

Universidade do Algarve
Faculdade de Ciências e Tecnologia

**Controlo de temperatura, pH e condutividade da
solução nutriente de uma estufa hidropónica**

Relatório do Projecto Final da Licenciatura em Eng. de Sistemas e
Computação

César Alexandre Domingues Teixeira, nº15850

Orientador: Prof. Doutor António E. B. Ruano

Faro, Setembro de 2003

Orientando

César Alexandre Domingues Teixeira

Orientador

Prof. António E. B. Ruano

Resumo

Uma das vertentes da agricultura moderna caracteriza-se por um cultivo sem solo, chamado hidropónico. Estas técnicas de cultivo identificam-se por possuir automatismos, que permitem o controlo da solução nutriente. Nas primeiras culturas hidropónicas esses automatismos aplicavam um controlo analógico “on-off”, embebido e de difícil alteração. O controlo digital mais recente permite a aplicação de algoritmos mais sofisticados e de fácil substituição.

O objectivo deste projecto é implementar paralelamente a um sistema de controlo analógico antigo, um novo sistema de controlo digital implementado por um PC. Esta alteração tem em vista a melhoria do controlo da solução hidropónica, servindo igualmente para a comparação dos dois sistemas.

Neste relatório é apresentado um sistema prático de actuação e aquisição para controlo da solução nutriente de uma cultura hidropónica.

Abstract

One of the aspects of modern agriculture is characterised for culture without soil, called hydroponic. These culture techniques are identified for possessing automatic systems, that allow the control of the nutrient solution. In the first hydroponic cultures these automatic systems apply an “ on-off ” analogic control, integrated and of difficult alteration. The more recent digital control allows the application of more intelligent algorithms of easy adaptation. The objective of this project is to implement, in a parallel way to a old analogic control system, a new digital control system, implemented on a PC. This alteration has in sight the improvement of the control of the hydroponic solution, serving equally for the comparison of the two systems.

A practical actuation and acquisition system, for control an hydroponic nutrient solution, is presented in this report.

Agradecimentos

Gostaria de agradecer ao Prof. Doutor António E. B. Ruano pelo convite para trabalhar neste projecto, e pelas condições proporcionadas para o seu desenvolvimento.

Especiais agradecimentos ao Eng. Pedro Frazão, pelo apoio e pela oportunidade de experimentação do sistema desenvolvido no Centro de Ciência Viva do Algarve.

Um grande obrigado ao Laboratório de Optoelectrónica, pela cedência do material que permitiu a realização de parte do “hardware”.

Agradecimentos à FCT, pela concessão da bolsa de iniciação à investigação científica (BIC), ao abrigo do projecto POCTI/33906/MGS/2000.

A todos os colegas de licenciatura e que trabalham no Centro de Sistemas Inteligentes, um muito obrigado pela companhia e apoio nas horas de maior “stress”.

Finalmente um agradecimento especial a minha família por sempre acreditarem em mim e por nunca me faltar nada durante o curso.

Índice

| | | |
|----------|--|-----------|
| 1 | Introdução | 1 |
| 1.1 | A hidroponia | 4 |
| 1.1.1 | Vantagens e desvantagens | 5 |
| 1.1.2 | A solução nutriente | 6 |
| 1.1.3 | NFT (Nutrient Film Technique) | 10 |
| 1.1.4 | Controlo hidropónico | 11 |
| 1.2 | Artigo relevante | 12 |
| 1.3 | Levantamento do sistema antigo | 14 |
| 2 | Implementação | 17 |
| 2.1 | Implementações de “hardware” | 17 |
| 2.1.1 | Placa de aquisição de dados | 17 |
| 2.1.2 | Bloco terminal de 50 posições | 20 |
| 2.1.3 | “Drivers” <i>TTL</i> | 20 |
| 2.1.4 | Elementos de comutação | 22 |
| 2.1.5 | Actuadores | 25 |

| | |
|---|-----------|
| <i>ÍNDICE</i> | vi |
| 2.1.6 Sensores | 26 |
| 2.1.7 “Data-logger” | 27 |
| 2.1.8 Aquisição do “sinal de rega” | 28 |
| 2.1.9 Porta paralela | 28 |
| 2.1.10 Fonte de alimentação | 30 |
| 2.1.11 Alteração do sistema antigo | 30 |
| 2.1.12 Interligação dos diversos dispositivos | 32 |
| 2.2 Implementações de “software” | 32 |
| 2.2.1 Linguagem de programação <i>Python</i> | 33 |
| 2.2.2 “Device drivers” | 34 |
| 2.2.3 Código de alto nível | 39 |
| 2.2.4 Interface gráfico | 39 |
| 2.2.5 Interligação dos diversos componentes de “software” | 41 |
| 3 Resultados | 43 |
| 4 Conclusão e trabalho futuro | 45 |
| Anexo A | 46 |
| A.1 Introdução | 47 |
| A.2 Instalação do “driver” | 47 |
| A.2.1 Pré-requisitos | 47 |
| A.2.2 Instalação passo a passo | 48 |
| A.2.3 Remoção do módulo | 48 |

| | |
|---|-----------|
| <i>ÍNDICE</i> | vii |
| A.3 Usando o “driver” | 48 |
| A.4 Acesso dos programas do utilizador ao dispositivo | 49 |
| A.4.1 read | 50 |
| A.4.2 write | 50 |
| A.4.3 ioctl | 50 |
| A.4.4 open e close | 54 |
| Anexo B | 55 |
| B.1 Introdução | 56 |
| B.2 Instalação do “driver” | 56 |
| B.2.1 Pré-requisitos | 56 |
| B.2.2 Instalação passo a passo | 57 |
| B.2.3 Remoção do módulo | 57 |
| B.3 Usando o “driver” | 58 |
| B.4 Acesso dos programas do utilizador ao dispositivo | 58 |
| B.4.1 read | 58 |
| B.4.2 write | 59 |
| B.4.3 ioctl | 59 |
| B.4.4 open e close | 59 |
| Anexo C | 61 |
| C.1 Introdução | 62 |
| C.2 Uso dos módulos | 62 |

| | | |
|----------------|--|-----------|
| C.2.1 | Pré-requisitos | 62 |
| C.2.2 | Como usar os módulos | 63 |
| C.3 | Descrição das funções de cada módulo | 63 |
| C.3.1 | Módulo adc44d | 63 |
| C.3.2 | Módulo adc44d_adc | 64 |
| C.3.3 | Módulo adc44d_dac | 67 |
| C.3.4 | Módulo adc44d_dio | 69 |
| Anexo D | | 72 |
| D.1 | Introdução | 73 |
| D.2 | Uso do módulo | 73 |
| D.2.1 | Pré-requisitos | 73 |
| D.2.2 | Como usar o módulo | 73 |
| D.3 | Descrição das funções do módulo | 74 |
| D.3.1 | fd=ctparport_open() | 74 |
| D.3.2 | ctparport_close(ctpar_fd) | 74 |
| D.3.3 | ctpar_enable_int(fd) | 74 |
| D.3.4 | ctpar_disable_int(fd) | 74 |
| D.3.5 | status=ctpar_get_int_stat(fd) | 75 |
| D.3.6 | val=ctparport_read_sel(fd,int_flag) | 75 |
| D.3.7 | ctparport_write(fd,value) | 75 |
| Anexo E | | 76 |

| | | |
|-------|--|----|
| E.1 | Introdução | 77 |
| E.2 | Uso do interface | 77 |
| E.2.1 | Pré-requisitos | 77 |
| E.2.2 | Como por interface a funcionar | 78 |
| E.2.3 | Como utilizar o interface | 79 |

Lista de Figuras

| | | |
|------|---|----|
| 1.1 | Diagrama esquemático da estrutura de controlo seguida | 3 |
| 1.2 | Relação entre o pH do solo e a absorção de nutrientes. | 8 |
| 1.3 | Implementação esquemática do sistema NFT | 10 |
| 1.4 | Circuito eléctrico do painel de controlo | 15 |
| 2.1 | Placa de aquisição de dados BCT ADC44D. | 18 |
| 2.2 | Pinos do <i>DIO</i> e respectiva função. | 19 |
| 2.3 | ”Layout“ do bloco terminal, em placa de circuito impresso. | 21 |
| 2.4 | ”Layout“ do circuito para os drivers <i>TTL</i> , em placa de circuito impresso. | 22 |
| 2.5 | Placa de suporte para os módulos opto-acoplados. | 23 |
| 2.6 | ”Layout“ do circuito para o relé de estado sólido, em placa de circuito impresso. | 25 |
| 2.7 | Disposição dos sensores no sistema de rega. | 26 |
| 2.8 | Diagrama de blocos do ”data-logger“. | 27 |
| 2.9 | Diagrama explicativo da porta paralela de um PC. | 29 |
| 2.10 | Diagrama do painel incluindo o comutador manual | 31 |
| 2.11 | Interligação dos diversos dispositivos de aquisição. | 32 |

| | | |
|------|---|----|
| 2.12 | Interligação dos diversos dispositivos de actuação. | 33 |
| 2.13 | Diagrama explicativo do enquadramento do <i>device-driver</i> para a placa ADC44D | 37 |
| 2.14 | Diagrama explicativo do enquadramento do <i>device-driver</i> para a porta paralela | 38 |
| 2.15 | Janela principal do Interface gráfico. | 40 |
| 2.16 | Apresentação gráfica do estado dos actuadores. | 40 |
| 2.17 | Diagrama explicativo da interligação dos diversos componentes de “software”. | 42 |
| E.1 | Tabelas da base de dados de teste. | 78 |
| E.2 | Janela principal do interface. | 79 |
| E.3 | Janelas de alteração das “Referências” e dos “Parâmetros de Controlo” . . . | 80 |
| E.4 | Mensagem de erro. | 80 |
| E.5 | Apresentação gráfica do estado dos actuadores. | 81 |
| E.6 | Janela de escolha dos gráficos. | 81 |
| E.7 | Gráfico da evolução dos valores (simulados) para a variável Temperatura . . | 82 |

Lista de Tabelas

| | | |
|-----|---|----|
| A.1 | Numeração dos dispositivos e convenção de nomes para o driver bct-adc-44d | 49 |
| A.2 | Valores possíveis do argumento para o ioctl ADC44D_DAC_ECHAN . . | 51 |
| A.3 | Valores possíveis do argumento para o ioctl ADC44D_DAC_SCHAN . . | 51 |
| A.4 | Valores possíveis do argumento para o ioctl ADC44D_ADC_SMODE . . | 52 |
| A.5 | Valores possíveis do argumento para o ioctl ADC44D_ADC_SGAIN . . | 52 |
| A.6 | Listagem e função das “control words” do DIO a funcionar no Modo 0. . . . | 54 |
| C.1 | Valores possíveis do argumento channels na função dac_enable_channels . | 68 |

Capítulo 1

Introdução

As culturas de plantas directamente no solo são limitadas por diversos factores, tais como: a difícil determinação da quantidade de fertilizantes a usar na rega, dependência da localidade da cultura (por exemplo: clima, solos áridos, má qualidade das águas, etc.), má gestão da água, maior perigo do aparecimento de doenças nas raízes, entre outros. O cultivo de plantas sem solo, conhecido como hidropónico, permite minimizar os problemas anteriormente citados. Numa cultura deste tipo todos os nutrientes que as plantas necessitam são fornecidos por uma solução balanceada. O correcto controlo desta solução contribui significativamente para um aumento ao nível da produção, não desprezando a qualidade. Este controlo é realizado em sistemas antigos por controladores analógicos, onde a estratégia de controlo resume-se a ligar ou desligar actuadores (funcionamento “on-off”). Estes controladores normalmente possuem uma única estratégia de controlo implementada em “hardware”, que por regra não pode ser alterada. A alteração da mesma implica possivelmente a substituição do controlador e de toda a electrónica adjacente. O controlo digital implementado por micro-processadores

possibilita a aplicação de novas estratégias de controlo, onde a alteração do comportamento do sistema passa simplesmente pela alteração de um programa de computador, não implicando qualquer substituição de “hardware”. Assim o processo de alteração da estratégia torna-se numa tarefa de fácil execução, permitindo simultaneamente o uso de estratégias de controlo mais robustas que o simples controlo analógico.

Este projecto tem como objectivo implementar paralelamente a um sistema de controlo analógico já existente, um novo sistema de controlo da solução nutriente, implementado por um P.C.¹. O sistema analógico encontra-se instalado num painel, dentro de uma estufa hidropónica localizada no Centro de Desenvolvimento de Ciências e Técnicas de Produção Vegetal (Faculdade de Engenharia Dos Recursos Naturais - Universidade do Algarve). Com este paralelismo o controlo pode ser realizado pelo sistema antigo ou pelo novo sistema, bastando para tal seleccionar o pretendido através de um comutador manual. O objectivo final é melhorar o controlo das variáveis da solução nutriente, usando técnicas mais avançadas de controlo. Este objectivo é conseguido através da programação de controladores digitais no P.C. A presença dos dois sistemas permite a comparação do comportamento do sistema antigo com o comportamento do novo sistema digital.

O sistema analógico controla o pH e a condutividade; o novo sistema, para além destas variáveis da solução, pretende controlar também a temperatura da mesma.

Os trabalhos realizados neste projecto passaram pelo levantamento do sistema já existente, tanto ao nível do sistema de controlo como ao nível do sistema de rega. Seguiu-se o projecto

¹Sigla inglesa para Personal Computer (P. C.)

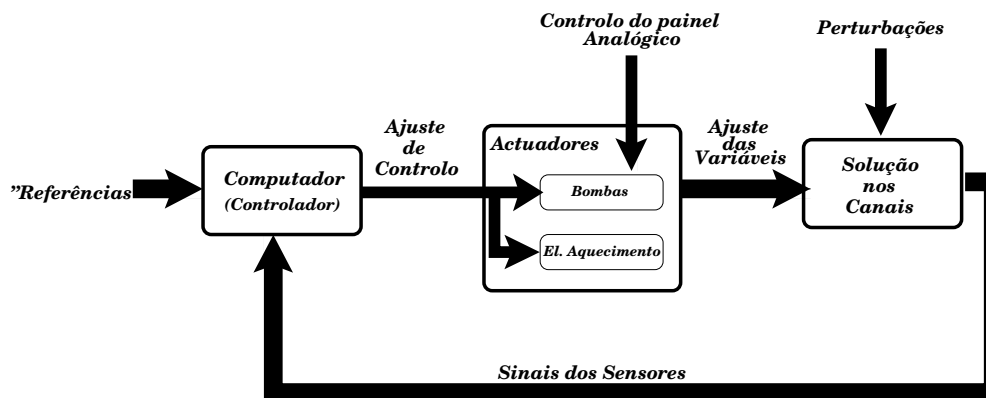


Figura 1.1: Diagrama esquemático da estrutura de controlo seguida

e realização de toda a electrónica de interface do P.C. com os actuadores e sensores. Por fim realizou-se todo o “software” necessário para permitir o controlo da electrónica desenvolvida e a apresentação dos dados adquiridos.

A programação de um controlador digital para as variáveis da solução tendo como modelo para os canais² uma série de funções de transferência de primeira ordem com tempos de atraso, tal como se tinha pensado no início do projecto, não foi possível dado aos normais atrasos na entrega dos componentes e na montagem dos canais de nutrientes.

Todo o sistema de actuação e aquisição foi desenhado tendo em vista a estrutura de controlo, que pode ser vista na Figura 1.1. Nesta figura é visível a relação entre o novo e o velho controlo. Pode também ser extraído desta figura que o sistema tratado neste projecto é do tipo **MIMO**³.

²Entende-se como canais a estrutura por onde circula a solução nutriente e onde são fixadas as plantas. Estes canais são característicos do método NFT (Nutrient Film Technique) apresentado na secção 1.1.3 e no qual se baseou este projecto.

³Multiple Input Multiple Output

Nas secções seguintes deste capítulo introdutório são introduzidos os conceitos básicos no que respeita a hidroponia, sendo também apresentado o sistema antes da actualização, e o resumo de um artigo no qual se pretende basear, no futuro, a modelação do comportamento da solução nutriente nos canais.

No capítulo 2 deste relatório são apresentadas as implementações realizadas, quer ao nível de “software”, quer ao nível de “hardware”. Segue no capítulo 3 a descrição dos resultados obtidos e finalmente no capítulo 4 são apresentadas as conclusões respectivas, sendo também indicadas sugestões para trabalho futuro.

1.1 A hidroponia

A hidroponia é um sistema de cultivo dentro ou fora de estufas, onde as plantas não crescem fixadas ao solo .

Os nutrientes que a planta precisa para seu desenvolvimento e produção são fornecidos somente por água [1] [2]. Neste tipo de cultivo as plantas são colocadas em canais ou em recipientes onde uma solução nutriente é introduzida, podendo as raízes das plantas estar ou não envolvidas num meio inerte. Em [3] são descritos cinco métodos de cultura hidropónica usados comercialmente: cultura em areia, cultura em “rockwool”⁴, cultura em cascalho, NFT (Nutrient Film Technique) e cultura em saco. Neste relatório será apenas abordado o método NFT, visto ser o método que se pretende utilizar futuramente na estufa.

A solução nutriente, após passar pelas raízes das plantas pode ser reaproveitada ou não. No primeiro caso diz-se que o sistema é re-circulatório, sendo esta topologia utilizada em

⁴Meio de crescimento inerte para as raízes, feito de fibras finas fabricadas a partir de rocha.

sistemas NFT e em sistemas que usam cascalho. No segundo caso trata-se de um sistema não re-circulatório ou em malha aberta, sendo esta topologia usada nos restantes métodos anteriormente referidos.

1.1.1 Vantagens e desvantagens

Este tipo de cultura pode trazer vários benefícios, quer para o produtor quer para o consumidor. Ao nível do consumidor pode-se referir: os produtos hidropónicos tendem a ser vendidos devidamente embalados, prevenindo o contacto com meios poluentes durante o transporte. A embalagem pode também servir para uma correcta identificação do produto (origem, nome do fabricante, etc.), certificando o produto.

Para o produtor inúmeras vantagens se levantam, tanto do ponto de vista económico, ambiental como do ponto de vista da qualidade de vida. Economicamente é de salientar uma redução de gastos em mão de obra e em máquinas, devido ao facto do processo de tratamento do solo ser desnecessário. Por outro lado, uma melhor gestão da água e dos nutrientes imposta pelo cultivo hidropónico, leva a uma poupança de recursos. Os produtos agrícolas tendem a ser mais uniformes e de melhor qualidade, facilitando a sua comercialização. A nível ambiental um menor número de pulverizações requeridas por estas culturas impõe uma menor poluição dos solos reflectindo-se também numa melhor qualidade de vida para o trabalhador agrícola, visto que está menos exposto a pesticidas. A qualidade de vida do trabalhador agrícola também é aumentada pelo facto deste tipo de trabalho ser mais limpo e mais leve.

Como inconvenientes há a apontar: os elevados custos iniciais impostos pela preparação do terreno para as estufas e para a construção das mesmas, a grande dependência da energia eléctrica que alimenta o vários sistemas de bombagem e controlo e o elevado conhecimento técnico necessário. Ainda há a referir a maior probabilidade de propagação de doenças pelo facto de, num sistema com re-circulação, onde existe uma grande população de plantas, um individuo doente poder contaminar parte da população [1].

1.1.2 A solução nutriente

A solução nutriente de uma estufa hidropónica é composta por nutrientes dissolvidos em água, onde o balanceamento dos nutrientes depende da espécie vegetal em questão. Estes nutrientes podem ser divididos em macro-nutrientes e micro-nutrientes. As plantas necessitam de 16 minerais para o seu crescimento, no entanto existem alguns que são mais importantes que outros. Entre os mais importantes pode-se citar: azoto (N), fósforo (P) e potássio (K), sendo estes conhecidos como macro-nutrientes. Além destes minerais também são considerados macro-nutrientes: Sulfato (S), Cálcio (Ca) e Magnésio (Mg). Os micro-nutrientes estão presentes em pequenas quantidades nas plantas. Fazem parte deste grupo os seguintes elementos: Boro (B), Cobre (Cu), Cobalto (Co), Ferro (Fe), Manganésio (Mn), Molibdénio (Mo) e Zinco (Zn). Industrialmente o controlo desta solução é feito essencialmente pela manutenção do seu pH, condutividade eléctrica e temperatura, sendo os macro e micro nutrientes introduzidos sobre a forma de sais, por exemplo: CaCl_2 , KH_2PO_4 , etc.

pH

O pH é uma medida do grau de acidez ou alcalinidade de uma solução. Esta medida é expressada como o logaritmo negativo da concentração de iões hidrogénio (H^+) numa solução ($pH = -\log_{10}([H^+])$). Se a concentração de iões hidrogénio sobe então o pH decresce, e vice-versa. A escala de pH compreende valores entre 0 e 14, correspondendo um pH entre 0 e 7 a uma solução ácida e entre 7 e 14 a uma solução básica ou alcalina. O valor 7 corresponde a substâncias neutras, como é o caso da água destilada.

No caso da hidroponia, um correcto controlo do pH é importante para manter uma absorção equilibrada dos diversos nutrientes. Numa cultura hidropónica um valor recomendado de pH, por vários autores, situa-se entre 6 e 7, ou seja, uma solução pouco ácida. Estes valores surgem pelo facto da absorção dos nutrientes ser optimizada neste intervalo de valores. A absorção de manganésio, cobre, zinco e especialmente ferro é reduzida a valores altos do pH. Por outro lado a valores baixos de pH existe uma pequena baixa na absorção de fósforo, potássio, cálcio e magnésio. As ideias anteriormente expostas podem ser sumariadas pela Figura 1.2 [4].

O pH da solução varia com a luz e temperatura ao longo do dia. A fotossíntese intensa durante as horas de luz provoca um aumento do pH, por outro lado ao anoitecer, quando a fotossíntese diminui, a respiração intensa causa uma diminuição do pH.

Um correcto controlo desta variável permite que ela se mantenha dentro dos valores recomendados, compensando as flutuações anteriormente referidas. Este controlo é feito pela adição de uma concentração diluída de ácido fosfórico ou nítrico (diminuição do pH) ou pela

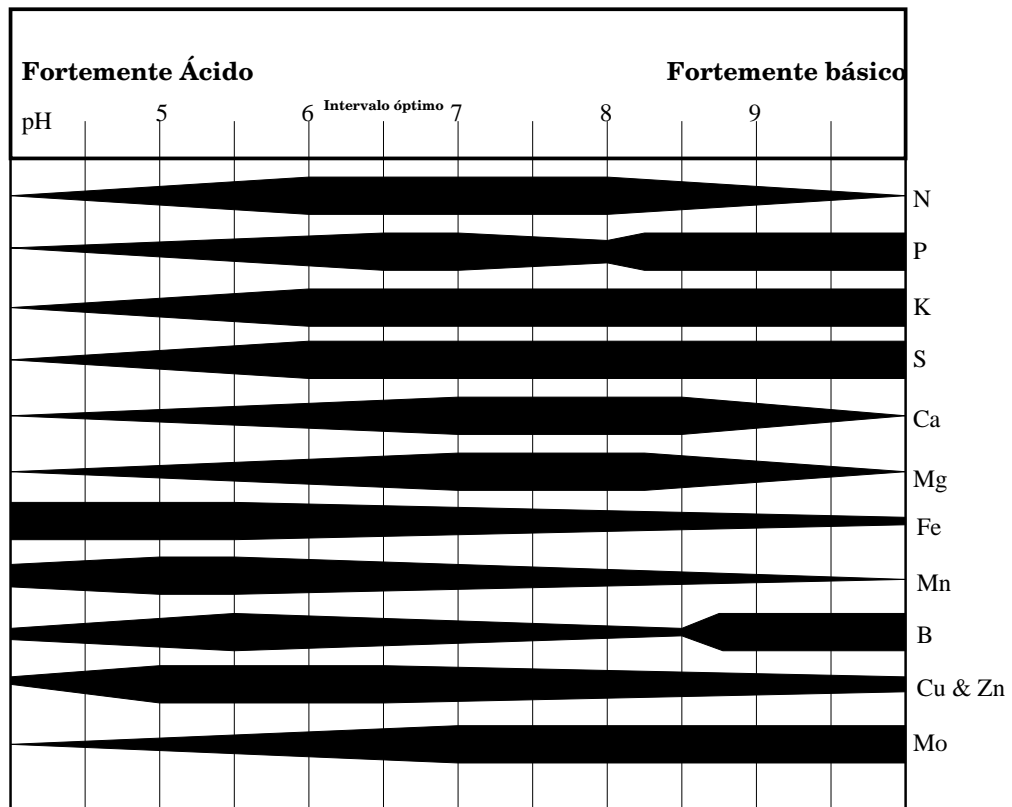


Figura 1.2: Relação entre o pH do solo e a absorção de nutrientes.

adição de uma solução diluída de hidróxido de potássio (aumento do pH) [5].

Condutividade eléctrica

A condutividade eléctrica é uma medida da facilidade de condução de corrente eléctrica numa solução aquosa. Esta está proporcionalmente relacionada com o total de sais dissolvidos, sendo deste modo um indicador eficaz da composição da solução hidropónica. A condutividade eléctrica é dada em microsiemens/cm ($\mu S/cm$), milisiemens/cm (mS/cm) ou micromho/cm ($\mu mho/cm$). Para esta variável da solução, ao contrário do pH, não existem valores recomendados, dado estes dependerem da altura do ano, da fase de crescimento da planta e da qualidade da água.

Ainda há a referir o facto da condutividade aumentar cerca de 2% por cada grau célsius da solução, o que leva a que os controladores desta variável possuem um compensador de temperatura integrado [3].

Temperatura

As flutuações da temperatura da solução hidropónica afectam o pH da solução, bem como a solubilidade dos nutrientes. Estudos indicam que um valor óptimo para a temperatura da solução situa-se no intervalo [20 C° e 22 C°]. Caso a temperatura da água saia fora deste intervalo muitos elementos de percurso (micro-nutrientes) tornam-se insolúveis, tendo o mesmo efeito que o uso de valores extremos para o pH [5].

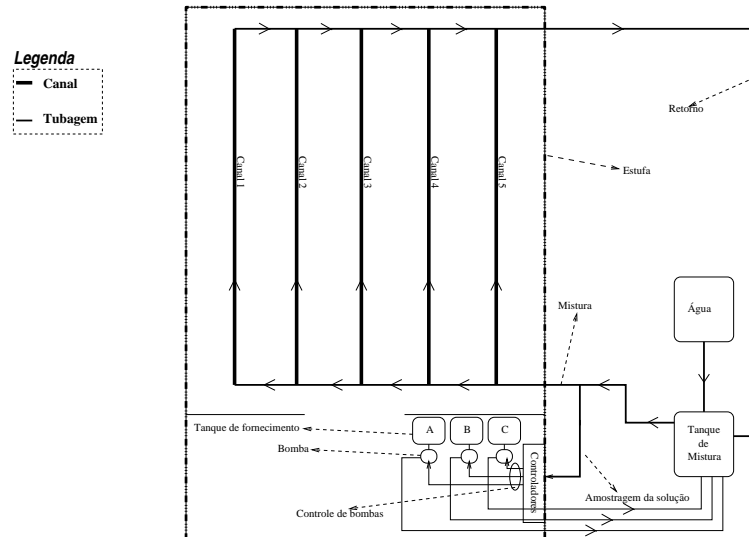


Figura 1.3: Implementação esquemática do sistema NFT

1.1.3 NFT (Nutrient Film Technique)

Este sistema/método foi desenvolvido por Cooper no Reino Unido em 1975 [3] e consiste num sistema onde as plantas não têm qualquer meio de sustentação ao nível das raízes, ao contrário dos outros métodos referidos em 1.1. As plantas são alimentadas por um fluxo contínuo de nutrientes, que flui ao longo de canais. No fim dos canais a solução não absorvida pelas plantas é re-introduzida no tanque de mistura original, sendo novamente misturada com novos nutrientes, caso seja necessário, e é de novo bombeada para os canais. O pH, condutividade e temperatura da solução são controlados por instrumentação electrónica. Os controladores actuam sobre as bombas e elementos de aquecimento de modo a transferir uma correcta quantidade de ácido e nutrientes dos tanques de fornecimento para o tanque de mistura e a manter uma correcta temperatura da solução. Os tanques de fornecimento contêm ácido e uma concentração padrão dos nutrientes (macro-nutriente e micro-nutrientes). Um diagrama explicativo deste sistema pode ser visto na Figura 1.3.

1.1.4 Controlo hidropónico

Os controladores mais usados neste tipo de sistemas, podem ser classificados como [3]:

- Controladores *On/Off*: A acção de controlo resulta simplesmente em ligar ou desligar um actuador.
- Controladores Moduladores: Estes controladores são normalmente usados para controlo de válvulas que recebem uma entrada pulsada, onde a abertura da válvula é proporcional à frequência dos pulsos ou à duração dos mesmos.
- Controladores Proporcionais: Este tipo de controladores fornecem uma acção de controlo proporcional ao desvio em relação à referência (erro).
- Controladores Implementados em Micro-processadores: Para implementar um controlo mais geral, ou seja, o controlo simultâneo de várias variáveis, com algoritmos de controlo mais sofisticado é por vezes utilizado um sistema micro-processador. Nestes sistemas os controladores são programas de computador (controlo digital) que recebem o estado do sistema através de circuitos de conversão de entrada e devolvem o resultado da computação para o ambiente através de circuitos de conversão de saída.

Os circuitos de conversão de entrada/saída são o interface entre o programa de computador (controlador) e o ambiente a controlar. Os conversores de entrada estão ligados aos sensores e convertem grandezas físicas contínuas em sinais digitais, "inteligíveis" pelo micro-processador. Por sua vez os conversores de saída realizam a operação inversa. Podem ser classificados como circuitos conversores os seguintes dispositivos: ADC (Analog to Digital Converters), DAC (Digital to Analog Converter), interfaces

de pulsos para interacção com sensores de saída pulsada e com actuadores de entrada pulsada e interfaces digitais para interacção com dispositivos que funcionam em estado “ON/OFF”.

Como já tinha sido referido anteriormente, o objectivo deste projecto é a implementação de um sistema de controlo digital, onde o controlador é implementado num sistema microprocessador, neste caso um PC.

1.2 Artigo relevante

Um artigo do ano de 1985 [6], expõe a modelação e controlo de um sistema hidropónico NFT. Neste trabalho são considerados relevantes para a modelação somente os canais de nutrientes, sendo os efeitos provocados pela bombagem e pelo tanque de mistura desprezados. No entanto é indicado como trabalho futuro a consideração destes aspectos na respectiva modelação. A modelação efectuada tem como base testes experimentais, realizados com uma substância de percurso, neste caso nitrato de potássio. Para exprimir matematicamente os resultados das experiências, ao fim de 28 semanas, foi considerado um modelo *ADZ* (*Agregated Dead Zone*). Este modelo foi concebido considerando balanços de massa de nitrato de potássio ao longo de um segmento do canal. O tanque de recolha é considerado como continuamente agitado, de onde resulta a seguinte equação diferencial:

$$\frac{dx(t)}{dt} = - \left(r + \frac{1}{T} \right) x(t) + \frac{1}{T} u(t - \tau) \quad (1.1)$$

onde $u(t)$ é a concentração do nutriente a entrada do segmento de canal, em $mg\,l^{-1}$, no tempo $t(s)$; $x(t)$ é a concentração ($mg\,l^{-1}$) à saída do segmento de canal; T é o tempo de residência do modelo ADZ em segundos, definido por $\frac{V_E}{Q}$ onde Q é o caudal e V_E é o volume considerado no modelo; τ é o tempo de atraso de transporte, isto é, o tempo entre a primeira chegada de nutrientes no início do segmento de canal e a primeira chegada no fim do segmento. Por último r é a taxa de decaimento dos nutrientes em hertz (Hz), que no caso particular deste sistema é função da taxa de absorção de nutrientes por parte das plantas. Deste modo o ganho em regime estacionário ($\frac{dx(t)}{dt} = 0$) é dado por:

$$G = \frac{1}{1 + rT} \quad (1.2)$$

Após uma discretização, 1.1 resulta em:

$$x_k = -a_1 x_{k-1} + b_0 u_{k-\delta} \quad (1.3)$$

onde δ é definido por $round(\frac{\tau}{\Delta t})$; $a_1 = -exp\left(-\frac{(1+rT)\Delta t}{T}\right)$ e $b_0 = G(1 + a_1)$. De referir que estas relações são exactas se u_k é constante durante Δt . A função de transferência discreta que relaciona a concentração no fim do segmento de canal, no instante k , com a concentração no início do segmento medida δ instantes antes, é dada por:

$$x_k = \frac{b_0 z^{-\delta}}{1 + a_1 z^{-1}} u_k \quad (1.4)$$

Um correcto número destas funções de transferências dispostas em série, levam à obtenção de um modelo para um canal completo.

Os valores de δ , a_1 e b_0 , assim como o correcto número de funções de transferência foram estimados usando o método recursivo IV (Instrumental Variable) (Young, 1984,1985).

Os resultados mostrados no artigo demonstram que o modelo ADZ representa bem o sistema NFT.

Os efeitos do tanque de recolha foram desprezados, como já tinha sido referido anteriormente, dado que na altura da publicação os testes ainda não eram significativos. Os autores referem que este facto não dificultou o desenho do sistema de controlo, visto que a resposta do referido tanque ser rápida.

Os resultados das experiências demonstram que a absorção de nutrientes por parte das plantas é baixo durante uma experiência, conseqüentemente a estimação de r é pouco significativa. Uma solução proposta pelos autores passa por considerar tempos longos para as experiências, no entanto no momento da elaboração do artigo ainda não tinha sido aplicada. Para a estimação de r os autores basearam-se em literatura técnica disponível de sistemas NFT, onde é indicado que a absorção de nutrientes está intimamente relacionada com a intensidade da luz. Os resultados obtidos para o ganho estimado em regime estacionário estão muito próximos da unidade, o que leva a propor que o ganho seja considerado mais baixo, para tomar em conta as perdas não consideradas.

1.3 Levantamento do sistema antigo

Para a compreensão do funcionamento do sistema de rega, foi feito um levantamento do mesmo. Este sistema apresentava todas as características do sistema NFT, apresentado em 1.1.3, com a diferença que no lugar de canais possuía sacos cheios de uma substância inerte. Esta diferença não impôs nenhum entrave no desenvolvimento do sistema de actuação e

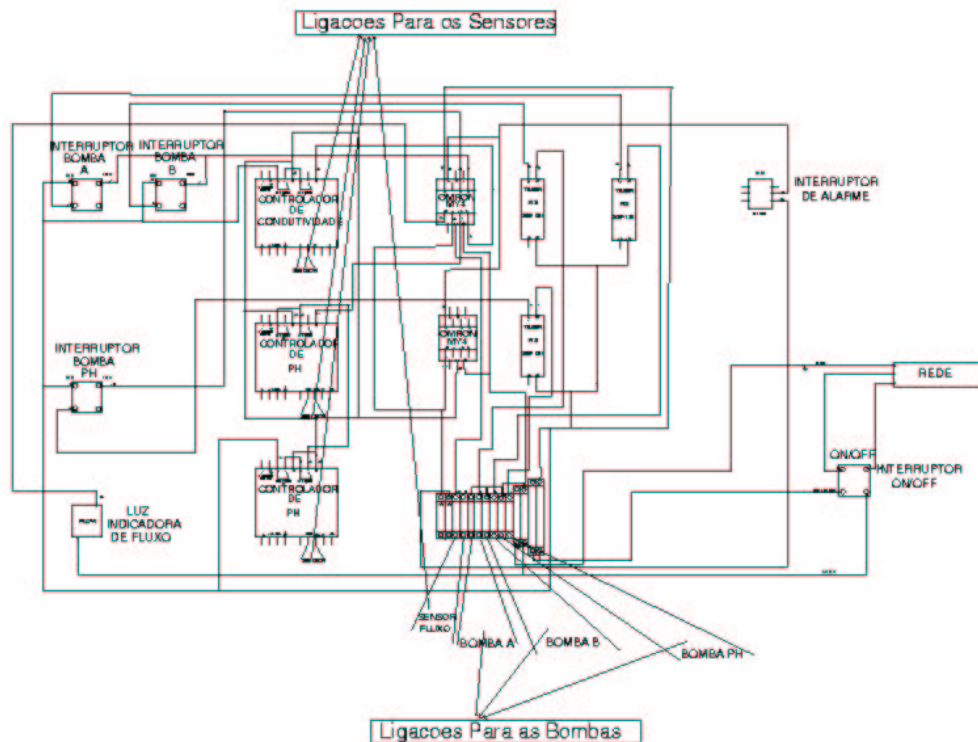


Figura 1.4: Circuito eléctrico do painel de controlo

aquisição, dado que estes sistemas podem ser usados em qualquer tipo de cultura hidrónica.

Da mesma forma foi feito um levantamento do circuito eléctrico do painel de controlo, tendo como objectivo o conhecimento dos pontos de interface do mesmo com os actuadores e sensores. Este conhecimento foi essencial para posteriormente se realizar o interface dos mesmos actuadores e sensores com o sistema controlado pelo PC. O levantamento deste circuito encontram-se representados na Figura 1.4, onde é apresentado os pontos de ligação com as bombas e com os sensores. Este circuito é composto basicamente por três controladores analógicos, por elementos de comutação dos actuadores (relés) e por interruptores manuais de selecção de funções. Dois dos controladores são de pH sendo o restante de condutividade. A presença de dois controladores de pH têm como fundamento um controlo mais seguro,

evitando injeções excessivas de ácido na solução.

Capítulo 2

Implementação

Neste capítulo são apresentados as implementações realizadas neste projecto. Na secção 2.1 são apresentadas as implementações de “hardware”, sendo na secção 2.2 apresentadas as implementações de “software”.

2.1 Implementações de “hardware”

Esta secção relata o “ hardware” utilizado neste projecto de fim de curso. Nas sub-secções seguintes são apresentadas as características dos diferentes dispositivos utilizados, sendo também descrito para cada dispositivo específico a sua interligação com os restantes.

2.1.1 Placa de aquisição de dados

A transformação da acção de controlo resultante do algoritmo em execução no PC, para um sinal eléctrico é feita, neste projecto por uma placa de aquisição de dados, do fabricante *Blue Chip Technology*, modelo *ADC44D*(Fig. 2.1).

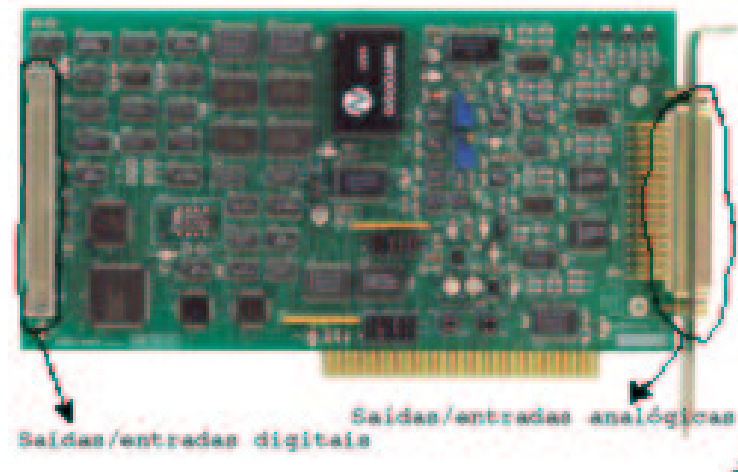


Figura 2.1: Placa de aquisição de dados BCT ADC44D.

Esta placa possui 24 entradas/saídas digitais, 3 temporizadores programáveis e 20 entradas e saídas analógicas.

Dois dos temporizadores possuem saída para o exterior, funcionando o terceiro como pré-escalador dos outros dois.

A conversão analógica-digital é assegurada por uma ADC com 12 bits de resolução. Esta possui 16 canais (entradas) simples, ou 8 canais (entradas) diferenciais. A ADC pode converter entradas com amplitude entre $\pm 50\text{mVolts}$ até $\pm 10\text{ Volts}$.

A conversão digital-analógica é feita por uma DAC de 4 canais, que podem fornecer uma tensão entre $\pm 10\text{ Volts}$ e uma corrente entre 0 e 20 mA. A resolução da DAC é de 12 bits.

As conversões analógicas (entradas ou saídas) podem ser feitas sobre controlo I/O, de interrupções ou DMA¹, podendo-se utilizar os temporizadores para controlo da taxa de operação.

A especificação detalhada desta placa pode ser vista em [7].

¹Direct Memory Access

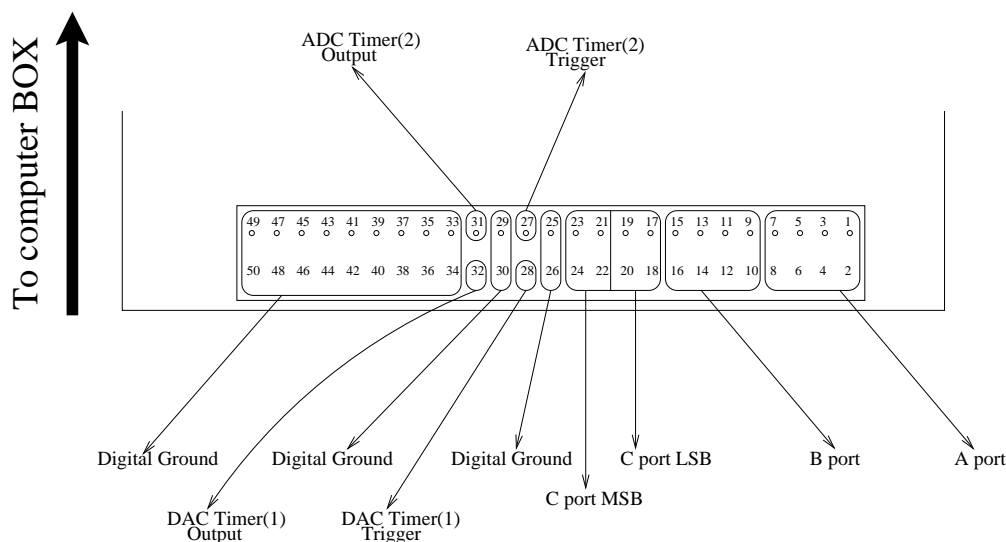


Figura 2.2: Pinos do *DIO* e respectiva função.

Neste projecto usam-se unicamente as conexões digitais (*DIO-Digital Input/Output*), dado que se pretende somente ligar ou desligar dispositivos. Estas conexões são apresentadas ao exterior através de um conector *IDC (Insulation Displacement Connector)*² de 50 pinos (Figura 2.2).

Os pinos de 1 a 24 são as saídas de um *NEC $\mu PD71055$* , que é equivalente ao *Intel i8255 PIO*³. Estes circuitos integrados fornecem conexões vocacionadas para enviarem ou receberem sinais *TTL*. A comunicação com estes dispositivos é feita através de 4 portos de 8 bits. Os primeiros 3 (A, B e C) podem ser configurados como entradas/saídas, sendo o quarto reservado para controlo. O *$\mu PD71055$* pode funcionar em um de 3 modos. No primeiro modo (Modo 0) os portos A, B e C são usados para simples operações de leitura e escrita, sem uso de “handshaking”. O segundo modo (Modo 1) permite a troca de dados através das portos A e B, sendo os sinais de “handshaking” fornecidos através do porto C. No último modo (Modo

²Conectores que fazem a conexão através do deslocamento do isolamento do condutor [8].

³Programmed Input/Output

3) os dados são transferidos pelo porto A, sendo o “handshaking” feito mais uma vez pelo porto C. Para este projecto foi considerado o funcionamento somente no Modo 0, dado este interface servir só para ligar e desligar actuadores, não sendo utilizado para nenhum tipo de comunicação avançada. As “Control Words”, assim como toda a documentação sobre estes dispositivos, encontram-se descritas em [9].

O *DIO* fornece ainda:

- Linhas de retorno digitais, essenciais para inter-actuar com qualquer circuito.
- Os sinais do temporizador do *ADC* e do *DAC*.

2.1.2 Bloco terminal de 50 posições

O *DIO* da placa de aquisição de dados apresenta um conector *IDC* de 50 pinos como meio de comunicação para o exterior. A ligação directa de fios individuais a este conector é uma tarefa difícil e pouco segura, dado que podem ocorrer curto-circuitos e más ligações, que podem afectar negativamente o funcionamento do sistema de actuação. Para resolver esta questão foi construído um bloco terminal de 50 posições, que faz a derivação do conector *IDC*, para terminais de fácil utilização.

Este dispositivo foi desenvolvido numa placa de circuito impresso, cujo “layout” pode ser visto na Figura 2.3.

2.1.3 “Drivers” *TTL*

O $\mu PD71055$ da placa de aquisição só pode inter-actuar com circuitos *TTL*, não podendo deste modo controlar os módulos de comutação opto-acoplados e o relé, usados neste tra-

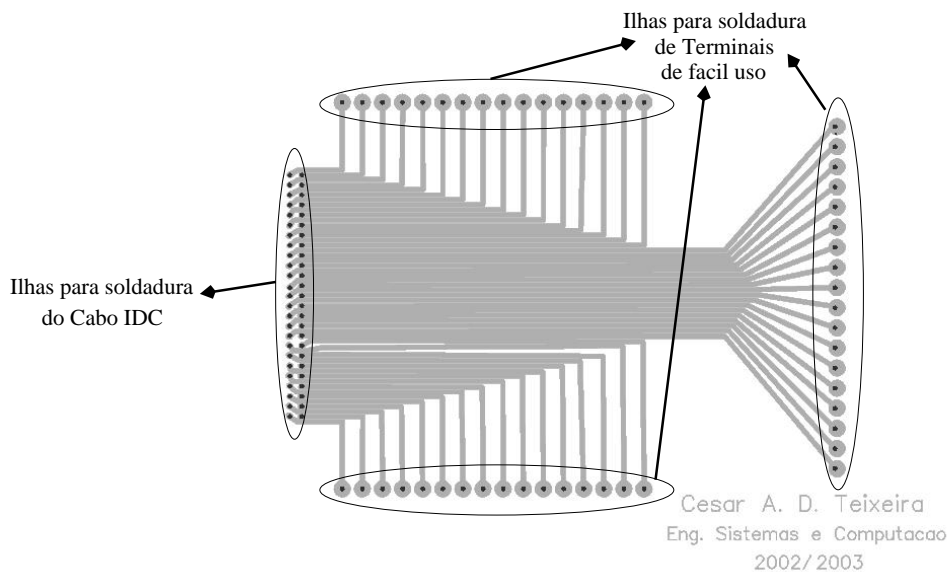


Figura 2.3: "Layout" do bloco terminal, em placa de circuito impresso.

balho. Para contornar este problema foi necessário o desenvolvimento de um circuito de adaptação.

Analisando os "datasheets" dos relés verificou-se que estes requeriam à entrada 10 mA e 15mA⁴ a 5V. Consultando a lista da família lógica em causa escolheu-se o integrado, cuja referência é *74LS06*. Este dispositivo engloba 6 "drivers" inversores, que conseguem comportar uma corrente de 40 mA no nível lógico inferior (saída em colector aberto), satisfazendo plenamente os requisitos. As especificações detalhadas deste circuito integrado podem ser consultadas em [10].

Este circuito foi projectado e concebido numa placa de circuito impresso, cujo "layout" pode

⁴Existem duas correntes de entrada, dado que existem dois tipos de relés, esta questão será explicada na subsecção (2.1.4)

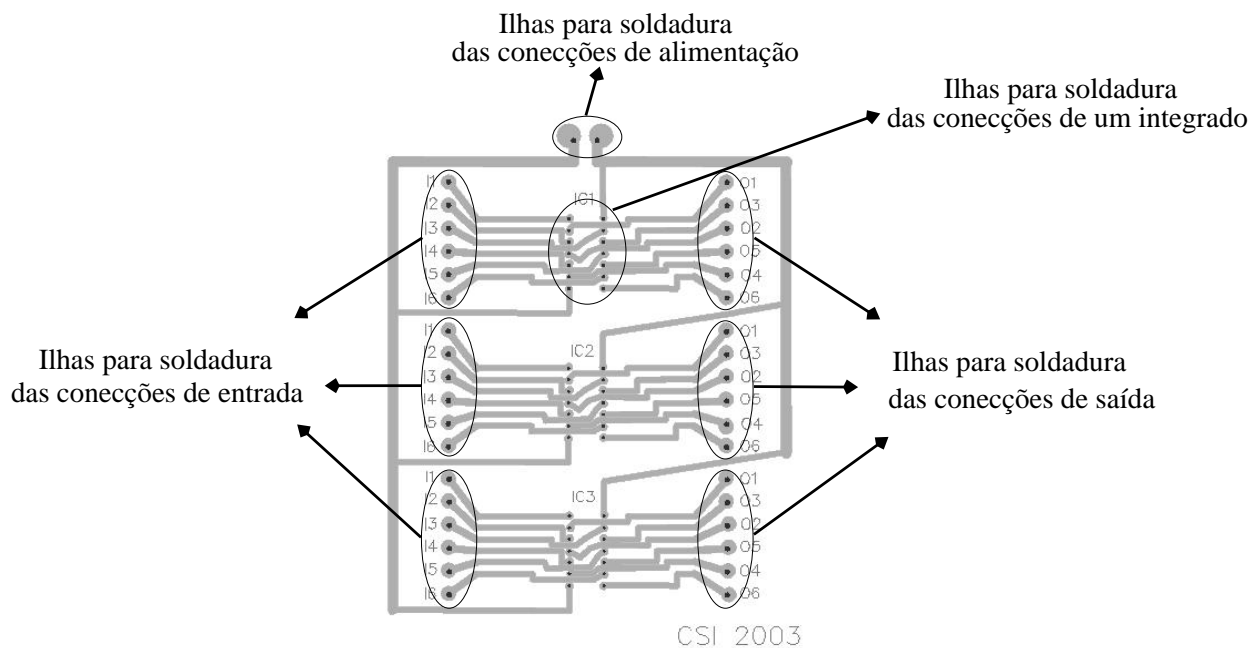


Figura 2.4: "Layout" do circuito para os drivers *TTL*, em placa de circuito impresso.

ser visto na Figura 2.4. Foi necessário a inclusão de 3 integrados *74LS06*, dado que o sistema pode conter até 17 elementos de comutação.

2.1.4 Elementos de comutação

Para a comutação dos actuadores foram comprados dois tipos de dispositivos: módulos opto-acoplados e relés de estado sólido.

Os módulos opto-acoplados são da *Crydom*, modelo *OAC5A* e são usados para actuar sobre as bombas de injeção de nutrientes, sendo deste modo usados 3 dispositivos deste tipo. Estes módulos apresentam uma corrente de entrada de 10 mA a 5V contínuos. A tensão de entrada pode variar entre 2.5V e 6V, dando-se a comutação aos 2V. À saída estes dispositivos podem comutar tensões nominais de 240V alternos, sendo a tensão máxima e mínima

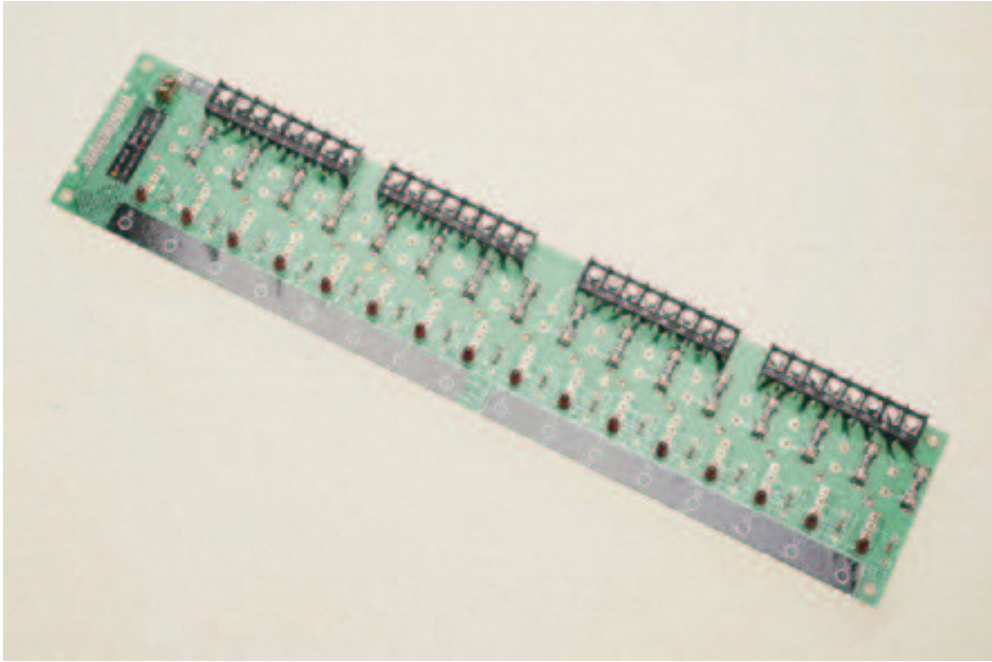


Figura 2.5: Placa de suporte para os módulos opto-acoplados.

admitidas de 280V e 24V alternos, respectivamente. A corrente máxima em estado “on” é de 3A, especificação que é suficiente para este projecto, como é referido na sub-secção 2.1.5. Os módulos aqui referidos são de saída (entrada DC - saída AC), mas também existem dispositivos deste tipo, que recebem à entrada uma tensão AC devolvendo uma tensão DC (módulos de entrada). Esta variante é útil para fazer a monitorização de linhas AC, sendo também usados neste projecto, como é explicado na sub-secção 2.1.8. A informação detalhada destes dispositivos pode ser consultada em [11].

Para fazer as conexões com os terminais destes dispositivos foi adquirida uma placa de suporte no distribuidor *RS-AMIDATA*, cuja referência é *RS632-118* (Fig. 2.5).

Esta placa possui 16 “sockets” onde os módulos se encaixam, sendo fixados através de um parafuso incorporados nos mesmos. Para este projecto são usados 4 posições, sendo as

restantes reservadas para uso futuro. Cada posição possui um *LED*⁵ que indica quando um módulo de saída está activado ou quando um módulo de entrada tem uma tensão alterna na sua entrada. Para protecção de cada dispositivo, a placa de suporte possui fusíveis de 3.15A. As conexões digitais (entradas dos módulos) com os “drivers” são feitas através de um conector *DIN*⁶ 41612 de 32 pinos. As saídas/entradas de potência são feitas através de terminais dotados de parafusos.

Além dos módulos anteriormente referidos, foi ainda necessário a aquisição de um relé de estado sólido, usado para actuar sobre uma resistência de aquecimento da solução nutriente (subsecção 2.1.5). A resistência requer uma corrente de aproximadamente 9A a 220V alternos, especificação que os módulos opto-acoplados não conseguiam cumprir.

O relé adquirido é do fabricante *Crydom*, modelo *PF240D25*. Este relé requer na entrada de controlo uma tensão entre 3V e 15V contínuos, onde a corrente à tensão nominal de 5V é de 15mA. Esta especificação é cumprida através dos drivers *TTL*, citados anteriormente. À saída este dispositivo consegue comutar tensões alternas entre 12V e 280V, sendo a corrente máxima admitida de 10A ou 25A (valores RMS), dependendo das condições de refrigeração. Para um melhor esclarecimento sobre o dispositivo aconselha-se a consulta da informação disponível em [12].

O suporte para este relé, ao contrário dos módulos opto-acoplados, foi construído para o efeito, numa placa de circuito impresso. Isto devido ao facto do referido relé não se encaixar na placa da *RS-AMIDATA* e da mesma não suportar correntes tão altas. O “layout” do suporte foi copiado da placa *RS* e encontra-se representado na Figura 2.6. Para permitir que o

⁵Light Emiting Diode

⁶Deutsches Institut für Normung

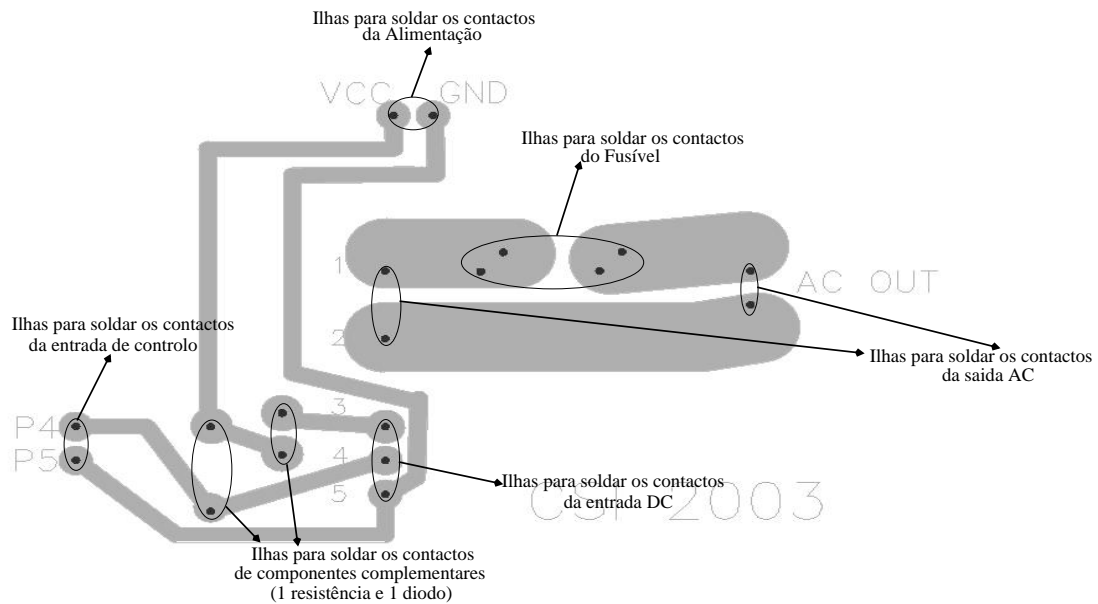


Figura 2.6: "Layout" do circuito para o relé de estado sólido, em placa de circuito impresso.

relé funcione em boas condições de refrigeração, permitindo assim a comutação de correntes mais elevadas, foi inserido o suporte numa caixa de uma fonte de alimentação avariada, onde a ventoinha proporciona uma circulação de ar constante, refrigerando o relé.

2.1.5 Actuadores

As bombas de injeção de nutrientes já se encontravam no sistema antigo, sendo assim partilhadas pelo sistema antigo e pelo novo sistema. A selecção de controlo é feita através de um comutador manual.

Estas bombas pneumáticas são fabricadas pela *RENA* e requerem para funcionamento 6 Watts, sendo este requisito completamente satisfeito pelos módulos opto-acoplados.

A resistência de aquecimento requer uma potência de 2000 Watts, sendo para tal necessário

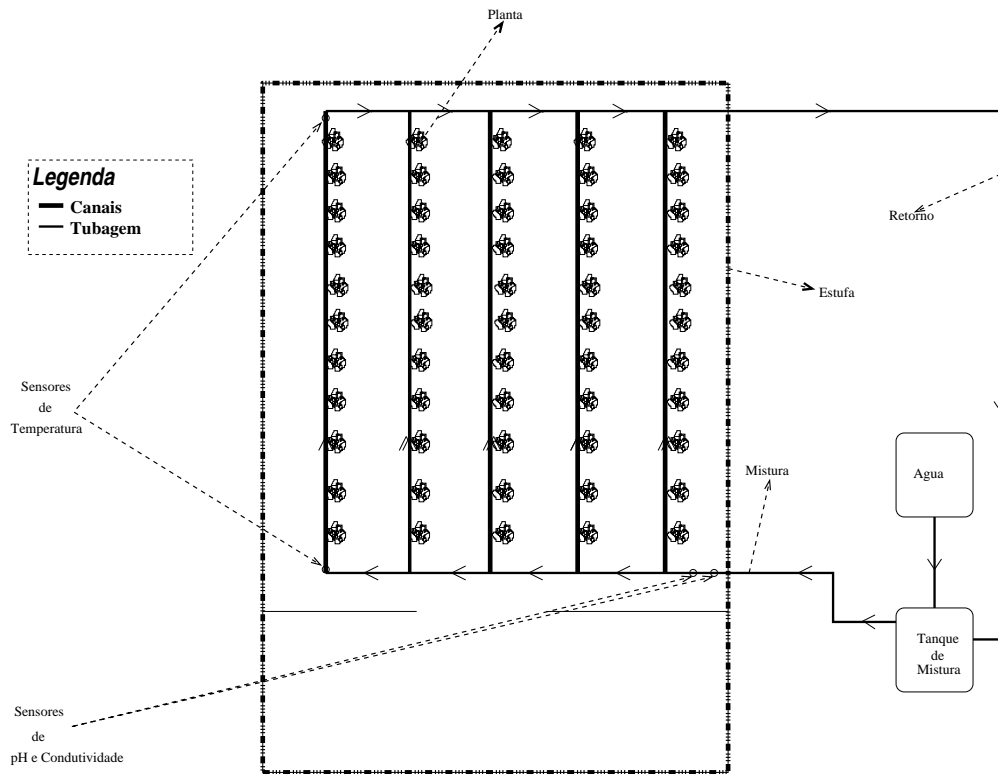


Figura 2.7: Disposição dos sensores no sistema de rega.

um elemento de comutação especial, como já foi referido na sub-secção 2.1.4. Este elemento de aquecimento possui um termostato de protecção, que consegue comutar correntes até 16 A, numa gama de temperaturas entre $0C^o$ e $40C^o$.

2.1.6 Sensores

Para este projecto foram usados sensores de 3 tipos: temperatura, pH e condutividade. A futura disposição dos sensores no sistema pode ser vista na Figura 2.7. Estes dispositivos foram adquiridos juntamente com o “data-logger” (sub-secção 2.1.7) e são fabricados pelo fabricante *Envirodata Australia Pty Ltd.*. Este sensores fornecem à saída uma onda quadrada, onde a grandeza física medida é proporcional à frequência da onda.

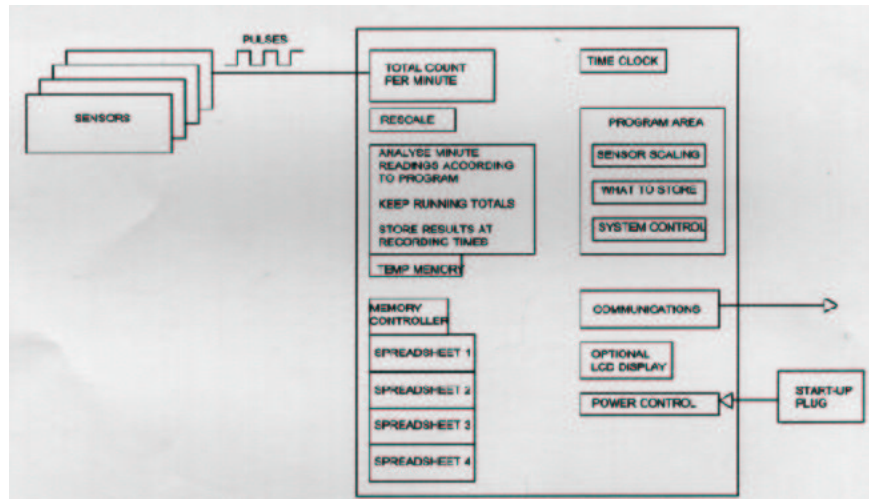


Figura 2.8: Diagrama de blocos do "data-logger".

2.1.7 "Data-logger"

Para a aquisição de dados dos sensores foi usado um "data-logger" *Easidata MK-4 16CH* do mesmo fabricante dos sensores. O diagrama de blocos deste aparelho é mostrado na Figura 2.8.

Este "data-logger" possui 16 canais e acumula o sinal de cada sensor durante um período de um minuto, calculando um valor médio de cada grandeza medida. Estes valores são usados para extrair valores médios, máximos, mínimos e totais durante longos períodos de tempo. A frequência e o tipo de dados a armazenar são determinados pelo "software" a correr na "PROGRAM AREA" (Fig. 2.8). Após a fase de aquisição e tratamento, os dados são armazenados numa das 4 memórias, sendo posteriormente usado o sistema de comunicação para fazer a transferência os dados para um PC ou para qualquer dispositivo que possua um interface série [13].

2.1.8 Aquisição do “sinal de rega”

O sistema de controlo da solução nutriente só deve actuar quando a etapa de rega⁷ se inicia. Para se determinar o tempo de acção do controlador foi necessário fazer uma monitorização do “sinal de rega”. Este sinal é gerado no sistema antigo, através de um sensor de fluxo. Este sensor indica a passagem e a ausência de fluxo através da passagem ou não de corrente alterna (220V). Para a aquisição deste sinal para o PC foi usado um módulo opto-acoplado da *Crydom*, modelo *IAC5A*. Este módulo já tinha sido referido na sub-secção 2.1.4 como “módulos de entrada”. Estes apresentam à saída uma diferença de potencial dependente da alimentação (neste caso 5V) quando não existe uma tensão alterna na entrada e uma diferença de potencial nula, quando a mesma tensão existe. A ligação deste com o computador é feita através de um dos pinos destinados para leitura na porta paralela do PC. Para este fim foi desenvolvido um novo driver para a referida porta, que será exposto na sub-secção 2.2.2. A razão para não se usar o “data-logger” para a aquisição deste sinal, advém do facto do mesmo ser de alarme e necessitar de uma monitorização permanente.

2.1.9 Porta paralela

Um computador pessoal possui uma porta paralela que normalmente é usada para comunicar com periféricos, como é o caso da impressora. Esta porta pode no entanto ser usada para outros fins, como é o caso de aplicações de controlo.

O integrado usado pela porta paralela é do mesmo tipo do usado no DIO da placa de aquisição

⁷Esta etapa é ditada por um temporizador, que comanda também a bomba de injeção de água no tanque e a bomba que injecta a solução nutriente nos canais.

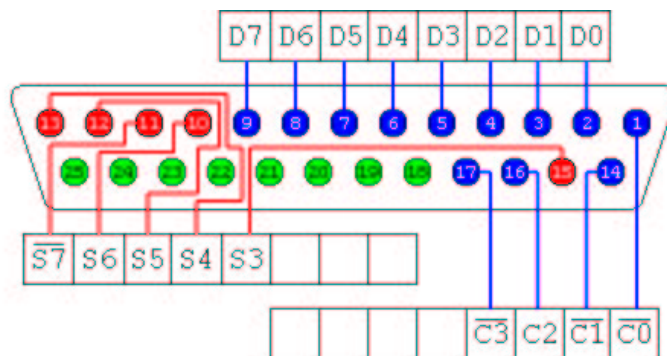


Figura 2.9: Diagrama explicativo da porta paralela de um PC.

de dados, anteriormente referida. O diagrama explicativo da porta paralela encontra-se apresentada na Figura 2.9.

O acesso a esta porta é feito através de 3 portos consecutivos de 8 bits. O primeiro é denominado de porto de dados e encontra-se normalmente no endereço 0x378 [14] [15], correspondendo cada bit aos pinos de 2 a 9. Este porto é somente de saída de dados, não sendo permitidas operações de leitura. O segundo porto encontra-se no endereço 0x379 e é denominado de porto de estado. Este permite ler dados do pino 10 ao 13 e também do 15. A saída de dados ainda pode ser feita por mais 4 pinos (1, 14, 16 e 17), escrevendo para o porto de controlo, que se encontra no endereço 0x37a. O bit 4 deste porto é denominado de “IRQ ENABLE” e permite activar ou não o accionamento de uma interrupção para esta porta. Normalmente a interrupção utilizada é a número 7, sendo esta accionada quando o nível lógico passa de 1 para 0 no pino 10. Os restantes bits deste registo encontram-se reservados ou permitem controlar o restante funcionamento da porta paralela. Por último os pinos de 18 a 25 são de retorno (“terra”).

O modo de funcionamento anteriormente explicado refere-se ao funcionamento “standard”

da porta paralela, denominado de SPP⁸. Neste modo determinados pinos encontram-se reservados somente para entrada ou somente para saída, não sendo permitida a comunicação bi-direccional pelo mesmo pino. Existem outros modos de funcionamento (EPP⁹, ECP¹⁰), que permitem uma comunicação mais avançada [16]. Estes saem fora do âmbito deste trabalho, não sendo desta maneira explicados no relatório.

2.1.10 Fonte de alimentação

Para alimentar toda a electrónica anteriormente citada, foi adquirida uma fonte de alimentação estável. Esta fonte foi adquirida, mais uma vez, à *RS-AMIDATA* e consegue fornecer 8A a 5V contínuos.

2.1.11 Alteração do sistema antigo

Para permitir o funcionamento do sistema antigo e do novo sistema, foi adicionado um comutador manual ao painel. Este comutador permite comutar o controlo para o sistema analógico ou para o sistema controlado pelo PC. A Figura 2.10 mostra a integração do comutador manual no painel. O sistema antigo também possui sensores de pH e condutividade, mas devido ao facto de estes se encontrarem degradados e da documentação dos controladores (que fazem interface com os sensores) não se encontrar disponível, optou-se por comprar novos sensores que serão ligados directamente ao “data-logger”.

⁸Standard Parallel Port

⁹Enhanced Parallel Port

¹⁰Extended Capabilities Port

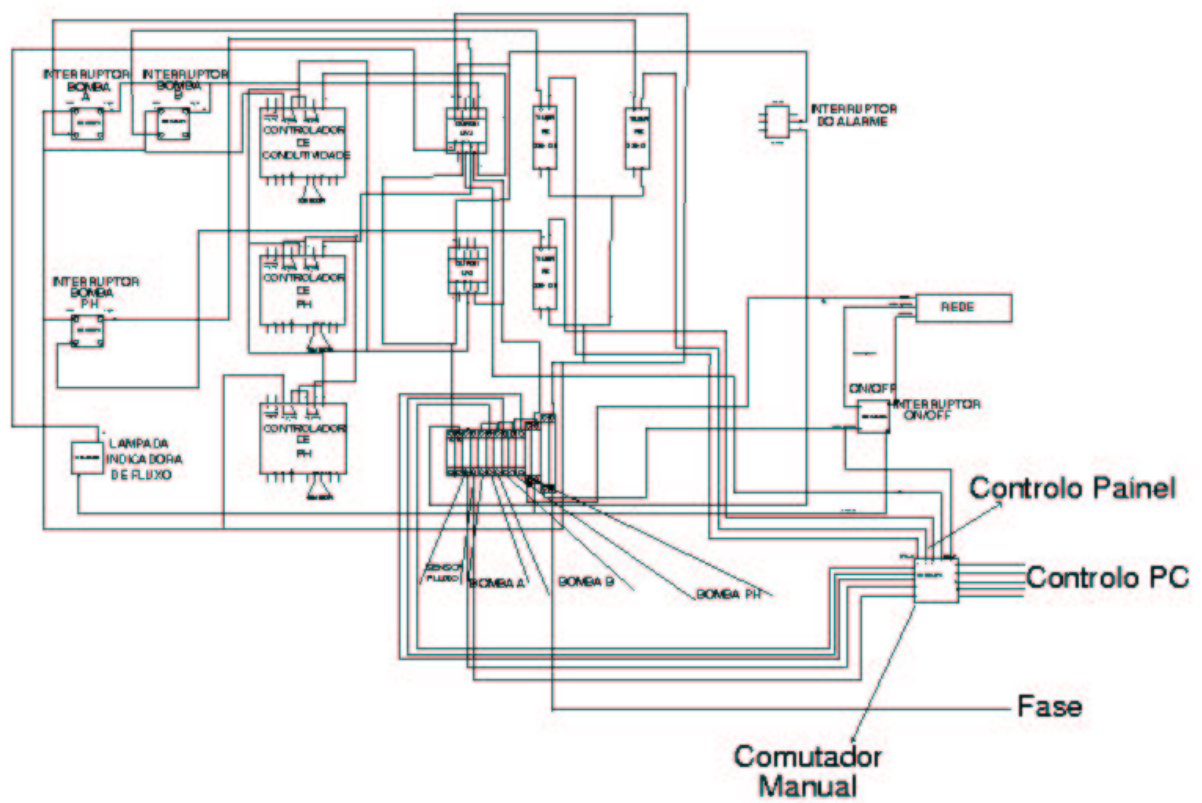


Figura 2.10: Diagrama do painel incluindo o comutador manual

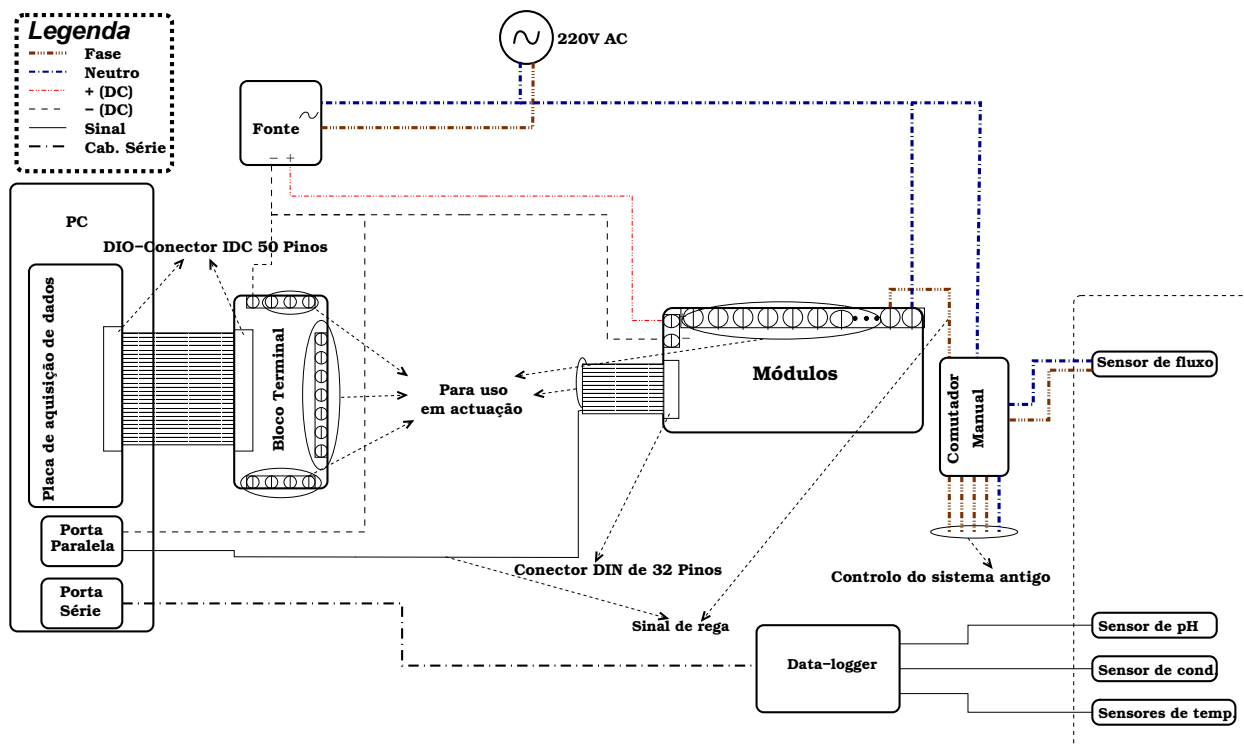


Figura 2.11: Interligação dos diversos dispositivos de aquisição.

2.1.12 Interligação dos diversos dispositivos

Nesta subsecção é mostrada uma descrição gráfica (Figuras 2.11 e 2.12) geral da interligação dos diversos dispositivos, referidos anteriormente. Esta descrição tem em vista o esclarecimento de alguma dúvida que possa ter surgido das explicações anteriores. Para esta descrição gráfica ser inteligível, optou-se por separar o sistema em actuação e aquisição. Obviamente que na implementação prática estes dispositivos encontram-se todos ligados em conjunto.

2.2 Implementações de “software”

Esta secção descreve o “software” realizado neste projecto de fim de curso. Para o desenvolvimento do mesmo foram utilizadas as linguagens *Python* e *C*. A linguagem *C* foi utilizada

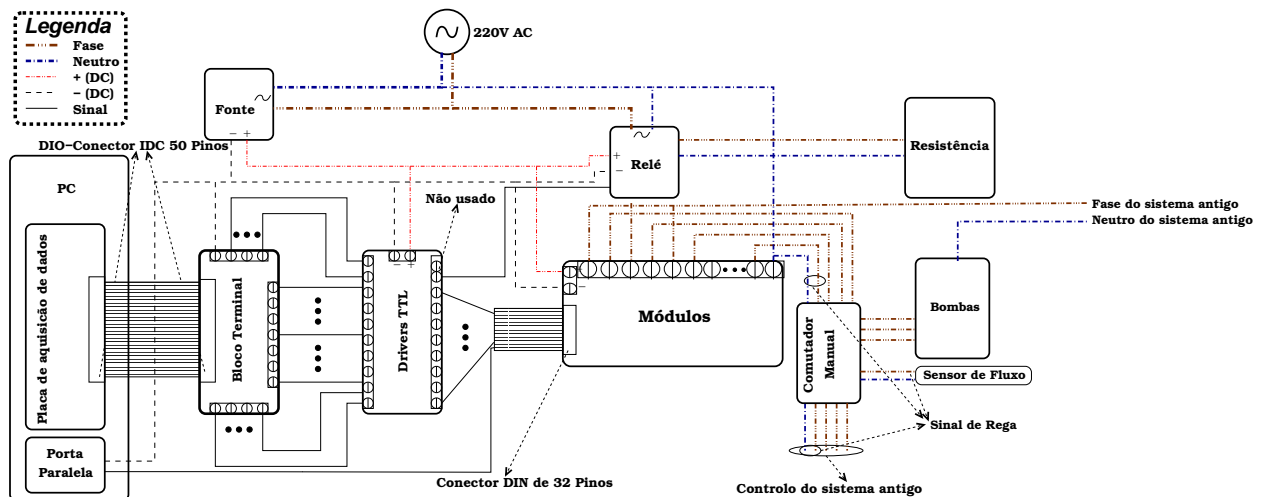


Figura 2.12: Interligação dos diversos dispositivos de actuação.

para o desenvolvimento dos “device-drivers”, ou seja, para comunicação de baixo nível com a placa de aquisição de dados e com a porta paralela. Esta escolha foi realizada pelo facto desta linguagem possuir bibliotecas que permitem o acesso directo ao dispositivo, mantendo a sua rapidez característica. Para o desenvolvimento de “software” de alto nível foi escolhido o *Python*.

2.2.1 Linguagem de programação *Python*

O *Python* é uma linguagem interpretada de “scripting” de domínio público, sendo simultaneamente uma linguagem de alto nível (Very High Level Language (VHLL)), e uma linguagem dinâmica orientada a objectos (Object Oriented Dynamic Language (OODL)) [17]. Pode ser comparada ao *TCL*, *Perl*, *Scheme* ou *Java*. Esta linguagem possui módulos, classes, excepções, tipos de dados de alto nível e tipagem dinâmica. A sua grande portabilidade é conseguida pelos interfaces que possui com muitas “system calls” e bibliotecas e pela

propriedade de ser extensível através da adição de novos módulos implementados numa linguagem compilada como por exemplo o *C* ou o *C++*. A apresentação de informação gráfica é facilmente conseguida por interfaces com diversos sistemas de janelas.

Um número extenso de módulos já desenvolvidos permitem: o tratamento simplificado de “arrays” (*Numeric.py*), cálculo de “FFTs” e operações de álgebra linear, através do interface com as bibliotecas *FFTPACK* e *LAPACK* do *Fortran* (*fft.py* e *LinearAlgebra.py*).

Todas as propriedades anteriormente referidas, e muitas mais, fazem desta linguagem uma ferramenta óptima para um desenvolvimento rápido e eficiente de aplicações em engenharia.

2.2.2 “Device drivers”

Introdução

Um “device-driver” pode ser visto como uma camada de software entre um dispositivo¹¹ e o Sistema Operativo (SO). A função desta camada é ocultar completamente o funcionamento do dispositivo, permitindo que este seja comandado única e exclusivamente por primitivas standard do Sistema Operativo (SO). As primitivas são independentes do driver, sendo mapeadas para um dispositivo específico através do mesmo. Por exemplo a primitiva *open* pode ser usada para abrir um ficheiro de texto, como pode ser usada para activar uma placa de aquisição de dados. O programador de “drivers” deve ter como preocupação principal o desenvolvimento de produtos livres de políticas. Um “driver” deve realizar somente funções básicas de acesso ao dispositivo, deixando as políticas e condicionalismos para as camadas superiores de “software” (no espaço do utilizador). A implementação de políticas ao nível

¹¹Um dispositivo pode ser uma porção de hardware, ou mesmo de software

do “driver” é arriscado, dado que o driver é realizado ao nível do *kernel* e uma boa política para um utilizador pode ser uma má política para outro [19].

Em Unix são distinguidas três classe de dispositivos, sendo estas:

- “Character devices”.

Podem ser acedidos como uma cadeia de caracteres (como um ficheiro, na directoria */dev*). Os “drivers” para este tipo de dispositivos são chamados de “char drivers” e normalmente implementam as “System Calls”: “open”, “close”, “write” e “read”.

- “Block devices”.

Como os “character devices”, são acedidos por nós do sistema de ficheiros, na directoria */dev*. Este tipo de dispositivos diferem dos “character devices” na gestão interna feita pelo kernel, sendo somente acedidos em múltiplos de um bloco (normalmente um bloco é um kilobyte ou outra potência de dois). Um dispositivo deste tipo pode alojar um sistema de ficheiros, fazendo parte desta classe, por exemplo: os discos rígidos, drives de cdroms e drives de disketes.

- “Network interfaces”.

Permitem a transacção de dados entre computadores. Estes dispositivos não são orientados para acessos através de cadeias de caracteres, não sendo deste modo facilmente mapeados por um nó do sistema de ficheiros. Estes interfaces são nomeados pelo SO como “ethx”, onde “x” é o número associado ao interface, sendo as “system calls” “read” e “write” substituídas por “system calls” relacionadas com a transferência de pacotes.

O SO *Linux* considera outros tipos de dispositivos, de menor importância, não sendo deste modo referidos neste trabalho.

“Driver” para a placa *ADC44D*

Como a placa de aquisição de dados escolhida para actuação não vinha acompanhada de “driver” para *Linux* (muito poucas placas de aquisição de dados trazem driver para *Linux*), foi necessário o desenvolvimento do mesmo.

A placa *ADC44D* pode ser encarada como um “character device”, sendo deste modo desenvolvido um “char driver” para a mesma.

Dado a explicação do funcionamento do “driver” ao nível do kernel ser extensa, neste relatório optou-se por fazer uma pequena explicação geral, de como este actua como intermediário entre o SO e a placa *ADC44D*. Esta explicação vai ser feita, através da análise da Figura 2.13.

Foram criados três nós no sistema de ficheiros (*/dev/adc44d-dac*, */dev/adc44d-adc* e */dev/adc44d-dio*). Estes foram criados usando a “system call” **mknod**, da seguinte forma: **mknod /dev/adc44d-* c 60 minor**. A letra (c) indica que o nó criado está relacionado com um *char driver*. O número 60 refere-se ao *major number*. Este número identifica o driver associado a um dispositivo ou conjunto de dispositivo. É usado pelo kernel no momento da abertura para a atribuição de um dispositivo ou conjunto de dispositivos, ao respectivo driver. Para este driver foi escolhido o *major* 60 dado ser um dos números disponíveis para uso local (não registado)¹². A variável **minor** representa um número, denominado de *minor*

¹²Os números nos intervalos [60, 63], [120, 127] e [240, 254] podem, da mesma forma, ser usados para uso

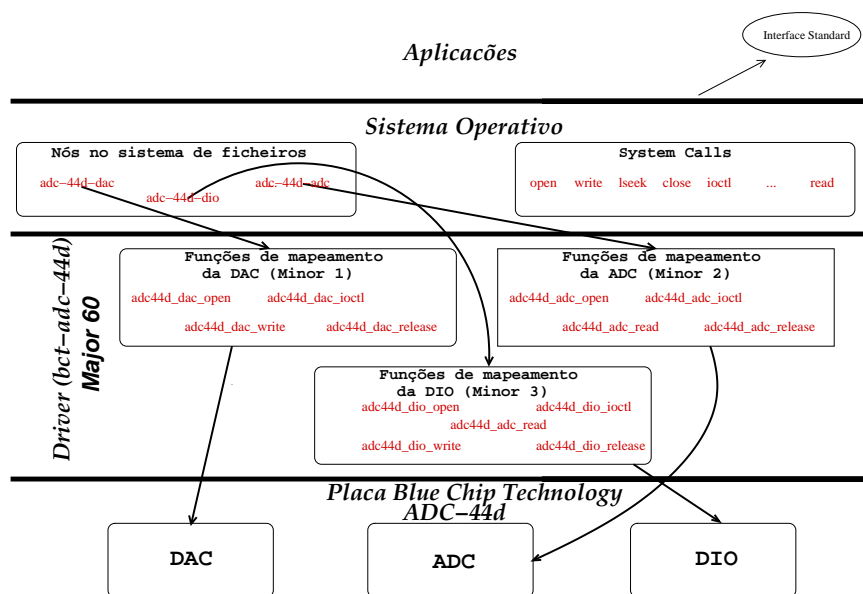


Figura 2.13: Diagrama explicativo do enquadramento do *device-driver* para a placa ADC44D *number*. Este é usado somente pelo driver, identificado pelo *major number*. O driver diferencia os vários dispositivos sob seu controlo, pelo seu *minor number*. No nosso caso usou-se a seguinte convenção: DAC-*minor 1*, ADC-*minor 2* e DIO-*minor 3*.

O sistema operativo, ao executar a *system call open*, identifica qual o driver correspondente a um dispositivo ou conjunto de dispositivos, através do *major* associado ao nó do sistema de ficheiros que se pretende abrir, atribuindo de seguida o comando do dispositivo ao respectivo “driver”. Da mesma forma, o “driver”, identifica qual o dispositivo específico, que se pretende abrir, através do *minor*. Após estas identificações, é executada a função de *open* (`adc44d.dac.open`, `adc44d.adc.open` e `adc44d.dio.open`), que executa na placa os procedimentos de abertura específicos do dispositivo em causa. Estas funções diferem de driver para driver, mapeando a acção da “system call open”, para um dispositivo específico.

No fim da operação de abertura, a “system call open” devolve um inteiro (“file descriptor”),
 local.

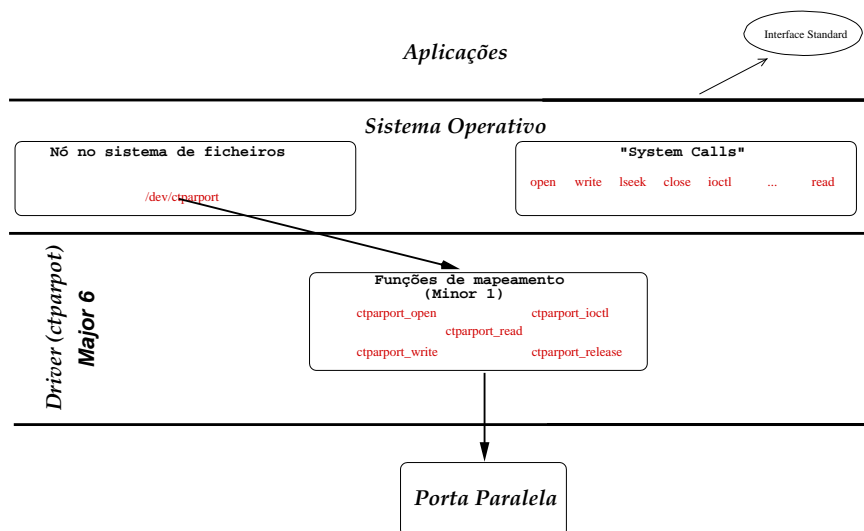


Figura 2.14: Diagrama explicativo do enquadramento do *device-driver* para a porta paralela que identifica um dispositivo aberto. Através deste inteiro, pode-se executar qualquer *system call* que possua uma função de mapeamento definida no driver.

“Driver” para a porta paralela do PC

O driver desenvolvido para a porta paralela usa a mesma arquitectura que o driver desenvolvido para a placa de aquisição de dados, sendo neste caso usado o “major number” 6 (número reservado para a porta paralela). O dispositivo pode ser acedido através do nó `/dev/ctparport`. O uso de primitivas do SO sobre este nó permite: a escrita de dados através do porto de dados (primitiva *write*), a leitura de dados através do porto de estado (primitiva *read*) e o accionamento da interrupção através da monitorização do pino 10. A função de tratamento da interrupção (Interrupt Handler) apenas permite o desbloqueio da operação de leitura, quando a interrupção surge.

O diagrama explicativo deste driver encontra-se exposto na Figura 2.14.

2.2.3 Código de alto nível

Como foi dito em 2.2.2, um “device driver” deve ser o mais simples possível, deixando as políticas para camadas superiores de “software”. Para este projecto foi desenvolvido código de alto nível que implementa as referidas políticas ou condicionalismos, e que por sua vez é utilizado como intermediário entre os programas do utilizador e o “driver”. Este código foi desenvolvido em *Python* e a sua função resume-se a:

1. Abrir nós do sistema de ficheiros que tenham correspondência com os “drivers”, anteriormente referidos. Por exemplo: `/dev/adc44d-dac`, `/dev/ctparport`, etc.
2. Executar as primitivas (open, close, ioctl, read e write) do SO que possuam uma função de mapeamento nos “drivers”.

2.2.4 Interface gráfico

A chave para uma fácil adopção de um sistema de controlo digital frequentemente reside no seu emprego como apresentador de informação. É fundamental que o operador do processo tenha um quadro claro do estado do processo, e que seja fácil a sua interacção com o processo. [18]

Para este projecto foi desenvolvido em *Python Gtk* um interface gráfico (Fig. 2.15) que permite a apresentação e a alteração dos valores das referências (pH, Condutividade e Temperatura) e dos valores de alguns dos parâmetros de controlo (intervalo de cálculo e actuação). Os valores dos sensores são apresentados em caixas de texto, sendo também possível a sua apresentação de uma forma gráfica. Por último este interface apresenta o estado dos actu-



Figura 2.15: Janela principal do Interface gráfico.



Figura 2.16: Apresentação gráfica do estado dos actuadores.

adores (bombas e resistência) e do sensor de fluxo, como se mostra nas Figura 2.15 e 2.16.

Os valores das referências e dos parâmetros de controlo, apresentados neste interface, são alterados numa base de dados e lidos da mesma. Estes serão usados futuramente para o desenvolvimento do algoritmo de controlo. A mesma base de dados armazena os valores dos sensores e dos erros. Os valores dos sensores são introduzidos através de um programa, que por sua vez lê a memória do “data-logger”. O estado do sensor de fluxo é lido da

porta paralela do computador, que por sua vez está ligada ao referido sensor, como se mostra em 2.1.12. Por último o estado dos actuadores é lido a partir da placa aquisição de dados.

2.2.5 Interligação dos diversos componentes de “software”

Esta sub-secção tem como objectivo expor de uma forma gráfica (Fig. 2.17), a interligação final dos diversos componentes de “software”.

Alguns dos componentes expostos na Figura 2.17 ainda não se encontram desenvolvidos, ou estão parcialmente desenvolvidos. Estes componentes são o algoritmo de controlo e a base de dados. O algoritmo de controlo ainda não se encontra desenvolvido pelo facto de ter havido atrasos na entrega do equipamento utilizado na elaboração do sistema de actuação e aquisição e pelo facto dos canais de nutrientes não se encontrarem instalados na referida estufa, até à data da elaboração deste relatório. A base de dados encontra-se em parte desenvolvida bastando adicionar apenas algumas tabelas.

Para o teste do interface gráfico foi usada uma base de dados com informação fictícia, e simulações em laboratório do estado dos actuadores e sensor de fluxo.

O programa de leitura do “Data-logger” foi desenvolvido em projectos anteriores, sendo futuramente usado neste trabalho.

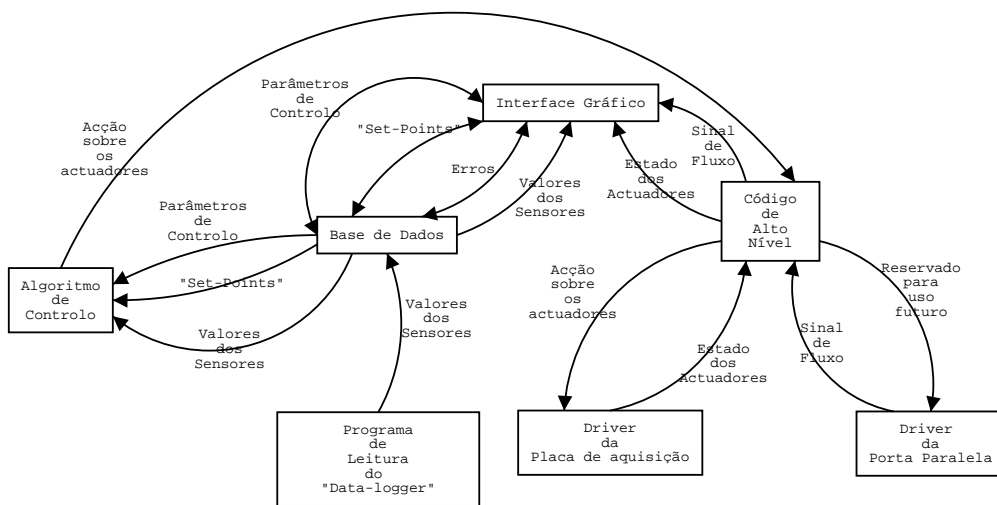


Figura 2.17: Diagrama explicativo da interligação dos diversos componentes de “software”.

Capítulo 3

Resultados

Até ao momento da escrita deste relatório encontram-se completamente desenvolvidos o sistema de actuação e aquisição. Estes sistemas englobam toda a electrónica referida anteriormente, bem como todo o “software” de suporte. Por falta de tempo estes sistemas não foram testados na estufa referida na introdução. O teste realizou-se em laboratório e numa estufa experimental que se encontra no Centro de Ciência Viva do Algarve (CCVA).

Os testes realizados na estufa do CCVA permitiram avaliar o sistema a funcionar num ambiente real, servindo deste modo para limar algumas imperfeições que não foram detectadas em laboratório, das quais se salientam alguma fragilidade na presença de transitórios na rede pública e a dificuldade de comutação dos módulos opto-acoplados, em algumas situações. Para resolver o problema da fragilidade face aos transitório, substituiu-se a fonte referida em 2.1.10 pela fonte do PC. Esta substituição foi possível, dado as tensões e correntes fornecidas por esta serem suficientes para alimentar a electrónica em questão. Estas fontes possuem um melhor sistema de filtragem que a fonte comprada. A dificuldade de comutação dos módulos

advém do facto de estes terem que descarregar algumas capacidade internas, para se dar a comutação. Quando a resistência da carga aos bornos é muito elevada esta descarga é lenta, necessitando para tal de uma carga adicional. Foram então introduzidas resistências, na ordem das dezenas de $k\Omega$, nos módulos que apresentavam dificuldades de comutação, solucionando assim o problema.

Após a resolução das imperfeições anteriormente referidas o sistema funcionou durante quinze dias sem falhas.

Capítulo 4

Conclusão e trabalho futuro

Como já foi salientado anteriormente um dos objectivos deste projecto passava pela modelação dos canais de nutrientes de um sistema NFT, e pela implementação de um controlador para a solução nutriente nos canais. Tais objectivos não foram possíveis de atingir, por razões referidas em 2.2.5. Pode-se dizer que neste projecto foi desenvolvido todo o “hardware” e “software”, ou seja, todo o sistema de actuação e aquisição. Este sistema forma a base que irá permitir a integração de algoritmos de controlo da solução nutriente.

Pode-se concluir a partir dos testes referidos em 3, que o sistema de actuação e aquisição funcionam na prática, demonstrando a validade do trabalho.

Para trabalho futuro, após à instalação dos canais de nutrientes na estufa da Universidade, segue-se a implementação e teste do equipamento. Seguindo-se finalmente a modelação dos canais e o desenvolvimento de um algoritmo de controlo.

Anexo A

bct-adc-44d

Manual do utilizador

César A. D. Teixeira & Pedro Frasão

Centro de Sistemas Inteligentes

Faculdade de Ciências e Tecnologia

Universidade do Algarve

Campus de Gambelas

8000 Faro

Resumo

O módulo para o *Linux Kernel* **bct-adc-44d.o** permite o acesso à placa de aquisição de dados **Blue Chip Technology ADC-44D**. São permitidos o acesso e controlo do: DAC,

ADC e DIO. O controlo dos “timers” não é permitido até ao momento.

A.1 Introdução

O módulo **bct-adc-44d.o** é um “character device driver” que permite o suporte da placa de aquisição de dados **Blue Chip Technology ADC-44D**, por parte do *Linux Kernel*. A documentação desta placa pode ser consultada em [7].

As conversões analógicas e digitais são feitas sobre controlo I/O, não sendo até ao momento suportado o controlo através de DMA ou interrupções.

A.2 Instalação do “driver”

A.2.1 Pré-requisitos

Para instalar o driver é necessário:

- uma placa de aquisição de dados **Blue Chip Technology ADC44D** para o barramento **ISA**
- o ficheiro **bct-adc-44d.tar.gz**;
- um sistema *Linux/Intel* com um *kernel 2.4.x* ;
- suporte para módulos no *Kernel*, os “header files” do *Kernel*, o *gcc* e o utilitário *insmod* e seus relacionados;

A.2.2 Instalação passo a passo

Para uma instalação fácil é aconselhável seguir os seguintes passos:

- descompactar o ficheiro:

```
tar -xzvf bct-adc-44d.tar.gz
```

- entrar no directório **bct-adc-44d** e executar **make clean** seguido de **make**

- introduzir o módulo **bct-adc-44d.o** no *Kernel*:

```
insmod bct-adc-44d.o [adc44d_base_port=porto base]
```

O parâmetro “adc44d_base_port” define o porto base. Se este parâmetro for omitido o porto base por defeito é **0x300**.

A.2.3 Remoção do módulo

Para remover o módulo basta executar:

```
rmmod bct-adc-44d
```

A.3 Usando o “driver”

Após a inserção do módulo no *Kernel* é necessário criar nós no sistema de ficheiros. Esta criação é feita, pelo utilizador **root**, através do comando **mknod** da seguinte forma:

```
mknod /dev/<nome> c <major> <minor>
```

O **nome** pode ser qualquer um, mas sugere-se que seja usado os nomes descritos na Tabela A.1.

O **major** escolhido para o driver é o 60, ou seja, um dos números disponíveis para uso local (não registado)¹.

Os **minors** associados a cada dispositivo podem também ser vistos na Tabela A.1. Resu-

Tabela A.1: Numeração dos dispositivos e convenção de nomes para o driver **bct-adc-44d**

| Dispositivo | minor | Nó no sistema de ficheiros |
|-------------|--------------|----------------------------|
| DAC | 1 | <i>/dev/adc44d - dac</i> |
| ADC | 2 | <i>/dev/adc44d - adc</i> |
| DIO | 3 | <i>/dev/adc44d - dio</i> |

mindo, a sequência de comandos para criar os diversos nós no sistema de ficheiros é:

```
mknod /dev/adc44d-dac c 60 1
```

```
mknod /dev/adc44d-adc c 60 2
```

```
mknod /dev/adc44d-dio c 60 3
```

Os nós criados só podem ser acedidos pelo utilizador root. Caso se queira dar privilégios a outros utilizadores tem que se utilizar o comando **chmod**.

A.4 Acesso dos programas do utilizador ao dispositivo

Os programas do utilizador comunicam através de simples operações de I/O, usando as primitivas do SO.

¹Os números nos intervalos [60, 63], [120, 127] e [240, 254] podem, da mesma forma, ser usados para uso local.

A.4.1 read

Esta primitiva pode ser usada sobre o nó correspondente ao DIO e ao ADC. O valor de **size_t count** deve ser 3 (bytes) no caso do DIO e 2 (bytes) no caso do ADC. Caso contrário a operação de leitura não tem efeito e é retornado 0 .

A.4.2 write

Esta primitiva pode ser usada sobre o DAC e DIO, sendo o valor do parâmetro **size_t count** 2 e 3 (bytes), respectivamente. Caso o valor deste parâmetro não seja o correcto, a operação de escrita não tem efeito sendo retornado **-EMSGSIZE**.

A.4.3 ioctl

Nota: Para usar a primitiva **ioctl** é necessário incluir a biblioteca **bct-adc-44d.h**.

Os **ioctl's** definidos e sua operação, encontram-se listados abaixo (um terceiro argumento é necessário caso seja indicado, este argumento é em qualquer caso do tipo **unsigned char ***).

- **ioctl's** relacionados com o DAC
 - **ADC44D_DAC_RST**: faz um “reset” ao DAC.
 - **ADC44D_DAC_TRANS_SMODE**: define o modo de transferência do DAC, servindo também para accionar a interrupção. Para isso é necessário um terceiro argumento, este argumento deve ser: **0**-Transferência através de simples operações de I/O, **0x01**-Accionamento da interrupção e **0x02**-Transferência através de DMA.

- **ADC44D_DAC_GCHAN**: faz com que a primitiva **ioctl** devolva o canal actual do DAC.
- **ADC44D_DAC_ECHAN**: selecciona quais os canais que podem ser usados, para tal deve ser passado como argumento um dos valores listados na Tabela A.2.

Tabela A.2: Valores possíveis do argumento para o **ioctl ADC44D_DAC_ECHAN**

| Valor | Canais activos |
|-------|----------------|
| 0x00 | Todos |
| 0x01 | 1, 2 e 3 |
| 0x02 | 0, 2 e 3 |
| 0x03 | 2 e 3 |
| 0x04 | 0, 1 e 3 |
| 0x05 | 1 e 3 |
| 0x06 | 0 e 3 |
| 0x07 | 3 |
| 0x08 | 0, 1, 2 |
| 0x09 | 1 e 2 |
| 0x0a | 0 e 2 |
| 0x0b | 2 |
| 0x0c | 0 e 1 |
| 0x0d | 1 |
| 0x0e | 0 |
| 0x0f | Nenhum |

- **ADC44D_DAC_SCHAN**: selecciona qual o canal a ser usado. Deve ser passado um dos valores apresentados na Tabela A.3.

Tabela A.3: Valores possíveis do argumento para o **ioctl ADC44D_DAC_SCHAN**

| Valor | Canal |
|-------|-------|
| 0x00 | 0 |
| 0x01 | 1 |
| 0x02 | 2 |
| 0x03 | 3 |

- **ADC44D_DAC_SREF_MODE**: activa ou desactiva o DAC. Deve ser passado como argumento adicional : **0x00**- DAC desactivada ou **0x04**- DAC activa.
- **ioctl**'s relacionados com o ADC
 - **ADC44D_ADC_RST** : faz um “reset” ao ADC.
 - **ADC44D_ADC_SMODE**: selecciona o modo de funcionamento para o ADC. Deve ser passado como terceiro argumento, um dos valores expostos na Tabela A.4.

Tabela A.4: Valores possíveis do argumento para o **ioctl ADC44D_ADC_SMODE**

| Valor | Modo | Descrição |
|-------|------|--|
| 0x00 | 0 | Funcionamento por operações individuais de I/O |
| 0x01 | 1 | Funcionamento por reactivação automática |
| 0x03 | 2 | Funcionamento em função do Timer |

- **ADC44D_ADC_SGAIN**: selecciona qual o ganho a ser usado nas leituras. O valor a passar deve ser um dos mostrados na Tabela (A.5).

Tabela A.5: Valores possíveis do argumento para o **ioctl ADC44D_ADC_SGAIN**

| Valor | Ganho |
|-------|-------------|
| 0x00 | 1 (Defeito) |
| 0x01 | 2 |
| 0x02 | 10 |
| 0x03 | 100 |

- **ADC44D_ADC_DTIMER**: desactiva o timer.
- **ADC44D_ADC_ETIMER**: activa o timer.
- **ADC44D_ADC_SING**: define os canais como não diferenciais.

- **ADC44D_ADC_DIFF**: define os canais como diferenciais.
 - **ADC44D_ADC_SIO**: define a transferência de informação através de operações de I/O.
 - **ADC44D_ADC_SDMA**: define a transferência de informação através de DMA (não funcional até ao momento).
 - **ADC44D_ADC_SCHAN**: selecciona um dos canais. Deve ser passado um número de **0** a **15** para o funcionamento não diferencial ou entre **0** e **7** para o funcionamento diferencial.
 - **ADC44D_ADC_GCHAN**: devolve o canal actual do ADC.
 - **ADC44D_ADC_SCHAN_MAN**: define o modo manual de selecção de canais.
 - **ADC44D_ADC_SCHAN_AUTO**: define o modo automático de selecção de canais.
 - **ADC44D_ADC_START**: desencadeia uma conversão no modo 0 ou uma sequência de conversões no modo 1.
 - **ADC44D_ADC_DT_READY**: devolve o estado da conversão (**0**-Completa ou **1**- Em progresso).
- **ioctl's** relacionados com o DIO
 - **ADC44D_DIO_SMOD0_CW**: define a “control word” no Modo 0. As “control words” definem se as diferentes portas (A, B, C low e C high) são de entrada ou saída. A configuração imposta por cada uma encontra-se listada na Tabela A.6.

Tabela A.6: Listagem e função das “control words” do DIO a funcionar no Modo 0.

| Control word | Porta | | | |
|--------------|---------|---------|---------|---------|
| | A | B | C high | C low |
| 0x80 | Saída | Saída | Saída | Saída |
| 0x81 | Saída | Saída | Saída | Entrada |
| 0x82 | Saída | Entrada | Saída | Saída |
| 0x83 | Saída | Entrada | Saída | Entrada |
| 0x88 | Saída | Saída | Entrada | Saída |
| 0x89 | Saída | Saída | Entrada | Entrada |
| 0x8A | Saída | Entrada | Entrada | Saída |
| 0x8B | Saída | Entrada | Entrada | Entrada |
| 0x90 | Entrada | Saída | Saída | Saída |
| 0x91 | Entrada | Saída | Saída | Entrada |
| 0x92 | Entrada | Entrada | Saída | Saída |
| 0x93 | Entrada | Entrada | Saída | Entrada |
| 0x98 | Entrada | Saída | Entrada | Saída |
| 0x99 | Entrada | Saída | Entrada | Entrada |
| 0x9A | Entrada | Entrada | Entrada | Saída |
| 0x9B | Entrada | Entrada | Entrada | Entrada |

- `ADC44D_DIO_GMOD0_CW`: devolve a “control word” que está activa.

A.4.4 open e close

Estas primitivas estão definidas para todos os dispositivos anteriormente referidos. A primitiva **open** devolve um descritor, que corresponde a um dispositivo aberto . Por exemplo para o DAC temos: `fd=open("/dev/adc44d-dac","w")`. Este descritor é usado pelas restantes primitivas para actuarem sobre o dispositivo.

A primitiva **close** fecha um dispositivo aberto, através do descritor correspondente (ex: `close(fd)`)

Anexo B

ctparport

Manual do utilizador

César A. D. Teixeira

Centro de Sistemas Inteligentes

Faculdade de Ciências e Tecnologia

Universidade do Algarve

Campus de Gambelas

8000 Faro

Resumo

O módulo para o *Linux Kernel* **ctparport.o** permite o acesso à porta paralela do PC. São permitidas operações de leitura, escrita e de controlo da interrupção.

B.1 Introdução

O módulo **ctparport** é um “character device driver” que permite o suporte da porta paralela do PC, por parte do *Kernel Linux*. A documentação desta porta pode ser consultada em [14] [15].

O driver suporta a escrita de dados através do porto de dados (porto base), leitura de dados através do porto de status (porto base +1) e o accionamento da interrupção através da monitorização do pino 10. O “Interrupt Handler” apenas permite o desbloqueio da operação de leitura quando a interrupção surge.

B.2 Instalação do “driver”

B.2.1 Pré-requisitos

Para instalar o driver é necessário:

- um PC com porta paralela;
- o ficheiro **ctparport.tar.gz**;
- um sistema *Linux/Intel* com um *kernel 2.4.x* ;

- suporte para módulos no *Kernel*, os “header files” do *Kernel*, o *gcc* e o utilitário *insmod* e seus relacionados;

B.2.2 Instalação passo a passo

Para uma instalação fácil é aconselhável seguir os seguintes passos:

- descompactar o ficheiro:

```
ctparport.tar.gz
```

- entrar no directório `ctparport` e executar **make clean** seguido de **make**
- introduzir o módulo **ctparport.o** no *Kernel*:

```
insmod ctparport.o [par_io_base=porto base][ctparport_irq=número da in-  
terrupção]
```

O parâmetro opcional “par_io_base” define o porto base. Se este parâmetro for omitido o porto base por defeito é **0x378**. Por sua vez o parâmetro opcional “ctparport_irq” define o irq a utilizar para a porta paralela. O número standard para esta porta é o 7, sendo deste modo aconselhável o seu uso. A omissão deste parâmetro na introdução do módulo no *Kernel* simplesmente desactiva o funcionamento através de interrupções.

B.2.3 Remoção do módulo

Para remover o módulo basta executar:

```
rmmod ctparport
```

B.3 Usando o “driver”

Após a inserção do módulo no *Kernel*, é necessário criar nós no sistema de ficheiros. Esta criação é feita, pelo utilizador **root**, através do comando **mknod** da seguinte forma:

```
mknod /dev/<nome> c <major> <minor>
```

O **nome** pode ser qualquer um, mas sugere-se que seja usado **/dev/ctparport**.

O **major** escolhido é o 6 dado ser o número reservado para esta porta.

Neste caso o “driver” só controla um dispositivo o que leva a que só exista um **minor**. Nesta situação foi escolhido o **minor** 1.

Por exemplo, o comandos para criar o nó no sistema de ficheiros é:

```
mknod /dev/ctparport c 6 1
```

O nó assim criado só pode ser acedido pelo utilizador **root**. Caso se queira dar privilégios a outros utilizadores tem que se utilizar o comando **chmod**.

B.4 Acesso dos programas do utilizador ao dispositivo

Os programas do utilizador comunicam através de simples operações de I/O, usando as primitivas do SO.

B.4.1 read

O uso desta primitiva faz com que seja lido um byte do porto de estado.

B.4.2 write

O uso desta primitiva faz com que seja escrito um byte para o porto de dados. O valor de `size_t count` deve ser 1 (byte). Caso contrário a operação não tem efeito sendo retornado `-EMSGSIZE`.

B.4.3 ioctl

Nota: Para usar a primitiva `ioctl` é necessário incluir a biblioteca `ctparport.h`.

Os `ioctl`'s definidos encontram-se listados abaixo (um terceiro argumento é necessário caso seja indicado, este argumento é em qualquer caso do tipo `unsigned char *`).

- `CTPARPORT_INT_ON`: acciona a interrupção para a porta paralela, caso tenha sido definido um IRQ no momento da inserção do módulo no *Kernel*.
- `CTPARPORT_INT_OFF`: desactiva a interrupção para a porta paralela.
- `CTPARPORT_GET_INT_BIT`: devolve o estado da interrupção (0- Desactivada ou 1-Activa).

B.4.4 open e close

A primitiva `open` devolve um descritor que corresponde a um dispositivo aberto . Para este dispositivo temos: `fd=open("/dev/ctparport","w+")`. Este descritor é usado pelas restantes primitivas para actuarem sobre o dispositivo.

A primitiva **close** fecha um dispositivo aberto, através do descritor correspondente (ex: **close(fd)**)

Anexo C

Código de Alto Nível para a placa ADC44D

Manual do utilizador

César A. D. Teixeira

Centro de Sistemas Inteligentes

Faculdade de Ciências e Tecnologia

Universidade do Algarve

Campus de Gambelas

8000 Faro

Resumo

Neste manual de utilizador é apresentado um conjunto de módulos *Python*, que fazem interface com o “driver” **bct-adc-44d**. Este conjunto de módulos permitem o controlo da placa

de aquisição de dados a partir de programas escritos em *Python*.

C.1 Introdução

O “Código de Alto Nível para a placa ADC44D” é composto por um conjunto de módulos. Existindo um módulo para cada dispositivo (DAC, ADC e DIO). Além destes existe ainda um módulo geral, que permite realizar um “reset” completo à placa. Estes módulos usam as primitivas do SO e implementam os condicionalismos e políticas de programação (no espaço do utilizador) que não são devem ser implementadas ao nível do driver (espaço do *Kernel*), facilitando o uso e controlo da placa **ADC44D** por parte de programas *Python*

C.2 Uso dos módulos

C.2.1 Pré-requisitos

Para usar os módulos é necessário:

- ter o módulo **bct-adc-44d.o** instalado;
- ter os nós: `/dev/adc44d-dac`, `/dev/adc44d-adc` e `/dev/adc44d-dio`; a “apontar” para o driver **bct-adc-44d** (em caso de dúvida num dos tópicos anteriores, é aconselhável consultar o manual do utilizador do driver, descrito no Anexo A).
- o ficheiro **adc44d.tar.gz**;
- um interpretador de *Python*.

C.2.2 Como usar os módulos

Para uma utilização fácil é aconselhável seguir os seguintes passos:

- descompactar o ficheiro:

```
adc44d.tar.gz
```

- Copiar os módulos presentes na directoria **adc44d**, para o directório de desenvolvimento do código. Outra alternativa é adicionar a “path” onde os módulos se encontram através de **sys.path.append(<path>)** após **import sys**, no código *Python*;
- importar o módulo que se deseja através de **import <módulo>** ou **from <módulo> import <object>**. Por exemplo caso se queira aceder à DAC basta adicionar no programa **from adc44d_dac import <object>** ou **import adc44d_dac**.

C.3 Descrição das funções de cada módulo

C.3.1 Módulo **adc44d**

```
adc44d_reset(adc_file)
```

Esta função faz um reset completo à placa ADC44D, incluindo os registos e a lógica. Caso esta operação não seja bem sucedida é gerada a excepção **BOARD_RESET** (ver definição em **adc44d.py**).

C.3.2 Módulo `adc44d_adc`

Caso não seja dito nada em contrário, é sempre levantada a excepção `ADC_IOERR`, em caso de erro nas funções descritas abaixo.

`adc=adc_open()`

Esta função abre o nó no sistema de ficheiros `/dev/adc44d-adc`, devolvendo o respectivo descritor (`adc`). Se a operação não for bem sucedida é gerada a excepção `ADC_OPEN`.

`adc_close(adc)`

Esta função fecha o ficheiro `/dev/adc44d-adc`, identificado pelo descritor `adc`. Em caso de erro é levantada a excepção `ADC_CLOSE`.

`adc_reset(adc)`

Esta função faz um “reset” ao ADC.

`adc_set_mode(adc,mode)`

Esta função define o modo de operação do ADC. É possível definir 3 modos, mas neste momento a escolha do terceiro modo não tem qualquer utilidade, visto que os “timers” não são considerados nesta versão do driver . Os modos válidos são: funcionamento por operações individuais de I/O (`mode=0`) e funcionamento por reactivação automática (`mode=1`). No último modo, referido anteriormente, a ADC é automaticamente reactivada após a leitura da última amostra. Caso seja definido o terceiro modo de funcionamento (`mode=3`) é gerada a excepção `ADC_INTERR`.

adc_set_gain(adc,gain)

Esta função define o ganho da ADC. É possível definir 4 ganhos (ver Anexo A): ganho por defeito (**gain=1**), ganho 2 (**gain=2**), ganho 10 (**gain=10**) e ganho 100 (**gain=100**).

Caso seja definido um outro ganho é assumido o valor por defeito, ou seja 1.

adc_enable_timer(adc) e adc_disable_timer(adc)

As funções de activação e desactivação do “timer” encontram-se definidas, mas de momento não têm qualquer utilidade, visto que o modo 3 de funcionamento não é considerado.

adc_set_single(adc)

Esta função define o funcionamento não diferencial para a ADC. Isto implica a existência 16 canais.

adc_set_differential(adc)

Esta função define o funcionamento diferencial para a ADC. Isto implica a existência de 8 canais em vez de 16.

adc_set_io(adc):

Esta função configura o ADC para transferir a informação, através de operações de I/O.

adc_set_dma(adc):

Esta função configura o ADC para transferir a informação, através de DMA.

adc_set_channel(adc,channel)

Esta função permite a selecção do canal para o ADC. O valor de **channel** pode ser um valor entre **0** e **15** no modo não deferencial ou um valor entre **0** e **7** no modo diferencial.

channel=adc_get_channel(adc)

Esta função devolve o canal (**channel**) actualmente em uso.

adc_set_channel_man(adc)

Esta função define a selecção manual dos canais.

adc_set_channel_auto

Esta função define a selecção automática dos canais.

adc_start_conv(adc)

Esta função desencadeia uma conversão no modo 0 (I/O), ou iniciando uma sequência de conversões no modo 1 (reactivação automática).

ready=adc_data_ready(adc)

Esta função devolve **1** se os dados estão prontos para leitura ou **0** caso contrário.

value=adc_read(adc)

Esta função lê os 12 bits da ADC sem verificar se os dados estão ou não prontos para leitura, ou seja, limita-se a ler um registo.

value=adc_recv(adc,mode,gain,channel)

Esta função realiza todos os passos necessários para uma conversão correcta. Esses passos são: definição do modo de funcionamento, definição do ganho, escolha do canal, início da conversão e verificação de fim de conversão.

C.3.3 Módulo `adc44d_dac`

Caso não seja dito nada em contrário, é sempre levantada a excepção **DAC_IOERR**, em caso de erro nas funções descritas abaixo.

dac=dac_open()

Esta função abre o nó no sistema de ficheiros `/dev/adc44d-dac`, devolvendo o respectivo descritor (**dac**). Se a operação não for bem sucedida é gerada a excepção **DAC_OPEN**.

dac_close(dac)

Esta função fecha o ficheiro `/dev/adc44d-dac`, identificado pelo descritor **dac**. Em caso de erro é levantada a excepção **DAC_CLOSE**.

dac_reset(dac)

Esta função faz um “reset” ao DAC.

channel=dac_get_channel(dac)

Esta função devolve o canal (**channel**) em uso no DAC.

dac_select_channel(dac,channel)

Esta função permite seleccionar um dos 4 canais disponíveis. Esta selecção é feita atribuindo ao parâmetro **channel** o número do canal.

dac_enable_channels(dac,channels)

Esta função define quais os canais do DAC que são activados. O parâmetro **channel** pode assumir um dos valores listados na Tabela (C.1).

Tabela C.1: Valores possíveis do argumento **channels** na função **dac_enable_channels**.

| Valor | Canais activos |
|-------|----------------|
| 0 | Nenhum |
| 1 | 0 |
| 2 | 1 |
| 3 | 0 e 1 |
| 4 | 2 |
| 5 | 0 e 2 |
| 6 | 1 e 2 |
| 7 | 0, 1, 2 |
| 8 | 3 |
| 9 | 0 e 3 |
| 10 | 1 e 3 |
| 11 | 0, 1 e 3 |
| 12 | 2 e 3 |
| 13 | 0, 2 e 3 |
| 14 | 1, 2 e 3 |
| 15 | Todos |

dac_ref_on(dac)

Esta função activa o DAC.

dac_ref_off(dac)

Esta função desactiva o DAC.

dac_write(dac,value)

Esta função limita-se a escrever **value** para o registo de dados do DAC, servindo como função de base para outras funções. Caso se queira escrever correctamente para o DAC, deve-se usar a função **dac_send(dac,channel,value)**.

dac_send(dac,channel,value)

Esta função implementa todos os passos necessários a uma correcta actualização do canal (**channel**) escolhido. Esta usa a função **dac_write(dac,value)** para escrever para o registo de dados do DAC. **value** deve conter um inteiro entre 0 (correspondente a uma tensão de saída de -10 V) e 4095 (correspondente a uma tensão de saída de 10 V).

C.3.4 Módulo **adc44d_dio**

Caso não seja dito nada em contrário, é sempre levantada a excepção **DIO_IOERR**, em caso de erro nas funções descritas abaixo.

dio=dio_open()

Esta função abre o nó no sistema de ficheiros **/dev/adc44d-dio**, devolvendo o respectivo descritor (**dio**). Se a operação não for bem sucedida é gerada a excepção **DIO_OPEN**.

dio_close(dio)

Esta função fecha o ficheiro `/dev/adc44d-dio`, identificado pelo descritor **dio**. Em caso de erro é levantada a excepção **DIO_CLOSE**.

dio_set_mode0_cword(dio,word)

Esta função define a “control word” (**word**) para o DIO a funcionar no modo 0. Existem mais dois modos de funcionamento, mas estes não são considerados pelo “driver”. As “control words” definem quais dos portos do DIO são para entrada ou para saída. A listagem dos valores que **word** pode assumir, pode ser vista na Tabela (A.6).

dio_get_mode0_cword(dio)

Esta função devolve a “control word” activa no DIO a funcionar no modo 0.

dio_write(dio,value)

Esta função escreve um inteiro de 24 bits (**value**) para os registos de dados do DIO, ou seja, afecta todos os portos. Esta função é usada, como função de base, pelas funções de escrita apresentadas de seguida.

dio_<porta>_send(dio,value)

As funções **dio_<porta>_send(dio,value)** escrevem o valor **value** para um dos portos do DIO (A, B, C, CL ou CH). Por exemplo para a porta A a função chama-se **dio_a_send(dio,value)**.

dio_send(dio,value)

Esta função escreve um inteiro de 24 bits para o DIO, ou seja, afecta todos os portos. Esta é uma melhoria da função **dio_write(dio,value)** e deve ser usada sempre que se queira escrever para todos os registos de dados do DIO.

value=dio_read(dio)

Esta função devolve um inteiro de 24 bits, resultante da leitura de todos os portos do DIO. Tal como a função **dio_write(dio,value)** é usada como função de base por outras funções de leitura apresentadas de seguida.

value=dio_<porta>_rcv(dio)

As funções **dio_<porta>_rcv(dio)** lêem o valor **value** de um dos portos do DIO (A, B, C, CL ou CH). Por exemplo para o porta A a função chama-se **dio_a_rcv(dio)**.

value=dio_rcv(dio)

Esta função lê um inteiro de 24 bits do DIO, ou seja, afecta todos os portos. Esta é uma melhoria da função **value=dio_read(dio)** e deve ser usada sempre que se queira ler de todos os portos do DIO.

Anexo D

Código de Alto Nível para a porta paralela

Manual do utilizador

César A. D. Teixeira

Centro de Sistemas Inteligentes

Faculdade de Ciências e Tecnologia

Universidade do Algarve

Campus de Gambelas

8000 Faro

Resumo

Neste manual de utilizador é apresentado um módulo *Python*, que faz interface com o “driver” **ctparport**. Este permite o controlo da porta paralela a partir de programas escritos em

Python.

D.1 Introdução

O “Código de Alto Nível para a porta paralela” é composto por um módulo que usa as primitivas do SO e implementa os condicionalismos e políticas de programação (no espaço do utilizador) que não são devem ser implementadas ao nível do driver (espaço do *Kernel*), permitindo que programas *Python* consigam aceder e controlar facilmente a porta paralela.

D.2 Uso do módulo

D.2.1 Pré-requisitos

Para usar o módulo é necessário:

- o módulo **ctparport.o** devidamente instalado no *Kernel*;
- o nó **/dev/ctparport** criado e a “apontar” para o driver **ctparport**. (Em caso de dúvida é aconselhável a leitura do manual do utilizador do driver, que se encontra no Anexo B);
- o ficheiro **ctparport_py.tar.gz**;
- um interpretador de *Python*.

D.2.2 Como usar o módulo

Para uma utilização fácil é aconselhável seguir os seguintes passos:

- descompactar o ficheiro:
ctparport_py.tar.gz
- Copiar o módulo presente na directoria **ctparport_py**, para o directório de desenvolvimento do código. Outra alternativa é adicionar a “path” onde o módulo se encontra através de **sys.path.append(<path>)** após **import sys** no código *Python*.
- importar o módulo através de **import ctparport** ou **from ctparport import <objecto>**.

D.3 Descrição das funções do módulo

D.3.1 **fd=ctparport_open()**

Esta função abre o nó **/dev/ctparport**, devolvendo o descritor correspondente.

D.3.2 **ctparport_close(ctpar_fd)**

Esta função fecha o ficheiro **/dev/ctparport**.

D.3.3 **ctpar_enable_int(fd)**

Esta função activa a interrupção da porta paralela.

D.3.4 **ctpar_disable_int(fd)**

Esta função desactiva a interrupção da porta paralela.

D.3.5 `status=ctpar_get_int_stat(fd)`

Esta função devolve `status=0` se a interrupção está desactivada ou `status=1` se a interrupção está activa.

D.3.6 `val=ctparport_read_sel(fd,int_flag)`

Esta função lê a “select line” da porta paralela. Se o parâmetro `int_flag` for `0` significa que a interrupção está desactivada, e a função retorna imediatamente o valor lido. Caso `int_flag` seja `1` significa que a interrupção está activa, o que leva a que esta função fique bloqueada até que a interrupção seja levantada.

D.3.7 `ctparport_write(fd,value)`

Esta função escreve o valor `value` para o porto de dados da porta paralela. O valor `value` deve ser um valor entre `0` e `255` (8 bits).

Anexo E

Interface Gráfico

Manual do utilizador

César A. D. Teixeira

Centro de Sistemas Inteligentes

Faculdade de Ciências e Tecnologia

Universidade do Algarve

Campus de Gambelas

8000 Faro

Resumo

Neste manual de utilizador é apresentado um interface gráfico programado em *Python Gtk*, que permite: alterar as referências, alterar os parâmetros de controlo, apresentar o valor dos

sensores, apresentar os valores dos erros, o estado dos actuadores e do “sensor de fluxo”.

E.1 Introdução

O “Interface Gráfico” permite ao operador do processo de controlo da solução nutriente, a alteração e verificação dos parâmetros de uma forma intuitiva. A apresentação e alteração dos parâmetros de controlo e das referências são feitas através de caixas de texto. Os valores dos sensores são apresentados em caixas de texto, sendo também possível a sua apresentação de uma forma gráfica. Por último este interface apresenta o estado dos actuadores (bombas e resistência) e do sensor de fluxo através de figuras.

Os valores dos sensores, dos erros e dos parâmetros de controlo são resultantes de uma pesquisa a uma base de dados, sendo o estado dos actuadores e do sensor de fluxo obtidos através do código explicado nos Anexos C e D.

E.2 Uso do interface

E.2.1 Pré-requisitos

Para usar o interface é necessário:

- ter os módulos **bct-adc-44d.o** e **ctparport.o** instalados;
- ter os nós: `/dev/adc44d-dac`, `/dev/adc44d-adc`, `/dev/adc44d-dio` e `/dev/ctparport`; a “apontar” para os drivers **bct-adc-44d** e **ctparport** (em caso de dúvida num dos tópicos anteriores, é aconselhável consultar os manuais do utilizador dos drivers, descritos nos Anexos A e B).

- o ficheiro **interface.tar.gz**.
- um interpretador de *Python*.
- os módulos *Python*: *gtk*, *GtkExtra*, *gtkextra* e *psycpg*.
- PostgreSQL.

E.2.2 Como por interface a funcionar

Para uma utilização fácil é aconselhável seguir os seguintes passos:

- criar uma base de dados PostgreSQL com as tabelas apresentadas na Figura E.1, caso esta não exista. Atenção: Esta base de dados é para efeitos de teste, contendo deste

| par_cont | setpoints | erro | sensores |
|---|---|---|---|
| +calc_int: real +aqui_int: real +date: date | +setp_ph: real +setp_cond: real +setp_temp: real +date: date | +erro_ph: real +erro_cond: real +erro_temp: real +date: date | +sens_ph: real +sens_cond: real +sens_temp: real +date: date |

Figura E.1: Tabelas da base de dados de teste.

modo as tabelas mínimas para o interface funcionar;

- colocar o servidor de base de dados *postmaster* a funcionar:

```
postmaster -i -D <directoria da base de dados>;
```

- descompactar o ficheiro:

```
interface.tar.gz;
```

- entrar na directoria **interface** gerada e executar **python proj.py**.

E.2.3 Como utilizar o interface

Janela principal

Ao executar `python proj.py` é mostrada a janela visualizada na Figura E.2.



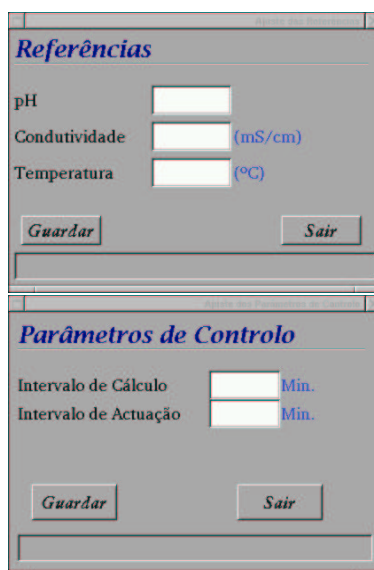
Figura E.2: Janela principal do interface.

Esta janela mostra os dados instantâneos e os botões de acesso a outras janelas.

Alterar as “Referências” ou os “Parâmetros de Controle”

Para alterar as referências ou os parâmetros de controlo, basta clicar no botão **Alterar** na área **Referências** ou na área **Parâmetros de Controlo**, da janela principal. Após esta operação é mostrado no ecrã uma das janelas mostradas na Figura E.3.

Nestas janelas basta escolher com o rato um ou vários campos (pH, Condutividade, Temperatura, Intervalo de Cálculo ou Intervalo de Actuação), e introduzir os valores. Para actualizar os valores na base de dados basta pressionar o botão **Guardar**. Se alguns dos



The image shows two screenshots of a software interface. The top window is titled "Referências" and contains three input fields: "pH", "Condutividade (mS/cm)", and "Temperatura (°C)". Below these fields are two buttons: "Guardar" and "Sair". The bottom window is titled "Parâmetros de Controlo" and contains two input fields: "Intervalo de Cálculo Min." and "Intervalo de Actuação Min.". Below these fields are two buttons: "Guardar" and "Sair".

Figura E.3: Janelas de alteração das “Referências” e dos “Parâmetros de Controlo”

campos contiverem caracteres não numéricos, ou valores fora dos intervalos admissíveis para as variáveis, os valores não são guardados. Nesta situação é gerada uma mensagem de erro (Fig. E.4), e o controlo do programa é de novo devolvido às janelas anteriormente referidas. Para sair de qualquer uma das janelas basta pressionar o botão **Sair**, perdendo-se os dados introduzidos caso estes não tenham sido previamente guardados.

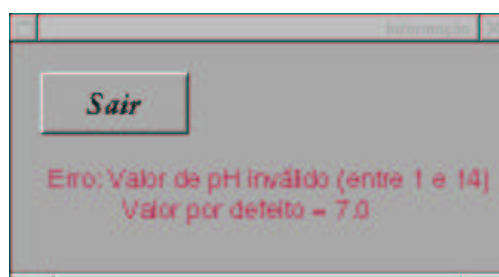


Figura E.4: Mensagem de erro.

Visualização do estado dos actuadores

Para visualizar o estado dos actuadores basta clicar sobre o botão **Estado Actuadores**.

Posteriormente é mostrada a janela exposta na Figura E.5.



Figura E.5: Apresentação gráfica do estado dos actuadores.

Visualização dos valores dos sensores de uma forma gráfica

A visualização da evolução dos valores dos sensores, pode ser vista graficamente pressionando o botão **Ver Gráficos** na janela principal. De seguida é mostrada a janela exposta na Figura E.6.



Figura E.6: Janela de escolha dos gráficos.

A partir desta janela pode ser seleccionado o gráfico que se pretende visualizar. Por exemplo se for pressionado o botão **Temperatura** é mostrado o gráfico visualizado na Figura E.7.

Estes gráficos mostram valores numa janela deslizante de dimensão $10 \times$ "Intervalo de cálculo".

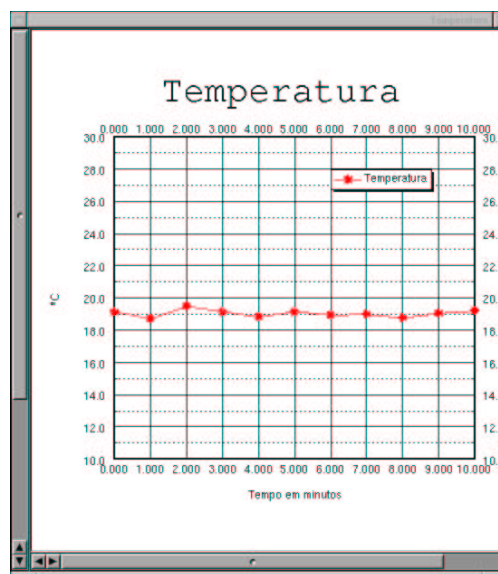


Figura E.7: Gráfico da evolução dos valores (simulados) para a variável **Temperatura**.

Bibliografia

- [1] R. R. do Carmo Junior, “O que É hidroponia.” [Online]. Available: <http://www.terravista.pt/bilene/7810/oque.htm>
- [2] H. Warehouse, “What is hydroponics?” [Online]. Available: <http://www.hydroponics.webcentral.com.au/what/default.htm>
- [3] M. Schwarz, *Soilless Culture Management*. Springer-Verlag, 1995.
- [4] P. Howard M. Resh, *Hydroponic Food Production*, 5th ed. Woodbridge Press Publishing Company, 1995.
- [5] S. Carruthers, *Hydroponic Gardening*, J. Patrick, Ed. Lothian Books, 1993.
- [6] P. Young, M. A. Behzadi, A. Chotai, and P. Davis, “The modelling and control of nutrient film systems.”
- [7] *ADC-44d Multi-Function Analogue Input/Output Card, User Manual*, Blue Chip Technology Ltd. [Online]. Available: http://www.bluechiptechnology.co.uk/drivers/ADC-44D_user_manual.pdf

- [8] Cabosul, “Glossário, especificações de termos técnicos.” [Online]. Available: <http://www.cabosul.com.br/glossario/glossario.htm>
- [9] “82c55a chmos programmable peripheral interface,” Intel, datasheet. [Online]. Available: <ftp://download.intel.com/design/periphrl/datashts/23125604.pdf>
- [10] “Sn54ls06, sn74ls06, sn74ls16 hex inverter buffers/drivers with open-collector high-voltage outputs,” Texas Instruments, datasheet. [Online]. Available: <http://focus.ti.com/lit/ds/symlink/sn74ls06.pdf>
- [11] “Series oac,(s)moac, x4oac ac; output modules,” Crydom, datasheet. [Online]. Available: http://www.crydom.com/pdf/io_oac.pdf
- [12] “Series pf, PowerFinTM up to 25amp ”120/240, 380, 480 vac ”ac output,” Crydom, datasheet. [Online]. Available: <http://www.crydom.com/pdf/pf.pdf>
- [13] *Easidata MK4 Handbook*, Environdata Austrália Pty. Ltd., 44 Percy Street, Warwick, Queensland, Austrália, 4370, rev. 181200.
- [14] I. Harries, “Interfacing to the ibm-pc parallel printer port.” [Online]. Available: <http://www.doc.ic.ac.uk/~ih/doc/par/>
- [15] P. H. Anderson, “Use of a pc printer port for control and data acquisition.” [Online]. Available: <http://et.nmsu.edu/~etti/fall96/computer/printer/printer.html>
- [16] W. N. Engineering, “Parallel port background.” [Online]. Available: <http://www.fapo.com/porthist.htm>
- [17] M. Lutz, *Programming Python*. O’ Reilly, 1996.

- [18] H. Daniel, “Resumo das aulas de sistemas de tempo real,” 2003, resumo das Aulas Teóricas da Cadeira de Sistemas de Tempo Real.
- [19] J. C. Alessandro Rubini, *Linux Device Drivers, 2nd Edition*. O’ Reilly, 2001.
- [20] M. C. Daniel P. Bovet, *Understanding the Linux Kernel*. O’ Reilly, 2000.
- [21] F. L. D. J. Guido van Rossum, *Python Tutorial*, 2002.
- [22] M. Pilgrim, *Dive Into Python*, 2002.
- [23] F. T. A. Alpi, *Cultivo en invernadero*. Mundi-Prensa, 1991.