(D1^[i-1],1),x(D2^[i],1),y(D1^[i],1),1);
            Until (i = ndados) or KeyPressed;
          end;
    '2' : begin
            If s <> 'S' then MinMax2;
            WriteV(71,3,tity[canal[i]] + ' ' + unidy[canal[i]]);
            GoToXY(50,23); Write(titx[canal[i]],' ',unidx[canal[i]]);
            i := 0;
            Repeat
              i := i + 1;
              Draw(x(DT^[i-1],1),y(D1^[i-1],1),x(DT^[i],1),y(D1^[i],1),1);
            Until (i = ndados) or KeyPressed;
          end;
    '3' : begin
            If s <> 'S' then MinMax3
            else cg := can;
            WriteV(71,3,tity[cg] + ' ' + unidy[cg]);
            GoToXY(50,23); Write(titx[cg],' ',unidx[cg]);
            i := 0;
            Repeat
              i := i + 1;
              If cg = canal[1] then
                Draw(x(DT^[i-1],1),y(D1^[i-1],1),x(DT^[i],1),y(D1^[i],1),1);
              If cg = canal[2] then
                Draw(x(DT^[i-1],1),y(D2^[i-1],1),x(DT^[i],1),y(D2^[i],1),1);
              If cg = canal[3] then
                Draw(x(DT^[i-1],1),y(D3^[i-1],1),x(DT^[i],1),y(D3^[i],1),1);
            Until (i = ndados) or KeyPressed;
          end;
    '4' : begin
            If s <> 'S' then MinMax4
            else
              begin
                primeiro := nrc; ultimo := nrc; passo := 1;
```



```
                cg := can;
              end;
            WriteV(71,3,tity[cg] + ' ' + unidy[cg]);
            GoToXY(50,23); Write(titx[cg] + ' ' + unidx[cg]);
            j := primeiro;
            Repeat
              i := 0;
              Repeat
                i := i + 1;
                Case cg of
                  0    : Draw(x(mV[i-1],1),y(DV1^[i-1,j],1),
                              x(mV[i],1),y(DV1^[i,j],1),1);
                  1..6 : Draw(x(mV[i-1],1),y(DV2^[i-1,j],1),
                              x(mV[i],1),y(DV2^[i,j],1),1);
                end; { case }
              Until (i = ndados) or KeyPressed;
              j := j + passo;
            Until KeyPressed or (j > ultimo);
          end;
    end; { case }
    Sound(1760); Delay(200); NoSound;
    If s <> 'S' then
      begin
        Repeat
          Read(Kbd,tecla);
        Until tecla = CR;
        TextMode;
      end;
End;

Overlay Procedure Manipula;

var jacalc, terminou : Boolean;
    tx,ty,op : Char;
    vlrconvy, vlrconvx, cte : Real;
    unix,titx,uniy,tity : String8;
    cg,nrc : Integer;

procedure Permuta;

var funcao : PonteiroR;
    tit    : String8;
    un     : String2;

Begin
  funcao := fcx;
  fcx := fcy;
  fcy := funcao;
  tit := titx;
  titx := tity;
  tity := tit;
  un := unix;
  unix := uniy;
  uniy := un;
End;
```

```
procedure MinMax;

var i : Integer;

Begin
  minx := fcx^[0]; maxx := fcx^[0]; miny := fcy^[0]; maxy := fcy^[0];
  For i := 1 to ndados do
    begin
      If minx ) fcx^[i] then minx := fcx^[i];
      If maxx ( fcx^[i] then maxx := fcx^[i];
      If miny ) fcy^[i] then miny := fcy^[i];
      If maxy ( fcy^[i] then maxy := fcy^[i];
    end;
  xmin := minx; xmax := maxx; ymin := miny; ymax := maxy;
  Janela(3,12,48,20);
  Write(' Escala eixo X: minimo (',unix,') -) '); ReadRe(CR,minx,0,3);
  Write('                maximo (',unix,') -) '); ReadRe(CR,maxx,0,3);
  Write(' Escala eixo Y: minimo (',uniy,') -) '); ReadRe(CR,miny,0,3);
  Write('                maximo (',uniy,') -) '); ReadRe(CR,maxy,0,3);
  If maxy = miny then
    begin
      miny := miny - miny / 10.0;
      maxy := maxy + maxy / 10.0;
    end;
  If maxx = minx then
    begin
      minx := minx - minx / 10.0;
      maxx := maxx + maxx / 10.0;
    end;
  fx[i] := 1.0 * (dir - esq) / (maxx - minx);
  fy[i] := 1.0 * (baixo - cima) / (maxy - miny);
  Quadro(miny, maxy, minx, maxx);
  GoToXY(3,24);
  Write('Abscissa -) ',titx,': min. = ',xmin:0:3,'   max. = ',xmax:0:3);
  GoToXY(3,25);
  Write('Ordenada -) ',tity,': min. = ',ymin:0:3,'   max. = ',ymax:0:3);
End;

procedure ErroFunc(mensagem : String80);

Begin
  ClrScr; Sound(440); Delay(50); Sound(880); Delay(50); Sound(1760);
  Delay(50); NoSound; TextBackground(7); TextColor(0);
  GoToXY(23 - length(mensagem) div 2,5);  Write(mensagem);
  GoToXY(6,14); TextBackground(0); TextColor(7);
  Write('...Aperte uma tecla para continuar...');
  Repeat Until KeyPressed;
End;

procedure Converte(f : Char; cv : PonteiroR; var fc : PonteiroR);

var i : Integer;

Begin
```



```
Case f of
  '1' : begin
          For i := 0 to ndados do
            fc^[i] := cv^[i];
        end;
  '2' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] (= 0 then
                begin
                  ErroFunc('NUMERO MENOR OU IGUAL A ZERO'); EXIT;
                end;
              fc^[i] := ln(cv^[i]);
            end;
        end;
  '3' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] (= 0 then
                begin
                  ErroFunc('NUMERO MENOR OU IGUAL A ZERO'); EXIT;
                end;
              fc^[i] := ln(cv^[i])/ln(10);
            end;
        end;
  '4' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] ) 87 then
                begin
                  ErroFunc('NUMERO MUITO GRANDE'); EXIT;
                end;
              fc^[i] := exp(cv^[i]);
            end;
        end;
  '5' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] ) 37 then
                begin
                  ErroFunc('NUMERO MUITO GRANDE'); EXIT;
                end;
              fc^[i] := exp(cv^[i] * ln(10));
            end;
        end;
  '6' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] (= 0 then
                begin
                  ErroFunc('NUMERO MENOR OU IGUAL A ZERO'); EXIT;
                end;
              fc^[i] := sqrt(cv^[i]);
            end;
        end;
```

```
    '7' : begin
          For i := 0 to ndados do
            begin
              If abs(cv^[i]) ) 9.99E+18 then
                begin
                  ErroFunc('NUMERO MUITO GRANDE'); EXIT;
                end;
              fc^[i] := sqr(cv^[i]);
            end;
        end;
    '8' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] = 0 then
                begin
                  ErroFunc('NUMERO MENOR OU IGUAL A ZERO'); EXIT;
                end;
              fc^[i] := 1/(cv^[i]);
            end;
        end;
    '9' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] (= 0 then
                begin
                  ErroFunc('NUMERO MENOR OU IGUAL A ZERO'); EXIT;
                end;
              fc^[i] := -ln(cv^[i]);
            end;
        end;
    '0' : begin
          For i := 0 to ndados do
            begin
              If cv^[i] (= 0 then
                begin
                  ErroFunc('NUMERO MENOR OU IGUAL A ZERO'); EXIT;
                end;
              fc^[i] := -ln(cv^[i])/ln(10);
            end;
        end;
    'A' : begin
          For i := 0 to ndados do
            fc^[i] := cv^[i] + cte;
        end;
    'B' : begin
          For i := 0 to ndados do
            fc^[i] := cv^[i] * cte;
        end;
  end; { case }
End;

procedure Funcoes;

var posx,posy : Integer;
```



```
Begin
  Janela(30,1,77,25);
  WriteLn('     F(x)                  F(y)');
  WriteLn;
  WriteLn(' 1) x                  1) y');
  WriteLn(' 2) Ln(x)              2) Ln(y)');
  WriteLn(' 3) Log(x)             3) Log(y)');
  WriteLn(' 4) Exp(x)             4) Exp(y)');
  WriteLn(' 5) 10^x               5) 10^y');
  WriteLn(' 6) Sqr(x)             6) Sqr(y)');
  WriteLn(' 7) x^2                7) y^2');
  WriteLn(' 8) 1/x                8) 1/y');
  WriteLn(' 9) -Ln(x)             9) -Ln(y)');
  WriteLn(' 0) -Log(x)            0) -Log(y)');
  WriteLn(' A) + cte.             A) + cte.');
  WriteLn(' B) * cte.             B) * cte.');
  WriteLn;
  Write('Entre com o nro. da funcao do eixo X  -) ');
  posx := WhereX; posY := WhereY;
  Repeat
    GoToXY(posx,posy);
    ReadLn(tx);  tx := UpCase(tx);
  Until tx in ['0'..'9','A','B'];
  Write('Titulo do eixo X -) '); ReadLn(titx);
  Write('Unidade do eixo X -) '); ReadLn(unix);
  If tx in ['A','B'] then
    begin
      Write('Entre o vlr. da cte. -) '); ReadRe(CR,cte,0,3);
    end;
  Write('Entre com o nro. da funcao do eixo Y  -) ');
  posx := WhereX; posY := WhereY;
  Repeat
    GoToXY(posx,posy);
    ReadLn(ty);  ty := UpCase(ty);
  Until ty in ['0'..'9','A','B'];
  Write('Titulo do eixo Y -) '); ReadLn(tity);
  Write('Unidade do eixo Y -) '); ReadLn(uniy);
  If ty in ['A','B'] then
    begin
      Write('Entre o vlr. da cte. -) '); ReadRe(CR,cte,0,3);
    end;
End;

procedure AvisoM;

Begin
  Janela(30,10,55,13);
  Sound(2000); Delay(200); NoSound;
  Write('Precisa definir funcoes');
  Repeat Until KeyPressed;
end;


Begin
  jacalc := FALSE; cg := canal[1]; cte := 1.0; nrc := primeiro;
```

```
Repeat
  Janela(1,1,80,25);
  GoToXY(23,1); WriteM('... MENU DE TRATAMENTO MATEMATICO ...',CR);
  If not(jacalc) then
    begin
      GoToXY(1,3); WriteLn('*  Atencao: Se os dados nao foram arquivados',
                           ' arquive-os antes de prosseguir  *');
      GoToXY(1,6); WriteM(' (ESC) abandona ',#8);
      Read(Kbd,tecla);
      If tecla = ESC then EXIT;
      GoToXY(1,3); ClrEol;
      GoToXY(1,6); ClrEol;
    end;
  Janela(3,3,33,11);
  WriteM('1 - Definir funcoes',CR);
  WriteM('2 - Graficar',CR);
  WriteM('3 - Permutar as coordenadas',CR);
  WriteM('4 - Terminar',CR);
  WriteLn;
  Write('         Opcao -) '); Read(Kbd,op);
  op := UpCase(op);
  If op in ['1'..'4','D','G','P','T'] then Write(op);
  Case op of
    '1','D' : begin
                Funcoes;
                If not(jacalc) then
                  Case modo of
                    '1' : begin
                            vlrconvx := vlrmax[canal[2]] / fundoesc[canal[2]];
                            vlrconvy := vlrmax[canal[1]] / fundoesc[canal[1]];
                            ClrScr;
                            GoToXY(17,11); Write('...AGUARDE...');
                            For i := 0 to ndados do
                              begin
                                fcx^[i] := D2^[i] * vlrconvx;
                                fcy^[i] := D1^[i] * vlrconvy;
                              end;
                          end;
                    '2' : begin
                            vlrconvy := vlrmax[canal[1]] / fundoesc[canal[1]];
                            ClrScr;
                            GoToXY(17,11); Write('...AGUARDE...');
                            For i := 0 to ndados do
                              begin
                                fcx^[i] := DT^[i];
                                fcy^[i] := D1^[i] * vlrconvy;
                              end;
                          end;
                    '3' : begin
                            Janela(3,12,45,20);
                            Write('Canal a ser manipulado (');
                            For i := 1 to ncanais do Write(canal[i],',');
                            Write(#8,') -) ');
                            ReadIn(CR,cg);
                            vlrconvy := vlrmax[cg] / fundoesc[cg];
```



```
                            ClrScr;
                            GoToXY(15,4); Write('...AGUARDE...');
                            For i := 0 to ndados do
                              begin
                                fcx^[i] := DT^[i];
                                If cg = canal[1] then
                                  fcy^[i] := D1^[i] * vlrconvy;
                                If cg = canal[2] then
                                  fcy^[i] := D2^[i] * vlrconvy;
                                If cg = canal[3] then
                                  fcy^[i] := D3^[i] * vlrconvy;
                              end;
                          end;
                    '4' : begin
                            Janela(3,12,45,20);
                            Write('Canal a ser manipulado (',canal[1],
                                  ' ou ',canal[2],') -) ');
                            ReadIn(CR,cg);
                            Write('Nro. do ciclo -) '); ReadIn(CR,nrc);
                            vlrconvy := vlrmax[cg] / fundoesc[cg];
                            ClrScr;
                            GoToXY(15,4); Write('...AGUARDE...');
                            For i := 0 to ndados do
                              begin
                                fcx^[i] := mV[i];
                                If cg = canal[1] then
                                  fcy^[i] := DV1^[i,nrc] * vlrconvy;
                                If cg = canal[2] then
                                  fcy^[i] := DV2^[i,nrc] * vlrconvy;
                              end;
                          end;
                  end ( case )
                else
                  begin
                    ClrScr;
                    GoToXY(17,11); Write('...AGUARDE...');
                  end;
                Converte(tx,fcx,fcx);
                Converte(ty,fcy,fcy);
                jacalc := TRUE;
                Sound(1760); Delay(100); NoSound;
              end;
    '2','G' : begin
                If jacalc then
                  begin
                    MinMax;
                    WriteV(71,3,tity + ' ' + uniy);
                    GoToXY(50,23); Write(titx + ' ' + unix);
                    i := 0;
                    Repeat
                      i := i + 1;
                      Draw(x(fcx^[i-1],1),y(fcy^[i-1],1),
                           x(fcx^[i],1),y(fcy^[i],1),1);
                    Until (i = ndados) or KeyPressed;
                    Sound(1760); Delay(200); NoSound;
```

```
                Repeat
                  Read(Kbd,tecla);
                Until tecla = CR;
                TextMode;
              end
            else AvisoM;
          end;
      '3','P' : begin
            If jacalc then
              begin
                Permuta;
                Sound(1760); Delay(200); NoSound;
              end
            else AvisoM;
          end;
    end; ( case )
  Until op in ['4','T'];
End;

Procedure Superpoe;

var i,ncur,cg,nrc   : Integer;
    nomes           : Array[1..20] of String15;

Begin
  Janela(10,10,60,20);
  ncur := 1;
  ClrScr; GoToXY(15,1); WriteM('...SUPERPOSICAO...',CR);
  WriteM(' (ESC) - abandona ',#8);
  Read(Kbd,tecla);
  If tecla = ESC then EXIT;
  Repeat
    GoToXY(2,2); Write('Numero de curvas (max.=20) -) ');
    ReadIn(CR,ncur);
  Until (ncur ) 0) and (ncur (= 20);
  For i := 1 to ncur do
    begin
      GoToXY(2,3); ClrEol;
      Write('nome ',i,' -) '); LeNome(nomes[i],#32);
    end;
  WriteLn;
  For i := 1 to ncur do
    begin
      nome := nomes[i];
      Recolhe('S');
      If erro then EXIT;
      If i = 1 then
        begin
          cg := canal[i]; nrc := primeiro;
          Case modo of
            '1' : begin
                    minx := -fundoesc[canal[2]]; maxx := fundoesc[canal[2]];
                    miny := -fundoesc[canal[1]]; maxy := fundoesc[canal[1]];
                  end;
            '2' : begin
```



```
                    minx := 0; maxx := tpotot;
                    miny := -fundoesc[canal[1]]; maxy := fundoesc[canal[1]];
                  end;
            '3' : begin
                    Write(' Canal a ser graficado (');
                    For j := 1 to ncanais do Write(canal[j],',');
                    Write(#8,') -) '); ReadIn(CR,cg);
                    minx := 0; maxx := tpotot;
                    miny := -fundoesc[cg]; maxy := fundoesc[cg];
                  end;
            '4' : begin
                    Write(' Canal a ser graficado (',canal[1],' ou ',canal[2],
                          ') -) '); ReadIn(CR,cg);
                    Write(' Nro. do ciclo (1 a ',nc,') -) '); ReadIn(CR,nrc);
                    minx := vini; maxx := vfi;
                    miny := -fundoesc[cg]; maxy := fundoesc[cg];
                  end;
          end; ( case )
          Write(' Minimo valor no eixo X -) '); ReadRe(CR,minx,0,3);
          Write(' Maximo valor no eixo X -) '); ReadRe(CR,maxx,0,3);
          Write(' Minimo valor no eixo Y -) '); ReadRe(CR,miny,0,3);
          Write(' Maximo valor no eixo Y -) '); ReadRe(CR,maxy,0,3);
          fx[i] := 1.0 * (dir - esq) / (maxx - minx);
          fy[i] := 1.0 * (baixo - cima) / (maxy - miny);
          Quadro(miny,maxy,minx,maxx);
        end;
      Grafica('S',cg,nrc);
    end;
  Repeat
    Read(Kbd,tecla);
  Until tecla = CR;
End;

Procedure Menu;

Begin
  TextMode; TextColor(14);  Janela(1,1,80,25);
  Window(1,1,80,25); GoToXY(32,1); Write('PROGRAMA MULTICAN');
  GoToXY(25,24); Write('(C) VALNIR F. JULIANO  -  1990/91');
  Janela(2,3,78,23);  Janela(3,4,33,21);
  GoToXY(9,1); WriteM('... MENU ...',CR); WriteLn;
  WriteM(' 1 - Aquisicao de dados',CR);
  WriteM(' 2 - Salvar dados',CR);
  WriteM(' 3 - Recolher dados do disco',CR);
  WriteM(' 4 - superposicao de Curvas',CR);
  WriteM(' 5 - Diretorio',CR);
  WriteM(' 6 - Graficar',CR);
  WriteM(' 7 - Manipulacao matematica',CR);
  WriteM(' 8 - Terminar',CR);
  GoToXY(10,11); Write('Opcao ... ');
End;


BEGIN
  ClrScr;
```

```
fazaquis := FALSE;
Mark(marcaheap);
D1 := NIL;
New(D1); New(D2); New(D3); New(DT); New(DV1); New(DV2); New(fcx); New(fcy);
GetDir(0,drv);
Case drv[1] of
  'A'         : drive := 'B';
  'B','C','D' : drive := 'A';
end; { case }
Menu;
Repeat
  Read(Kbd,opcao); opcao := UpCase(opcao);
  If opcao in ['1'..'7','A','S','R','C','D','G','M','T'] then Write(opcao);
  Case opcao of
    '1','A' : begin
                If fazaquis then Aquisicao
                else
                  begin
                    ClrScr;
                    Window(1,1,80,25);
                    TelaInicio('MULTI01');
                    Repeat
                      Read(Kbd,tecla);
                    Until tecla = CR;
                    TelaInicio('MULTI02');
                    GoToXY(31,25); TextColor(14);
                    Write('...A G U A R D E...');
                    GoToXY(60,25); Write('Processando...');
                    CalcCont; Inicializa;
                    DeterminaTempo;
                    GoToXY(5,25); ClrEol;
                    Sound(1000); Delay(100); Sound(500);
                    Delay(100); NoSound;
                    GoToXY(30,25); TextColor(31);
                    Write('...APERTE UMA TECLA...');
                    Repeat Until KeyPressed;
                    fazaquis := TRUE;
                  end;
                Menu;
              end;
    '2','S' : begin
                Janela(10,4,65,18);
                WriteLn('Qual o nome do arquivo? (maximo 8 caracteres)');
                LeNome(nome,CR);
                If modo = '4' then
                  begin
                    primeiro := 1; ultimo := nc; passo := 1;
                    Write('Primeiro ciclo -) '); ReadIn(CR,primeiro);
                    If (primeiro < 1) or (primeiro > nc) then
                      primeiro := 1;
                    Write('Ultimo ciclo -) '); ReadIn(CR,ultimo);
                    If (ultimo < primeiro) or (ultimo > nc) then
                      ultimo := nc;
                    Write('Passo -) '); ReadIn(CR,passo);
                    If passo <= 0 then passo := 1;
```



```
                  end;
                WriteLn('Arquivo de dados: ',nome,'.MDA');
                If modo in ['1'..'3'] then
                  WriteLn('  "        "    ',nome,'.MDT');
                If modo = '4' then
                  begin
                    WriteLn('  "        "    ',nome,'.MCI');
                    WriteLn('  "        "    ',nome,'.MVO');
                  end;
                WriteLn('Arquivo de parametros: ',nome,'.PAR');
                If Confirma('Confirma? (S/N)') then Salva;
                Menu;
              end;
    '3','R' : begin
                Janela(10,4,65,17);
                WriteLn('Qual o nome do arquivo? (maximo 8 caracteres)');
                LeNome(nome,CR);
                WriteLn('Arquivo de dados: ',nome,'.MDA');
                WriteLn('  "        "  : ',nome,'.MDT');
                WriteLn('  "        "    ',nome,'.MCI (modo Y x V)');
                WriteLn('  "        "    ',nome,'.MVO      "     ');
                WriteLn('Arquivo de parametros: ',nome,'.PAR');
                If Confirma('Confirma? (S/N)') then Recolhe(' ');
                Menu;
              end;
    '4','C' : begin
                Superpoe; Menu;
              end;
    '5','D' : begin
                Diretorio; Menu;
              end;
    '6','G' : begin
                Grafica(' ',1,1); Menu;
              end;
    '7','M' : begin
                Manipula; Menu;
              end;
    'B','T' : begin
                Janela(30,9,54,12);
                If not Confirma('Confirma saida? (S/N) ') then
                  begin
                    opcao := ' ';
                    Menu;
                  end;
              end;
  end; { case }
Until opcao in ['T','B'];
Release(marcaheap);
Window(1,1,80,25); ClrScr;
END.
```

A1.4. PROGRAMA VOLTAM

```
Program VOLTAM;

{ Este programa recolhe os dados de voltametria ciclica e permite obter carga,}
{ Eo, E1/2, Ep1/2, delta Ep, ipa e ipc e fazer superposicao de voltamogramas. }

  Const  versao = '2.0';

  Type String15    =  String[15];
       String10    =  String[10];
       String8     =  String[8];
       String2     =  String[2];
       RegArquivo  =  Record
                        ddx,ddy  :  Array[1..400] of Real;
                        nro      :  Integer;
                      end;

  Var  registro    :  RegArquivo;
       arqdado     :  File of RegArquivo;
       D1          :  Array[0..1600,1..20] of Integer Absolute $3400:$0000;
       D2          :  Array[0..1600,1..20] of Integer Absolute $4400:$0000;
       D3          :  Array[0..1600,1..20] of Integer Absolute $5400:$0000;
       Q           :  Array[0..4010] of Real Absolute $6400:$0000;
       mV          :  Array[0..1600] of Integer Absolute $6400:$CF08;
       csh         :  Array[1..69] of Integer;
       csv         :  Array[1..156] of Integer;
       i,nc,
       imax,imin,
       iini,ifi,
       vmax,
       vmin,
       vini, vfi,
       corrente,
       intervalo,
       incremento,
       espera,
       primeiro,
       ultimo,
       passo,
       nq, tpo,
       tpoa,tpod,
       tpotot,
       potin,potfi,
       npulsos,
       maxcorrente :  Integer;
       tecla,
       tecnica     :  Char;
       ganho       :  Byte;
       operador,
       amostra     :  String[20];
       data        :  String8;
       tipo,
       unidade,unid :  String2;
       tempo,
       tempopto,
       velo, incr,
```



```
       fatortt, fs,
       fEi         :  Real;
       vunid       :  Byte Absolute unid;
       erro        :  Boolean;


procedure Graphics;                                    external 'GRAPH.BIN';
procedure GraphMode;                                   external Graphics[0];
procedure GraphColorMode;                              external Graphics[3];
procedure HiRes;                                       external Graphics[6];
procedure HiResColor(Color: Integer);                  external Graphics[9];
procedure GraphBackground(Color: Integer);             external Graphics[15];
procedure GraphWindow(X1,Y1,X2,Y2: Integer);           external Graphics[18];
procedure Plot(X,Y,Color: Integer);                    external Graphics[21];
procedure Draw(X1,Y1,X2,Y2,Color: Integer);            external Graphics[24];
procedure GetPic(var Buffer; X1,Y1,X2,Y2: Integer);    external Graphics[36];
procedure PutPic(var Buffer; X,Y: Integer);            external Graphics[39];
procedure ClearScreen;                                 external Graphics[60];


Procedure Borda(x1,y1,x2,y2 : Integer);

  Begin
    GraphWindow(0,0,639,199);
    Draw(x1,y1,x2,y1,1);
    Draw(x2,y1,x2,y2,1);
    Draw(x2,y2,x1,y2,1);
    Draw(x1,y1,x1,y2,1);
  End;


Procedure Janela(x1,y1,x2,y2 : Integer);


  Var    i    : Integer;

  Begin
     Window (1,1,80,25);
     If (y2 = 25) and (x2 = 80) then y2 := 24;
     GoToXY (x1,y2); Write (#212);
     For i := x1+1 to x2 - 1 do Write(#205);
     Write (#190);
     GoToXY (x1,y1); Write (#213);
     For i := x1+1 to x2 - 1 do Write(#205);
     Write (#184);
     For i := y1+1 to y2 - 1 do
       begin
         GoToXY (x1,i); Write(#179);
         GoToXY (x2,i); Write(#179);
       end;
     If y2 - y1 > 2 then
       begin
         Window (x1+1,y1+1,x2-1,y2-1);
         ClrScr;
       end
```

```
    else Window (x1+1,y1,x2-1,y2);
  End;


Procedure Menu;

  Begin
    ClrScr;
    Janela(1,1,80,25);
    GoToXY(8,22);
    Write('PROGRAMA  * VOLTAM * - Versao ',versao,'          (C) VFJ - 1990');
    Janela(4,3,34,22);
    GoToXY(8,1);
    Write('MENU INICIAL');
    GoToXY(8,2);
    Write('============');
    GoToXY(4,4);
    Write('A - Arquivo em disco');
    GoToXY(4,6);
    Write('M - Manipulacao dos dados');
    GoToXY(4,8);
    Write('R - Resultados');
    GoToXY(4,10);
    Write('V - Varios Voltamogramas');
    GoToXY(4,12);
    Write('S - Salvar dados');
    GoToXY(4,14);
    Write('D - Diretorio');
    GoToXY(4,16);
    Write('F - Finalizar');
  End;


Procedure NumeroInteiro(spc:Integer; c1,c2:Char; Var numero_inteiro : Integer);

Var   tecla         : Char;
      vlr           : String[20];
      err, i        : Integer;
      endvlr        : Byte absolute vlr;


Begin
  Repeat
    Read(Kbd,tecla); tecla := UpCase(tecla);
    Case tecla of
      '-',
      '0'..'9' : begin
                   For i := 1 to spc do
                     Write(' ');
                   GoToXY(WhereX-spc,WhereY);
                   i := 1; vlr[i] := tecla; Write(vlr[i]);
                   Repeat
                     Read(Kbd,tecla);
                     Case tecla of
                       '-',
```



```
                       '0'..'9' : begin
                                    if (tecla = '-')
                                    and ((pos('-',vlr) <> 0) or (i >= 1))
                                      then Write(#7)
                                    else
                                      begin
                                        i := i + 1;
                                        vlr[i] := tecla;
                                        Write(vlr[i]);
                                      end;
                                  end;
                       #8       : begin
                                    If i > 0 then
                                      begin
                                        i := i - 1;
                                        Write(#8,#32,#8);
                                        vlr[i + 1] := #32;
                                      end
                                    else
                                      Write(#7);
                                  end;
                     end;
                   endvlr := i;
                   Until (tecla = #13);
                   val(vlr,numero_inteiro,err);
                 end;
      end;
    Until (tecla = #13);
    Write(c1,c2);
  End;


Procedure NumeroReal(spc : Integer; c1,c2 : Char; Var numero_real : Real);

Var   tecla         : Char;
      vlr           : String[20];
      err, i        : Integer;
      endvlr        : Byte absolute vlr;


Begin
  Repeat
    Read(Kbd,tecla); tecla := UpCase(tecla);
    Case tecla of
      '-',
      '0'..'9' : begin
                   For i := 1 to spc do
                     Write(' ');
                   GoToXY(WhereX-spc,WhereY);
                   i := 1; vlr[i] := tecla; Write(vlr[i]);
                   Repeat
                     Read(Kbd,tecla); tecla := UpCase(tecla);
                     Case tecla of
                       '+','-',
                       '.','E',
```

```
                '0'..'9' : begin
                           if ((tecla = '-') and (pos('-',vlr) <> 0)
                             and (vlr[i] <> 'E'))
                           or ((tecla = '+') and (vlr[i] <> 'E'))
                           or ((tecla = 'E') and (pos('E',vlr) <> 0))
                           or ((tecla = ',') and (pos(',',vlr) <> 0))
                             then Write(#7)
                           else
                             begin
                               i := i + 1;
                               vlr[i] := tecla;
                               Write(vlr[i]);
                             end;
                         end;
                #8       : begin
                             If i > 0 then
                               begin
                                 i := i - 1;
                                 Write(#8,#32,#8);
                                 vlr[i + 1] := #32;
                               end
                             else
                               Write(#7);
                           end;
              end;
            endvlr := i;
            Until (tecla = #13);
            val(vlr,numero_real,err);
          end;
    end;
  Until (tecla = #13);
  Write(c1,c2);
End;


Procedure Aviso(informa : String15);

  Begin
    Sound(2000); Delay(50); Sound(1000); Delay(50); Sound(500); Delay(50);
    NoSound;
    WriteLn('...Arquivo de ',informa,' nao encontrado...');
    WriteLn('   ...Aperte uma tecla para continuar...');
    Repeat Until KeyPressed;
  End;


Overlay Procedure Arquivo(tipo : Char; nomearq : String[0; var er : Boolean);

  Const LF = #10;
        CR = #13;

  Type RegCiclo = Record
                    Dados : Array[0..1600] of Integer;
                  end;
       RegmV    = Record
```



```
                   mV : Array[0..1600] of Integer;
                 end;
       RegParametros = Record
                         unidadeP    : String[2];          { CA e V }
                         AmostraP,                         { T }
                         OperadorP   : String[20];         { T }
                         DataP       : String8;            { T }
                         tecnicaP    : Char;               { T }
                         fatorttP    : Real;               { T }
                         ganhoP      : Byte;               { T }
                         intervaloP,                       { T }
                         tpoP,                             [ T }
                         tpoaP,                            { CA e CP }
                         tpodP,                            { CA e CP }
                         tpototP,                          { CA e CP }
                         npulsosP,                         { CA }
                         ncP,                              { V }
                         vmaxP,                            { CP }
                         vminP,                            { CP }
                         viniP,                            { CA e V }
                         vfiP,                             { CA e V }
                         correnteP   : Integer;            { CA e V }
                         veloP,                            { V }
                         incrP       : Real;               { V }
                         primeiroP,                        { V }
                         ultimoP,                          { V }
                         passoP      : Integer;            { V }
                         tempoptoP,                        { CC }
                         tempoP      : Real;               { CA e CP }
                         incrementoP,                      { CC }
                         nqP,                              { CC }
                         iiniP,                            { CP }
                         ifiP  : Integer;                  { CP }
                       end;


  var    Parametros    : RegParametros;
         ArqParametros : file of RegParametros;
         Onda          : RegmV Absolute $6400:$CF08;
         ArqOnda       : file of RegmV;
         nome,drv      : String[10];
         tecla,drive   : Char;
         i, ciclo,
         controle      : Integer;


  procedure LeCiclos;

    var UmCiclo      : RegCiclo;
        ArqCiclo     : file of RegCiclo;

    begin
      Assign(ArqCiclo,nome + '.DAT');
{$I-}
      Reset(ArqCiclo);
```

```
{$I+}
      If IOResult <> 0 then
        begin
          Aviso('dados');
          controle := 1;
          er := TRUE;
          EXIT;
        end;
      ciclo := primeiro;
      Repeat
        Read(ArqCiclo,UmCiclo);
        Case ciclo of
          1..20   : For i := 0 to 1600 do D1[i,ciclo] := UmCiclo.Dados[i];
          21..40  : For i := 0 to 1600 do D2[i,ciclo-20] := UmCiclo.Dados[i];
          41..60  : For i := 0 to 1600 do D3[i,ciclo-40] := UmCiclo.Dados[i];
          end; { case }
        ciclo := ciclo + passo;
      Until ( ciclo > ultimo);
      Close (ArqCiclo);
      Assign(ArqOnda,nome + '.VOL');     { recupera a matriz aV }
{$I-}
      Reset(ArqOnda);
{$I+}
      If IOResult <> 0 then
        begin
          Aviso('dados');
          controle := 1;
          er := TRUE;
          EXIT;
        end;
      Read(ArqOnda,Onda);
      Close(ArqOnda);
    end;

  procedure LeParametros;

    begin
      Assign(ArqParametros,nome + '.PAR');
{$I-}
      Reset(ArqParametros);
{$I+}
      If IOResult <> 0 then
        begin
          Aviso('parametros');
          controle := 1;
          er := TRUE;
          EXIT;
        end;
      Read(ArqParametros,Parametros);
      With Parametros do
        begin
          unidade    := unidadeP;
          Amostra    := AmostraP;
          Operador   := OperadorP;
          Data       := DataP;
```



```
          tecnica    := tecnicaP;
          fatortt    := fatorttP;
          ganho      := ganhoP;
          intervalo  := intervaloP;
          tpo        := tpoP;
          tpoa       := tpoaP;
          tpod       := tpodP;
          tpotot     := tpototP;
          npulsos    := npulsosP;
          nc         := ncP;
          vmax       := vmaxP;
          vmin       := vminP;
          vini       := viniP;
          vfi        := vfiP;
          corrente   := correnteP;
          velo       := veloP;
          incr       := incrP;
          primeiro   := primeiroP;
          ultimo     := ultimoP;
          passo      := passoP;
          tempopto   := tempoptoP;
          tempo      := tempoP;
          incremento := incrementoP;
          nq         := nqP;
          iini       := iiniP;
          ifi        := ifiP;
        end;  { With }
      Close(ArqParametros);
    end;


Begin
  controle := 0; er := FALSE;
  If tipo = 'A' then
    begin
      Janela(5,4,60,20);
      Repeat
        ClrScr; GoToXY(12,1);
        Write('RECOLHE DADOS GRAVADOS EM DISCO');
        GoToXY(2,3); Write('<ESC> - Abandona  '); Read(Kbd,tecla);
        If tecla = #27 then EXIT;
        GetDir(0,drv);
        Case drv[1] of
        'A'         : drive := 'B';
        'B','C','D' : drive := 'A';
        end;
        GoToXY(2,3); ClrEol;
        Write('Nome do arquivo (max. 8 caracteres) = '); ReadLn(nome);
        If nome[2] <> ':' then nome := drive + ':' + nome;
        If Pos('.',nome) > 0 then
          nome := Copy(nome,1,Pos('.',nome)-1);
        For i := 1 to Length(nome) do nome[i] := UpCase(nome[i]);
        WriteLn(CR,LF,' Arquivo de dados: ',nome + '.DAT',LF);
        WriteLn('                  ',nome + '.VOL',LF);
        WriteLn(' Arquivo de parametros: ',nome + '.PAR',LF);
```

```
          WriteLn(' Confirma? (S/N)'); Read(kbd,tecla);
        Until (UpCase(tecla) = 'S');
        LeParametros;
        If er then EXIT;
        ClrScr;
        WriteLn(' Amostra: ',Amostra,CR,LF,' Operador: ',Operador,LF);
        WriteLn(' Tipo de Arquivo : Voltametria');
        WriteLn(' Escala de corrente: ',corrente,' ',unidade);
        WriteLn(' Varredura de ',vini,' a ',vfi,' mV');
        WriteLn(' velocidade: ',velo:5:2,' mV/s');
        WriteLn(' numero de ciclos: ',nc);
        If nc > 1 then
          begin
            WriteLn(' primeiro ciclo:   ',primeiro);
            WriteLn(' ultimo ciclo:     ',ultimo);
            WriteLn(' incremento:       ',passo);
          end;
        LeCiclos;
        If er then EXIT;
        WriteLn(CR,LF,'  ...Aperte uma tecla...');
        Repeat Until KeyPressed;
      end
    else
      begin
        nome := nomearq;
        LeParametros;
        If er then EXIT;
        LeCiclos;
        If er then EXIT;
      end;
  End;

Procedure SeparaOverlay1;

  Begin
  End;

Overlay Procedure Manipula(modo : Char; nro_voltas : Byte);

  Var   tecla, opcao                          :  Char;
        xmin, ymin, xmax, ymax, iniciox, fimx,
        inicioy, fimy,passax, passay,
        divx, divy,posx, posy, x, y, i, j,
        i1, i2, e1, e2, nciclo,
        controlaIntegral, controlaVoltam,
        controlaAmpero, controlaPotenc        :  Integer;
        normaly, normalx, inicioqx, fimqx,
        inicioqy, fimqy, Qmax, Qmin,
        cargaliq, epa, epc, ipa, ipc, ypa, ypc,
        ye, yd, xe, xd, fx, fy, td , m        :  Real;


  function px(x : Integer) : Integer;

    begin
```



```
      px := Round(fx*(x - xmin));
    end;


  function py(y : Integer) : Integer;

    begin
      py := Round(fy*(ymax - y));
    end;


  procedure Linha(x1,y1,x2,y2 : Integer);

    begin
      Draw(px(x1),py(y1),px(x2),py(y2),1);
    end;


  procedure Condicoes;

    begin
      GoToXY(posx,posy);
      If (tipo <> 'Q') and (modo <> 'S') then
        begin
          If (corrente = 2000) or (corrente = 1000) then
            begin
              Write(' (max=',(imax*fEi):0:0,')');
              WriteLn(' (min=',(imin*fEi):0:0,')');
            end;
          If corrente = 200 then
            begin
              Write(' (max=',(imax*fEi):0:1,')');
              WriteLn(' (min=',(imin*fEi):0:1,')');
            end;
          If corrente = 20 then
            begin
              Write(' (max=',(imax*fEi):0:2,')');
              WriteLn(' (min=',(imin*fEi):0:2,')');
            end;
          If corrente = 2 then
            begin
              Write(' (max=',(imax*fEi):0:3,')');
              WriteLn(' (min=',(imin*fEi):0:3,')');
            end;
          Write('Veloc. de varredura = ',velo:5:2,' mV/s')
        end
      else
        If tipo = 'Q' then
          begin
            WriteLn(' (Q total = ',(Q[nq]*fEi):10,')');
            Write('Veloc. de varredura = ',velo:5:2,' mV/s');
          end;
    end;
```

```
procedure Integral;

  var  i, j   :  Integer;

  begin
    Qmax := 0; Qmin := 0;
    If (vini < vfi) then
      If (e1 < e2) then
        begin
          i1 := Round((e1 - vini) / incr);
          i2 := Round((e2 - vini) / incr);
        end
      else
        begin
          i1 := Round(((vfi - e1) / incr) + tpo div 2) + 1;
          i2 := Round(((vfi - e2) / incr) + tpo div 2) + 1;
        end
    else
      If (e1 > e2) then
        begin
          i1 := Round((e1 - vini) / incr);
          i2 := Round((e2 - vini) / incr);
        end
      else
        begin
          i1 := Round(((vfi - e1) / incr) + tpo div 2) + 1;
          i2 := Round(((vfi - e2) / incr) + tpo div 2) + 1;
        end;
    Q[0] := 0;  j := 1;
    Case nciclo of
      1..20  :  For i := i1 + 1 to i2 do
                begin
                  Q[j] := abs(mv[i]-mV[i-1]) *
                          ((D1[i,nciclo]+D1[i-1,nciclo])/2);
                  Q[j] :=  Q[j] / velo + Q[j-1];
                  If Qmax <= Q[j] then Qmax := Q[j];
                  If Qmin >= Q[j] then Qmin := Q[j];
                  j := j + 1;
                end;
      21..40 :  For i := i1 + 1 to i2 do
                begin
                  Q[j] := abs(mv[i]-mV[i-1]) *
                          ((D2[i,nciclo-20]+D2[i-1,nciclo-20])/2);
                   Q[j] :=  Q[j] / velo + Q[j-1];
                   If Qmax <= Q[j] then Qmax := Q[j];
                   If Qmin >= Q[j] then Qmin := Q[j];
                   j := j + 1;
                end;
      41..60 :  For i := i1 + 1 to i2 do
                begin
                  Q[j] := abs(mv[i]-mV[i-1]) *
                          ((D3[i,nciclo-40]+D3[i-1,nciclo-40])/2);
                  Q[j] :=  Q[j] / velo + Q[j-1];
                  If Qmax <= Q[j] then Qmax := Q[j];
                  If Qmin >= Q[j] then Qmin := Q[j];
```



```
                  j := j + 1;
                end;
    end; ( case )
    nq := j - 1; Sound(2440); Delay(50); Sound(1220); Delay(50);
    Sound(610); Delay(50); NoSound; controlaIntegral := 1;
  end;


procedure Cursor;

  var     ch                  :  Array [1..69] of Integer;
          cv                  :  Array [1..156] of Integer;
          poscx,poscy,incr    :  Integer;
          mov                 :  Char;


  const   ESC = #27;


  procedure novapos;

    begin
      GetPic(ch,0,poscy,500,poscy);
      GetPic(cv,poscx,0,poscx,150);
    end;

  procedure antpos;

    begin
      PutPic(ch,0,poscy);
      PutPic(cv,poscx,150);
    end;

  procedure novocs;

    begin
      PutPic(csh,0,poscy);
      PutPic(csv,poscx,150);
    end;


  begin
    incr := 1; poscx := 250; poscy := 75; novapos; novocs;
    epa := 0.0;  epc := 0.0;  m := 0.0;  ye := 0;  yd := 0;
    xe := 0.0;  xd := 0.0;  ipa := 0.0;  ipc := 0.0;
    GoToXY(70,25); Write('csr -> ',incr:2);
    Repeat
      Read(kbd,mov);
      If (mov = ESC) and KeyPressed then
        begin
          Read(kbd,mov);
          Case mov of
    (F1)   #59  :  begin      ( determina Epa )
                     epa := xmin + poscx / fx;  ypa := ymax - poscy / fy;
                     Sound(2000); Delay(50); Sound(1000); Delay(50);
```

```
                Sound(500); Delay(50); NoSound;
              end;
(F2)  #60 :   begin      { determina Epc }
                epc := xmin + poscx / fx;  ypc := ymax - poscy / fy;
                Sound(3000); Delay(50); Sound(2000); Delay(50);
                Sound(1000); Delay(50); NoSound;
              end;
(F3)  #61 :   begin{determina coef. angular da reta que passa por Eo}
                If (epa <> epc) then
                 begin
                   m := (ypa - ypc)/(epa - epc);
                   GoToXY(70,5); Write('  ANODO');
                   GoToXY(70,6); Write('          ');
                   GoToXY(70,6); Write('E = ',Round(epa));
                   GoToXY(70,9); Write(' CATODO');
                   GoToXY(70,10); Write('          ');
                   GoToXY(70,10); Write('E = ',Round(epc));
                   GoToXY(70,13); Write('          ');
                   GoToXY(70,13); Write('Eo = ',Round(epa-ypa/m));
                   Sound(1760); Delay(50); Sound(880); Delay(50);
                   Sound(440); Delay(50); NoSound;
                 end;
              end;
(F4)  #62 :   begin
                If (ipa <> 0.0) and (ipc <> 0.0) then
                  begin
                    GoToXY(70,15); Write('Ipa/Ipc =');
                    GoToXY(72,16); Write(abs(ipa/ipc):0:4);
                    Sound(1760); Delay(50); Sound(880); Delay(50);
                    Sound(440); Delay(50); NoSound;
                  end;
              end;
(F5)  #63 :   begin       { determina coordenada a esquerda }
                ye := ymax - poscy / fy; xe := xmin + poscx / fx;
                Sound(2000); Delay(50); Sound(1000); Delay(50);
                Sound(500); Delay(50); NoSound;
              end;
(F6)  #64 :   begin       { determina coordenada a direita }
                yd := ymax - poscy / fy; xd := xmin + poscx / fx;
                Sound(2000); Delay(50); Sound(1000); Delay(50);
                Sound(500); Delay(50); NoSound;
              end;
(F7)  #65 :   begin    { calcula ipa liquida }
                If (xe < xd) then
                 begin
                   m := (ye - yd)/(xe - xd);
                   ipa := (ypa - (ye + m * (epa - xe))) * fEi;
                   GoToXY(70,7); Write('          '); GoToXY(70,7);
                   If corrente >= 200 then Write('i = ',ipa:0:1);
                   If corrente = 20 then Write('i = ',ipa:0:2);
                   If corrente = 2 then Write('i = ',ipa:0:3);
                   registro.ddy[registro.nro] := ipa;
                   Sound(3000); Delay(50); Sound(2000); Delay(50);
                   Sound(1000); Delay(100);
                   Sound(1760); Delay(50); Sound(880); Delay(50);
```



```
                   Sound(440); Delay(50); NoSound;
                 end;
              end;
(F8)  #66 :   begin    { calcula ipc liquida }
                If (xe < xd) then
                  begin
                    m := (ye - yd)/(xe - xd);
                    ipc := (ypc - (ye + m * (epc - xe))) * fEi;
                    GoToXY(70,11); Write('          '); GoToXY(70,11);
                    If corrente >= 200 then Write('i = ',ipc:0:1);
                    If corrente = 20 then Write('i = ',ipc:0:2);
                    If corrente = 2 then Write('i = ',ipc:0:3);
                    registro.ddy[registro.nro] := ipc;
                    Sound(3000); Delay(50);
                    Sound(2000); Delay(50); Sound(1000); Delay(100);
                    Sound(1760); Delay(50); Sound(880); Delay(50);
                    Sound(440); Delay(50); NoSound;
                  end;
              end;
(F9)  #67 :   begin
                incr := incr div 2; { increm. do desloc. do cursor }
                If incr < 1 then incr := 1
                else
                  Sound(440); Delay(50); NoSound;
                GoToXY(70,25); Write('csr -> ',incr:2);
              end;
(F10) #68 :   begin
                incr := incr * 2; { increm. do desloc. do cursor }
                If incr > 64 then incr := 64
                else
                  Sound(440); Delay(50); NoSound;
                GoToXY(70,25); Write('csr -> ',incr:2);
              end;
(esq) #75 :   begin
                antpos; poscx := poscx - incr;
                If poscx < 1 then
                  begin
                    Sound(440); Delay(50); NoSound; poscx := 500;
                  end;
                novapos; novocs; GoToXY(70,23); Write('          ');
                GoToXY(70,23); Write('E = ',Round(xmin+poscx/fx));
              end;
(dir) #77 :   begin
                antpos;  poscx := poscx + incr;
                If poscx > 500 then
                  begin
                    Sound(440); Delay(50); NoSound; poscx := 1;
                  end;
                novapos; novocs; GoToXY(70,23); Write('          ');
                GoToXY(70,23); Write('E = ',Round(xmin+poscx/fx));
              end;
(cima) #72 :  begin
                antpos;  poscy := poscy - incr;
                If poscy < 1 then
                  begin
```

```
                Sound(440); Delay(50); NoSound; poscy := 150;
              end;
            novapos; novocs; GoToXY(70,24); Write('          ');
            GoToXY(70,24);
            If corrente >= 200 then
              Write('i = ',((ymax - poscy / fy)*fEi):0:1);
            If corrente = 20 then
              Write('i = ',((ymax - poscy / fy)*fEi):0:2);
            If corrente = 2 then
              Write('i = ',((ymax - poscy / fy)*fEi):0:3);
          end;
(baixo)  #80 : begin
            antpos; poscy := poscy + incr;
            If poscy > 150 then
              begin
                Sound(440); Delay(50); NoSound; poscy := 1;
              end;
            novapos; novocs; GoToXY(70,24); Write('          ');
            GoToXY(70,24);
            If corrente >= 200 then
              Write('i = ',((ymax - poscy / fy)*fEi):0:1);
            If corrente = 20 then
              Write('i = ',((ymax - poscy / fy)*fEi):0:2);
            If corrente = 2 then
              Write('i = ',((ymax - poscy / fy)*fEi):0:3);
          end;
 (F11)   #84 : begin    { calcula a carga anodica liquida }
            If (xe < xd) then
              begin
                m := (ye - yd)/(xe - xd);
                e1 := Round(xe); e2 := Round(xd); nciclo := 1;
                Sound(3000); Delay(50);Sound(2000); Delay(50);
                Sound(1000); Delay(50); NoSound;
                GoToXY(70,19); Write('aguarde...'); Integral;
                cargaliq := (m/2)*(xd*xd-xe*xe)+(ye-m*xe)*(xd-xe);
                cargaliq := (Q[nq]-cargaliq/velo)*fEi;
                GoToXY(70,18); Write('CARGA ANOD.');
                GoToXY(70,19); Write('          ');
                GoToXY(70,19); Write(cargaliq:8,unidade[1],'C');
                registro.ddy[registro.nro] := cargaliq;
              end;
          end;
 (F12)   #85 : begin    { calcula a carga catodica liquida }
            If (xe < xd) then
              begin
                m := (ye - yd)/(xe - xd);
                e1 := Round(xd); e2 := Round(xe); nciclo := 1;
                Sound(3000); Delay(50);Sound(2000); Delay(50);
                Sound(1000); Delay(50); NoSound;
                GoToXY(70,22); Write('aguarde...'); Integral;
                cargaliq := (m/2)*(xd*xd-xe*xe)+(ye-m*xe)*(xd-xe);
                cargaliq := (Q[nq]-cargaliq/velo)*fEi;
                GoToXY(70,21); Write('CARGA CAT.');
                GoToXY(70,22); Write('          ');
                GoToXY(70,22); Write(cargaliq:8,unidade[1],'C');
```



```
                registro.ddy[registro.nro] := cargaliq;
              end;
            end;
 (F20)   #93 : begin
                antpos;      { esconde o cursor }
                GoToXY(70,23); Write('          ');
                GoToXY(70,24); Write('          ');
                GoToXY(70,25); Write('        ');
              end;
          end; { case }
        end;
      Until (mov = ESC);
    end;

procedure EixoVolt(xc,xu,yc,yu : String2);

  procedure EixoY;

    var  vy  :  Real;

    begin
      If (corrente = 2000) or (corrente = 1000) then
        begin
          vy := (ymin/normaly)*fEi;
          Write('Eixo ',yc,' : ',vy:0:0);
          vy := (ymax/normaly)*fEi;
          Write(' a ',vy:0:0,' ',yu);
          vy := (divy/normaly)*fEi;
          Write(' com divisoes de ',vy:0:0,' ',yu);
        end;
      If corrente = 200 then
        begin
          vy := (ymin/normaly)*fEi; Write('Eixo ',yc,' : ',vy:0:1);
          vy := (ymax/normaly)*fEi; Write(' a ',vy:0:1,' ',yu);
          vy := (divy/normaly)*fEi;
          Write(' com divisoes de ',vy:0:1,' ',yu);
        end;
      If corrente = 20 then
        begin
          vy := (ymin/normaly)*fEi; Write('Eixo ',yc,' : ',vy:0:2);
          vy := (ymax/normaly)*fEi; Write(' a ',vy:0:2,' ',yu);
          vy := (divy/normaly)*fEi;
          Write(' com divisoes de ',vy:0:2,' ',yu);
        end;
      If corrente = 2 then
        begin
          vy := (ymin/normaly)*fEi; Write('Eixo ',yc,' : ',vy:0:3);
          vy := (ymax/normaly)*fEi; Write(' a ',vy:0:3,' ',yu);
          vy := (divy/normaly)*fEi;
          Write(' com divisoes de ',vy:0:3,' ',yu);
        end;
    end;

  begin
    tipo := yc; Window(1,1,80,25); HiRes; ClearScreen;
```

```
Borda(6,43,514,197); GraphWindow(10,45,510,195);
Draw(0,100,500,100,1); GetPic(csh,0,100,500,100);
Draw(300,0,300,150,1); GetPic(csv,300,0,300,150);
ClearScreen; GoToXY(1,1); Write('Amostra : ',amostra);
GoToXY(29,1); Write('Operador : ',operador);
GoToXY(64,1); WriteLn('Data : ',data);
divx := ((xmax - xmin) div 200) * 10;     ( calcula as divisoes )
If divx = 0 then divx := 10;
divy := ((ymax - ymin) div 200) * 10;
If divy = 0 then divy := 10;
Write('Eixo ',xc,' : ',Round(xmin/normalx),' a ',Round(xmax/normalx));
WriteLn(' ',xu,' com divisoes de ',Round(divx/normalx),' ',xu);
EixoY; posx := WhereX; posy := WhereY; Condicoes;
passay := 0;
If xmin > 0 then passay := xmin;
If xmax < 0 then passay := xmax;
passax := 0;
If ymin > 0 then passax := ymin;
If ymax < 0 then passax := ymax;
Linha(xmin,passax,xmax,passax);                    ( Eixo X )
Draw(px(xmax),py(passax),px(xmax)-7,py(passax)-2,1);
Draw(px(xmax),py(passax),px(xmax)-7,py(passax)+2,1);
Linha(passay,ymin,passay,ymax);                    ( Eixo Y )
Draw(px(passay),py(ymax),px(passay)-5,py(ymax)+3,1);
Draw(px(passay),py(ymax),px(passay)+5,py(ymax)+3,1);
x := Round((px(passay)+10)/639*79)+1;              ( escreve yc )
GoToXY(x,5);Write(yc);
y := Round((py(passax)+45)/199*23)+1;              ( escreve xc )
GoToXY(66,y);Write(xc);
x := passay;                    ( desenha as divisoes do eixo x )
While ( x < (xmax - divx/2) ) do
  begin
    Draw(px(x),py(passax)+2,px(x),py(passax)-2,1);
    x := x + divx;
  end;
x := passay;
While ( x > xmin ) do
  begin
    Draw(px(x),py(passax)+2,px(x),py(passax)-2,1);
    x := x - divx;
  end;
y := passax;                    ( desenha as divisoes do eixo y )
While ( y > ymin ) do
  begin
    Draw(px(passay)-4,py(y),px(passay)+4,py(y),1);
    y := y - divy;
  end;
y := passax;
While ( y < ymax - divy/2 ) do
  begin
    Draw(px(passay)-4,py(y),px(passay)+4,py(y),1);
    y := y + divy;
  end;
end;
```



```
procedure Limites;

  var pilha  :  Integer;

  begin
    If xmin > xmax then
      begin
        pilha := xmin; xmin := xmax; xmax := pilha;
      end;
    If ymin > ymax then
      begin
        pilha := ymin; ymin := ymax; ymax := pilha;
      end;
    fx := 500.0/(xmax - xmin); fy := 150.0/(ymax - ymin);
  end;


procedure Escalas(xc,yc : Char);

  var  iny, fiy  :  Real;

  begin
    GoToXY(1,6); WriteLn('Escalas :'); WriteLn;
    Write(' Eixo ',xc,' - Inicio = ',iniciox);
    GoToXY(20,WhereY); NumeroInteiro(5,#13,#10,iniciox);
    Write('          Fim    = ',fimx);
    GoToXY(20,WhereY); NumeroInteiro(5,#13,#10,fimx);
    iny := inicioy*fEi; fiy := fimy*fEi;
    Write(' Eixo ',yc,' - Inicio = ',iny:8);
    GoToXY(20,WhereY); NumeroReal(8,#13,#10,iny);
    inicioy := Round(iny/fEi);
    Write('          Fim    = ',fiy:8);
    GoToXY(20,WhereY); NumeroReal(8,#13,#10,fiy);
    fimy := Round(fiy/fEi);
  end;


procedure EscalasQE;

  var iny, fiy  :  Real;

  begin
    GoToXY(1,7); WriteLn('Escalas :'); WriteLn;
    Write(' Eixo E - Inicio = ',iniciox);
    GoToXY(20,WhereY); NumeroInteiro(4,#13,#10,iniciox);
    Write('          Fim    = ',fimx);
    GoToXY(20,WhereY); NumeroInteiro(4,#13,#10,fimx);
    iny := inicioqy*fEi;  fiy := fimqy*fEi;
    Write(' Eixo Q - Inicio = ',iny:8);
    GoToXY(20,WhereY); NumeroReal(8,#13,#10,iny);
    inicioqy := iny/fEi;
    Write('          Fim    = ',fiy:8);
    GoToXY(20,WhereY); NumeroReal(8,#13,#10,fiy);
    fimqy := fiy/fEi;
```

```
  end;

procedure IxE;

  function ftc(fat : Integer) : Integer;

    var ftcon : Real;

    begin
      ftcon := fs * fat;
      If ftcon > 2000.0 then ftcon := 2000.0;
      If ftcon < -2000.0 then ftcon := -2000.0;
      ftc := Round(ftcon);
    end;


  var  pri, ult, inc  :  Integer;

  begin
    If modo = 'A' then
      begin
        Janela(46,5,76,19); WriteLn;
        pri := primeiro; ult := ultimo; inc := passo;
        Write(' Nro. do 1o. ciclo = ',pri);
        GoToXY(22,WhereY); NumeroInteiro(2,#13,#10,pri);
        Write(' Nro. do ultimo ciclo = ',ult);
        GoToXY(25,WhereY); NumeroInteiro(2,#13,#10,ult);
        If ult > ultimo then ult := ultimo;
        If ult < primeiro then ult := primeiro;
        Write(' Incremento = ',inc);
        GoToXY(15,WhereY); NumeroInteiro(2,#13,#10,inc);
        If inc <= 0 then inc := 1;
      end
    else
      begin
        pri := 1;  inc := 1;  ult := 1;
      end;
    If controlaVoltam = 0 then
      begin
        imax := -2000;  imin := 2000;
        j := pri;
        Repeat
          Case j of
            1..20    :  For i := 0 to tpo do
                          begin
                            If imax < D1[i,j] then imax := D1[i,j];
                            If imin > D1[i,j] then imin := D1[i,j];
                          end;
            21..40   :  For i := 0 to tpo do
                          begin
                            If imax < D2[i,j] then imax := D2[i,j];
                            If imin > D2[i,j] then imin := D2[i,j];
                          end;
```



```
            41..60   :  For i := 0 to tpo do
                          begin
                            If imax < D3[i,j-40] then imax:=D3[i,j-40];
                            If imin > D3[i,j-40] then imin:=D3[i,j-40];
                          end;
          end; { case }
          j := j + inc;
        Until (j > ult);
        controlaVoltam := 1;
      end;
    iniciox := vini; fimx := vfi;
    inicioy := imin - abs(imin div 10);
    fimy := imax + abs(imax div 10);
    If fimy = inicioy then fimy := fimy + 10;
    If modo = 'A' then  Escalas('E','i')
    else
      If modo = 'S' then
        begin
          iniciox := potin;
          fimx := potfi;
          inicioy := -2000;
          fimy := 2000;
        end;
    xmin := iniciox; xmax := fimx; ymin := inicioy; ymax := fimy; Limites;
    If modo = 'S' then
      begin
        If nro_voltam = 1 then
          begin
            amostra := 'SUPERPOSICAO'; Window(1,1,80,25);
            EixoVolt('E','mV','i',unid);
          end;
      end
    else
      begin
        Window(1,1,80,25); EixoVolt('E','mV','i',unidade);
      end;
    j := pri;
    Repeat
      Case j of
        1..20  : For i := 1 to tpo do
                   Linha(mV[i-1], ftc(D1[i-1,j]), mV[i],ftc(D1[i,j]));
        21..40 : For i := 1 to tpo do
                   Linha(mV[i-1],ftc(D2[i-1,j-20]),mV[i],ftc(D2[i,j-20]));
        41..60 : For i := 1 to tpo do
                   Linha(mV[i-1],ftc(D3[i-1,j-40]),mV[i],ftc(D3[i,j-40]));
      end;  { case }
      j := j + inc;
    Until (j > ult) or KeyPressed;
    Sound(1440); Delay(50); Sound(720); Delay(50);
    Sound(360); Delay(50); NoSound;
    If modo <> 'S' then
      begin
        Cursor; TextMode;
      end;
  end;
```

```
procedure QxE;

  begin
    If (Qmax > 30000.0) or (Qmin < -30000.0) then
      begin
        If Qmax > 30000.0 then normaly := 30000.0 / Qmax;
        If Qmin < -30000.0 then normaly := -30000.0 / Qmin;
      end
    else normaly := 1.0; iniciox := e1; fimx := e2;
    inicioqy := Qmin; fimqy := Qmax;
    If fimqy = inicioqy then fimqy := fimqy + 1.0;
    EscalasQE; xmin := iniciox; xmax := fimx;
    ymin := Round(inicioqy * normaly); ymax := Round(fimqy * normaly);
    Window(1,1,80,25); Limites;
    If UpCase(unidade[1]) = 'M' then EixoVolt('E','mV','Q','mC');
    If UpCase(unidade[1]) = 'U' then EixoVolt('E','mV','Q','uC');
    j := 1; i := i1 + 1;
    Repeat
      Linha(mV[i-1],Round(Q[j-1]*normaly),mV[i],Round(Q[j]*normaly));
      j := j + 1;  i := i + 1;
    Until (i > i2) or KeyPressed;
    Sound(1440); Delay(50); Sound(720); Delay(50);
    Sound(360); Delay(50); NoSound;
    Repeat Until KeyPressed;  TextMode;
  end;


Begin
  td := 1/fatortt;                ( tempo de aquisiao por dado em s )
  normalx := 1.0;  normaly := 1.0;  ( fator de normalizaao para Qxt )
  controlaIntegral := 0; controlaAmpero := 0;
  controlaVoltam := 0; controlaPotenc := 0;
  If modo = 'A' then
    begin
      Repeat
        Menu; Janela(26,4,56,20);
        GoToXY(5,1); Write('MANIPULAAO DOS DADOS');
        GoToXY(5,2); Write('=========== === =====');
        GoToXY(1,4); Write('Graficos(s)  :  ');
        WriteLn('(I) - i x E');
        WriteLn('              (Q) - Q x E');
        GoToXY(4,14); Write('<ESC> - Menu Inicial');
        Read(Kbd,tecla); tecla := UpCase(tecla);
        Case tecla of
          'I' :  ixE;
          'Q' :  begin
                   If controlaIntegral = 0 then
                     begin
                       Janela(46,5,76,19);
                       e1 := vini;  e2 := vfi;  nciclo := 1;
                       GoToXY(10,1); Write('INTEGRACAO');
                       GoToXY(10,2); Write('==========='); WriteLn;
                       WriteLn(' Limites :');
```



```
                       Write('    E Inicial = ',e1);
                       GoToXY(17,WhereY); NumeroInteiro(6,#13,#10,e1);
                       Write('    E  Final  = ',e2);
                       GoToXY(17,WhereY); NumeroInteiro(6,#13,#10,e2);
                       Write('    nro. do ciclo = ',nciclo);
                       GoToXY(21,WhereY); NumeroInteiro(2,#13,#10,nciclo);
                       Integral; ClrScr;
                       GoToXY(10,1); Write('RESULTADOS');
                       GoToXY(10,2); Write('==========='); WriteLn; WriteLn;
                       Write(' Carga maxima: ',(Qmax*fEi):10);
                       WriteLn(' ',unidade[1],'C');
                       Write(' Carga minima: ',(Qmin*fEi):10);
                       WriteLn(' ',unidade[1],'C');
                     end;
                   QxE;
                 end;
          end;   { case }
        Until (tecla = #27);
      end
    else
      ixE;
  End;

Procedure MostraResultados;

var  i     :  Integer;
     tecla :  Char;

  Begin
    If registro.nro = 0 then EXIT;
    Janela(1,1,80,25);
    For i := 1 to registro.nro do
      begin
        WriteLn('x = ',registro.ddx[i]:10,'          y = ',registro.ddy[i]:10);
        If ((i/20) = (i div 20)) and (i < registro.nro) then
          begin
            GoToXY(27,22); Write('... Aperte uma tecla ...');
            Repeat Until KeyPressed;
            ClrScr;
          end;
      end;
    Sound(2000); Delay(50); Sound(1000); Delay(50); Sound(500); Delay(50);
    NoSound; GoToXY(27,22); Write('... Aperte uma tecla ...');
    Repeat Until KeyPressed;
  End;


Procedure Varios;

var  tecla,modo,
     drive       : Char;
     drv         : String[10];
     nrovolt, i  : Byte;
     fcor        : Real;
     volt        : Array[1..30] of String[10];
```

```
Begin
  unid := 'mA';  maxcorrente := 200; fs := 1.0;
  potin := -2000;  potfi := 2000;
  Janela(26,4,66,20);
  GoToXY(12,1); Write('VARIOS VOLTAMOGRAMAS');
  GoToXY(12,2); Write('====== ============');
  GoToXY(13,4); Write('<ESC> - Abandona');
  Read(Kbd,tecla); tecla := UpCase(tecla); If tecla = #27 then EXIT;
  GoToXY(2,4); ClrEol; Write('Maximo de voltamogramas = 30');
  GoToXY(2,6); Write('Nro. de voltamogramas = '); Read(nrovolt);
  If nrovolt > 30 then nrovolt := 30; If nrovolt < 1 then nrovolt := 1;
  GoToXY(2,8); Write('(S)uperposicao ou (D)ados --> ');
  Repeat
    Read(Kbd,modo);  modo := UpCase(modo);
  Until (modo = 'S') or (modo = 'D');
  Write(modo);
  GetDir(0,drv);
  Case drv[1] of
  'A'         : drive := 'B';
  'B','C','D' : drive := 'A';
  end;
  For i := 1 to nrovolt do
    begin
      GoToXY(2,10); ClrEol; Write(i,' Arquivo:');
      Repeat
        GoToXY(4,11); ClrEol; Write('Nome = '); ReadLn(volt[i]);
      Until pos('.',volt[i]) = 0;
      If pos(':',volt[i]) = 0 then volt[i] := drive + ':' + volt[i];
    end;
  If modo = 'S' then
    begin
      ClrScr;
      GoToXY(12,1); Write('VARIOS VOLTAMOGRAMAS');
      GoToXY(12,2); Write('====== ============');
      GoToXY(2,4); Write('Potenc. Inicial = ',potin);
      GoToXY(20,4); NumeroInteiro(5,#8,#8,potin);
      GoToXY(2,5); Write('Potencial Final = ',potfi);
      GoToXY(20,5); NumeroInteiro(5,#8,#8,potfi);
      GoToXY(2,7); Write('Corrente Maxima = ',maxcorrente);
      GoToXY(20,7); NumeroInteiro(3,#8,#8,maxcorrente);
      GoToXY(2,8); Write('Unidade = ',unid);
      Repeat
        GoToXY(12,8); Read(unid);
        If (unid[1] <> 'm') and (unid[1] <> 'u') then Write(#7);
      Until (unid[1] = 'u') or (unid[1] = 'm');
      vunid := 2;
    end;
  If modo = 'D' then
    begin
      For i := 1 to nrovolt do
        begin
          Arquivo('D',volt[i],erro);
          If erro = FALSE then
            begin
```



```
              fEi := corrente / 2000.0; registro.nro := i;
              registro.ddx[i] := velo; Manipula('D',i);
            end
          else
            begin
              registro.ddx[i] := 0;  registro.ddy[i] := 0;
            end;
        end;
    end
  else
    begin
      For i := 1 to nrovolt do
        begin
          Arquivo('S',volt[i],erro);
          If erro = FALSE then
            begin
              fEi := maxcorrente / 2000.0;
              If unid <> unidade then
                begin
                  If (unidade = 'uA') and (unid = 'mA') then
                    fcor := 0.001;
                  If (unidade = 'mA') and (unid = 'uA') then
                    fcor := 1000.0;
                end
              else  fcor := 1.0;
              fs := fcor * corrente / maxcorrente ;  Manipula('S',i);
            end;
        end;
      Repeat Until KeyPressed;
      TextMode;
    end;
End;


Procedure Salva;

var tecla,drive     : Char;
    nomedado,drv    : String[10];

  Begin
    If registro.nro = 0 then EXIT;
    Janela(30,5,62,10);
    GoToXY(11,1); Write('SALVA DADOS'); GoToXY(11,2); Write('===== =====');
    GoToXY(2,4); Write('<ESC> - Abandona');
    Read(Kbd,tecla); tecla := UpCase(tecla);
    If tecla = #27 then EXIT;
    GetDir(0,drv);
    Case drv[1] of
    'A'         : drive := 'B';
    'B','C','D' : drive := 'A';
    end;
    Repeat
      GoToXY(2,4); ClrEol; Write('Nome do Arquivo = '); Read(nomedado);
    Until (pos('.',nomedado) = 0);
    If pos(':',nomedado) = 0 then nomedado := drive + ':' + nomedado;
```

```
  Assign(Arqdado,nomedado + '.dat');
  Rewrite(Arqdado);
  Write(Arqdado,registro);
  Close(Arqdado);
End;


Overlay Procedure Diretorio;

  Const  TextSize = 200;

  Type
    String80      =    String[80];
    String15      =    String[15];
    String5       =    String[5];
    String3       =    String[3];
    LineSize      =    String[15];
    TextArrayType =    Array[1..TextSize] of LineSize;

  Var
    FileSpec    :   String80;
    Path        :   String80;
    Name,drv    :   String15;
    AttrList    :   String5;
    DayName     :   String3;
    TextArray   :   TextArrayType;
    FileBytes,
    totbytes    :   Real;
    Code,j,k    :   Integer;
    Attr, Yr,
    Mon, Day,
    Hr, Min,
    Sec, DayNum,
    ndl         :   Byte;
    tecla,
    opcao,drive :   Char;


  procedure FileInfo(FileSpec : String80; Var FileBytes : Real;
                     Var Yr, Mon, Day, Hr, Min, Sec : Byte;
                     Var AttrList : String5);

     Type
       RegList  =  Record
                     AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                   end;
       FileData  =  Record
                      Reserved  :  Array[1..21] of Byte;
                      Attr      :  Byte;
                      Time, Date, SizeLo, SizeHi  :  Integer;
                    end;

     var  Reg  :  RegList;
          DTA  :  Array[1..43] of Byte;
          FD   :  FileData Absolute DTA;
```



```
  begin
    Reg.DX := ofs(DTA);
    Reg.DS := seg(DTA);
    Reg.AX := $1A00;
    MSDOS(Reg);
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.CX := $16;
    Reg.AX := $4E00;
    MSDOS(Reg);
    If Lo(Reg.AX) <> 0 then
      begin
        FileBytes := -1.0;
        EXIT
      end;
    FileBytes := Lo(FD.SizeLo) + 256.0 * Hi(FD.SizeLo) +
                 65536.0 * Lo(FD.SizeHi) + 16777216.0 * Hi(FD.SizeHi);
    Yr := (Hi(FD.Date) shr 1) + 80;
    If Yr > 99 then
      Yr := Yr - 100;
    Mon := Lo(FD.Date) shr 5 + (Hi(FD.Date) and $01 shl 3);
    Day := Lo(FD.Date) and $1F;
    Hr := Hi(FD.Time) shr 3;
    Min := Lo(FD.Time) shr 5 + (Hi(FD.Time) and $07 shl 3);
    Sec := (Lo(FD.Time) and $1F) shl 1;
    AttrList := '     ';
    If FD.Attr and $20 = $20 then
      AttrList[1] := 'A';
    If FD.Attr and $10 = $10 then
      AttrList[2] := 'D';
    If FD.Attr and $02 = $02 then
      AttrList[3] := 'H';
    If FD.Attr and $01 = $01 then
      AttrList[4] := 'R';
    If FD.Attr and $04 = $04 then
      AttrList[5] := 'S';
  end;


procedure NextFile(FileSpec : String80; Attr : Byte;
                   var Code : Integer; Var Name : String15);

  Type
    RegList  =  Record
                  AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                end;

  var  j      :   Integer;
       Reg    :   RegList;
       DTA    :   Array[1..43] of Byte;

  begin
    If (Code < 0) or (Code > 1) then
```

```
  begin
    Code := -1;
    EXIT;
  end;
If Code = 0 then
  begin
    Reg.DX := ofs(DTA);
    Reg.DS := seg(DTA);
    Reg.AX := $1A00;
    MSDOS(Reg);
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.CX := Attr;
    Reg.AX := $4E00;
    MSDOS(Reg);
    If Lo(Reg.AX) <> 0 then
      begin
        Name := '';
        Code := 2;
        EXIT;
      end;
    If DTA[22] and $10 <> 0 then
      Name := '[D]'
    else
      Name := '';
    j := 31;
    While DTA[j] <> 0 do
      begin
        Name := Name + Chr(DTA[j]);
        j := j + 1;
      end;
    Code := 1;
  end
else
  begin
    Reg.DX := ofs(DTA);
    Reg.DS := seg(DTA);
    Reg.AX := $4F00;
    MSDOS(Reg);
    If Lo(Reg.AX) = 0 then
      begin
        If DTA[22] and $10 <> 0 then
          Name := '[D]'
        else
          Name := '';
        j := 31;
        While DTA[j] <> 0 do
          begin
            Name := Name + Chr(DTA[j]);
            j := j + 1;
          end;
        Code := 1;
      end
    else
```



```
      begin
        Name := '';
        Code := 2;
      end;
    end;
  end;

procedure FileDel(FileSpec : String80; Var Code : Integer);

  Type
    RegList = Record
                AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
              End;

  var Reg : RegList;

  begin
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.AX := $4100;
    MSDOS(Reg);
    Code := Lo(Reg.AX);
    If (Reg.Flags and $01) = 0 then
      Code := 0;
  end;


procedure SortHT(Var TextArray : TextArrayType;
                 LineCount, Position, Width : Integer);

  Label
    H1, H2, H3, H4, H5, H6, H7, H8;

  var  i, j,
       l, r     :   Integer;
       ThisLine :   LineSize;

  begin
    If LineCount <= 1 then EXIT;
    H1: L := LineCount div 2 + 1;
        R := LineCount;
    H2: If L > 1 then
          begin
            L := L - 1;
            ThisLine := TextArray[L];
          end
        else
          begin
            ThisLine := TextArray[R];
            TextArray[R] := TextArray[1];
            R := R - 1;
            If R = 1 then
              begin
                TextArray[1] := ThisLine;
```

```
            EXIT;
          end;
        end;
    H3: j := L;
    H4: i := j;
        j := j + j;
        If j > R then goto H7;
        If j = R then goto H6;
    H5: If Copy(TextArray[j], Position, Width) <
          Copy(TextArray[j + 1], Position, Width) then j := j+1;
    H6: TextArray[i] := TextArray[j];
        goto H4;
    H7: j := i;
        i := j div 2;
    H8: If (Copy(ThisLine, Position, Width) <=
          Copy(TextArray[i], Position, Width)) or (j = L) then
        begin
          TextArray[j] := ThisLine;
          goto H2;
        end
      else
        begin
          TextArray[j] := TextArray[i];
          goto H7;
        end;
  end;


function Tab(TabCol : Integer) : Char;

  begin
    If TabCol > 79 then
      TabCol := 79
    else
      If TabCol < 1 then
        TabCol := 1;
    GoToXY(TabCol + 1,WhereY);
    Tab := #8;
  end;


Begin
  Janela(5,4,55,20);
  GetDir(0,drv);
  Case drv[1] of
  'A'         : drive := 'B';
  'B','C','D' : drive := 'A';
  end;
  Path := drive + ':';
  Attr := $16;
  Code := 0;
  j := 0;
  opcao := 'M';
  Janela(17,6,43,18);
  GoToXY(11,1);
```



```
  Write('OPCOES');
  GoToXY(11,2);
  Write('======');
  GoToXY(2,4);
  Write('(M) - Mostra Arquivos');
  GoToXY(2,6);
  Write('(D) - Deleta Arquivos');
  GoToXY(2,8);
  Write('<ESC> - Abandona   ');
  Repeat
    Read(Kbd,opcao);
    opcao := UpCase(opcao);
  Until (opcao = 'M') or (opcao = 'D') or (opcao = #27);
  If opcao = #27 then EXIT;
  Window(6,5,54,18);  ClrScr;
  GoToXY(1,1);
  If opcao = 'D' then
    Write('DRIVE:ARQUIVOS =      ')
  else
    Write('DRIVE:ARQUIVOS = ',drive,':*.DAT');
  GoToXY(18,1);
  Read(Kbd,tecla);
  tecla := UpCase(tecla);
  If (tecla <> #13) and (tecla <> #32) then
    begin
      Write('       ');
      GoToXY(18,1);
      Write(tecla);
      ReadLn(FileSpec);
      FileSpec := tecla + FileSpec;
    end
  else
    begin
      If opcao = 'D' then
        FileSpec := drive + ':   '
      else
        FileSpec := drive + ':*.DAT';
      WriteLn;
    end;
  If Copy(FileSpec,1,2) = 'A:' then Path := 'A:';
  If Copy(FileSpec,1,2) = 'C:' then Path := 'C:';
  If Copy(FileSpec,1,2) = 'D:' then Path := 'D:';
  If (Length(FileSpec) = 2) then
    If (FileSpec[2] = ':') then
      begin
        If opcao = 'D' then
          FileSpec := '.:   '
        else
          FileSpec := '.:*.DAT';
      end;
  If FileSpec[2] <> ':' then FileSpec := '..' + FileSpec;
  FileSpec := Path + copy(FileSpec,3,80);
  Repeat
    NextFile(FileSpec, Attr, Code, Name);
    If Code = 1 then
```

```
    begin
      j := j + 1;
      TextArray[j] := Name;
    end;
Until (Code <> 1);
WriteLn(j,' Arquivos encontrados');
SortHT(TextArray, j, 1, 15);
If opcao = 'D' then
  begin
    ndl := 0;
    For k := 1 to j do
      begin
        FileSpec := Path + TextArray[k];
        FileDel(FileSpec, Code);
        If Code <> 0 then
          begin
            WriteLn(FileSpec,' nao deletado');
            ndl := ndl + 1;
          end;
      end;
    WriteLn(j - ndl,' arquivos deletados');
  end
else
  begin
    totbytes := 0;
    For k := 1 to j do
      begin
        FileSpec := TextArray[k];
        If FileSpec[1] = '[' then
          FileSpec := Copy(FileSpec, 4, 12);
        FileSpec := Path + FileSpec;
        FileInfo(FileSpec,FileBytes, Yr,Mon,Day, Hr,Min,Sec, AttrList);
        Write(TextArray[k], Tab(14), FileBytes:8:0, Tab(24));
        If FileBytes < 0.0 then
          WriteLn('... Arquivo nao encontrado ...')
        else
          begin
            Write(Day:2,'/',Mon:2,'/',Yr, Tab(34));
            Write(Hr:2,':');
            If Min < 10 then WriteLn('0',Min) else WriteLn(Min:2);
            totbytes := totbytes + FileBytes;
          end;
        If ((k/12) = (k div 12)) and (k < j) then
          begin
            WriteLn('  ... Aperte uma tecla ...');
            Read(Kbd,tecla);
          end;
      end;
    WriteLn('       ',totbytes:8:0,' bytes ocupados');
  end;
Sound(1760); Delay(50);
Sound(880); Delay(50);
Sound(440); Delay(50);
NoSound;
Write('       ... [Enter] - volta ao MENU ...');
```



```
  Read(tecla);
End;


BEGIN
  passo := 1;  primeiro := 1;        (Default para Voltametria)
  registro.nro := 0;
  Repeat
    Repeat
      fs := 1.0;  Menu;
      Read(Kbd,tecla);
      tecla := UpCase(tecla);
      Case tecla of
        'A'  :  begin
                  Arquivo('A','',erro);
                  If erro = FALSE then
                    fEi := corrente / 2000.0;
                end;
        'M'  :  Manipula('A',1);
        'D'  :  Diretorio;
        'R'  :  MostraResultados;
        'V'  :  Varios;
        'S'  :  begin
                  Salva;
                end;
      end;  { case }
    Until (tecla = 'F');
    Janela(30,10,60,12);
    GoToXY(2,2);
    Write('Confirma saida?  (S/N)  ');
    Read(Kbd,tecla);
    tecla := UpCase(tecla);
  Until (tecla = 'S');
  Window(1,1,80,25); ClrScr;
END.
```

A1.5. PROGRAMA CRONOAM

```
Program CRONOAM;

{-------------------------------------------------------------------------}
{   Este programa manipula os dados de cronoamperometria de modo a obter  }
{ ix(1/SQRT(t)), Q, Qxt e QxSQRT(t), permitindo tambem a superposicao de  }
{ cronoamperogramas e cronocoulogramas.                                   }
{                                              VFJ/1991                   }
{-------------------------------------------------------------------------}

  Const LF = #10;
        CR = #13;
        versao = '1.0';

  Type String80    =  String[80];
       String15    =  String[15];
       String10    =  String[10];
       String8     =  String[8];
       String2     =  String[2];
       Ponteiro1   =  ^TipoVet1;
       TipoVet1    =  Array[0..32000] of Integer;
       Ponteiro2   =  ^TipoVet2;
       TipoVet2    =  Array[0..4010] of Real;

  Var  Crono       :  Ponteiro1;
       Q           :  Ponteiro2;
       t           :  Ponteiro2;
       csh         :  Array[1..69] of Integer;
       csv         :  Array[1..156] of Integer;
       i,
       imax,imin,
       vini, vfi,
       corrente,
       intervalo,
       incremento,
       nq, tpo,
       tpoa,tpod,
       tpotot,
       npulsos,
       linha,coluna :  Integer;
       tecla,
       tecnica     :  Char;
       operador,
       amostra     :  String[20];
       data        :  String8;
       unidade     :  String2;
       Qmin, Qmax,
       fatortt,
       incr,fEi,
       teapopto    :  Real;
       vunidade    :  Byte absolute unidade;
       erro        :  Boolean;
       marcaheap   :  ^Char;


procedure Graphics;                          external 'GRAPH.BIN';
```



```
procedure HiRes;                             external Graphics[6];
procedure GraphWindow(X1,Y1,X2,Y2: Integer); external Graphics[18];
procedure Plot(X,Y,Color: Integer);          external Graphics[21];
procedure Draw(X1,Y1,X2,Y2,Color: Integer);  external Graphics[24];
procedure GetPic(var Buffer; X1,Y1,X2,Y2: Integer); external Graphics[36];
procedure PutPic(var Buffer; X,Y: Integer);  external Graphics[39];
procedure ClearScreen;                       external Graphics[60];

Procedure Borda(x1,y1,x2,y2 : Integer);

Begin
  GraphWindow(0,0,639,199);
  Draw(x1,y1,x2,y1,1); Draw(x2,y1,x2,y2,1);
  Draw(x2,y2,x1,y2,1); Draw(x1,y1,x1,y2,1);
End;

Procedure Janela(x1,y1,x2,y2 : Integer);

Var   i,
      maxx,
      maxy   : Integer;

Begin
  Window (1,1,80,25);
  GoToXY (x1,y1);
  Window (x1,y1,x2,y2);
  ClrScr; maxx := x2 - x1 + 1; maxy := y2 - y1 + 1;
  GoToXY (1,maxy); Write (#212);
  For i := 2 to maxx - 1 do Write(#205);
  Write (#190);
  GoToXY (1,1); Write (#213);
  For i := 2 to maxx - 1 do Write(#205);
  Write (#184);
  For i := 2 to maxy - 2 do
    begin
      GoToXY (1,i); Write(#179);
      GoToXY (maxx,i); Write(#179);
    end;
  GoToXY (2,2); Window (x1+1,y1+1,x2-1,y2-2);
  linha := y1 ; coluna := x1 + 1;
End;

Procedure Menu;

Begin
  ClrScr; Janela(1,1,80,25);
  GoToXY(8,22);
  Write('PROGRAMA  "CRONOAM" - Versao ',versao,'      (C) VFJ - 1991');
  Janela(4,3,34,22);
  GoToXY(8,1); Write('MENU INICIAL');
  GoToXY(8,2); Write('============');
  GoToXY(4,5); Write('A - Arquivo em disco');
  GoToXY(4,7); Write('R - Resultados');
  GoToXY(4,9); Write('S - Superpor curvas');
  GoToXY(4,11); Write('D - Diretorio');
```

```
  GoToXY(4,13); Write('F - Finalizar');
End;


Procedure ReadIn(ci : Char; var x : Integer);

type par = Record
             car   : Char;
             atrib : Byte;
           end;

     TipoLinha = Array[1..80] of par;
     TipoTela  = Array[1..25] of TipoLinha;

var numero           : String[20];
    codigo, spc, c   : Integer;
    tela             : TipoTela absolute $B800:$0000;

Begin
  Str(x, numero);
  Write(numero);
  Repeat Until keypressed;
  Repeat
    spc := length(numero);
    GoToXY(WhereX - spc, WhereY);
    For c := WhereX to WhereX + spc do
      tela[linha + WhereY,coluna + c].car := ' ';
    Read(numero); val(numero, x, codigo);
  Until codigo = 0;
  spc := length(numero); GoToXY(WhereX - spc, WhereY); Write(x);
  If ci = CR then Writeln;
End;

Procedure ReadRe(ci : Char; var x : Real; spc, dec : Integer);

type par = Record
             car   : Char;
             atrib : Byte;
           end;

     TipoLinha = Array[1..80] of par;
     TipoTela  = Array[1..25] of TipoLinha;

var numero      : String[20];
    codigo,c,xi : Integer;
    tela        : TipoTela absolute $B800:$0000;

Begin
  xi := WhereX; Str(x, numero);
  Write(x:spc:dec);
  Repeat Until keypressed;
  Repeat
    GoToXY(xi,WhereY);
    For c := WhereX to WhereX + spc do
      tela[linha + WhereY,coluna + c].car := ' ';
```



```
    Read(numero);
    val(numero, x, codigo);
    If codigo <> 0 then spc := length(numero);
  Until codigo = 0;
  GoToXY(xi, WhereY);  Write(x:spc:dec);
  If ci = CR then Writeln;
End;

Procedure Aviso(mensagem : String80);

Begin
  Janela((39 - length(mensagem) div 2),11,(41 + length(mensagem) div 2),14);
  Write(mensagem);
  Sound(2000); Delay(50); Sound(1000); Delay(50); NoSound;
  Repeat Until KeyPressed;
End;

Overlay Procedure Disco(n : String[10]; var er : Boolean);

Type RegCrono = Record
                  Cronod : Ponteiro;
                end;
     RegParametros = Record
                       unidadeP    : String[2];          { CA e V }
                       AmostraP,                         { T }
                       OperadorP   : String[20];         { T }
                       DataP       : String8;            { T }
                       tecnicaP    : Char;               { T }
                       fatorttP    : Real;               { T }
                       ganhoP      : Byte;               { T }
                       intervaloP,                       { T }
                       tpoP,                             { T }
                       tpoaP,                            { CA e CP }
                       tpodP,                            { CA e CP }
                       tpototP,                          { CA e CP }
                       npulsosP,                         { CA }
                       ncP,                              { V }
                       vmaxP,                            { CP }
                       vminP,                            { CP }
                       viniP,                            { CA e V }
                       vfiP,                             { CA e V }
                       correnteP   : Integer;            { CA e V }
                       veloP,                            { V }
                       incrP       : Real;               { V }
                       primeiroP,                        { V }
                       ultimoP,                          { V }
                       passoP      : Integer;            { V }
                       tempoptoP,                        { CC }
                       tempoP      : Real;               { CA e CP }
                       incrementoP,                      { CC }
                       nqP,                              { CC }
                       iiniP,                            { CP }
                       ifiP  : Integer;                  { CP }
                     end;
```

```
var    Parametros    : RegParametros;
       ArqParametros : file of RegParametros;
       nome, drv     : String[0];
       tecla,drive   : Char;
       ASCII         : Boolean;
       i,controle    : Integer;


procedure LeCrono;

var  UmCrono       : RegCrono;
     ArqCrono      : file of RegCrono;
     ArqCronoASCII : file of Integer;
     i             : Integer;

begin
  If ASCII then
    begin
      Assign(ArqCronoASCII,nome + '.DAT');
      {$I-}
      Reset(ArqCronoASCII);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Arquivo de dados nao encontrado');
          controle := 1;
          Close(ArqCronoASCII);
          EXIT;
        end;
      {$I-}
      For i := 0 to tpotot do Read(ArqCronoASCII,Crono^[i]);
      {$I+}
      If IOResult <> 0 then
        begin
          Aviso('Problemas na leitura');
          controle := 1;
          Close(ArqCronoASCII);
          EXIT;
        end;
      Close(ArqCronoASCII);
    end
    else
      begin
        Assign(ArqCrono,nome + '.DAT');
        {$I-}
        Reset(ArqCrono);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Arquivo de dados nao encontrado');
            controle := 1;
            Close(ArqCrono);
            EXIT;
          end;
```



```
        {$I-}
        Read(ArqCrono,UmCrono);
        {$I+}
        If IOResult <> 0 then
          begin
            Aviso('Problemas na leitura');
            controle := 1;
            Close(ArqCrono);
            EXIT;
          end;
        With UmCrono do Crono := Cronod;
        Close(ArqCrono);
      end;
  end;


procedure LeParametros;

begin
  Assign(ArqParametros,nome + '.PAR');
  {$I-}
  Reset(ArqParametros);
  {$I+}
  If IOResult <> 0 then
    begin
      Aviso('Arquivo de parametros nao encontrado');
      controle := 1;
      Close(ArqParametros);
      EXIT;
    end;
  {$I-}
  Read(ArqParametros,Parametros);
  {$I+}
  If IOResult <> 0 then
    begin
      Aviso('Problemas na leitura');
      controle := 1;
      Close(ArqParametros);
      EXIT;
    end;
  With Parametros do
    begin
      unidade    := unidadeP;
      Amostra    := AmostraP;
      Operador   := OperadorP;
      Data       := DataP;
      tecnica    := tecnicaP;
      fatortt    := fatorttP;
      tpo        := tpoP;
      tpoa       := tpoaP;
      tpod       := tpodP;
      tpotot     := tpototP;
      npulsos    := npulsosP;
      vini       := viniP;
      vfi        := vfiP;
```

```
      corrente   := correnteP;
      tempopto   := tempoptoP;
      incremento := incrementoP;
      nq         := nqP;
    end;  ( With )
  Close(ArqParametros);
end;

Begin
  controle := 0;  ASCII := TRUE; er := FALSE;
  GetDir(0,drv);
  Case drv[1] of
    'A'         : drive := 'B';
    'B','C','D' : drive := 'A';
  end; ( case )
  If n = ' ' then
    begin
      Janela(5,4,60,20);
      Repeat
        ClrScr;
        WriteLn('RECOLHE DADOS GRAVADOS EM DISCO',CR,LF);
        Write(' (ESC) - Abandona  '); Read(Kbd,tecla);
        If tecla = #27 then EXIT;
        GoToXY(1,WhereY);
        WriteLn('1) Recolhe em ASCII      2) Recolhe em binario   ');
        Repeat
          Read(Kbd,tecla); tecla := UpCase(tecla);
        Until tecla in ['1','2'];
        If tecla = '1' then ASCII := TRUE
        else ASCII := FALSE;
        WriteLn('Qual o nome do arquivo? (maximo 8 caracteres)'); Readln(nome);
        If nome[2] () ':' then nome := drive + ':' + nome;
        If Pos('.',nome) ) 0 then
          nome := Copy(nome,1,Pos('.',nome)-1);
        For i := 1 to Length(nome) do nome[i] := UpCase(nome[i]);
        WriteLn('Arquivo de dados: ',nome + '.DAT',LF);
        WriteLn('Arquivo de parametros: ',nome + '.PAR',LF);
        WriteLn('Confirma? (S/N)'); Read(kbd,tecla);
      Until (UpCase(tecla) = 'S');
      LeParametros;
      If controle = 1 then
        begin
          er := TRUE;
          EXIT;
        end;
      If tecnica = 'A' then
        begin
          ClrScr;
          WriteLn('Amostra: ',Amostra,CR,LF,'Operador: ',Operador,LF);
          Write('Tipo de arquivo: ');
          WriteLn('Crono-Amperometria');
          WriteLn('Escala de corrente: ',corrente,' ',unidade);
          WriteLn('Tempo antes do pulso = ',(tpoa*fatortt):8:0);
          WriteLn('Tempo do pulso = ',(tpo*fatortt):8:0);
          WriteLn('Tempo apos o pulso = ',(tpod*fatortt):8:0);
```



```
          LeCrono;
        end
      else
        begin
          WriteLn('Este arquivo nao e de cronoamperometria');
          er := TRUE;
        end;
      If controle = 1 then EXIT;
      WriteLn(CR,LF,'...Aperte uma tecla...');
      Repeat Until KeyPressed;
    end
  else
    begin
      nome := n;
      If nome[2] () ':' then nome := drive + ':' + nome;
      If Pos('.',nome) ) 0 then
        nome := Copy(nome,1,Pos('.',nome)-1);
      LeParametros;
      If (controle = 1) or (tecnica () 'A') then
        begin
          er := TRUE;
          EXIT;
        end;
      LeCrono;
      If controle = 1 then
        begin
          er := TRUE;
          EXIT;
        end;
    end;
End;

Overlay Procedure Resultados(s : Char; nc, xs,ys : Integer);

  Var  cs                 : Array[1..18] of Integer;
       unid,tipoy,tipox   : String8;
       tecla, opcao       : Char;
       xmin, ymin,
       xmax, ymax,
       iniciox,fimx,
       inicioy,fimy,
       passax,passay,
       mult,
       divx, divy,
       posx,posy,
       x,y,i,j,
       l1, l2, l3, l4     :  Integer;
       iniciogx,fimgx,
       iniciogy,fimgy,
       fx, fy             :  Real;
       contInt,contAmp    :  Boolean;


  function px(x : Integer) : Integer;
```

```
  begin
    px := Round(fx*(x - xmin));
  end;


function py(y : Integer) : Integer;

  begin
    py := Round(fy*(ymax - y));
  end;


procedure Linha(x1,y1,x2,y2 : Integer);

  begin
    Draw(px(x1),py(y1),px(x2),py(y2),1);
  end;


procedure Condicoes;

  begin
    GoToXY(posx,posy);
    If (tipoy () '@') and (s = ' ') then
      begin
        If (corrente = 2000) or (corrente = 1000) then
          begin
            Write(' (max=',(imax*fEi):0:0,')');
            WriteLn(' (min=',(imin*fEi):0:0,')');
          end;
        If corrente = 200 then
          begin
            Write(' (max=',(imax*fEi):0:1,')');
            WriteLn(' (min=',(imin*fEi):0:1,')');
          end;
        If corrente = 20 then
          begin
            Write(' (max=',(imax*fEi):0:2,')');
            WriteLn(' (min=',(imin*fEi):0:2,')');
          end;
        If corrente = 2 then
          begin
            Write(' (max=',(imax*fEi):0:3,')');
            WriteLn(' (min=',(imin*fEi):0:3,')');
          end;
      end;
    If s () ' ' then WriteLn;
    If (tipoy = '@') and (tipox = 't') and (s = ' ') then
      WriteLn(' (carga total = ',(Q^[nq]*fEi):8,')')
    else If tipox = 't1/2' then WriteLn;
    If s = ' ' then
      begin
        Write('Tempo de aquisicao = ');
        Write((((tpo+tpod)*npulsos + tpoa)*fatortt):0:0,' s');
        Write(' (E inicial = ',vini,'mV) (E final = ',vfi,'mV)');
```



```
      end;
  end;


procedure EixoVolt(xc,xu,yc,yu : String8);

  procedure EixoY;

    var  vy  :  Real;

    begin
      If (corrente = 2000) or (corrente = 1000) then
        begin
          vy := ymin * fEi;
          Write('Eixo ',yc,' : ',vy:0:0);
          vy := ymax * fEi;
          Write(' a ',vy:0:0,' ',yu);
          vy := divy * fEi;
          Write(' com divisoes de ',vy:0:0,' ',yu);
        end;
      If corrente = 200 then
        begin
          vy := ymin * fEi;
          Write('Eixo ',yc,' : ',vy:0:1);
          vy := ymax * fEi;
          Write(' a ',vy:0:1,' ',yu);
          vy := divy * fEi;
          Write(' com divisoes de ',vy:0:1,' ',yu);
        end;
      If corrente = 20 then
        begin
          vy := ymin * fEi;
          Write('Eixo ',yc,' : ',vy:0:2);
          vy := ymax * fEi;
          Write(' a ',vy:0:2,' ',yu);
          vy := divy * fEi;
          Write(' com divisoes de ',vy:0:2,' ',yu);
        end;
      If corrente = 2 then
        begin
          vy := ymin * fEi;
          Write('Eixo ',yc,' : ',vy:0:3);
          vy := ymax * fEi;
          Write(' a ',vy:0:3,' ',yu);
          vy := divy * fEi;
          Write(' com divisoes de ',vy:0:3,' ',yu);
        end;
    end;

  begin
    tipoy := yc; tipox := xc;
    Window(1,1,80,25); HiRes; ClearScreen;
    Borda(6,43,514,197); GraphWindow(10,45,510,195);
    Draw(0,100,500,100,1); GetPic(csh,0,100,500,100);
```

```
Draw(300,0,300,150,1); GetPic(csv,300,0,300,150);
ClearScreen;
GoToXY(1,1);
Write('Amostra : ',amostra);
GoToXY(29,1);
Write('Operador : ',operador);
GoToXY(64,1);
WriteLn('Data : ',data);
divx := ((xmax - xmin) div 200) * 10;  { calcula as divisoes }
If divx = 0 then divx := 1;
divy := ((ymax - ymin) div 200) * 10;
If divy = 0 then divy := 1;
Write('Eixo ',xc,' : ',xmin,' a ',xmax);
WriteLn(' ',xu,' com divisoes de ',divx,' ',xu);
EixoY;
posx := WhereX; posy := WhereY;
Condicoes;
passay := 0;
If xmin ) 0 then passay := xmin;
If xmax ( 0 then passay := xmax;
passax := 0;
If ymin ) 0 then passax := ymin;
If ymax ( 0 then passax := ymax;
Linha(xmin,passax,xmax,passax);                    { Eixo X }
Draw(px(xmax),py(passax),px(xmax)-7,py(passax)-2,1);
Draw(px(xmax),py(passax),px(xmax)-7,py(passax)+2,1);
Linha(passay,ymin,passay,ymax);                    { Eixo Y }
Draw(px(passay),py(ymax),px(passay)-5,py(ymax)+3,1);
Draw(px(passay),py(ymax),px(passay)+5,py(ymax)+3,1);
x := Round((px(passay)+10)/639*79)+1;              { escreve yc }
GoToXY(x,5);Write(yc);
y := Round((py(passax)+45)/199*23)+1;              { escreve xc }
GoToXY(66,y);Write(xc);
x := passay;                      { desenha as divisoes do eixo x }
While ( x ( (xmax - divx/2) ) do
  begin
    Draw(px(x),py(passax)+2,px(x),py(passax)-2,1);
    x := x + divx;
  end;
x := passay;
While ( x ) xmin ) do
  begin
    Draw(px(x),py(passax)+2,px(x),py(passax)-2,1);
    x := x - divx;
  end;
y := passax;                      { desenha as divisoes do eixo y }
While ( y ) ymin ) do
  begin
    Draw(px(passay)-4,py(y),px(passay)+4,py(y),1);
    y := y - divy;
  end;
y := passax;
While ( y ( ymax - divy/2 ) do
  begin
    Draw(px(passay)-4,py(y),px(passay)+4,py(y),1);
```



```
    y := y + divy;
  end;
end;


procedure Limites;

  var pilha : Integer;

  begin
    If xmin ) xmax then
      begin
        pilha := xmin;
        xmin := xmax;
        xmax := pilha;
      end;
    If ymin ) ymax then
      begin
        pilha := ymin;
        ymin := ymax;
        ymax := pilha;
      end;
    fx := 500.0/(xmax - xmin);
    fy := 150.0/(ymax - ymin);
  end;


procedure Escalas(xc,yc : string8);

  var  iny, fiy  :  Real;

  begin
    GoToXY(1,6);
    WriteLn('Escalas :');
    WriteLn;
    Write(' Eixo ',xc,' - Inicio = '); ReadIn(CR,iniciox);
    Write('         Fim    = '); ReadIn(CR,fimx);
    iny := inicioy*fEi; fiy := fimy*fEi;
    Write(' Eixo ',yc,' - Inicio = '); ReadRe(CR,iny,8,2);
    inicioy := Round(iny/fEi);
    Write('         Fim    = '); ReadRe(CR,fiy,8,2);
    fimy := Round(fiy/fEi);
  end;


procedure EscalasQ;

  var  iny, fiy  :  Real;

  begin
    GoToXY(1,7);
    WriteLn('Escalas :');
    WriteLn;
    Write(' Eixo t - Inicio = '); ReadRe(CR,inicioqx,8,2);
    Write('         Fim    = '); ReadRe(CR,fimqx,8,2);
```

```
    iny := inicioqy*fEi;  fiy := fimqy*fEi;
    Write(' Eixo Q - Inicio = '); ReadRe(CR,iny,8,2);
    inicioqy := iny/fEi;
    Write('         Fim    = '); ReadRe(CR,fiy,8,2);
    fimqy := fiy/fEi;
  end;


procedure Integral(modo : Char; comeco, fim : Integer);

  var  i, j   :  Integer;

  begin
    incremento := Round(1 + (fim - comeco)/4000.0);
    i := comeco + 1 + incremento;
    j := 2;
    tempopto := incremento * fatortt;  { tempo correspondente a um ponto }
    If modo <> 'g' then
      begin
        Qmax := 0; Qmin := 0;
      end;
    Q^[0] := 0;
    Q^[1] := Crono^[comeco + 1] * fatortt;
    t^[0] := 0;
    t^[1] := fatortt;
    Repeat
      Q^[j] := Q^[j-1] + Crono^[i] * tempopto;{ em sub-unidades de C }
      t^[j] := t^[j-1] + tempopto;
      If modo <> 'g' then
        begin
          If Qmax <= Q^[j] then Qmax := Q^[j];
          If Qmin >= Q^[j] then Qmin := Q^[j];
        end;
      i := i + incremento;
      j := j + 1;
    Until (i > fim);
    nq := j - 1;
    If UpCase(unidade[1]) = 'M' then unid := 'mC';
    If UpCase(unidade[1]) = 'U' then unid := 'uC';
    If modo = 't' then
      begin
        ClrScr;
        GoToXY(10,1); Write('RESULTADOS');
        GoToXY(10,2); Write('==========='); WriteLn; WriteLn;
        If unid = 'mC' then
          begin
            WriteLn(' Carga maxima : ',(Qmax*fEi):8,' mC');
            WriteLn(' Carga minima : ',(Qmin*fEi):8,' mC');
          end;
        If unid = 'uC' then
          begin
            WriteLn(' Carga maxima : ',(Qmax*fEi):8,' uC');
            WriteLn(' Carga minima : ',(Qmin*fEi):8,' uC');
          end;
      end;
```



```
    Sound(2440); Delay(50); Sound(1220); Delay(50); NoSound;
    contInt := TRUE;
  end;


procedure Qxt(modo, tip : Char);

  begin
    If (abs(Qmax) > 30000.0) or (abs(Qmin) > 30000.0) then
      begin
        For i := 0 to nq do Q^[i] := Q^[i] / 1000.0;
        Qmax := Qmax / 1000.0;
        Qmin := Qmin / 1000.0;
        If unid = 'mC' then unid := 'C';
        If unid = 'uC' then unid := 'mC';
        If (abs(Qmax) > 30000.0) or (abs(Qmin) > 30000.0) then
          begin
            For i := 0 to nq do Q^[i] := Q^[i] / 1000.0;
            Qmax := Qmax / 1000.0;
            Qmin := Qmin / 1000.0;
            If unid = 'C' then unid := 'kC';
            If unid = 'mC' then unid := 'C';
          end;
      end;
    If modo = 'r' then
      For i := 0 to nq do t^[i] := sqrt(t^[i]);
    inicioqx := 0; fimqx := t^[nq];
    inicioqy := Qmin; fimqy := Qmax;
    If fimqy = inicioqy then fimqy := fimqy + 1.0;
    If (nc = 0) and (tip = 'i') and (s = ' ') then EscalasQ;
    If s <> ' ' then
      begin
        xmin := 0; xmax := xs;
        ymin := -ys; ymax := ys;
      end
    else
      begin
        xmin := Round(inicioqx); xmax := Round(fimqx);
        ymin := Round(inicioqy); ymax := Round(fimqy);
      end;
    If (nc = 1) and (modo = 'r') then
      If abs(ymax) > abs(ymin) then ymin := -ymax
      else ymax := -ymin;
    If (abs(xmax - xmin) < 330) and (xmax < 330) then
      begin
        If (abs(xmax - xmin) < 33) and (xmax < 33) then
          begin
            mult := 1000;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end
        else
          begin
            mult := 100;
            xmin := xmin * mult;
```

```
            xmax := xmax * mult;
          end;
      end
    else mult := 1;
    Window(1,1,80,25);
    Limites;
    If (tip = '1') and (modo = 't') then
      begin
        If mult = 1000 then EixoVolt('t','ms','Q',unid);
        If mult = 100 then EixoVolt('t','cs','Q',unid);
        If mult = 1 then EixoVolt('t','s','Q',unid);
      end;
    If (tip = '1') and (modo = 'r') then
      begin
        If mult = 1000 then EixoVolt('ti/2','m(s1/2)','Q',unid);
        If mult = 100 then EixoVolt('ti/2','c(s1/2)','Q',unid);
        If mult = 1 then EixoVolt('ti/2','s1/2','Q',unid);
      end;
    For i := 1 to nq do
      Linha(Round(t^[i-1] * mult),Round(Q^[i-1]),
            Round(t^[i] * mult), Round(Q^[i]));
    If modo = 'r' then
      For i := 0 to nq do t^[i] := sqr(t^[i]);
    Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
  end;


procedure Ixt;

  begin
    If s = ' ' then
      begin
        If not(contAmp) then
          begin
            imax := Crono^[0];  imin := Crono^[0];
            For i := 1 to tpotot do
              begin
                If imax < Crono^[i] then imax := Crono^[i];
                If imin > Crono^[i] then imin := Crono^[i];
              end;
            contAmp := TRUE;
          end;
        If (imin > 0) then inicioy := 0
        else inicioy := imin - abs(imin div 10);
        If (imax < 0) then fimy := 0
        else fimy := imax + abs(imax div 10);
        If fimy = inicioy then fimy := fimy + 10;
        iniciox := 0; fimx := Round(tpotot * fatortt);
        Escalas('t','i');
      end
    else
      begin
        iniciox := 0; fimx := xs;
        inicioy := -ys; fimy := ys;
      end;
```



```
    xmin := iniciox; xmax := fimx; ymin := inicioy; ymax := fimy;
    If (abs(xmax - xmin) < 330) and (xmax < 330) then
      begin
        If (abs(xmax - xmin) < 33) and (xmax < 33) then
          begin
            mult := 1000;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end
        else
          begin
            mult := 100;
            xmin := xmin * mult;
            xmax := xmax * mult;
          end;
      end
    else mult := 1;
    Window(1,1,80,25);
    Limites;
    If (nc = 0) or ((nc = 1) and (s <> ' ')) then
      begin
        If mult = 1000 then EixoVolt('t','ms','i',unidade);
        If mult = 100 then EixoVolt('t','cs','i',unidade);
        If mult = 1 then EixoVolt('t','s','i',unidade);
      end;
    If s = ' ' then
      begin
        i := Round(iniciox / fatortt) + 1;
        Repeat
          Linha(Round((i-1) * fatortt * mult), Crono^[i-1],
                Round(i * fatortt * mult), Crono^[i]);
          i := i + 1;
        Until (i > Round(fimx / fatortt)) or KeyPressed;
        Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
      end
    else
      begin
        For i := 1 to tpotot do
          Linha(Round((i-1) * fatortt * mult), Crono^[i-1],
                Round(i * fatortt * mult), Crono^[i]);
        Sound(1440); Delay(50); Sound(720); Delay(50); NoSound;
      end;
  end;


procedure Cursor;

  var     ch                  :  Array [1..69] of Integer;
          cv                  :  Array [1..156] of Integer;
          li,l2,l3,l4,
          poscx,poscy,incr    :  Integer;
          mov                 :  Char;
          grx, gry            :  String8;
          c1, c2, c3, c4      :  Boolean;
```

```
const    ESC = #27;


procedure novapos;

   begin
     GetPic(ch,0,poscy,500,poscy);
     GetPic(cv,poscx,0,poscx,150);
   end;

procedure antpos;

   begin
     PutPic(ch,0,poscy);
     PutPic(cv,poscx,150);
   end;

procedure novocs;

   begin
     PutPic(csh,0,poscy);
     PutPic(csv,poscx,150);
   end;


begin
  incr := 1; poscx := 250; poscy := 75; novapos; novocs;
  GoToXY(70,25); Write('csr -) ',incr:2);
  c1 := FALSE; c2 := FALSE; c3 := FALSE; c4 := FALSE;
  l1 := 0; l2 := 0; l3 := 0; l4 := 0; gry := 'i'; grx := 't';
  Repeat
    Read(kbd,mov);
    If (mov = ESC) and KeyPressed then
      begin
        Read(kbd,mov);
        Case mov of
   (F1)   #59  :  begin
                    If (c1 and c2) and (l1 (= l2 - 1) then
                      begin
                        Integral('i',l1,l2);
                        c3 := TRUE;
                        Sound(2000); Delay(50); Sound(1000); Delay(50);
                        Sound(500); Delay(50); NoSound;
                      end;
                  end;
   (F2)   #60  :  begin
                    If c3 then
                      begin
                        gry := '0'; grx := 't';
                        nc := 1;
                        Gxt('t','i');
                        nc := 0;
                        novapos; novocs;
                      end;
```



```
                  end;
   (F3)   #61  :  begin
                    If c3 then
                      begin
                        gry := '0'; grx := 't1/2';
                        nc := 1;
                        Gxt('r','1');
                        nc := 0;
                        novapos; novocs;
                      end;
                  end;
   (F4)   #62  :  begin
                    If (c1 and c2 and c3 and c4)
                        and (l1 (= l2 - 1) and (l3 (= l4 - 1) then
                      begin
                        nc := 1;
                        gry := '0'; grx := 't1/2';
                        Integral('i',l1,l2);
                        Gxt('r','1');
                        Integral('g',l3,l4);
                        Gxt('r','2');
                        nc := 0;
                        novapos; novocs;
                        Sound(1760); Delay(50); Sound(880); Delay(50);
                        Sound(440); Delay(50); NoSound;
                        contInt := FALSE;
                      end;
                  end;
   (F5)   #63  :  begin       { determina 1 coordenada a esquerda }
                    l1 := Round(((xmin + poscx / fx) / fatortt) / mult);
                    Sound(2000); Delay(50); Sound(1000); Delay(50);
                    Sound(500); Delay(50); NoSound;
                    c1 := TRUE;
                  end;
   (F6)   #64  :  begin       { determina 1 coordenada a direita }
                    l2 := Round(((xmin + poscx / fx) / fatortt) / mult);
                    Sound(2000); Delay(50); Sound(1000); Delay(50);
                    Sound(500); Delay(50); NoSound;
                    c2 := TRUE;
                  end;
   (F7)   #65  :  begin       { determina 2 coordenada a esquerda }
                    l3 := Round(((xmin + poscx / fx) / fatortt) / mult);
                    Sound(1760); Delay(50); Sound(880); Delay(50);
                    Sound(440); Delay(50); NoSound;
                    c3 := TRUE;
                  end;
   (F8)   #66  :  begin       { determina 2 coordenada a direita }
                    l4 := Round(((xmin + poscx / fx) / fatortt) / mult);
                    Sound(1760); Delay(50); Sound(880); Delay(50);
                    Sound(440); Delay(50); NoSound;
                    c4 := TRUE;
                  end;
   (F9)   #67  :  begin
                    incr := incr div 2; { increm. do desloc. do cursor }
                    If incr ( 1 then incr := 1
```

```
                else
                  Sound(880); Delay(50); NoSound;
                GoToXY(70,25); Write('csr -) ',incr:2);
              end;
(F10)   #68 : begin
                incr := incr * 2; { increm. do desloc. do cursor }
                If incr ) 64 then incr := 64
                else
                  Sound(440); Delay(50); NoSound;
                GoToXY(70,25); Write('csr -) ',incr:2);
              end;
(esq)   #75 : begin
                antpos; poscx := poscx - incr;
                If poscx ( 1 then
                  begin
                    Sound(440); Delay(50); NoSound; poscx := 500;
                  end;
                novapos; novocs; GoToXY(70,23); Write('          ');
                GoToXY(70,23);
                Write(grx,' = ',Round(xmin+poscx/fx));
              end;
(dir)   #77 : begin
                antpos;  poscx := poscx + incr;
                If poscx ) 500 then
                  begin
                    Sound(440); Delay(50); NoSound; poscx := 1;
                  end;
                novapos; novocs; GoToXY(70,23); Write('          ');
                GoToXY(70,23);
                Write(grx,' = ',Round(xmin+poscx/fx));
              end;
(cima)  #72 : begin
                antpos;  poscy := poscy - incr;
                If poscy ( 1 then
                  begin
                    Sound(440); Delay(50); NoSound; poscy := 150;
                  end;
                novapos; novocs; GoToXY(70,24); Write('          ');
                GoToXY(70,24);
                If corrente )= 200 then
                  Write(gry,' = ',((ymax - poscy / fy)*fEi):0:1);
                If (corrente ( 200) and (corrente )= 20) then
                  Write(gry,' = ',((ymax - poscy / fy)*fEi):0:2);
                If corrente ( 20 then
                  Write(gry,' = ',((ymax - poscy / fy)*fEi):0:3);
              end;
(baixo) #80 : begin
                antpos;  poscy := poscy + incr;
                If poscy ) 150 then
                  begin
                    Sound(440); Delay(50); NoSound; poscy := 1;
                  end;
                novapos; novocs; GoToXY(70,24); Write('          ');
                GoToXY(70,24);
                If corrente )= 200 then
```



```
                  Write(gry,' = ',((ymax - poscy / fy)*fEi):0:1);
                If (corrente ( 200) and (corrente )= 20) then
                  Write(gry,' = ',((ymax - poscy / fy)*fEi):0:2);
                If corrente ( 20 then
                  Write(gry,' = ',((ymax - poscy / fy)*fEi):0:3);
              end;
(F20)   #93 : begin
                antpos;      { esconde o cursor }
                GoToXY(70,23); Write('          ');
                GoToXY(70,24); Write('          ');
                GoToXY(70,25); Write('         ');
              end;
          end; { case }
        end;
      Until (mov = ESC);
    end;


procedure LimitesIntegral;
var tin,tfim : Real;
Begin
  Janela(46,5,76,19);
  GoToXY(10,1); Write('INTEGRACAO');
  GoToXY(10,2); Write('==========');
  WriteLn;
  tin := tpoa * fatortt; tfim := tpotot * fatortt;
  Write(' Inicio (s) -) '); ReadRe(CR,tin,8,2);
  Write(' Fim    (s) -) '); ReadRe(CR,tfim,8,2);
  Integral('t',Round(tin / fatortt),Round(tfim / fatortt));
End;

Begin
  contInt := FALSE;  contAmp := FALSE;
  If s = ' ' then
    begin
      Repeat
        Menu;
        Janela(26,4,56,20);
        GoToXY(5,1);
        Write('MANIPULACAO DOS DADOS');
        GoToXY(5,2);
        Write('========== === =====');
        GoToXY(1,4);
        Write('Grafico(s) : ');
        WriteLn('(A) - i x t');
        WriteLn('            (Q) - Q x t');
        WriteLn('            (R) - Q x t(1/2)');
        GoToXY(4,14);
        Write('(ESC) - Menu Inicial');
        Read(Kbd,tecla);
        tecla := UpCase(tecla);
        Case tecla of
          'A' : begin
                  ixt;
                  Cursor;
```

```
                TextMode;
              end;
        'Q' : begin
                If not(contInt) then LimitesIntegral;
                Qxt('t','i');
                Repeat Until KeyPressed;
                TextMode;
              end;
        'R' : begin
                If not(contInt) then LimitesIntegral;
                Qxt('r','i');
                Repeat Until KeyPressed;
                TextMode;
              end;
      end;  ( case )
    Until (tecla = #27);
  end
  else
    Case s of
      'I' : ixt;
      'Q' : begin
              If nc = 1 then Integral('i',tpoa,tpotot)
              else Integral('g',tpoa,tpotot);
              If nc = 1 then Qxt('t','i')
              else Qxt('t','2');
            end;
      'R' : begin
              If nc = 1 then Integral('i',tpoa,tpotot)
              else Integral('g',tpoa,tpotot);
              If nc = 1 then Qxt('r','i')
              else Qxt('r','2');
            end;
    end;
End;

Overlay Procedure Diretorio;

  Const  TextSize = 200;

  Type
    String80      =   String[80];
    String15      =   String[15];
    String5       =   String[5];
    String3       =   String[3];
    LineSize      =   String[15];
    TextArrayType =   Array[1..TextSize] of LineSize;

  Var
    FileSpec    :  String80;
    Path        :  String80;
    Name,drv    :  String15;
    AttrList    :  String5;
    DayName     :  String3;
    TextArray   :  TextArrayType;
    FileBytes,
```



```
    totbytes    :  Real;
    Code,j,k    :  Integer;
    Attr, Yr,
    Mon, Day,
    Hr, Min,
    Sec, DayNum,
    ndl         :  Byte;
    tecla,drive,
    opcao       :  Char;


procedure FileInfo(FileSpec : String80; Var FileBytes : Real;
                   Var Yr, Mon, Day, Hr, Min, Sec : Byte;
                   Var AttrList : String5);

  Type
    RegList  =  Record
                  AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                end;
    FileData  =  Record
                   Reserved  :  Array[1..21] of Byte;
                   Attr      :  Byte;
                   Time, Date, SizeLo, SizeHi  :  Integer;
                 end;

  var  Reg  :  RegList;
       DTA  :  Array[1..43] of Byte;
       FD   :  FileData Absolute DTA;

  begin
    Reg.DX := ofs(DTA);
    Reg.DS := seg(DTA);
    Reg.AX := $1A00;
    MSDOS(Reg);
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.CX := $16;
    Reg.AX := $4E00;
    MSDOS(Reg);
    If Lo(Reg.AX) <> 0 then
      begin
        FileBytes := -1.0;
        EXIT
      end;
    FileBytes := Lo(FD.SizeLo) + 256.0 * Hi(FD.SizeLo) +
                 65536.0 * Lo(FD.SizeHi) + 16777216.0 * Hi(FD.SizeHi);
    Yr := (Hi(FD.Date) shr 1) + 80;
    If Yr > 99 then
      Yr := Yr - 100;
    Mon := Lo(FD.Date) shr 5 + (Hi(FD.Date) and $01 shl 3);
    Day := Lo(FD.Date) and $1F;
    Hr := Hi(FD.Time) shr 3;
    Min := Lo(FD.Time) shr 5 + (Hi(FD.Time) and $07 shl 3);
    Sec := (Lo(FD.Time) and $1F) shl 1;
```

```
    AttrList := '     ';
    If FD.Attr and $20 = $20 then
      AttrList[1] := 'A';
    If FD.Attr and $10 = $10 then
      AttrList[2] := 'D';
    If FD.Attr and $02 = $02 then
      AttrList[3] := 'H';
    If FD.Attr and $01 = $01 then
      AttrList[4] := 'R';
    If FD.Attr and $04 = $04 then
      AttrList[5] := 'S';
  end;


procedure NextFile(FileSpec : String80; Attr : Byte;
                   var Code : Integer; Var Name : String15);

  Type
    RegList  =  Record
                  AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
                end;

  var  j       :  Integer;
       Reg     :  RegList;
       DTA     :  Array[1..43] of Byte;

  begin
    If (Code < 0) or (Code > 1) then
      begin
        Code := -1;
        EXIT;
      end;
    If Code = 0 then
      begin
        Reg.DX := ofs(DTA);
        Reg.DS := seg(DTA);
        Reg.AX := $1A00;
        MSDOS(Reg);
        FileSpec := FileSpec + Chr(0);
        Reg.DX := ofs(FileSpec[1]);
        Reg.DS := seg(FileSpec[1]);
        Reg.CX := Attr;
        Reg.AX := $4E00;
        MSDOS(Reg);
        If Lo(Reg.AX) <> 0 then
          begin
            Name := '';
            Code := 2;
            EXIT;
          end;
        If DTA[22] and $10 <> 0 then
          Name := '[D]'
        else
          Name := '';
        j := 31;
```



```
        While DTA[j] <> 0 do
          begin
            Name := Name + Chr(DTA[j]);
            j := j + 1;
          end;
        Code := 1;
      end
    else
      begin
        Reg.DX := ofs(DTA);
        Reg.DS := seg(DTA);
        Reg.AX := $4F00;
        MSDOS(Reg);
        If Lo(Reg.AX) = 0 then
          begin
            If DTA[22] and $10 <> 0 then
              Name := '[D]'
            else
              Name := '';
            j := 31;
            While DTA[j] <> 0 do
              begin
                Name := Name + Chr(DTA[j]);
                j := j + 1;
              end;
            Code := 1;
          end
        else
          begin
            Name := '';
            Code := 2;
          end;
      end;
  end;

procedure FileDel(FileSpec : String80; Var Code : Integer);

  Type
    RegList = Record
                AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
              End;

  var Reg : RegList;

  begin
    FileSpec := FileSpec + Chr(0);
    Reg.DX := ofs(FileSpec[1]);
    Reg.DS := seg(FileSpec[1]);
    Reg.AX := $4100;
    MSDOS(Reg);
    Code := Lo(Reg.AX);
    If (Reg.Flags and $01) = 0 then
      Code := 0;
  end;
```

```
procedure SortHT(Var TextArray : TextArrayType;
                 LineCount, Position, Width : Integer);

  Label
    H1, H2, H3, H4, H5, H6, H7, H8;

  var  i, j,
       l, r     :  Integer;
       ThisLine :  LineSize;

  begin
    If LineCount (= 1 then EXIT;
    H1: L := LineCount div 2 + 1;
        R := LineCount;
    H2: If L ) 1 then
          begin
            L := L - 1;
            ThisLine := TextArray[L];
          end
        else
          begin
            ThisLine := TextArray[R];
            TextArray[R] := TextArray[1];
            R := R - 1;
            If R = 1 then
              begin
                TextArray[1] := ThisLine;
                EXIT;
              end;
          end;
    H3: j := L;
    H4: i := j;
        j := j + j;
        If j ) R then goto H7;
        If j = R then goto H6;
    H5: If Copy(TextArray[j], Position, Width) (
           Copy(TextArray[j + 1], Position, Width) then j := j+1;
    H6: TextArray[i] := TextArray[j];
        goto H4;
    H7: j := i;
        i := j div 2;
    H8: If (Copy(ThisLine, Position, Width) (=
            Copy(TextArray[i], Position, Width)) or (j = L) then
          begin
            TextArray[j] := ThisLine;
            goto H2;
          end
        else
          begin
            TextArray[j] := TextArray[i];
            goto H7;
          end;
  end;
```



```
function Tab(TabCol : Integer) : Char;

  begin
    If TabCol ) 79 then
      TabCol := 79
    else
      If TabCol ( 1 then
        TabCol := 1;
    GoToXY(TabCol + 1,WhereY);
    Tab := #8;
  end;


Begin
  Janela(5,4,55,20);
  Path := 'A:';
  Attr := $16;
  Code := 0;
  j := 0;
  opcao := 'M';
  Janela(17,6,43,18);
  GoToXY(11,1);
  Write('OPCOES');
  GoToXY(11,2);
  Write('======');
  GoToXY(2,4);
  Write('(M) - Mostra Arquivos');
  GoToXY(2,6);
  Write('(D) - Deleta Arquivos');
  GoToXY(2,8);
  Write('(ESC) - Abandona   ');
  Repeat
    Read(Kbd,opcao);
    opcao := UpCase(opcao);
  Until (opcao = 'M') or (opcao = 'D') or (opcao = #27);
  If opcao = #27 then EXIT;
  GetDir(0,drv);
    Case drv[1] of
      'A'     : drive := 'B';
      'B','C' : drive := 'A';
    end; { case }
  Window(6,5,54,18);  ClrScr;
  GoToXY(1,1);
  If opcao = 'D' then
    Write('DRIVE:ARQUIVOS =      ')
  else
    Write('DRIVE:ARQUIVOS = ',drive,':*.DAT');
  Repeat
    GoToXY(18,1); Read(Kbd,tecla); tecla := UpCase(tecla);
  Until tecla in ['A','B','C',CR,#32];
  If (tecla () CR) and (tecla () #32) then
    begin
      Write('       ');
      GoToXY(18,1);
```

```
      drive := tecla; Write(drive);
      ReadLn(FileSpec);
      FileSpec := drive + FileSpec;
    end
  else
    begin
      If opcao = 'D' then
        FileSpec := drive + ':   '
      else
        FileSpec := drive + ':*.DAT';
      WriteLn;
    end;
  If Copy(FileSpec,1,2) = 'A:' then Path := 'A:';
  If Copy(FileSpec,1,2) = 'B:' then Path := 'B:';
  If Copy(FileSpec,1,2) = 'C:' then Path := 'C:';
  If Copy(FileSpec,1,2) = 'D:' then Path := 'D:';
  If (Length(FileSpec) = 2) then
    If (FileSpec[2] = ':') then
      begin
        If opcao = 'D' then
          FileSpec := '.:   '
        else
          FileSpec := '.:*.DAT';
      end;
  If FileSpec[2] <> ':' then FileSpec := '..' + FileSpec;
  FileSpec := Path + copy(FileSpec,3,80);
  Repeat
    NextFile(FileSpec, Attr, Code, Name);
    If Code = 1 then
      begin
        j := j + 1;
        TextArray[j] := Name;
      end;
  Until (Code <> 1);
  WriteLn(j,' Arquivos encontrados');
  SortHT(TextArray, j, 1, 15);
  If opcao = 'D' then
    begin
      ndl := 0;
      For k := 1 to j do
        begin
          FileSpec := Path + TextArray[k];
          FileDel(FileSpec, Code);
          If Code <> 0 then
            begin
              WriteLn(FileSpec,' nao deletado');
              ndl := ndl + 1;
            end;
        end;
      WriteLn(j - ndl,' arquivos deletados');
    end
  else
    begin
      totbytes := 0;
      For k := 1 to j do
```



```
          begin
            FileSpec := TextArray[k];
            If FileSpec[1] = 'C' then
              FileSpec := Copy(FileSpec, 4, 12);
            FileSpec := Path + FileSpec;
            FileInfo(FileSpec,FileBytes, Yr,Mon,Day, Hr,Min,Sec, AttrList);
            Write(TextArray[k], Tab(14), FileBytes:8:0, Tab(24));
            If FileBytes < 0.0 then
              WriteLn('... Arquivo nao encontrado ...')
            else
              begin
                Write(Day:2,'/',Mon:2,'/',Yr, Tab(34));
                Write(Hr:2,':');
                If Min < 10 then WriteLn('0',Min) else WriteLn(Min:2);
                totbytes := totbytes + FileBytes;
              end;
            If ((k/12) = (k div 12)) and (k < j) then
              begin
                WriteLn('  ... Aperte uma tecla ...');
                Read(Kbd,tecla);
              end;
          end;
        WriteLn('       ',totbytes:8:0,' bytes ocupados');
      end;
    Sound(1760); Delay(50);
    Sound(880); Delay(50);
    Sound(440); Delay(50);
    NoSound;
    Write('       ... [Enter] - volta ao MENU ...');
    Read(tecla);
  End;

Procedure Superpoe;
var ncs,i     : Integer;
    n         : Array[1..10] of String[10];
    ax, ay    : Real;
    modo,un   : Char;

Begin
  ncs := 1;
  Repeat
    Janela(10,4,60,20);
    GoToXY(20,1); Write('SUPERPOSICAO');
    GoToXY(20,2); WriteLn('============');
    GoToXY(2,4);  Write('(ESC) - Abandona');
    Read(Kbd,tecla);
    If tecla = #27 then EXIT;
    Repeat
      GoToXY(2,4); ClrEol;
      Write('Nro. de curvas (max.=10) = '); ReadLn(' ',ncs);
    Until ncs in [1..10];
    WriteLn; WriteLn;
    Write('  (I) - ixt    (Q) - Qxt   (R) - Qxt(1/2)   ');
    Repeat
      Read(Kbd,modo);  modo := UpCase(modo);
```

```
    Until modo in ['I','Q','R'];
    For i := 1 to ncs do
      begin
        GoToXY(2,8); ClrEol;
        Write('Nome do arquivo ',i,' -) '); Read(nCi]);
      end;
    WriteLn; WriteLn;
    Write(' Confirma? (S/N) ');
    Repeat
      Read(kbd,tecla); tecla := UpCase(tecla);
    Until tecla in ['S','N'];
  Until tecla = 'S';
  Disco(n[1],erro);
  If erro then EXIT;
  un := UpCase(unidade[1]);
  amostra := 'SUPERPOSICAO';
  fEi := corrente / 2000.0;
  ax := tpotot * fatortt; ay := 1.0 * corrente;
  GoToXY(2,12); Write('Escala: tempo    = '); ReadRe(CR,ax,8,2);
  If modo = 'i' then
    begin
      GoToXY(2,14); Write('       corrente = '); ReadRe(' ',ay,8,2);
    end
  else
    begin
      GoToXY(2,14); Write('       carga    = '); ReadRe(' ',ay,8,2);
    end;
  ClrScr;
  GoToXY(17,6); Write('...AGUARDE...');
  Resultados(modo,1,Round(ax),Round(ay/fEi));
  For i := 2 to ncs do
    begin
      Disco(n[i],erro);
      If erro then EXIT;
      If UpCase(unidade[1]) <> un then
        begin
          Aviso('unidades de corrente incompativeis');
          EXIT;
        end;
      Resultados(modo,i,Round(ax),Round(ay/fEi));
    end;
  Repeat Until KeyPressed;
End;

BEGIN
  Mark(marcaheap);
  Crono := nil;
  New(Crono);
  New(Q);
  New(t);
  Repeat
    Menu;
    Read(Kbd,tecla);
    tecla := UpCase(tecla);
    Case tecla of
```



```
      'A' : begin
              Disco(' ',erro); fEi := corrente/2000.0;
            end;
      'R' : Resultados(' ',0,0,0);
      'D' : Diretorio;
      'S' : Superpoe;
      'F' : begin
              Janela(16,10,46,15);
              GoToXY(2,2); Write('Confirma saida?  (S/N)  ');
              Read(Kbd,tecla);
              tecla := UpCase(tecla);
              If tecla = 'S' then tecla := 's';
            end;
    end;  ( case )
  Until (tecla = 's');
  Window(1,1,80,25); ClrScr;
  Release(marcaheap);
END.
```

APÊNDICE 2
ENSAIOS COM PADRÕES

Nesta seção encontram-se tabelados os valores de E° obtidos com soluções de $FeCl_3$, $K_3Fe(CN)_6$ e $Fe(C_5H_5)_2$ conforme indicados pela literatura.[32]

Os valores foram obtidos por medidas de voltametria cíclica com eletrodos de Pt (auxiliar e trabalho) e calomelano (referência).

solução	E° (mV) obtido	E° (mV) lit.
$FeCl_3$	779	770
$K_3Fe(CN)_6$	694	690
$Fe(C_5H_5)_2$	320	307