Sistema Integrado de Aquisição de Dados para Física Nuclear

Tese de Mestrado

José Luís Malaquias

Bolseiro do Programa Ciência

Departamento de Física

Faculdade de Ciências e Tecnologia

Universidade de Coimbra

À Minha Mãe

Conteúdo

-

Agradecimentos

Não é invulgar, numa tese de mestrado, começar-se os agradecimentos pelo orientador de mestrado. No meu caso, não serei excepção, mas terei de o fazer por três motivos diferentes. Assim, é com muito prazer que agradeço ao Prof. Carlos Correia a orientação propriamente dita, sempre pronta e preciosa, a autoria da placa de aquisição sobre a qual este sistema se baseia, sem a qual me teria perdido nos meandros do hardware, mas sobretudo o excelente ambiente de trabalho que tem sabido inspirar no Laboratório de Electrónica e Instrumentação, a cujos membros estendo também os meus agradecimentos.

Este mestrado foi financiado pela Junta Nacional de Investigação Científica e Tecnológica, no âmbito do programa Ciência, à qual deixo aqui os meus agradecimentos pelo auxílio financeiro prestado.

Gostaria também de exprimir os meus mais sinceros agradecimentos ao Centro de Fusão Nuclear e ao seu presidente, Prof. Carlos Varandas, por me ter concedido a oportunidade de concluir o projecto descrito nesta tese e ter demonstrado tanta boa vontade perante os sucessivos atrasos sofridos.

Seria injusto não mencionar aqui também o valioso contributo que me foi dado pelo Paulo Amilcar e pelo Pedro Almeida – dois finalistas da licenciatura em Engenharia Física com quem foi sempre um prazer trabalhar.

Por fim, a um título mais pessoal, gostaria de agradecer à minha esposa Raquel todo o apoio e encorajamento que me deu durante os momentos de maior desânimo e a boa disposição com que sempre me tem acompanhado.

Resumo

É descrito um sistema integrado de aquisição de dados para utilização em experiências na área de física nuclear. O sistema gira em torno de uma placa de aquisição de sinal para PC que inclui módulos de analisador multicanal, contador multicanal e analisador de sinal, além de um processador digital de sinal de vírgula flutuante, encarregue da gestão da placa. Foi desenvolvido o software de baixo nível que corre no processador digital de sinal e um pacote de alto nível que corre no computador hospedeiro e em diversos outros computadores que podem participar do sistema através de uma rede. Todo o pacote de alto nível inclui uma interface gráfica de fácil utilização, baseada nos sistemas operativos Windows NT e Windows 95. A visualização dos dados adquiridos é possível a partir de qualquer ponto da rede.

Abstract

An integrated data acquisition system to be used in nuclear physics experiments is described. The system is based upon a PC expansion board which includes a multichannel analyzer, a multichannel scaler, a signal analyzer, besides a floating point digital signal processor, in charge of the board management. The low level software running on the digital signal processor and a high level package running on the host computer, and on other computers that can take part of the system through a computer network, have been developed in the course of this project. The whole high level package includes a graphical user friendly interface, based on the Windows NT and Windows 95 operating systems. The acquired data visualization can be performed from any point of the computer network.

Introdução

Os Processadores Digitais de Sinal

No mundo da electrónica actual, são cada vez mais omnipresentes os processadores digitais de sinal. Trata-se de microprocessadores, já de si extremamente rápidos, que foram especialmente concebidos e optimizados para o processamento de sinais electrónicos, frequentemente em tempo real. Podem ser encontrados em diversos tipos de equipamento de telecomunicações (telemóveis, modems inteligentes, centrais comutadoras digitais, etc.), de audiovisual, de instrumentação médica e num número sempre crescente de outras aplicações.

Também no campo da electrónica nuclear, os processadores digitais de sinal fizeram já a sua aparição, pela sua especial aptidão para tratar sinais rápidos e sobre eles efectuar vários tipos de tratamento, em tempo real ou quase real. É, de facto, vulgar encontrar esse tipo de circuito ligado a conversores analógicos–digitais, com recurso ou não a uma prévia filtragem analógica.

A presença de um circuito processador num sistema electrónico, quer se destine ou não à electrónica nuclear, confere ainda ao sistema em causa um certo grau de "inteligência" que lhe permite maior flexibilidade e versatilidade. As vantagens são por demais evidentes, tendo já sido discutidas em diversa literatura. Uma solução de software é sempre mais flexível, permite uma maior padronização dos componentes utilizados, a sua actualização é menos dispendiosa e torna possível a implementação de algoritmos de tratamento de sinal

mais complexos. Esta última vantagem, em especial, é especialmente relevante quando existe uma grande quantidade de dados a transmitir aos módulos de monitorização de um sistema. Nesses casos, a existência de um processador de sinal permite que os dados adquiridos sofram um primeiro tratamento de selecção/redução e sejam transmitidos à fase seguinte já "digeridos" e com um maior conteúdo informativo.

Uma vantagem adicional, não desprezável, da presença de um processador de sinal num sistema de aquisição de dados tem a ver com o facto de um único circuito poder substituir, com vantagem, uma profusão de outros circuitos, os quais efectuariam por hardware operações que passam a ser efectuadas por software. O espaço precioso ganho na placa de circuitos com essa substituição, aliado à maior flexibilidade de operações da solução por software em processador de sinal, irá permitir a inclusão de um maior número de funções num mesmo sistema de aquisição de dados, sem um grande acréscimo de custos ou de complexidade. Assim, por exemplo, um sistema de aquisição multicanal, que tradicionalmente seria implementado em uma ou mais placas de circuito impresso, passa a poder partilhar uma única placa com outras funções de monitorização do sinal, de temporização, de comunicação com outros sistemas, etc.

O Computador Pessoal

Um outro desenvolvimento extraordinário a que temos vindo a assistir nos últimos anos é o dos computadores pessoais e de secretária. Uma arquitectura, em especial, sofreu um enorme desenvolvimento desde o seu aparecimento no início da década de 80. Trata-se da arquitectura que derivou do IBM PC original, sob a designação genérica de «compatíveis PC». De um modo geral, trata-se de computadores baseados numa família de microprocessadores específica, a 80x86 da Intel, com uma especificação comum de bus de

periféricos e várias normas de concepção que lhes permitem partilhar a maior base de software do mundo, com programas que vão do puro entretenimento às aplicações científicas mais sérias – cálculo científico, processamento e visualização gráfica, simulação, desenho técnico e industrial, etc. Entre essas aplicações, não poderia deixar de figurar também a aquisição de dados, nomeadamente na área da electrónica nuclear.

A existência de um bus comum de periféricos – normalmente designado por bus ISA, nas suas versões XT de 8 bits e AT de 16 bits (uma terceira versão de 32 bits designada por EISA nunca teve muito sucesso) – permitiu que, desde muito cedo, vários fabricantes independentes de instrumentação e electrónica apresentassem no mercado placas de expansão do PC que o transformavam num versátil sistema de aquisição, processamento e apresentação de dados. Essas placas tipicamente encaixam directamente no interior do PC, nas entradas do bus ISA, existindo por vezes um segundo módulo exterior ao computador, ligado directamente à placa de expansão interna. Na sua grande maioria, as primeiras placas que surgiram, não possuíam poder de processamento próprio, dependendo exclusivamente do PC para a sua operação. Numa configuração típica, possuiriam um ou mais conversores analógicos–digitais, capazes de ser lidos directamente a partir do próprio PC, através de um conjunto de endereços do seu espaço de periféricos, expressamente reservados para esse efeito.

As Interfaces Gráficas

Na sequência do enorme sucesso que gozou a arquitectura PC ao longo da última década, emergiu uma outra classe de produtos aliada ao desenvolvimento do computador pessoal. Trata-se das interfaces gráficas dos novos sistemas operativos para PC. Destinadas a tornar mais simples e acessível a utilização dos computadores pessoais por parte de leigos, essas

interfaces gráficas começaram a popularizar-se nos computadores Macintosh da Apple. Por exigência do enorme número de utilizadores não especializados que entretanto surgiu no campo da arquitectura dos compatíveis PC, a Microsoft lançou no mercado o Microsoft Windows. Este, de início, pouco mais era do que uma *shell* (concha) que isolava o utilizador das complexidades do próprio sistema operativo da Microsoft, o MS-DOS. O programa oferecia ao utilizador uma rudimentar interface gráfica, que pouco mais permitia do que lançar, por meio da utilização de um rato, as vulgares aplicações DOS que nenhuma alteração sofriam, no que respeita à sua facilidade de utilização. Muito poucas aplicações foram desenvolvidas fora da própria Microsoft para tirar partido das facilidades gráficas oferecidas pelo MS Windows. Só com a versão 3.0 do Windows, a Microsoft conseguiu produzir um sistema realmente útil e eficaz e alcançou o sucesso que hoje é reconhecido a toda a gama de produtos Windows. Pela primeira vez, essa versão oferecia vantagens convincentes quer aos programadores de aplicações quer aos utilizadores finais. Essas vantagens incluíam uma gestão de memória muito melhorada, a possibilidade de executar várias aplicações simultaneamente, tirando partido do modo de operação protegido dos processadores Intel que se seguiram ao 80386, várias facilidades gráficas dificilmente obteníveis em DOS, gestão comum de impressoras e outros periféricos, fácil troca de informação entre aplicações, entre muitas outras vantagens estratégicas para o desenvolvimento de aplicações. Com a versão 3.0 do Windows, a Microsoft lançou no mercado um produto híbrido, pois tratava-se de mais do que uma simples concha, ao efectuar a gestão de memória, de alguns periféricos, etc., mas que ainda não se qualificava propriamente como sistema operativo, pois não efectuava operações básicas de um sistema

operativo, como a gestão de ficheiros, para as quais exigia a presença simultânea, no computador, do sistema operativo MS-DOS, com o qual trabalhava em íntima cooperação.

Na sequência do sucesso do Windows 3.0, a Microsoft lançou várias actualizações no mercado que vieram acrescentar ao produto já existente outras características, como a gestão de rede, capacidades multimédia, utilização de canetas digitais, interligação de objectos, etc.

Só, porém, com o lançamento do Windows NT a Microsoft lançou um verdadeiro sistema operativo completo, já incluindo a interface gráfica e todas as funções que se esperam de um sistema operativo, capaz de tirar partido da arquitectura de 32 bits dos processadores Intel do 80386 em diante. Especialmente vocacionado para utilizações mais sérias, em estações de trabalho de alto desempenho, esse sistema operativo, com soberbas capacidades de ligação em rede, uma gestão de processamento paralelo muito eficaz e perfeitamente portátil para outros tipos de processador, veio abrir as portas a tipos inteiramente novos de aplicações, até então reservadas a computadores de grande porte, em ambiente VAX ou Unix.

Recentemente, foi lançado um novo produto da linha Microsoft Windows: o sistema operativo Windows 95, introduzido em Agosto de 1995. Trata-se de um verdadeiro sistema operativo completo, também de 32 bits, com uma interface de utilizador completamente revista, mas capaz de correr as aplicações de 32 bits concebidas para Windows NT. Embora se trate de um sistema operativo bastante mais aligeirado (não possui as primitivas de segurança, as características de servidor e outras possibilidades oferecidas pelo seu *irmão mais velho*), inclui já muitas das características introduzidas pelo

Windows NT, como sejam o processamento multitarefa, a gestão de rede integrada e uma interface de aplicações (a chamada API - a parte do sistema operativo que interactua com uma aplicação) em tudo semelhante à Win32 concebida para o NT. Tudo indica que, com este sistema operativo, a Microsoft tenha pretendido apresentar uma plataforma de transição para o Windows NT, destinando-se o Windows 95 ao grosso volume de vendas do mercado doméstico e dos computadores menos potentes, enquanto o NT ficaria, por agora, reservado às aplicações mais sérias, em máquinas mais potentes. Consegue-se assim uma simbiose entre os dois sistemas, pois o maior volume de vendas que se espera para o Windows 95, permitirá a criação de uma biblioteca de software bastante mais vasta que poderá simultaneamente correr em Windows NT, deixando o caminho aberto à expansão de ambas as plataformas. A suposição de o Windows 95 funcionar como uma espécie de campo de ensaios e meio de angariação de mercado para o Windows NT confirma-se pelo facto de a próxima versão 4.0 do Windows NT já incluir uma interface gráfica em tudo semelhante à do Windows 95. O esforço de convergência dos dois sistemas é notório.

A Conjugação de Três Tecnologias

Na sequência da enorme divulgação que tem acompanhado as três tecnologias acima descritas – processadores digitais de sinal, computadores pessoais e sistemas operativos com interfaces gráficas avançadas – os preços dos produtos delas derivados têm registado enormes descidas, em virtude da massificação da sua produção. Variadíssimos produtos destinados ao grande consumo, agrupando diversos aspectos dessas três tecnologias têm surgido no mercado. Entre eles, destacam-se placas de som para PC, com capacidades de geração de efeitos sonoros quase infindáveis e possibilidade de edição visual de pistas; placas de comunicação telefónica que, por meio de um único circuito, efectuam funções de

fax, atendedor de chamadas, comunicação de dados, videoconferência, comunicação por rede GSM, etc.; osciloscópios digitais capazes de abrir um sem número de ecrãs virtuais no monitor de um PC, entre dezenas de outras aplicações que todos os dias surgem nas revistas da especialidade.

Também o objectivo do trabalho de mestrado apresentado nesta tese foi o de, tirando partido das três tecnologias descritas, produzir um sistema de aquisição de dados para electrónica nuclear que aliasse a potência e a versatilidade dos processadores digitais de sinal à facilidade de utilização das novas interfaces gráficas para PC, tudo isso na plataforma económica e amplamente disponível de um computador pessoal.

Assim, foi produzida uma placa segundo as especificações do bus ISA da arquitectura IBM PC, englobando no mesmo espaço as funções de analisador multicanal, dependente de um ADC de 16 bits, osciloscópio digital, dependente de um ADC de 50 MHz, e contador multicanal, baseado num conjunto de contadores digitais. Todos esses três módulos trabalham sob o controlo de um processador digital de sinal Texas Instruments TMS 320C31, o qual é ainda responsável pelas comunicações com o computador hospedeiro (PC). A gestão dos módulos é efectuada em paralelo, sendo por isso possível a aquisição de dados simultânea por parte dos três elementos que compõem a placa. É da responsabilidade do processador digital de sinal (PDS) a recolha, primeiro tratamento e armazenamento dos dados adquiridos nos vários módulos. Perante solicitações do computador hospedeiro, esses dados são posteriormente encaminhados para o PC, na justa medida das necessidades de visualização e armazenamento definitivo de resultados por parte do utilizador.

No que toca a visualização, essa é garantida do lado do PC por três programas gráficos diferentes, correspondendo cada um deles às três funções da placa acima descritas. A função desses programas será a de, por um lado receber os comandos de controlo e activação vindos do utilizador, por outro lado apresentar gráfica e analiticamente os resultados obtidos, na forma mais conveniente para o utilizador. Cada um desses programas comunica com a placa por intermédio de um quarto programa, unicamente responsável pela inicialização da placa e comunicação posterior entre os três programas de visualização e o programa de gestão que executa no PDS. Esse quarto programa actua, assim, como uma espécie de central telefónica que agrupa, num só canal, as solicitações de três programas de índole diversa. As vantagens dessa separação da aplicação em três programas diferentes são a melhor gestão da largura de banda de comunicação entre o PDS e o computador hospedeiro, com menores riscos de interferência mútua entre diferentes funções do sistema; a possibilidade de execução simultânea, em janelas gráficas distintas, das três componentes, sendo, inclusivamente, possível executar simultaneamente mais do que uma instância do mesmo programa (o programa de visualização do multicanal poderá estar a executar quatro vezes, cada uma delas pondo em evidência um aspecto diferente do espectro adquirido); a possibilidade de execução remota dos programas de visualização, que poderão espalhar-se por vários computadores ligados, por uma rede, ao computador hospedeiro (é, assim, possível um cientista monitorar, do seu gabinete, uma experiência que decorre no laboratório, três andares mais abaixo).

Todos os programas descritos foram desenvolvidos no ambiente Windows NT, por ser a melhor plataforma gráfica de 32 bits disponível no início dos trabalhos. Porém, foi sempre tida em conta a futura compatibilidade com o sistema operativo Windows 95, em cujas

sucessivas versões beta foram sendo efectuadas experiências de execução. Isso fez com que o sistema final ficasse apto, desde o início, a executar nas duas plataformas de que se espera maior divulgação ao longo da próxima década.

Capítulo 1

Descrição Geral do Sistema

Conforme foi dito na introdução, o sistema projectado é essencialmente constituído por três grandes componentes:

1. A placa de aquisição, construída à volta do TMS 32031.
2. O computador, de arquitectura PC AT com bus ISA.
3. O software de interface gráfica

A forma como todas essas componentes se relacionam pode ser vista na Figura 1.

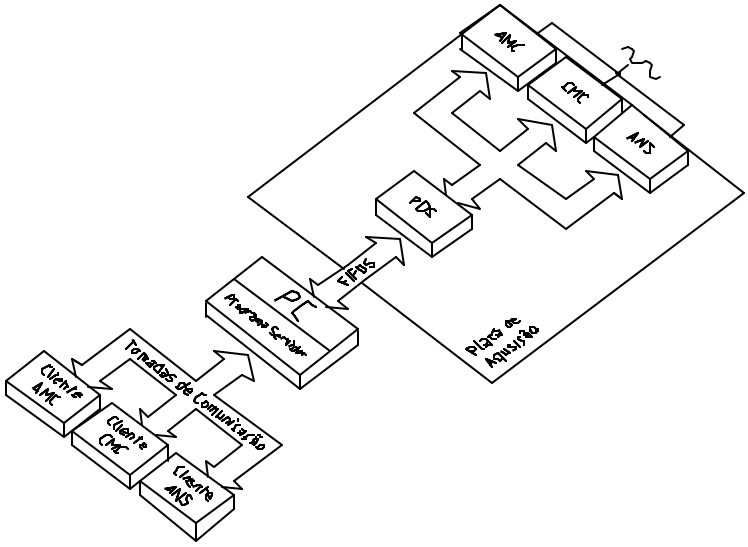


Figura 1 – Esquema de blocos do sistema de aquisição de dados.

Conforme se pode ver, o PC comunica com a placa, essencialmente, através de um conjunto de duas memórias FIFO, orientadas respectivamente em cada um dos sentidos da comunicação. Embora o PC tenha ainda hipótese de controlar e monitorar a placa por meio de alguns portos de leitura e de escrita que não estão assinalados, o grosso da comunicação é efectuada através das memórias FIFO. Estas permitem um esquema de comunicação assíncrono bastante flexível entre o PC e a placa, ainda que tenham sempre associados os inconvenientes de um acesso obrigatoriamente sequencial. Por outro lado, não sofrem as complexidades de um esquema de acesso aleatório em memória partilhada. Pode-se, assim, dizer que serão um bom compromisso entre uma razoável performance – largura de banda teórica de 200 Mbps – e uma grande simplicidade de interfaceamento, que se resume a um mapeamento num porto do lado do PC e num endereço de memória do lado do PDS.

A separação que figura no diagrama entre o PDS e os vários módulos da placa é apenas conceptual, dado que o próprio PDS é parte integrante do funcionamento de cada um desses módulos. Contudo, conforme se verá mais adiante, na descrição de cada componente, o processo de aquisição de dados por parte dos módulos recorre ao PDS unicamente no contexto das interrupções. O tempo de processamento normal do processador é unicamente dedicado à comunicação com o PC, conforme ilustra a figura. A pedido deste último, o PDS obtém imagens dos últimos dados adquiridos em memória pelos vários módulos e fornece-os ao PC através do FIFO. É claro que, paralelamente, estará a actualizar a informação residente em memória, sempre que haja pedidos de interrupção que denunciem a existência de novos dados adquiridos. Esse comportamento, porém, é transparente para o PC.

Do lado do PC, o programa responsável pelas comunicações é o chamado programa servidor. Esse programa é o primeiro responsável pela gestão geral do sistema, sendo só através dele que se pode activar e tirar partido da placa. A sua primeira função, quando da activação do sistema, é obter um programa de gestão do PDS do disco e descarregá-lo, através do FIFO, na memória da placa. O TMS 32031 foi escolhido pela sua capacidade de carregar dinamicamente um programa ao arrancar. Tudo foi preparado para que esse programa seja o que provém do FIFO. Evita-se assim a necessidade de programar uma EPROM para cada placa e obtém-se uma enorme flexibilidade quanto à programação do PDS. Esta pode ser alterada em qualquer altura e é mesmo possível, um dia, distribuir actualizações de software do PDS em disquete, por modem ou por rede, já durante a fase de comercialização do sistema.

Após o arranque do programa carregado pelo PDS, o PC e o PDS iniciam um ciclo de escuta mútua das memórias FIFO, à espera de uma comunicação do outro lado. Esse ciclo não impede o funcionamento normal do PDS, pois como já se disse todas as suas funções, à excepção da comunicação com o PC, são efectuadas por meio de interrupções. Também não impede o funcionamento normal do PC, pois o programa servidor tem uma estrutura de execução paralela multi-filar, dedicando um dos seus fios de processamento exclusivamente à escuta do FIFO. Começa, pois, a dar frutos a opção por um sistema operativo com capacidades multitarefa.

Regra geral, a iniciativa da comunicação entre o PC e o PDS parte do PC. Este deixa, tipicamente, uma sequência de bytes no FIFO, os quais são interpretados pelo PDS como uma espécie de código máquina, em que cada instrução corresponde a uma operação específica que o PC pretende ver efectuada. Esses comandos poderão ter um ou mais

parâmetros que, nesse caso, constarão também do pedido, podendo ainda carecer de resposta. Exemplos de comandos poderão ser: ordens para activar, suspender, reactivar ou desligar um dos módulos; pedidos de actualização de dados recentemente adquiridos; pedidos de diagnóstico, etc. Perante essas instruções, o PDS começa por descodificá-las para, em seguida, invocar as subrotinas apropriadas de execução dos respectivos comandos. Dados que tenham eventualmente sido solicitados são, por sua vez, depositados no FIFO de resposta, onde o PC os irá ler e encaminhar para a aplicação que originou o pedido.

Poderá parecer estranha a designação de "servidor" para a aplicação que corre no PC, pois perante o PDS ela parece comportar-se como cliente. No entanto, conforme se pode ver no canto inferior do diagrama da Figura 1, o mesmo programa relaciona-se com outros três tipos de programas diferentes. Os clientes, respectivamente, do analisador multicanal, do contador multicanal e do analisador de sinal. Conforme se disse na introdução, mais do que um programa de cada tipo pode estar a correr simultaneamente, sendo esse precisamente um dos motivos da escolha desta arquitectura cliente–servidor. A comunicação entre o programa servidor e os diversos programas clientes é efectuada através de *tomadas*, um método muito usado actualmente em qualquer comunicação entre um servidor (p. ex. uma base de dados) e os seus clientes (p. ex. programas de acesso aos dados armazenados). Uma das principais vantagens é a flexibilidade que permite quanto à localização dos diversos programas. A localização do programa cliente é irrelevante. Desde que este saiba nomear a localização na rede em que se situa o outro extremo da sua tomada de comunicação com o servidor, poderá funcionar tão bem como se residisse no mesmo computador.

Com base nos comandos que vai recebendo do utilizador, cada um dos programas cliente irá redigir sequências de comandos destinadas ao PDS. Essas sequências são transmitidas através da respectiva tomada para o programa servidor, o qual funcionará como uma simples antena retransmissora para o PDS. Caso haja pedido de resposta, o próprio servidor se encarregará de aguardar os dados provindos do PDS e colocá-los na mesma tomada de onde proveio o pedido, a fim de chegarem ao programa cliente. Esse se encarregará de produzir a visualização mais apropriada dos dados recebidos no ecrã do utilizador. Eventualmente, poderá ainda haver lugar a algum processamento posterior dos dados, já no programa cliente. No entanto, o sistema é suficientemente flexível para que, caso se ache mais conveniente, esse processamento possa ser feito no próprio PDS. Desde que o software descarregado de início no PDS preveja a interpretação de um maior leque de comandos, não será difícil conseguir que os dados fluam do PDS para a aplicação cliente já um pouco mais "digeridos". É, pois, possível balancear o esforço de processamento dos dados entre o PDS – amplamente vocacionado para esse tipo de tarefa, mas eventualmente sobrecarregado em caso de número excessivo de interrupções ou pedidos de clientes – e o programa cliente – que poderá estar a correr numa estação de trabalho muito veloz e até baseada em outro sistema operativo como o UNIX ou o VMS.

Capítulo 2

O Hardware

O hardware do sistema aqui descrito gira em torno do TMS 32031 da Texas Instruments. Esse processador é responsável pela gestão dos diversos módulos da placa, pela comunicação com o PC e por qualquer eventual processamento numérico intensivo de que haja necessidade.

A Comunicação do PC com a Placa

A comunicação entre o PDS e o PC processa-se através de um conjunto de duas memórias FIFO complementares, correspondendo a cada um dos sentidos da comunicação. Toda a transferência de dados de e para o PC é feita através de escritas e de leituras nas respectivas memórias FIFO. Adicionalmente, tanto o PC como o DSP têm acesso à sinalização de conteúdo das memórias FIFO, sabendo em cada momento se essas possuem dados, se encontram cheias pela metade ou se estão completamente cheias.

Além dos dados que lhe chegam através do FIFO, o PC tem ainda acesso, por intermédio de um porto de leitura e outro de escrita, a informação variada sobre o estado da placa. Além da sinalização de conteúdo das memórias FIFO, já mencionada, o PC tem acesso a duas bandeiras de sinalização do PDS, muito úteis durante a fase de desenvolvimento e diagnóstico. Por último, pode ainda ler nesse porto o estado da linha de "hold acknowledge" do PDS. Finalmente, o PC dispõe também de um porto de escrita, através

do qual pode controlar alguns aspectos do funcionamento da placa. Assim, através desse porto pode provocar um cancelamento e inicialização das memórias FIFO, do próprio PDS e das linhas de interrupção deste último. Tem ainda capacidade de provocar uma interrupção do PDS, sempre que seja forçoso chamar a sua atenção (na actual versão do software, essa capacidade só é utilizada no carregamento inicial do programa do PDS) e de colocar o PDS em estado de espera.

As Interrupções do PDS

Das quatro interrupções que fazem parte da arquitectura do TMS 32031, uma delas, conforme já se viu é utilizada pelo próprio PC. As três interrupções restantes são utilizadas pelos três diferentes módulos da placa, a saber, o módulo analisador multicanal, o módulo contador multicanal e o módulo analisador de sinal.

Toda a arquitectura de funcionamento da placa se baseia nessas interrupções. Em modo de processamento normal, fora das rotinas de interrupção, o PDS mais não faz do que responder aos comandos e solicitações do PC, encaminhando-os para os módulos apropriados. A aquisição de dados propriamente dita tem apenas lugar numa lógica de resposta a eventos. Esses eventos serão a chegada de um novo impulso, no caso do módulo multicanal, a expiração de um período de contagem, no caso do módulo contador multicanal, e o enchimento de um FIFO, no caso do módulo analisador de sinal. Em resposta a esses eventos, o gestor de interrupções do PDS transfere o controlo para software pertencente ao respectivo módulo e os dados entretanto disponíveis são recolhidos.

O Hardware do Módulo Analisador Multicanal

O hardware do módulo analisador multicanal é, talvez, o mais complexo existente na placa. Começa num circuito de formatação do sinal de entrada, ao qual se segue uma porta linear de entrada, de concepção bastante engenhosa, que permite o isolamento de todos os circuitos a jusante de quaisquer novos impulsos que cheguem à placa durante o período em que se efectua a conversão. A concepção dessa porta linear de entrada com base apenas em componentes discretos permite que, a um bom isolamento, não se junte o efeito adverso de uma distorção do sinal provocada por uma porta linear de estado sólido. Seguidamente, o sinal encontra um circuito prolongador de impulsos, o qual garante a manutenção do nível máximo atingido por cada impulso durante o tempo que demora a conversão por parte do ADC (cerca de 15 μs). Finalmente, segue-se a etapa de conversão propriamente dita. Essa é constituída por um par ADC – DAC, que implementa o método da escala deslizante para reduzir os erros de linearidade diferencial inerentes ao ADC. O ADC utilizado é o ADC 700 da Burr-Brown, um ADC de aproximações sucessivas de 16 bits. O DAC utilizado é o DAC 700, também da Burr-Brown, que foi especialmente concebido para trabalhar em conjunto com o ADC 700. Aliás, o seu núcleo é idêntico ao do DAC contido no interior do ADC 700, ligado ao registo de aproximações sucessivas.

A articulação do analisador multicanal com o PDS processa-se da seguinte forma. Sempre que entra na porta aberta do multicanal um novo impulso, é gerado um sinal de pico no momento em que o prolongador de impulsos entra em funcionamento. A lógica da placa provoca então uma interrupção ao PDS e fecha a porta linear de entrada. A rotina de serviço do PDS que trata as interrupções provindas do analisador multicanal, depois de verificar que está tudo em ordem, dá então sinal ao ADC para iniciar a conversão, depois

de ter introduzido no DAC um valor de deslocamento apropriado à implementação da escala deslizante. O serviço dessa interrupção termina aí, libertando o PDS para a execução de outras tarefas pendentes durante a conversão do ADC. Após o sinal de fim de conversão (tipicamente 13-14 µs depois), a lógica da placa gera uma nova interrupção do PDS. A mesma rotina de serviço da interrupção é outra vez invocada, mas desta feita recorda-se que foi anteriormente chamada para dar início à conversão. Assim, o seu procedimento será diferenciado, executando antes uma rotina de leitura do valor adquirido. Após a leitura do valor contido no ADC, é-lhe descontado, por meio de cálculos efectuados em vírgula flutuante no PDS, o deslocamento introduzido pelo DAC que implementa o método da escala deslizante. O valor obtido é depois convertido para o número de bits a que se está a converter (esse número é definido no momento da inicialização do multicanal pelo respectivo programa cliente). O histograma contido na memória da placa é, por fim, actualizado, incrementando o canal correspondente ao valor de conversão obtido.

O Hardware do Módulo Contador Multicanal

O módulo contador multicanal tem uma constituição muito simples, baseada em poucos componentes, todos eles integrados. O coração do sistema é, evidentemente, um contador. Neste caso, trata-se de um conjunto de dois contadores de oito bits, ligados em cascata num contador de 16 bits (Estudos efectuados entretanto, levaram-nos a constatar a necessidade de introdução de um terceiro contador, para elevar o número total para 24 bits. Essa alteração será introduzida na próxima versão da placa). A lógica da placa define, segundo um parâmetro definido pelo PDS num dos seus portos de saída, se a origem dos impulsos que entram no contador é exterior à placa, entrando como sinal TTL através da

ficha de nove pinos, ou interior à placa, correspondendo ao sinal de pico discutido a propósito do analisador multicanal.

Independentemente da origem do sinal que entra no contador, cada novo impulso incrementa o registo do contador de uma unidade. Ao fim de certos intervalos de tempo pré–especificados, os quais mais uma vez poderão ser definidos quer por uma lógica exterior quer pelo conjunto de temporizadores existentes na placa, é efectuada uma interrupção do PDS. A rotina de serviço às interrupções provindas do contador multicanal não faz mais do que ler o valor do registo do contador, ao qual tem acesso por estar mapeado na sua memória, activando simultaneamente um circuito de reinicialização do contador. O valor lido é armazenado na memória interna da placa, numa posição correspondente ao canal temporal que acaba de ser lido.

O procedimento repete-se um número de vezes correspondente ao número de canais temporais solicitado pelo programa cliente do contador multicanal.

No final, ficará armazenado em memória o espectro temporal correspondente ao número de impulsos registado em cada intervalo de tempo.

O Hardware do Módulo Analisador de Sinal

O hardware do módulo analisador de sinal gira em torno de um ADC muito rápido, de 12 bits. As frequências de aquisição possíveis deste ADC vão até aos 50 MHz. É ainda possível uma enorme gama de outras frequências de aquisição, seleccionáveis por software através do programa cliente do analisador sinal.

Dado que, a adquirir à frequência máxima, o PDS não poderia acompanhar a velocidade do ADC e que, mesmo a frequências mais baixas, a arquitectura da placa não permitiria que

o PDS se dedicasse exclusivamente ao analisador de sinal, foi necessário intercalar uma memória tampão entre o ADC e o PDS. Essa memória tampão é constituída por duas memórias FIFO de $8\ k \times 8$ bits cada uma, alternando uma com a outra devido ao facto de, individualmente, não poderem suportar a frequência máxima do ADC..

No momento em que essas memórias ficam cheias, é accionada a interrupção do PDS correspondente ao módulo analisador de sinal. A rotina de serviço a essa interrupção procede à leitura do conteúdo das memórias FIFO, copiando os dados relevantes para uma zona de memória interna da placa. Os dados ficam assim disponíveis para o PDS, a fim de lhes aplicar qualquer tratamento numérico ou proceder ao seu envio para o PC, quando para tal solicitado pelo programa cliente do analisador de sinal.

Outras Componentes do Hardware

Uma componente do hardware que não está atribuída a nenhum módulo específico, mas que poderá ter aplicação em todos eles é o conjunto de temporizadores da placa. Trata-se de dois circuitos integrados 8254 da Intel, cada um dos quais possui internamente três temporizadores. Embora não esteja à partida definida uma aplicação definitiva para todos eles, estes temporizadores vêm permitir à placa o controlo do factor temporal da aquisição de dados.

A primeira aplicação dos temporizadores é, claro está, a fixação dos intervalos de contagem do contador multicanal, conforme foi já referido durante a discussão desse módulo.

Também no caso do analisador multicanal, os temporizadores terão uma importância crucial, pois permitirão a aquisição de histogramas por períodos de tempo pré--determinados, a avaliação de tempos mortos e tempos reais de aquisição, etc.

No caso do analisador de sinal, não existem por agora aplicações concretas dos temporizadores. Numa versão futura da placa, porém, em que se espera que existam circuitos de desencadeamento da aquisição do analisador de sinal, os temporizadores poderão vir a ser úteis na implementação de desencadeamentos com atraso da aquisição.

Capítulo 3

O Software do PDS

O software do PDS encontra-se dividido em quatro grandes blocos, conforme se pode ver na Figura 2. O primeiro, constituído pelo programa principal e subrotinas correspondentes, é basicamente um ciclo interpretador de comandos. Sempre que o PDS se encontra em processamento normal, o programa principal monitoriza as bandeiras de conteúdo das memórias FIFO, à espera da presença de quaisquer dados.

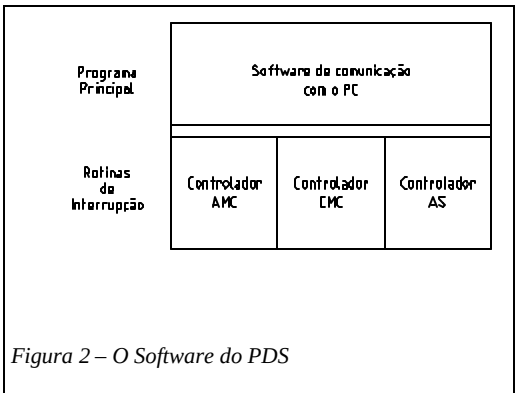


Figura 2 – O Software do PDS

A partir do momento em que o programa que corre no PC emite um comando para o FIFO, o interpretador de comandos detecta-o, recolhe-o e dá início ao processo de descodificação.

A descodificação do comando é efectuada por meio da divisão do primeiro byte do comando em campos de bits, dos quais os primeiros dois bits (bits AA na Figura 3) identificam o destinatário do comando. Como destinatários possíveis dos comandos, teremos o próprio processador – para carregar/descarregar secções, funções de diagnóstico e outras –, o analisador multicanal – para transmitir comandos de

activação/desactivação, recolher dados e configurar a aquisição –, o contador multicanal e ainda o analisador de sinal – ambos para fins idênticos aos do analisador multicanal.

A descodificação do comando é efectuada por fases. Após a descodificação do primeiro campo de dois bits (AA), é chamada uma subrotina correspondente a cada um dos quatro possíveis destinatários do comando. Cada uma dessas quatro subrotinas recebe o comando como parâmetro e dar-lhe-á um tratamento diferenciado. Para isso, cada uma delas procederá à análise dos campos de bits subsequentes. De um modo geral, as subrotinas dividirão os seus comandos em grandes conjuntos de comandos (bits BB) e subcomandos (bits CC, etc.), procedendo assim à descodificação faseada do comando por campos de bits. Consoante o tipo de comando resultante da descodificação, poderá haver necessidade de leitura de outros parâmetros. Esses serão enviados, sempre que necessário, sequencialmente para o FIFO, logo a seguir ao byte de comando ao qual correspondem.

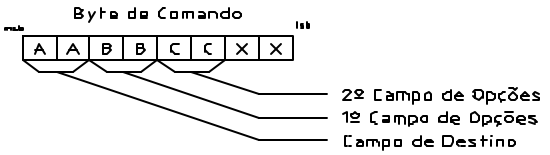


Figura 3 – Estrutura do byte de comando

O Descodificador Geral

O primeiro módulo possível de destino dos comandos do PC é o próprio PDS. Nesse grupo, incluem-se comandos de diagnóstico, envio de secções adicionais de programa (sempre que não seja possível carregar nas memórias FIFO a totalidade do programa no processo de arranque) e suspensão do funcionamento da placa. A esse tipo de comandos corresponde o código 00 nos bits AA do byte de comando. Os comandos possíveis

encontram-se descritos na Tabela 1, juntamente com os respectivos códigos e número de parâmetros esperados (em número de bytes).

Nome do Comando	Código (BB)	N.º Parâmetros
Diagnóstico	00	0
Envio de nova secção	01	2 + secção
Suspender operações	10	0
Retomar operações	11	0

Tabela 1 – Comandos destinados ao PDS.

O primeiro comando, de diagnóstico, tem como primeira função informar o programa servidor a correr no PC do bom funcionamento geral do programa PDS e da placa. Ao receber este comando, o programa que corre no PDS responde com uma sequência pré--estabelecida de bytes, que servem ao PC como garantia de que a placa está bem instalada, o programa está a correr no PDS e a comunicação é possível entre o PC e a placa.

A segunda função deste comando é a de permitir activar o programa servidor, no PC, "a quente", com o programa já a correr no PDS. Numa execução normal, o programa servidor que corre no PC descarrega o programa do PDS na placa e começa de imediato a comunicar com ela. No entanto, nem sempre terá que ser assim. Visto que a placa de aquisição tem uma total autonomia de funcionamento, continuando o respectivo programa sempre a correr enquanto lhe for fornecida alimentação por parte do PC, poderá haver ocasiões em que seja do interesse do utilizador abandonar o programa servidor do PC (para libertar espaço para outros programas, para efectuar a validação de entrada de outro utilizador no sistema, para efectuar uma reinicialização "a quente" do sistema, etc.), deixando porém a placa de aquisição sempre a funcionar em fundo. Depois, mais tarde, o

utilizador pretenderá retomar a execução no ponto em que a deixou. Para isso, pode executar de novo o programa servidor no PC, o qual através da função de diagnóstico toma conhecimento do estado actual de funcionamento da placa e retoma a operação no ponto em que foi interrompida a execução. Assim, é possível, por exemplo, efectuar uma aquisição que se prolongue por vários dias, deixando durante a noite apenas o computador ligado, ainda que num estado de poupança de energia. Num sistema de segurança controlada, como o Windows NT, o utilizador poderá mesmo sair da sua conta e deixar a aquisição a correr em fundo. No dia seguinte, só terá que executar de novo o programa servidor e este invocará a função de diagnóstico para conhecer o estado actual da placa e retomar a operação.

O segundo comando desta série serve para o envio de novas secções de código. A intenção original deste comando era a de possibilitar a utilização de programas com tamanhos superiores ao das memórias FIFO utilizados na comunicação entre o PC e a placa. De facto, o esquema actualmente implementado obriga a que, no momento do arranque do PDS, já o programa que aí irá correr esteja armazenado no FIFO de comunicação do PC para a placa. Na implementação actual isso impõe um limite de 8 kb para o tamanho total do programa e dos dados de inicialização. A solução encontrada foi o envio prévio de uma parcela do programa, capaz de funcionar como receptor da restante parte do programa. Dado que um programa PDS é dividido em secções, no envio prévio seriam enviadas unicamente as secções necessárias ao interpretador de comandos e à execução do comando de envio de secções. As outras secções seriam depois enviadas por intermédio deste comando. Após a codificação das principais funções do sistema, porém, verificou-se que a opção pela utilização exclusiva da linguagem assembler na programação

do PDS permitiu uma enorme compactação do código. Foi possível incluir todo a programação e dados de inicialização necessários ao funcionamento da placa num pacote cujo tamanho é inferior ao tamanho do FIFO de escrita do PC. Não se tornou assim necessária a utilização do comando de envio de secções. O comando, porém, foi mantido tendo em conta possíveis necessidades de expansão futura.

O terceiro e quarto comandos desta série são comandos de suspensão e reposição do sistema. Quando invocados, suspendem e retomam, respectivamente, todas as operações do sistema de aquisição de dados.

O Analisador Multicanal

Os comandos destinados a este módulo controlam toda a actividade do analisador multicanal. Entre os principais comandos encontram-se comandos de activação/ desactivação, acesso ao histograma, limpeza do espectro, etc.

Na Tabela 2 encontram-se sumariados os principais comandos do analisador multicanal.

Nome do Comando	Código (BB)	N.º Parâmetros
Arranque do AMC	00	1
Pausa / Retoma	01	0
Histograma	10	variável
Paragem do AMC	11	0

Tabela 2 – Comandos destinados ao analisador multicanal.

Mais uma vez, se chama a atenção para a subdivisão de cada um dos comandos anteriores, segundo o esquema apresentado no Apêndice A.

O primeiro comando desta série abrange todas as modalidades de arranque do analisador multicanal. Conforme já foi dito, o analisador multicanal pode funcionar em diversos modos

de temporização: por tempo indefinido, até perfazer um determinado número de contagens, ou até expirar um determinado período temporal, definido quer como tempo real quer como tempo vivo de aquisição.

Os diversos subcomandos abrangidos por este comando tratam respectivamente cada uma dessas hipóteses, pondo em andamento a aquisição do AMC no modo especificado. Até ao momento, infelizmente, encontra-se implementada a aquisição por tempo indefinido por dificuldades de hardware na operação dos temporizadores que se espera já estejam sanadas numa próxima versão da placa.

O comando de pausa/retoma suspende temporariamente a aquisição de dados por parte do AMC, até nova invocação do mesmo comando. A função deste comando é possibilitar ao utilizador, por meio de um programa cliente, a suspensão eventual da aquisição de um espectro durante um período em que motivos exteriores o forcem a interromper a experiência.

O comando de histograma abrange todos os subcomandos de pedido de informação ao módulo por parte de um programa cliente do analisador multicanal. De entre os comandos abrangidos, destacam-se aqueles que permitem ao cliente solicitar uma cópia do espectro adquirido até ao momento ou simplesmente de uma fracção desse espectro. É por meio deste grupo de comandos que um dado programa cliente pode actualizar a imagem do histograma que exibe ao utilizador no ecrã. Sempre que haja necessidade de refrescar a imagem, o programa emite um comando de pedido de histograma ao PDS, por intermédio do programa servidor, que lhe responde com a cópia actualizada do conteúdo de cada canal.

Por fim, o último comando desta série provoca a interrupção imediata do processo de aquisição do analisador multicanal. Este comando é usado, em circunstâncias normais, numa aquisição por tempo indefinido quando o utilizador julga chegado o momento de terminar a aquisição ou, a título excepcional, em outros modos de aquisição quando o utilizador decide terminar a aquisição antes de ser atingido o tempo pré-estabelecido de funcionamento.

O Contador Multicanal

No contador multicanal, a estrutura do interpretador de comandos é quase uma cópia fiel da estrutura anteriormente descrita para o analisador multicanal. Os comandos dividem-se, mais uma vez, em comandos de arranque nos vários modos, em comandos de suspensão/retoma, em comandos de paragem e em comandos de pedido de dados.

Os parâmetros de funcionamento do analisador multicanal e do contador multicanal têm uma estrutura ligeiramente diferente, em virtude das diferenças que existem entre os módulos. Essas diferenças serão explicadas nos capítulos relativos aos respectivos programas cliente.

O Analisador de Sinal

Conforme será explicado no capítulo 7, à data da escrita desta tese, a componente de hardware do módulo analisador de sinal não tinha ainda atingido uma configuração estável, estando ainda sobre a mesa diversas propostas de alteração a introduzir no esquema, nomeadamente um circuito de desencadeamento por hardware. Por esse motivo, não tinha ainda sido possível discutir exaustivamente as capacidades previstas para o módulo analisador de sinal.

Das ideias que, entretanto, foram já discutidas, algumas parecem, desde já, ponto assente. Assim, tudo parece indicar que o módulo analisador de sinal disporá de dois modos de funcionamento: modo osciloscópio digital e modo digitalizador de sinal.

No modo de funcionamento como osciloscópio digital, prevê–se que o utilizador possa dispor, na janela de uma aplicação cliente, de uma espécie de osciloscópio virtual, que lhe possa dar uma indicação do formato dos impulsos que penetram na placa. Isso poderá ser útil, por exemplo, em conjugação com outros módulos como o analisador multicanal, pois permitir–lhe–á observar o formato dos impulsos cujo espectro de amplitudes está a ser adquirido naquele módulo. Permitir–lhe–á, inclusivamente, variar alguns parâmetros da formatação do sinal, observando de imediato os efeitos da alteração na forma geral do sinal. Poderá também ser útil o osciloscópio definir um certo período de retenção do sinal. Assim, uma dada forma de onda que seja captada no ecrã do osciloscópio, manter–se–ia no ecrã durante esse período de retenção, independentemente de, entretanto, serem adquiridas outras formas de onda, correspondentes a outros tantos impulsos. Essa capacidade permitirá que, em cada momento, estejam a ser observados os últimos n impulsos que entraram na placa, tendo–se uma ideia qualitativa da variação estatística do formato dos impulsos.

Como é evidente, todas essas capacidades que seria interessante introduzir no modo de funcionamento como osciloscópio dependem da existência de um bom circuito de desencadeamento. Inicialmente, pensou–se em criar um circuito de desencadeamento virtual por software, através da aquisição contínua da forma de onda e da busca, no sinal adquirido, dos pontos em que se satisfazem as condições de desencadeamento. Conforme

se explica no capítulo 7, essa abordagem acarreta dificuldades que põem em causa o bom funcionamento de um osciloscópio.

No modo de funcionamento como digitalizador de sinal, o módulo analisador procura adquirir, a pedido do PDS, um número *n* de conversões do ADC que digitalizem uma dada forma de onda que entra na placa. A frequência de aquisição é controlada pelo PDS e o número máximo de conversões adquiridas depende apenas do tamanho total da memória tampão do módulo, a qual é constituída por dois FIFO que funcionam em alternância, pois nenhum deles suporta individualmente as frequências máximas de aquisição do ADC existente no módulo (50 MHz). Assim, no caso dos FIFO actualmente utilizados, a memória tampão total é de 16 kbytes, mas poderá ser aumentada com o aparecimento de memórias FIFO de maior capacidade.

No modo digitalizador de sinal, as capacidades a incluir no módulo são ainda objecto de discussão. Algumas delas, como a capacidade de calcular a FFT, auto–correlações, etc., é mais ou menos consensual. Outras capacidades de processamento que tirem partido das possibilidades oferecidas pelo processador digital de sinal já terão que ser analisadas à luz da discussão efectuada no capítulo 7 sobre o balanço de carga entre o processador digital de sinal, o processador do computador hospedeiro e os processadores dos computadores que correm aplicações clientes.

Quanto às consequências que tudo isso terá na estrutura do interpretador de comandos do módulo analisador de sinal, elas são ainda objecto de estudo. Um módulo analisador de sinal funciona em escalas de tempo bastante mais curtas do que as de um analisador multicanal ou mesmo de um contador multicanal. Enquanto nestes dois últimos interessa

observar uma distribuição estatística dos impulsos por canais temporais ou de energia, no analisador de sinal interessa a forma de cada impulso individual. Quer isso dizer que o modelo de visualização de dados dos dois outros módulos – com uma aquisição em permanência e pedidos esporádicos de refrescamento dos dados por parte dos clientes – não pode ser directamente transposto para o analisador de sinal.

A comunicação entre o módulo analisador de sinal e cada cliente não pode ser tão anónima como acontece com os outros módulos, em que o módulo se limita a fornecer partes de um histograma pedidos pelo cliente. No caso do analisador de sinal, a informação tem um carácter muito mais dinâmico, variando radicalmente de segundo a segundo, ou menos ainda. Entre cada duas activações do circuito de desencadeamento do osciloscópio, por exemplo, são adquiridas duas imagens em tudo diferentes. Por outro lado, o PDS não se pode limitar a fornecer a que estiver mais actualizada, a mais recente. O utilizador poderá estar mais interessado numa anterior.

Assim, o módulo terá que armazenar, em memória, todos os acontecimentos relevantes que ocorram no seu historial, até que todos os programas clientes que neles possam estar interessados, os tenham recebido numa actualização de dados. Para isso, o PDS terá que manter um registo de todas as aplicações clientes do analisador de sinal que estejam, em cada momento, a olhar para o módulo, para saber que dados cada uma delas já recebeu e quais é que podem, entretanto ser eliminados. Também poderá ter necessidade de manter um registo das informações que são pedidas por cada cliente, a fim de efectuar a respectiva aquisição e guardar esses dados até que o cliente solicitador os vá pedir.

Assim, entre os comandos a processar poderá haver comandos de registo de clientes, comandos de configuração do circuito de desencadeamento, comandos de indicação dos acontecimentos relevantes para cada cliente e que deverão ser guardados até que ele os peça, etc. Paralelamente, poderá haver comandos de pedido de efectuação de operações de processamento sobre os dados (FFT, convolução, correlação, etc.), embora mais uma vez se chame a atenção para a discussão do capítulo 7 sobre o balanço de carga entre o PDS e outros processadores e a pertinência de efectuar esse tipo de operações no PDS.

Este módulo e o respectivo conjunto de operações constitui um tema ainda em aberto.

As Rotinas Servidoras de Interrupções

Conforme foi dito em secções anteriores, toda a aquisição de dados propriamente dita se dá no contexto das interrupções do PDS. Assim, sempre que um módulo de aquisição dispõe de novos dados, alerta o PDS por meio da respectiva linha de interrupção, para que este recolha, trate e armazene esses dados. O momento em que essa interrupção se dá depende, obviamente, do módulo em questão. Assim, enquanto o analisador multicanal chega a interromper duas vezes o PDS para a recolha de um único impulso, o analisador de sinal limita-se a chamar o PDS após ter enchido um FIFO inteiro de dados; no caso do contador multicanal, a interrupção dá-se com uma frequência variável, correspondente ao período que medeia dois avanços de canal sucessivos.

O Serviço à Interrupção do Analisador Multicanal

Dado que o analisador multicanal é, por um lado, o módulo que terá uma maior latência inerente ao longo período de conversão do ADC de 16 bits, por outro, aquele que terá maiores necessidades de processamento de cada impulso individual, devido à implementação do método da escala deslizante, justifica-se amplamente que chegue a

desencadear duas interrupções ao PDS por cada impulso que recolhe. Essas duas interrupções servem para tratar duas fases distintas da conversão de um impulso. Numa primeira fase, logo a seguir à detecção de um pico na entrada do analisador multicanal, o PDS é interrompido para avaliar a oportunidade de converter esse impulso e para preparar o deslocamento da escala deslizante.

Nesta interrupção, o PDS começa por avaliar, por meio da consulta dos temporizadores da placa, se o impulso ainda se encontra dentro dos limites temporais estabelecidos pelo utilizador no caso das aquisições a termo certo (em contraponto às aquisições por tempo indefinido ou com termo associado ao número de contagens). Isso torna-se necessário porque, devido à interrupção de outro módulo, o analisador multicanal poderá não ter sido suspendido a tempo no período normal de processamento do PDS e estar a receber um impulso já depois do fim do prazo. De seguida, o PDS avalia, por meio de comparadores presentes na placa, se a amplitude do impulso se encontra dentro de certos limites pré-definidos e se está aquém ou além da amplitude de meia-escala. Caso a amplitude se encontre fora dos limites, o impulso é imediatamente descartado e o sistema limpo e preparado para a recolha do impulso seguinte. Caso contrário, o DAC700, o qual fornece o valor de deslocamento de amplitude, é inicializado com um valor tal que empurre a amplitude do impulso para baixo, se esta superar a meia-escala, ou para cima, se lhe ficar aquém. Para isso, o DAC 700 está associado a uma fonte de corrente, polarizada em sentido inverso ao do DAC, com uma intensidade equivalente a cerca de metade da amplitude máxima do DAC. Desse modo, a meia-escala inferior do DAC corresponde a débitos totais de corrente negativos por parte do conjunto DAC + Fonte de Corrente, a meia-escala superior corresponde a débitos positivos. Configurando o DAC com um valor

apropriado, consegue-se deslocar a amplitude a que o impulso será convertido para cima ou para baixo, consoante a sua posição inicial. Garante-se assim que o valor a que o impulso será convertido nunca sairá fora dos limites de funcionamento do ADC, o que provocaria perda de informação.

Se o sentido do deslocamento é escolhido com base na análise do comparador de meia-escala, a amplitude seria, em princípio, escolhida por meio de um gerador de números aleatórios. Garantir-se-ia, assim, que cada impulso, independentemente da sua amplitude, teria uma probabilidade equilibrada de ser convertido em qualquer canal do ADC, reduzindo os erros associados à não linearidade diferencial. Infelizmente, não é possível ter um gerador de números aleatórios a funcionar dentro do sistema. Mesmo um gerador de números pseudo–aleatórios implica um certo gasto de computação e nem sempre é fácil garantir certos parâmetros, como uma boa distribuição espectral. Dado que, porém, não se espera que existam grandes correlações temporais dos impulsos analisados por um analisador multicanal deste tipo, não se corre um grande risco se, em vez de um gerador de números aleatórios, se usar simplesmente uma função de rampa que percorra sucessivamente todos os valores de amplitude do deslocamento. Em cada conversão, o valor introduzido na escala deslizante é acrescido de uma unidade, regressando eventualmente novamente a zero.

Após as operações de preparação descritas, o valor da escala deslizante é armazenado, para compensação posterior. Também é activada uma bandeira que informará o PDS, durante a interrupção seguinte do multicanal, de que existe um impulso a ser processado e que, consequentemente, a interrupção seguinte corresponde à segunda fase da conversão, descrita a seguir. Por fim, é dada ordem de conversão ao ADC 700 e termina o serviço à

interrupção. Enquanto o ADC 700 prossegue, em paralelo, o trabalho de conversão do impulso, o PDS fica disponível para quaisquer outras tarefas da placa que exijam a sua atenção.

Na segunda fase da conversão, o PDS é de novo interrompido quando o ADC 700 termina a conversão do impulso. Quando se dá essa interrupção, a respectiva rotina de serviço sabe imediatamente que se trata de uma interrupção da segunda fase, pois na primeira fase foi accionada a bandeira correspondente. Nesta segunda fase, a rotina de serviço tem essencialmente duas tarefas a cumprir: recolher e tratar o valor produzido pelo ADC e limpar os circuitos analógicos para que estejam prontos a receber o impulso seguinte. Esta última tarefa começa de imediato, pois é necessário prever um certo tempo de descarga para limpeza do condensador associado ao circuito prolongador de impulsos. Assim, é ligada uma fonte de corrente que procede à referida descarga e que será desligada no final da rotina de serviço à interrupção.

Quanto à recolha da informação vinda do ADC, o processo desenrola-se em vários passos. Em primeiro lugar, são lidas sucessivamente as componentes mais significativa e menos significativa do valor do ADC. São ambas juntas num registo apropriado do PDS. A esse valor, é depois descontado digitalmente o deslocamento introduzido analogicamente pelo DAC 700 na implementação do método da escala deslizante. Dado que, infelizmente, o ADC 700 e o DAC 700 não têm uma variação equivalente ou, dito de outra maneira, uma variação analógica de uma unidade no DAC 700 não resulta numa variação digital de uma unidade no ADC 700, é necessário introduzir um coeficiente de compensação. Esse coeficiente é calculado durante as operações de calibração do sistema. Da nossa experiência recente, o coeficiente situa-se normalmente ligeiramente abaixo de 0,75. Seja

como for, o valor de deslocamento introduzido e armazenado na interrupção da primeira fase é multiplicado pelo coeficiente de compensação. Nessa operação dá frutos a opção tomada pela utilização de um PDS com possibilidade de cálculo em virgula flutuante, pois permite-nos um cálculo muito mais preciso do valor da compensação. Depois dessa reposição da escala, é feito um escalonamento do valor resultante para o número de bits ao qual o utilizador optou por adquirir o seu histograma. Por fim, é incrementado de uma unidade o canal do histograma correspondente ao valor resultante das operações anteriormente descritas e que nos dá a amplitude estimada do impulso. Para terminar, os circuitos analógicos são repostos no seu estado anterior à entrada do impulso no sistema e o analisador fica pronto para o tratamento do impulso seguinte. Fica completo o processamento da interrupção.

O Serviço à Interrupção do Contador Multicanal

A linha de interrupção correspondente ao contador multicanal é activada uma vez por cada impulso (de origem externa ou interna) de avanço de canal. A sua operação não poderia ser mais simples. Logo após a interrupção, é lido o valor dos contadores, correspondente ao número de impulsos contados durante o último intervalo temporal. A rotina, em seguida, descobre a posição da memória onde é armazenado esse canal e guarda aí o valor lido dos contadores. O processamento fica-se por aí.

O Serviço à Interrupção do Analisador de Sinal

Dado que o hardware do analisador de sinal goza de grande autonomia, podendo adquirir mais de dezasseis mil valores para dentro das memórias FIFO antes de necessitar de processamento por parte do PDS, a respectiva interrupção só se dá quando a acumulação

de valores é já tão grande que o justifique ou, numa futura versão da placa, quando é activado o circuito de desencadeamento por hardware.

Ao ser invocada a interrupção do analisador de sinal, a rotina de serviço correspondente desencadeia um processo de leitura em rajada do número de valores contidos nas memórias FIFO para que tiver sido programada. Dependendo do modo em que se esteja a operar, são vários os desenvolvimentos que se podem tomar a seguir.

- Quando se estiver em busca de um acontecimento específico, representado por uma certa amplitude do sinal, uma determinada transição de nível, etc., o circuito de desencadeamento, a incluir futuramente na placa, já terá sido activado e a interrupção terá resultado dessa activação. A rotina de serviço à interrupção terá que ir ler das memórias FIFO os dados correspondentes ao acontecimento e guardá-los num registo onde as aplicações clientes que estejam interessadas nesse tipo de acontecimento o possam ir consultar.

- Numa utilização em que se pretenda conhecer o aspecto do sinal imediatamente anterior ao acontecimento que provoca o desencadeamento, o FIFO terá que ter associado um circuito que o mantenha permanentemente carregado durante a aquisição, retirando–lhe um valor por cada nova escrita.. No momento em que é localizado o acontecimento de desencadeamento, a aquisição é congelada e a rotina de serviço à interrupção que é, entretanto, chamada, irá ler todos os valores de aquisição que ficaram para trás, armazenados nas memórias FIFO. Poderá ainda haver interesse em observar o que se passou antes e depois do acontecimento relevante. Nesse caso, o circuito associado às memórias FIFO terá que as manter na posição de meio cheias até à ocorrência do

acontecimento de desencadeamento. A partir desse momento, deverá permitir um enchimento normal das memórias FIFO até à sua capacidade máxima. No final, a primeira metade do FIFO conterá os dados relativos ao período que antecedeu o acontecimento, e a segunda metade os dados relativos ao período que lhe sucedeu.

- Numa utilização em modo digitalizador, a interrupção do PDS será invocada sempre que as memórias FIFO, em permanente aquisição, atinjam a posição de meio cheias. Nesse momento, a respectiva rotina de serviço irá ler das memórias FIFO um número de valores igual a metade da capacidade máxima e transferir esses valores para a memória interna da placa, onde poderão ser depois fornecidos às aplicações clientes que neles possam estar interessadas. No casos em que a frequência de conversões do ADC seja bastante inferior à velocidade de leitura das memórias FIFO pelo PDS, poderá ser mesmo possível a aquisição integral de sinais, num modo de gravador digital de sinal. Seria ainda necessário que uma eventual aplicação cliente pedisse os mesmos valores ao PDS a um ritmo tal que não esgotasse a memória interna da placa. Julga-se que esse modo de funcionamento será sustentável até frequências de aquisição da ordem dos 500 kHz, sem pôr em causa o normal funcionamento dos outros módulos da placa.

Capítulo 4

O Programa Servidor

Do lado do PC, o único interlocutor válido com a placa de aquisição é o chamado programa servidor. Trata-se, basicamente, de um programa de gestão do sistema, que fornece ao utilizador uma interface com as operações de manutenção da placa: inicialização, descarregamento do programa PDS na placa, gestão dos portos, etc.

Paralelamente, o programa servidor efectua uma outra importante função, à qual deve o seu nome. Quando instruído para tal pelo utilizador, o programa servidor expõe um canal de comunicação ao exterior, ao qual se poderão ir ligar os programas clientes localizados no mesmo computador ou em qualquer ponto da rede. O meio de comunicação utilizado é o das *tomadas de comunicação.* No entanto, a programação orientada por objectos utilizada na construção do programa permite a fácil substituição desse meio de comunicação por qualquer outro método (pipes, Windows DDE ou outra forma de comunicação entre aplicações). Os motivos que levaram à escolha das tomadas de comunicação foi o facto de serem independentes do protocolo de rede utilizado e do sistema operativo em que o programa corre.

A partir do momento em que o servidor expõe o seu canal de comunicação, qualquer programa cliente com acesso ao outro extremo da tomada poderá ligar-se ao canal, a fim de lhe enviar mensagens segundo um protocolo muito simples, mas altamente eficaz. As

mensagens, depois de recebidas pelo servidor (Ponto 1 da Figura 4), são de imediato encaminhadas para o FIFO de comunicação entre o PC e o PDS (Ponto 2). O programa servidor não efectua qualquer tipo de análise à exactidão da mensagem. É, pois, da responsabilidade do programa cliente formular pedidos válidos que possam ser correctamente interpretados pelo PDS. Este, no decorrer do seu ciclo de monitorização das memórias FIFO, detectará a presença da mensagem do programa cliente que lhe foi retransmitida pelo servidor. Será depois da responsabilidade do PDS descodificar a mensagem e dar-lhe o andamento mais conveniente.

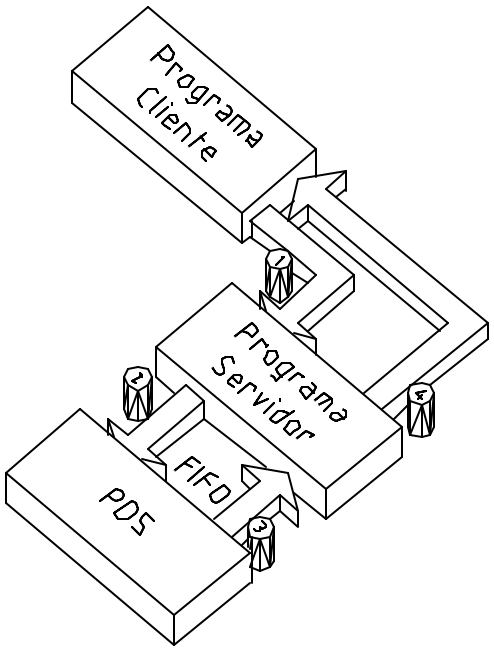


Figura 4 - Esquema da comunicação entre módulos.

Sempre que se trate de mensagens que careçam de resposta, o programa servidor manterá a ligação com o programa cliente que originou o pedido. Manter-se-á depois à escuta do seu próprio FIFO de leitura até que o PDS aí deposite a resposta correspondente ao pedido formulado pelo programa cliente (Ponto 3). Essa resposta será depois enviada através da tomada do pedido original até ao programa cliente que deu início ao processo (Ponto 4).

Após o envio da mensagem por parte do cliente, sempre que se trate de mensagens simples, ou o envio da resposta por parte do servidor, sempre que se trate de mensagens com pedido de resposta, o servidor dá por terminada a ligação com o cliente com o qual

efectuou a transacção. O canal de comunicação fica de novo exposto e pronto a receber novos pedidos de conexão por parte de outros clientes.

Como se pode ver, trata-se de um esquema de encaminhamento de informação muito simples, o que lhe dá uma enorme flexibilidade, mas também altamente eficaz. Permite ainda a existência simultânea de vários clientes de informação a interrogar a placa até ao limite de processamento desta.

A Programação por Objectos

É hoje ponto assente em todo o mundo da programação que a orientação por objectos deve ser a base da concepção de grandes aplicações. Traz vantagens ao programador, às equipas de desenvolvimento, à fiabilidade do código e à reutilização e actualização dos programas.

No campo das modernas interfaces gráficas, em que cada vez os sistemas se tornam maiores e mais complexos, torna-se impensável um único programador criar sozinho todo o código a que recorre o utilizador numa sessão típica frente ao sistema. Assim, cada vez mais funções são transferidas das aplicações para os próprios sistemas operativos, pois não faz sentido cada aplicação recriar independentemente características comuns a todas as aplicações. O próprio modo como as aplicações interagem com o utilizador através do sistema operativo tem vindo a conhecer uma grande padronização. Há, pois, todo o interesse em implementar formas facilitadas de reutilização de código e de apoio à distribuição da tarefa de programação por várias equipas. O código tem de ser suficientemente robusto e estanque para que uma outra entidade, frequentemente sem

contacto com os autores originais, possa tirar partido dele e integrá-lo de forma indolor no seu próprio código.

A programação orientada por objectos vem responder a todas essas exigências. As suas características amplamente modulares e estanques tornam fácil a reutilização de partes do código de uma aplicação em várias outras aplicações. De facto, a divisão dos programas em objectos, além de permitir uma representação do mundo exterior mais próxima da realidade, fornece um método de divisão lógica do código de um programa em unidades auto-contidas. Essas unidades necessitam apenas de conhecer mutuamente as interfaces externas que cada uma expõe. É por isso possível uma muito melhor manutenção do código, pois qualquer alteração necessária resume-se sempre ao objecto no qual ocorre, sendo reduzidas as possibilidades de provocar efeitos secundários em outros objectos. Por outro lado, é possível armazenar bibliotecas de objectos de utilização mais comum, a fim de a eles recorrer em vários programas. Por fim, é mesmo possível criar um mercado de objectos "pré-fabricados" que poderão poupar a um programador preciosas horas de trabalho. O programador terá apenas que recorrer a objectos de código disponíveis comercialmente, para os aspectos mais corriqueiros da sua aplicação, poupando os seus esforços para as características que a irão distinguir de todas as outras já existentes.

Em todos os programas escritos para a plataforma Windows ao longo deste projecto, foi amplamente utilizada uma dessas bibliotecas de objectos disponíveis no mercado. Trata-se da Microsoft Foundation Classes (MFC), na sua versão 3.0. Esta consiste numa biblioteca de classes (uma classe contém a definição geral das características de um objecto) vocacionada para a escrita de aplicações em ambiente gráfico, encapsulando de forma elegante e sistemática cerca de 60% de toda a funcionalidade disponibilizada por um

sistema operativo da família Windows. Foi ainda concebida com vista à portabilidade, pelo que uma aplicação baseada em MFC pode ser facilmente recompilada para Windows de 16 bits (Windows 3.1 e Windows for Workgroups 3.11), Windows de 32 bits (Windows 95 e Windows NT em plataforma 80x86, MIPS, Alpha, PowerPC) e até, com um mínimo de alterações, para a plataforma Macintosh.

Não sendo justificável descrever aqui em pormenor a estrutura intema de classes (em C++, uma classe está para um objecto como, numa linguagem estruturada clássica, um tipo de dados está para as próprias variáveis; pode-se, pois, dizer que um objecto é uma instância de uma classe) da MFC, convém chamar a atenção para uma característica que lhe foi introduzida com a versão 2.0.

O Documento e a Respectiva Vista

Uma aplicação MFC actual gira normalmente à volta de dois grandes objectos: o *documento* (Document) e a respectiva *vista* (View). O documento, cuja estrutura principal é definida pela própria MFC, na classe CDocument, é um repositório de toda a informação manipulada pela aplicação num dado momento. É também esse objecto que contém as principais funções que actuam sobre a informação nele contida. Porém, o documento não dispõe normalmente de quaisquer primitivas para interacção com o utilizador. Essa tarefa é deixada para o objecto vista, cujas principais características são definidas pela MFC na classe CView e que, conforme o próprio nome indica, oferece ao utilizador uma vista dos dados depositados no documento. Tem por isso que ter acesso ao objecto documento, a fim de dele poder retirar a informação necessária para a dar a conhecer ao utilizador. No entanto, o facto de ter acesso ao documento não quer dizer que deva ter acesso à sua estrutura interna, mas apenas à interface exterior que o documento pretenda expor. Desse

modo, o programador é livre de alterar a estrutura interna do objecto documento (ou, mais correctamente, do objecto por ele definido que herda as características da classe CDocument e que, para aquela aplicação específica, desempenha a função de documento), mantendo a interface exterior inalterada, não tendo essas alterações que se reflectir no objecto vista. Essa consideração é válida para a interacção entre quaisquer dois objectos. Uma metáfora interessante é a de um edifício antigo que se pretende remodelar por completo, mas cuja fachada principal tem de ser mantida por motivos históricos e para não desrespeitar o equilíbrio urbanístico do espaço envolvente. Um velho armazém do século XVIII, por exemplo, pode ser transformado num moderno e luxuoso banco por dentro, sem ter que se tocar na fachada e, por consequência, rever o plano urbanístico da rua onde se situa. O mesmo se passa no mundo da programação por objectos, em que se pode rever toda a estrutura interna do objecto, desde que não se toque na sua fachada. Essa facilidade está na base da possibilidade de reutilização de código oferecida pela programação orientada por objectos.

O objecto de vista é responsável, não só por dar a conhecer ao utilizador uma representação gráfica dos dados contidos no documento, mas também por registar as reacções do utilizador a esses dados e outras acções que o utilizador possa querer tomar de sua livre iniciativa. Para isso, o sistema operativo comunica à vista todas as acções empreendidas pelo utilizador (com o teclado, o rato, uma mesa digitalizadora, etc.) e que possam ser de interesse para a aplicação. Perante essas informações, a vista irá invocar comandos (designados por métodos) do documento, os quais irão afectar o seu conteúdo informativo, de acordo com as intenções do utilizador expressas nos comandos emitidos.

O Objecto de Comunicações

A fim de facilitar as comunicações entre aplicações e de cumprir o objectivo acima descrito de torná-las independentes do meio de comunicação utilizado (pipes, tomadas de comunicação, DDE, etc.), foi criada uma classe de objectos de comunicação entre aplicações. O objectivo dessa classe é encapsular todos os pormenores inerentes ao sistema de comunicações utilizado, a fim de que quaisquer alterações futuras na comunicação se restrinjam a uma única classe de objectos. A classe em causa tem o nome de CCommunication e foi integralmente desenvolvida no âmbito deste projecto.

Herança de Classes

Um outro conceito preponderante na programação orientada por objectos e que foi já mencionado anteriormente, é o conceito de herança de classes. Uma das ideias-base da programação orientada por objectos é a de definir classes de objectos que representem de forma estanque os objectos do mundo real. Um objecto que pretenda representar, por exemplo, uma peça de mobiliário, poderá armazenar informações sobre a sua cor, o tipo de madeira utilizado, os acabamentos, etc.

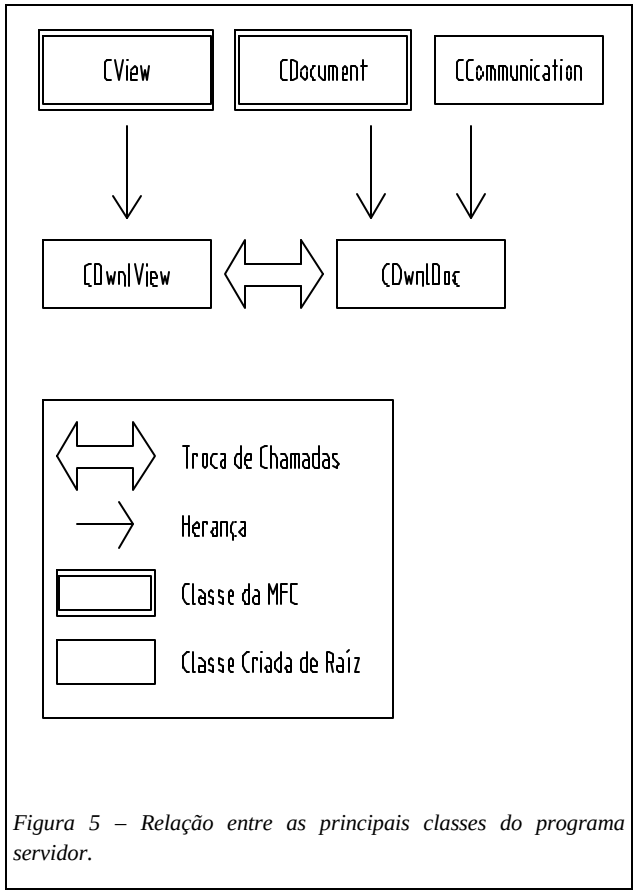


Figura 5 – Relação entre as principais classes do programa servidor.

Acontece que, no mundo real, referimo-nos aos objectos com variados graus de abstracção. A peça de mobiliário do exemplo citado é algo de muito geral, mas se nos referirmos a uma cadeira ou a uma mesa já estamos a concretizar melhor aquilo que pretendemos. Uma cadeira, por sua vez, pode tratar-se de uma cadeira de praia, uma cadeira de sala de jantar, uma cadeira de baloiço, etc. Contudo, uma cadeira de sala de jantar não deixará de ser uma cadeira e, por maioria de razão, não deixará de ser uma peça de mobiliário. Assim sendo, poderemos dizer que uma peça de mobiliário está num nível de abstracção superior ao de cadeira. Podemos também dizer que "cadeira" *herda* todas as propriedades de "peça de mobiliário" – não deixa de ter uma cor e um tipo de madeira, por exemplo –, mas acrescenta-lhe mais algumas ainda. Como propriedades comuns a todas as cadeiras do mundo, podemos pôr a altura dos pés, a área do assento, a existência ou não de braços, etc. Se estivéssemos a definir classes para representar todas essas entidades, poderíamos dizer que a classe "cadeira" herda da classe "peça de mobiliário", pois todas as propriedades da segunda passam também a existir na primeira, mas a recíproca não é verdadeira.

Também em C++ se começam por definir as classes mais abstractas e de propriedades comuns a um maior número de objectos. Depois, a partir dessas classes mais gerais, definem-se as classes mais específicas através de um mecanismo de herança, em que cada classe adquire as propriedades e os métodos das classes de quem descende. No caso concreto dos programas baseados nas Microsoft Foundation Classes, grande parte das classes utilizadas herdam as suas propriedades das classes definidas dentro da biblioteca MFC. Assim, conforme se pode ver na Figura 5, a classe CView da MFC está na origem da classe CDwnlView do programa servidor (o programa servidor tem o nome de

"download", pois a sua primeira função é a de descarregar o programa do PDS na placa). Já a classe CDwnlDoc – que efectua as funções de documento da nossa aplicação – é o resultado de um "casamento" um pouco estranho entre as classes CDocument da MFC e CCommunication, definida de raiz neste projecto. Os motivos que conduziram a essa opção têm a ver com a necessidade de um contacto muito íntimo entre o documento e as comunicações com outros programas. Fazendo o documento da nossa aplicação herdar simultaneamente as características do documento abstracto definido na MFC e da classe de comunicação, conseguimos ter num único objecto as capacidades de armazenamento de dados e de comunicação com outras aplicações. Paralelamente, não se perdem as vantagens da modularidade, pois todas as características de comunicação ficam definidas dentro do objecto CCommunication e as alterações que este sofra não afectarão o funcionamento de CDwnlDoc.

O Controlador de Dispositivo para NT

Poder-se-á nesta altura colocar a seguinte questão: se um objecto do tipo CCommunication (ou um seu derivado) garante as comunicações com as aplicações cliente, quem garantirá as comunicações com a placa de aquisição de dados?

Conforme já se referiu, as comunicações entre o PC e a placa são essencialmente mantidas através de duas memórias FIFO colocados em sentidos opostos entre os respectivos portos e, em menor grau, através de alguns portos de leitura/escrita do PC. Do ponto de vista do PC, tanto as memórias FIFO como os portos se encontram mapeados no respectivo espaço de I/O. Ora, aí surge um problema com certos sistemas operativos. Nos sistemas operativos mais simples como o MS-DOS, o Windows 3.x ou até mesmo o próximo Windows 95, o espaço de I/O encontra-se acessível, não sendo preciso mais do

que recorrer às instruções *in* e *out* do processador 80x86 para ler e escrever naqueles portos. Porém, no Windows NT, em que foi efectuado o desenvolvimento do nosso sistema, as coisas são um pouco mais complexas. Em primeiro lugar, o Windows NT pretende ser uma plataforma independente do processador. Quer isso dizer que o mesmo sistema operativo poderá estar a correr num processador 80x86, num MIPS, num PowerPC ou num Alpha. Cada um desses processadores terá uma filosofia diferente de acesso aos periféricos e uma arquitectura de hardware específica do sistema em causa. Ainda que vários sistemas baseados naqueles processadores possam vir equipados com um bus ISA e ser, portanto, capazes de tirar partido do nosso sistema, cada um deles acede ao bus ISA de maneira diferente. Por outro lado, o Windows NT pretende ser um sistema de alta fiabilidade para as chamadas missões críticas. Simultaneamente é um sistema multiutilizador protegido. Quer isso dizer que, em nenhum caso, poderá uma aplicação posta em execução por um utilizador pôr em risco a integridade do sistema. O sistema operativo tem de ser blindado e à prova de aplicações maliciosas. Ora, a única forma de garantir um tal requisito é impedir em absoluto às aplicações, quaisquer que elas sejam, o acesso aos níveis mais baixos do sistema operativo, nomeadamente os portos de hardware. Para isso, o processador executa as aplicações em modo protegido e qualquer tentativa de execução de uma instrução de acesso directo aos portos resulta na imediata suspensão da aplicação em falta.

Assim sendo, o único modo de um programa qualquer interagir com um dispositivo externo, com um qualquer elemento de hardware, é através da utilização de controladores de dispositivos, ou *device drivers* segundo a terminologia inglesa. Um controlador de dispositivo não é mais do que um segmento de código que, após ter sido instalado por um

administrador do sistema, passa a fazer parte do próprio sistema operativo. Quer isso dizer que esse código executa em modo privilegiado (já pode executar qualquer instrução) e que será automaticamente invocado pelo sistema operativo sempre que haja informação a transmitir para o dispositivo em causa.

Do ponto de vista do programa servidor, a placa de aquisição de dados passará a ser vista como um vulgar porto de alto nível (semelhante ao porto da impressora ou aos portos de comunicação série), com um nome de identificação perante o sistema. Sempre que tenha necessidade de aceder a esse porto, só terá que o abrir como um vulgar ficheiro e ler-lhe ou escrever-lhe, conforme seja necessário. Poderá ainda enviar-lhe sequências de controlo de dispositivo, que serão directamente interpretadas pelo nosso controlador.

A estrutura de um controlador de dispositivo num sistema multiutilizador e multitarefa é algo de muito complexo que não se justifica aprofundar aqui. Dar-se-á apenas uma ideia geral da sua concepção.

Assim, a fim de garantir a portabilidade do sistema para várias arquitecturas de hardware diferentes, o Windows NT inclui uma componente denominada "Camada de Abstracção de Hardware", ou "*Hardware Abstraction Layer* "(HAL) segundo a designação original em inglês. O objectivo dessa camada é isolar o sistema operativo das particularidades de uma plataforma específica. Todo o acesso ao hardware é efectuado segundo um esquema padronizado dentro do NT. Esse acesso é depois convertido pelo HAL num acesso apropriado à plataforma em causa. Desse modo, o próprio controlador de dispositivo usa uma "linguagem" padrão para aceder ao hardware, a qual permite ao controlador ser transportado de uma plataforma para outra com um mínimo de alterações.

Sempre que um programa a funcionar em modo protegido abre um dispositivo, o gestor de periféricos do sistema operativo informa desse facto o respectivo controlador de dispositivo que procederá às necessárias inicializações. A partir desse momento, o programa poderá enviar comandos de controlo ao controlador de dispositivo. Esses comandos poderão, inclusivamente, ser instruções individuais de leitura e escrita num porto específico. De início, em virtude da enorme dificuldade de escrever um controlador de dispositivo, tentou-se uma abordagem simplificada do controlo da placa, na qual os dados seriam transmitidos byte a byte por meio de comandos de controlo. O controlador de dispositivo era absolutamente minimalista e a única acção de que era capaz era a de efectuar essas leituras e escritas individuais em portos de hardware, sob as ordens de comandos emitidos pelo programa servidor. Porém, o facto de ter de efectuar uma chamada ao controlador de dispositivo, por via do sistema operativo, sempre que se pretendesse escrever um byte no porto, tornava a operação muito morosa. Embora o sistema funcionasse, a eficiência era calamitosa. A título de exemplo, a leitura de um histograma do analisador multicanal (cerca de 64k) chegava a demorar 16 segundos. Além disso, havia o inconveniente acrescido de o programa servidor ter de efectuar os envios byte por byte, em lugar de usar transferência s por segmentos de memória, semelhantes aos que se usam quando se lê ou se escreve num ficheiro.

Por esses motivos, julgou-se necessário investir algum tempo na escrita de um controlador de dispositivo completo, capaz de satisfazer pedidos de leitura e de escrita em rajada. O investimento resultou em pleno, pois os resultados foram notáveis. Os mesmos 64 k que antes demoravam cerca de 16 segundos, passaram então para tempos da ordem do décimo de segundo. As possibilidades que se abriram foram imensas, pois a nova largura de banda

passou a permitir facilidades como refrescamentos periódicos completos do histograma no PC, acessos simultâneos de vários clientes, etc.

O Objecto de Acesso ao Hardware

Visto que, conforme dissemos, o acesso à placa em Windows NT é mais complexo do que em Windows 95 – e são essencialmente essas as duas plataformas em que se pretende correr o sistema –, foi necessário pensar em criar duas versões do programa ou simplesmente uma única que actue consoante a plataforma em que estiver a correr. É claro que, em Windows 95, continuaria a ser possível escrever um controlador de dispositivo compatível com o que escrevemos para NT. Desse modo, o programa servidor seria sempre o mesmo e não se aperceberia da diferença de sistema operativo, pois teria sempre por baixo um controlador de dispositivo a aceitar o mesmo tipo de comandos. Porém, o enorme grau de dificuldade envolvido na escrita de um novo controlador de dispositivo aconselha a tirar partido do maior grau de relaxamento do Windows 95 relativamente aos acessos de hardware.

A fim, porém, de reduzir ao mínimo as diferenças entre um programa preparado para correr em Windows NT, outro para correr em Windows 95 ou um outro preparado para detectar a plataforma em que se encontra e actuar de forma apropriada, decidiu-se isolar todos os pormenores do acesso à placa num único objecto de interface exterior, comum a todos os casos. Esse objecto é definido pela classe CPort e encapsula todas as necessidades de interacção com a placa que o programa poderá ter.

Um objecto da classe CPort é definido como variável membro da classe CDwnlDoc. Desse modo, o documento da nossa aplicação passa a conter dentro de si uma "porta" de acesso

à placa, à qual pode aceder de forma transparente. A definição da classe CPort encarrega-se depois de tratar as especificidades de cada tipo de acesso à placa.

A Interface do Utilizador

A apresentação gráfica da aplicação servidora pode ser vista na Figura 6. Conforme se pode ver, a informação apresentada tem, sobretudo, um carácter de diagnóstico. Do lado esquerdo da área de apresentação da aplicação podem ser vistas diversas bandeiras que indicam o estado do servidor e do porto de acesso à placa. As três primeiras indicam se existe um programa de controlo do PDS carregado no servidor, se algum programa já foi descarregado na placa e, por fim, se o servidor se encontra receptivo à ligação de programas clientes. As outras oito bandeiras que se seguem indicam o estado dos diversos bits do porto de acesso à placa. Estas têm sobretudo a ver com o estado de ocupação das memórias FIFO, através das quais o computador comunica com o PDS. É através da consulta dessas bandeiras que o programa servidor sabe qual o momento indicado para enviar ou receber dados da placa.

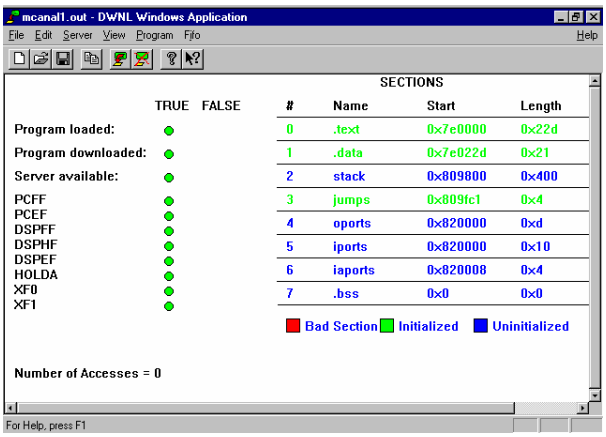


Figura 6 - Programa servidor.

Do lado direito da área de apresentação da aplicação é apresentada uma tabela com as secções que constituem o programa do PDS que se encontra presentemente em memória. As cores características de cada secção indicam o seu tipo perante o PDS.

No canto inferior esquerdo da área de apresentação é indicado o número total de acessos que as diversas aplicações cliente já efectuaram ao servidor desde a sua inicialização.

Os Comandos do Programa

Os comandos postos à disposição do utilizador no programa servidor residem todos no menu da aplicação, embora alguns deles sejam também acessíveis nos botões da barra de ferramentas.

Os comandos presentes no menu de ficheiro são os normais comandos para abrir e guardar um documento e para abandonar a aplicação. Neste caso, o documento será o programa do PDS que se pretende vir a descarregar na placa de aquisição. Esse programa será constituído por diversas secções, as quais, após a abertura do ficheiro, serão descritas na área de apresentação da aplicação.

No menu de edição existe apenas um comando de cópia que permite copiar o conteúdo informativo do ecrã para a área de transferência do sistema.

No menu do servidor encontram-se os comandos que permitem tornar o programa servidor disponível ou indisponível às aplicações clientes. Somente após o servidor ter sido colocado no estado de disponibilidade as aplicações clientes se poderão começar a ligar.

O menu de visualização contém opções para tornar visíveis ou não as barras de ferramentas e de estado da aplicação.

No menu de programa, o utilizador tem acesso aos comandos que lhe permitem descarregar o programa em memória na placa de aquisição e limpar da memória o programa carregado. Note–se que esta segunda opção não afecta em nada qualquer

programa que tenha sido entretanto descarregado na placa, procedendo apenas à limpeza do programa armazenado na aplicação servidora do PC.

A fim de auxiliar nas tarefas de diagnóstico da placa, o programa dispõe ainda de um menu de FIFO que permite, no essencial, ler e escrever manualmente valores isolados nas memórias FIFO de comunicação com a placa de aquisição.

Por fim, o menu de ajuda garante ao utilizador acesso ao sistema de ajuda em linha da aplicação.

Capítulo 5

O Cliente Analisador Multicanal

Conforme foi explicado anteriormente, a função dos programas cliente em geral e do cliente analisador multicanal em particular, neste sistema, é o de constituir uma janela sobre os dados que estão a ser adquiridos na placa de aquisição. Por outro lado, num dado momento, o experimentalista poderá estar interessado em observar várias janelas sobre os dados adquiridos. Poderá querer observar o princípio, o meio e o fim do histograma em diferentes janelas, por exemplo.

Dado que se pretendia uma flexibilidade total, no caso do analisador multicanal, sobre as secções do histograma que poderiam ser visualizadas em cada momento; sobre o grau de ampliação de cada uma delas; sobre o número de diferentes perspectivas do histograma; optou-se, numa primeira fase, por permitir a execução simultânea de várias instâncias do programa cliente. Evitava-se assim a complexidade inerente à gestão simultânea de muitas perspectivas dentro de um mesmo programa. Considerou-se bastante mais simples, para o experimentalista, abrir tantas instâncias do programa quantas deseje e, dentro de cada uma delas, escolher a perspectiva do histograma que melhor lhe convenha.

Tornou-se, pois, imperioso separar em dois programas separados a componente de visualização da componente de comunicação com a placa, para que assim todas as instâncias das aplicações de visualização pudessem ter igual oportunidade de acesso à placa

e aos dados adquiridos. Adoptou-se, pois, uma arquitectura cliente-servidor em que uma só aplicação servidora (discutida no capítulo anterior) comunica com a placa, mas satisfaz os pedidos de dados de uma ou mais aplicações cliente, interessadas em diferentes perspectivas dos dados adquiridos pela placa.

O passo seguinte, uma vez tomada a decisão de separar as componentes de gestão da placa e de visualização dos dados adquiridos, seria escolher o método de comunicação entre os clientes e o servidor. Ora, não acrescentando maior complexidade ao programa, não foi difícil escolher um protocolo de comunicação padrão que ultrapassasse o puro âmbito local do computador que alberga a placa de aquisição. De facto, não existe motivo nenhum para que a aplicação cliente e a aplicação servidora residam forçosamente no mesmo computador, embora muitas vezes assim aconteça. Escolhendo para método de ligação do programa cliente ao servidor as tomadas de comunicação padronizadas (*communication sockets*), disponíveis em todos os sistemas operativos modernos, conseguiu-se, não só uma comunicação muito eficiente entre o programa servidor e um grande número de clientes, mas também a possibilidade de correr as diversas aplicações em computadores separados, ligados apenas por uma rede. Dado que as tomadas de comunicação tratam da mesma forma um endereço de destino no mesmo computador ou em computadores distintos, a localização relativa do programa cliente relativamente ao programa servidor, tornou-se transparente.

Um grande número de possibilidades se abriu, na sequência desta escolha, com várias vantagens para o experimentalista. Por um lado, o exíguo espaço de ecrã do computador que alberga a placa não tem que ser partilhado por vários gráficos correspondentes a cada aplicação cliente. Havendo mais do que um computador disponível, todos eles podem estar

a observar o desenvolvimento de diferentes partes do histograma. Diversos operadores, ou mesmo outras aplicações, podem monitorar diferentes aspectos dos dados que são adquiridos. Além do mais, a monitorização não tem que ser efectuada localmente, podendo ser efectuada remotamente, a partir de um gabinete, de uma área de protecção radiológica, etc., a partir de onde haja uma ligação por rede ao computador que alberga a placa de aquisição.

As Diferentes Componentes do Programa

A estrutura básica do programa cliente do analisador multicanal segue os princípios básicos de uma aplicação baseada nas Microsoft Foundation Classes. Essa estrutura foi já enunciada para o programa servidor, não sendo aqui repetida. Tal como o programa servidor, o analisador multicanal é dividido entre dois grandes objectos: o documento e a vista.

O Documento

O documento da aplicação cliente do analisador multicanal é, antes de tudo, o repositório do histograma adquirido. O histograma em si é armazenado num vector de palavras duplas, definido pelas próprias MFC na classe CDWordArray. Trata–se de uma classe de objectos muito versátil, pois permite manipular um vector de valores de uma forma altamente flexível e com construções de alto nível. O objecto da classe CDWordArray por nós definido para armazenar o histograma é constituído como membro da classe CMcaDoc – a classe documento da aplicação cliente.

O documento da nossa aplicação armazena ainda outra informação de interesse para o seu processamento normal. Essa informação tem a ver com o estado de aquisição do módulo analisador multicanal e com o estado das comunicações com o programa servidor da placa

de aquisição. Porém, a informação fundamental, como é óbvio, será o conteúdo do histograma adquirido até ao momento, pois é essa que será guardada no ficheiro quando o utilizador decidir guardar o histograma para uso posterior.

A Vista

O objecto de vista, como é fácil compreender, terá aqui uma estrutura bastante mais complexa, visto que se trata de um programa eminentemente de visualização de dados. Para começar, o utilizador tem a liberdade de escolher entre dois tipos de perspectivas.

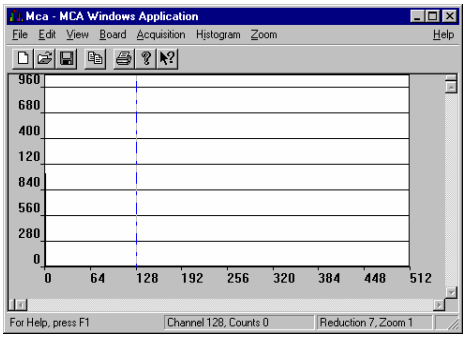


Figura 7 - Vista auto-ajustável.

Uma delas, auto-ajustável, procura mostrar em cada momento a totalidade do histograma actual, ajustando as suas dimensões às dimensões da janela escolhidas pelo utilizador, conforme se vê na Figura 7. O histograma exibe uma espécie de elasticidade que o leva a tomar a melhor forma para tirar partido de toda a área de visualização concedida pela janela. Infelizmente, no caso de histogramas mais compridos, com alguns milhares de pontos, a resolução habitual dos ecrãs de computador não permite mostrar numa única janela a totalidade da informação contida no histograma, ainda que essa janela ocupe a totalidade do ecrã. No entanto, mesmo nesses casos, o modo de visualização auto-ajustável constitui uma boa forma de se ter uma ideia geral do desenvolvimento do histograma no seu todo.

Para os casos em que seja necessária uma informação de maior pormenor, o programa permite um outro modo de visualização "deslizante", cujo exemplo pode ser visto na Figura 8. Neste modo, o grau de ampliação horizontal do histograma é ajustável desde uma largura mínima de 64 pontos até uma ampliação máxima de um canal por ponto do ecrã. Como é evidente, nessas maiores ampliações, deixa de ser possível ver a totalidade do histograma numa única janela, ou mesmo na totalidade do ecrã. Em virtude disso, o histograma passa a ser deslizante. A janela que o contém adquire uma barra de deslocamento horizontal que permite deslizá-lo até à zona que se pretenda observar em maior pormenor. Além dos deslizamentos para a esquerda e para a direita, este modo de visualização permite, como já se disse, que se defina o grau de ampliação com que se pretende observar o histograma. Todos os comandos de ampliação e actualização do histograma encontram-se disponíveis num menu que surge sempre que o utilizador pressione o botão direito do seu rato sobre o gráfico do histograma.

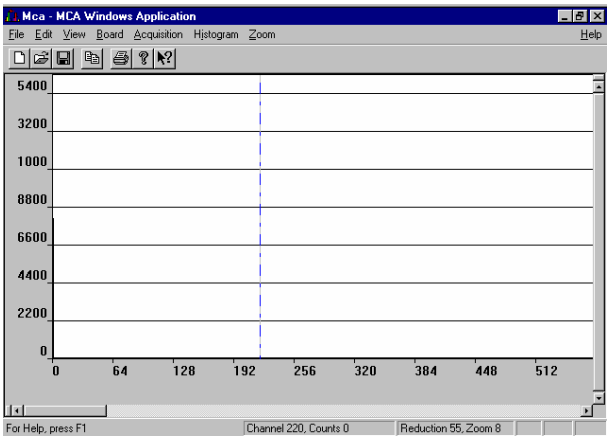


Figura 8 - Vista deslizante.

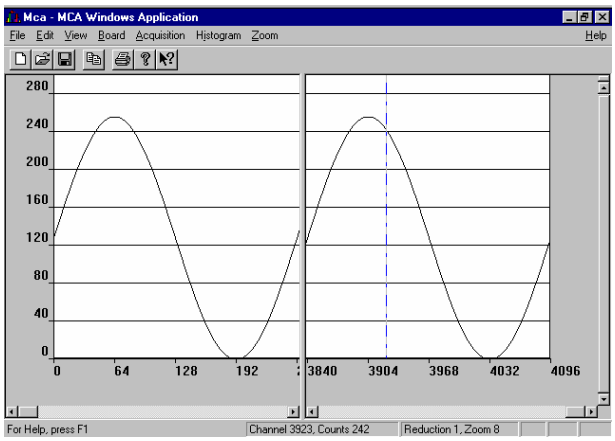


Figura 9 - Desdobramento em duas vistas.

No modo de observação deslizante, é ainda possível dividir a área de visualização de uma única janela em duas secções, para observação simultânea de duas porções do histograma, conforme se pode ver na Figura 9 (note-se que, neste caso, se trata de um histograma simulado, para melhor ilustrar a divisão).

Durante a visualização, um "fio de cabelo" azul designa o canal activo de visualização em cada momento (ver figura de cima). Esse canal activo desempenha duas funções principais. Em primeiro lugar, sempre que o utilizador solicite uma alteração do grau de ampliação do histograma, a nova vista gerada centrar–se–á na posição actual do fio de cabelo. Assim, se o utilizador pedir uma ampliação duas vezes maior (as ampliações possíveis são sempre em factores potências de dois), a nova vista aumentada terá o canal correspondente ao fio de cabelo na sua posição central.

Dado que a largura do fio de cabelo será sempre de um ponto do ecrã e só no grau de ampliação máxima existe uma relação de um para um entre os canais do histograma e os pontos do ecrã, nos graus de ampliação mais baixos, a posição ocupada pelo fio de cabelo corresponderá a mais do que um canal. Nesse caso, haverá mais do que um canal activo. Assim, a segunda aplicação do fio de cabelo será a de dar informações acerca dos canais activos sobre os quais o fio se encontra. Essa informação inclui o número dos canais e o

número total de contagens neles acumuladas e é apresentada na barra de estado do programa, colocada na parte inferior do ecrã.

Deslocando o fio de cabelo ao longo do ecrã, é possível ir tendo uma informação pormenorizada sobre os canais que vão sendo percorridos pelo fio de cabelo.

Os Comandos do utilizador

A generalidade dos comandos disponíveis na nossa aplicação encontra–se disponível dentro dos vários menus.

Alguns comandos mais utilizados podem ser encontrados ainda na barra de ferramentas existente na parte superior da aplicação, conforme se pode ver na Figura 10.

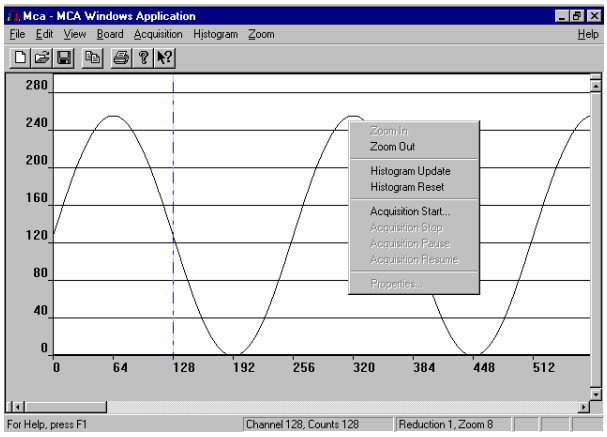


Figura 10 - Barra de ferramentas e menu de sobreposição.

Em qualquer ponto do histograma é possível invocar um menu de sobre-posição com alguns comandos mais frequentemente utilizados, conforme se pode ver também na Figura 10.

Os comandos da barra de ferramentas são, sobretudo, comandos de impressão, antevisão de impressão, etc. Os comandos do menu de sobre posição são comandos de regulação imediata do grau de ampliação, de comando da aquisição e de inicialização do histograma. Como é habitual em aplicações Windows 95, a última opção do menu de sobreposição permite aceder às propriedades do objecto apontado, neste caso o histograma.

Os comandos disponíveis no menu de ficheiro da aplicação são os habituais comandos que permitem criar, abrir, guardar, imprimir ou antever o documento, além do comando universal de saída do programa. A somar a esses, a aplicação acrescenta um comando de envio de correio que, no essencial, permite o envio codificado de uma cópia do documento actual para qualquer endereço de correio electrónico do mundo. Esta função funciona em conjunto com o sistema universal de correio do Windows 95.

No menu de edição aparece unicamente o comando de cópia que permite a cópia da imagem do histograma adquirido para a área de transferência do sistema operativo, onde poderá ser utilizada por qualquer aplicação de edição gráfica.

No menu de visualização, o utilizador pode escolher quais as barras do programa (barra de ferramentas e barra de estado) que pretende manter activas. Pode ainda optar neste menu entre os modos de visualização auto–ajustável e deslizante.

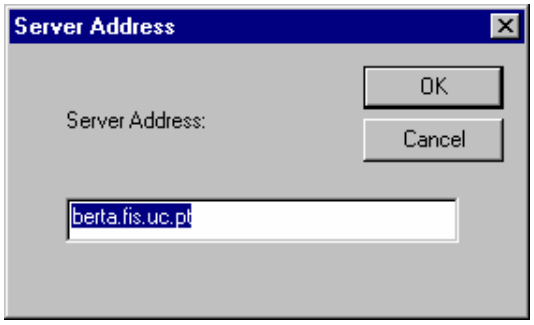


Figura 11 - Caixa de diálogo de ligação ao servidor.

No menu da placa, o utilizador pode estabelecer a ligação desta aplicação cliente do analisador multicanal ao programa servidor que gere a placa de aquisição. Ao invocar o comando de ligação, o utilizador é confrontado com a caixa de diálogo da Figura 11, na qual escreverá o endereço do computador onde executa o programa servidor. De notar que o endereço poderá ser o do próprio computador, nos casos em que as aplicações cliente e servidora correm no mesmo computador. De notar ainda que o endereço poderá ter um âmbito global, no caso em que ambos os

computadores estejam ligados, por exemplo, à internet, ou ter um âmbito meramente local, no caso, por exemplo, de uma intranet.

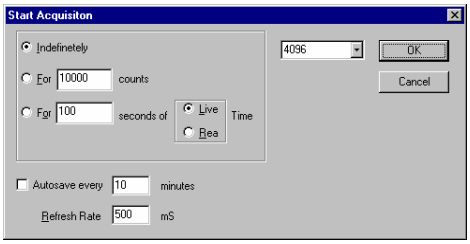


Figura 12 - Caixa de diálogo da aquisição.

No menu de aquisição encontram–se os comandos que permitem ao utilizador iniciar, terminar, suspender e retomar uma aquisição. A opção de iniciar a aquisição apresenta ao utilizador a caixa de diálogo da Figura 12, na qual lhe irão ser pedidos os parâmetros da aquisição.

O primeiro grupo de opções pergunta ao utilizador se pretende efectuar a aquisição por tempo indefinido (até ser invocado o comando de paragem), com termo definido ao fim de um determinado número de contagens, indicado na respectiva caixa, ou com termo definido ao fim de um certo período de tempo. No caso da última opção o utilizador é ainda convidado a indicar se pretende definir um tempo vivo de aquisição ou um tempo total de aquisição e qual a respectiva duração.

O segundo grupo de opções pede ao utilizador que indique a frequência com que quer ver o histograma guardado em disco durante a aquisição, para prevenir a eventualidade de uma quebra de energia provocar a perda de todo o histograma adquirido até ao momento da falha. É ainda pedida a frequência de actualização do ecrã pretendida pelo utilizador. No caso da activação simultânea de diversas aplicações clientes ligadas à mesma placa de aquisição, recomenda–se moderação na frequência de refrescamento do ecrã, a fim de evitar uma saturação do processador da placa. Um tempo médio de três segundos entre refrescamentos poderá ser razoável para a maior parte das aplicações.

No último grupo de opções pede–se ao utilizador que indique o número de canais que pretende ver incluídos no histograma. O número terá que ser da forma 2^n, em que *n* será o número de bits da conversão do ADC que serão aproveitados, num máximo de 16 e num mínimo de 6.

Por fim, ao premir o botão de OK, o utilizador põe em marcha a aquisição do analisador multicanal, através do envio para a placa do comando de início de aquisição, acompanhado pelos parâmetros escolhidos pelo utilizador na caixa de diálogo. Premindo o botão de cancelar, fecha–se a caixa de diálogo sem que seja enviada qualquer ordem para a placa.

No menu de histograma, o utilizador tem a possibilidade de pedir uma actualização extraordinária do histograma com os últimos dados adquiridos na placa. Pode ainda solicitar uma reinicialização do histograma, apagando toda a informação adquirida pelo analisador multicanal até então.

No menu de ampliação, o utilizador pode aumentar para o dobro ou diminuir para metade o grau de ampliação horizontal do histograma. Por outras palavras, pode decidir quantos canais, em potências de 2, serão representados por cada ponto do ecrã, no mínimo de um canal por ponto.

No menu de ajuda, o utilizador tem acesso à ajuda em linha disponibilizada pela aplicação.

Capítulo 6

O Cliente Contador Multicanal

Aplicações do Contador Multicanal

O programa cliente do contador multicanal foi desenvolvido com dois grandes campos de aplicação em mente. São eles o estudo de tempos de meia–vida de fenómenos físicos cuja intensidade segue uma evolução exponencial e o suporte da espectroscopia de Mossbauer.

No primeiro campo acima citado inclui–se o estudo de decaimentos radioactivos, nos quais a intensidade dos decaimentos vai diminuindo exponencialmente com o passar do tempo. Observando a evolução temporal da intensidade dos decaimentos é possível traçar um perfil do processo e, através dele, calcular o tempo de meia–vida associado ao processo nuclear que estiver a ocorrer.

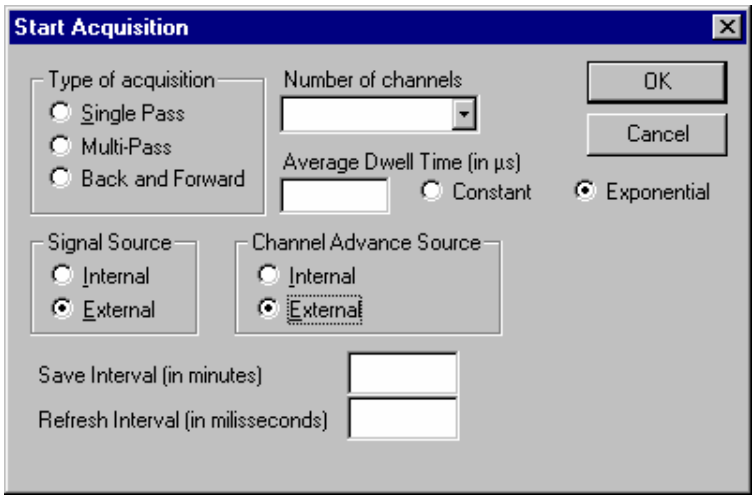
Um dos problemas a enfrentar no caso do estudo de decaimentos exponenciais é o da grande disparidade estatística que existe entre o início e o final de uma experiência, em virtude de o número de contagens nos últimos canais ser exponencialmente menor do que nos primeiros canais. Esse é um problema que, por vezes, se poderá resolver através da utilização de intervalos de tempo não uniformes entre os sucessivos avanços de canal.

No campo da espectroscopia de Mossbauer, um contador multicanal associado ao sistema permite traçar o perfil de intensidades ao longo do ciclo Mossbauer. Esse ciclo, conforme

se sabe, corresponde ao percurso cíclico descrito por um carreto no interior do sistema Mossbauer. Um dos problemas da espectroscopia Mossbauer tem a ver com a fraquíssima estatística (correspondente ao número de contagens por canal) que se consegue atingir ao longo de um único ciclo. A solução para esse problema passa pela necessidade de implementar um sistema de contagem cíclica do número de eventos ocorrido em cada ponto do período Mossbauer. Assim, cada canal do contador multicanal associado a uma experiência Mossbauer ficará sempre associado a uma fase específica do ciclo, contando o número total de eventos que ocorrem nessa fase do ciclo, independentemente do ciclo em que são registados. Desse modo, a estatística da aquisição estará sempre a aumentar com cada ciclo que passa, até ao momento em que todos os canais reunam uma estatística suficiente para que os cientistas possam tirar as suas conclusões. Quer isso tudo dizer que, para funcionar associado a uma experiência de Mossbauer, um contador multicanal terá que implementar um modo de funcionamento circular ou, em alternativa, em vai–vem e o seus circuitos terão que garantir um sincronismo perfeito entre o ciclo de contagens e o ciclo do carreto Mossbauer.

Estrutura Interna do Programa

A estrutura interna do programa cliente contador multicanal é em tudo semelhante à do cliente analisador multicanal. Como em todos os programas de alto nível desenvolvidos neste projecto, é mantida a estrutura do documento e respectiva vista. Também as componentes de comunicação do cliente com o servidor são uma réplica exacta das usadas no analisador multicanal. Não nos iremos, portanto, debruçar neste capítulo sobre as componentes do programa contador multicanal que são semelhantes às do analisador multicanal, mas apenas sobre as que são diferentes.



Na verdade, as únicas diferenças entre os dois programas residem nos parâmetros de inicialização do módulo e nas rotinas de exibição gráfica dos dados adquiridos.

Figura 13 – Parâmetros de inicialização do contador multicanal.

Quando o utilizador invoca o comando de activação do contador multicanal, é aberta a caixa de diálogo da Figura 13, solicitando-lhe os parâmetros da execução.

Parâmetros de Inicialização

A primeira coisa a definir, ao dar início a uma nova aquisição, será se a contagem de impulsos nos diferentes canais temporais será cíclica, em vai–vem ou de uma só passagem. Os primeiros dois casos, como se viu em cima, aplicam-se àquelas experiências em que existe uma equivalência temporal entre dois instantes separados de um dado período, como é o caso da espectroscopia Mossbauer.

O segundo caso, como também já foi visto, aplica-se nas situações em que se pretenda observar a evolução temporal de uma dada fonte de acontecimentos, como seja a evolução da actividade num decaimento radioactivo.

Considerando agora o caso da espectroscopia Mossbauer, conforme vimos em cima instantes equivalentes do ciclo deverão ser contados dentro do mesmo canal no histograma temporal. Assim, nesses casos, o resultado das contagens efectuadas em cada intervalo de

tempo, ao longo do ciclo, vão sendo armazenados em diferentes canais consecutivos, e os resultados de cada ciclo irão sendo sucessivamente adicionados aos resultados do ciclo anterior. Torna-se, pois, evidente a necessidade de um sincronismo absoluto entre os intervalos de tempo que medeiam os sucessivos avanços de canal e o intervalo de tempo próprio de cada ciclo. Ora essa necessidade de sincronismo leva-nos a um outro parâmetro que é necessário considerar no início de uma nova aquisição: a origem (externa ou interna) do sinal de avanço do canal de contagem.

Sempre que não haja grandes necessidades de sincronismo dos contadores com acontecimentos exteriores, a placa dispõe de um conjunto de temporizadores programáveis que são configurados para gerar internamente, com uma dada periodicidade, um sinal de avanço de canal. Nos casos em que seja fundamental sincronizar o histograma temporal com um conjunto de acontecimentos exteriores, tanto a passagem pelo canal zero do histograma como os avanços de canal seguintes deverão ser impostos externamente pelo sistema em estudo, através de duas linhas de sinal ligadas a um porto exterior da placa. No caso concreto do Mossbauer, o sistema controlador da montagem deverá, após o movimento de vai–vem do carrinho ter estabilizado, activar a linha de início de aquisição da placa no momento em que o carrinho passe pelo ponto médio do seu percurso. Depois disso, o mesmo sistema controlador deverá dividir a duração completa de cada ciclo do carreto Mossbauer em tantas partes quanto o número de canais com que se pretende construir o espectro. Ao fim de cada uma dessas fracções do período, o controlador deverá activar a linha de avanço de canal. No caso em que se opte por uma origem interna do sinal de avanço de canal, um dos parâmetros a fornecer ao programa cliente no momento do arranque será o intervalo de tempo entre dois avanços de canal sucessivos.

Em princípio, esse intervalo será constante, para que o número de contagens efectuadas em cada canal seja representativo da actividade no tempo que lhe corresponde. Há, porém, um caso em que será conveniente o intervalo gerado internamente não ser constante. Se considerarmos o caso do decaimento radioactivo exponencial, o número de contagens registadas nos últimos intervalos de tempo será drasticamente inferior ao número registado nos primeiros. Dado que o erro estatístico, no caso da distribuição considerada, é igual à raiz do número de contagens, o erro percentual será muito maior nos últimos canais, pois realizam menos contagens. Conforme foi discutido mais acima, uma solução seria a de aumentar artificialmente o número de contagens dos últimos canais, através de um aumento dos respectivos intervalos de tempo. Assim, se os intervalos de tempo crescerem exponencialmente, o número de contagens em cada canal poderá ser mantido aproximadamente constante ou, pelo menos, a variar apenas linearmente. Uma normalização conveniente das contagens registadas em cada canal, efectuada no final da aquisição, seria suficiente para repor a "verdade" da experiência, mantendo um erro estatístico aproximadamente constante do princípio ao fim do espectro. Essa é mais uma opção que o utilizador terá que tomar no momento do início da aquisição: a utilização de intervalos constantes ou de variação exponencial. É de notar que esta última capacidade não se encontra ainda implementada, pois a actual versão do hardware da placa ainda não a previa. Uma próxima versão da placa terá a capacidade de definição de intervalos de tempo não constantes criada de raiz. Por fim, no caso de o utilizador optar pela utilização de espaçamentos exponenciais, antes de ter início a aquisição será confrontado com uma segunda caixa de diálogo onde definirá os parâmetros da exponencial, nomeadamente a

respectiva constante temporal, a qual deverá ser da mesma ordem de grandeza da constante temporal do decaimento em estudo.

Nos casos em que há necessidade de sincronismo do avanço de canais com qualquer evento experimental exterior à placa – como é o caso do exemplo citado da espectroscopia de Mossbauer –, as opções de definição do espaçamento entre canais não estarão activas, aparecendo antes a sombreado a partir do momento em que o utilizador opta por uma origem externa.

Não é, porém, o sinal de avanço de canal o único cuja origem deverá ser definida. Também a própria origem dos impulsos de contagem tem que ser especificada. Mais uma vez, o utilizador terá que optar entre uma origem interna e outra externa. No caso de optar por uma origem interna, os impulsos que são contabilizados no contador são aqueles mesmos que chegam ao módulo do analisador multicanal. O sinal de entrada de impulso é desdobrado em dois destinos, correspondentes a cada um dos módulos. Dessa forma, é possível uma utilização complementar dos módulos contador multicanal e analisador multicanal, de tal modo que enquanto um traça o perfil energético dos impulsos que penetram na placa, o outro traça o perfil temporal. O cientista ficará, assim, de posse de uma informação mais completa. No caso de o utilizador optar por uma origem externa dos impulsos, o contador multicanal esperará a ocorrência de sinais TTL numa das linhas do seu porto externo, contando a existência de um impulso por cada sinal TTL entrado.

O último parâmetro importante a definir que esteja directamente ligado ao processo de aquisição é o número de canais que se pretende adquirir. No caso de uma aquisição de

passagem única, o termo da aquisição fica automaticamente definido uma vez especificado o intervalo entre avanços de canal e o número de canais que se pretende adquirir.

No caso da aquisição em ciclo ou em vai–vem, o final do processo de aquisição será uma questão deixada em aberto. Visto que não fica definido à partida quantas vezes o ciclo deverá ser percorrido, ficará à discrição do utilizador decidir quando o espectro já possui uma estatística suficiente para o estudo pretendido. Nesse momento, deverá instruir manualmente o contador multicanal para pôr termo à aquisição. Depois de receber esse comando do utilizador, o programa aguardará pelo final do ciclo que estiver entretanto a decorrer e porá então termo à aquisição.

Os Comandos do Utilizador

Devido às grandes semelhanças, já antes referidas, entre as interfaces de utilizador das aplicações clientes do analisador multicanal e do contador multicanal, não fará muito sentido sumariar de novo, neste capítulo, a função de cada comando da aplicação aqui abordada. As diferenças existentes a nível de comandos reduzem–se, essencialmente, aos parâmetros de arranque da aquisição, conforme foram descritos na secção anterior.

Isso não invalida que, de futuro, novos comandos não possam ser acrescentados a cada uma das aplicações, levando–as a divergir uma da outra. Esses comandos específicos de cada programa terão, no entanto, a ver provavelmente com o conteúdo da informação adquirida e permitirão uma exploração mais avançada dos módulos. Terão, em suma, a ver com os "extras" que distinguem um produto comercial dos seus concorrentes e que vão de encontro aos desejos dos utilizadores profissionais. Essas funções extra só poderão resultar de uma discussão interactiva com os utilizadores finais do produto – os físicos experimentais

–, sem a qual se corre o risco de incluir opções supérfluas e deixar de fora outras mais importantes. Considerou–se, porém, que numa primeira versão se deveria construir um sistema mais despido de funções, mas com um núcleo de funcionamento sólido, onde mais tarde se poderá ligar todas as opções de processamento que tirarão partido dos dados em bruto adquiridos pela versão actual. Essa versão básica poderá então servir de base à discussão necessária com os utilizadores finais que, depois de a ensaiarem, estarão mais habilitados a notar as faltas existentes.

Capítulo 7

O Cliente Analisador de Sinal

À data da escrita desta tese, o módulo de analisador de sinal encontrava–se ainda em fase de concepção.

Segundo a ideia original do sistema, uma das funções do analisador de sinal seria a de monitorar o sinal que chega aos outros módulos, nomeadamente o analisador multicanal. Este módulo permitiria, assim, evitar por exemplo a utilização de um osciloscópio exterior para monitorar o formato dos impulsos que chegam ao analisador multicanal. Funcionando, ele próprio, como osciloscópio, seria possível, durante uma aquisição do analisador multicanal, dispor de uma janela sobre o sinal. Com um sinal de desencadeamento apropriado e um tempo adequado de retenção do sinal, seria possível observar simultaneamente vários impulsos entrados, ficando–se com uma ideia da forma geral dos impulsos, conforme se pode ver na Figura 14.

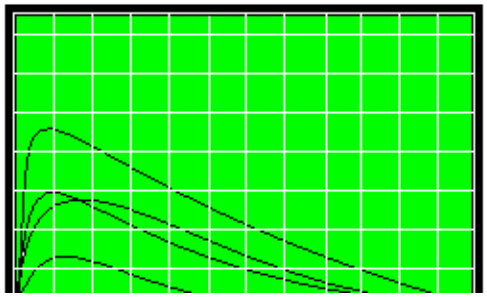


Figura 14 - Retenção simultânea de vários impulsos.

A ideia que inicialmente esteve na base do funcionamento do analisador de sinal seria a de ligar o ADC rápido a um conjunto de duas memórias FIFO que absorveriam, alternadamente, o resultado das conversões sucessivas. Esse artifício torna–se necessário, pois a velocidade de aquisição de um FIFO isolado não lhe permitiria acompanhar o ADC no seu modo de funcionamento mais rápido (50 MHz). Desse modo, a capacidade da memória tampão será o resultado da soma das capacidades das duas memórias FIFO. No momento em que ambas as memórias FIFO se encontrariam cheias, seria gerada uma interrupção ao PDS, o qual procederia então à leitura das memórias FIFO e a um eventual tratamento do sinal que tivesse sido armazenado naquela memória tampão. O circuito de desencadeamento vulgarmente existente em qualquer osciloscópio seria, neste caso, simulado por software. Sempre que se encontrasse definida pelo utilizador uma condição de desencadeamento, o PDS, ao ser interrompido, percorreria rapidamente o conteúdo dos FIFO em busca dos pontos em que fosse satisfeita a condição de desencadeamento. Quando os encontrasse, retiraria os pontos vizinhos da aquisição e transmiti–los–ia, através dos canais normais, para as aplicações cliente que poderiam desenhar o esboço do impulso numa janela do utilizador.

Foi constatada, porém, uma falha nesta estratégia. No caso de aquisições bastante rápidas, não é possível ao PDS ler os FIFO a um ritmo superior àquele a que o ADC neles escreve. Nesses casos, haverá períodos temporais da aquisição que ficarão em branco, pois coincidirão com o período em que o PDS está a analisar o conteúdo dos FIFO em busca de situações de desencadeamento. Poderá acontecer que uma dessas situações de

desencadeamento ocorra precisamente no período de tempo em que a aquisição se encontra suspensa. Nesse caso, o utilizador que confia que o acontecimento de desencadeamento por ele definido não passará em claro pelo sistema, estará a ser traído na sua confiança. O acontecimento poderá, de facto, acontecer e não ser detectado. No modo de aquisição mais rápido, a probabilidade de não detecção poderá ser de 80%, se admitirmos que o ADC é quatro vezes mais rápido a escrever nos FIFO do que o PDS a lê-los e a analisar o respectivo conteúdo, o que se trata de uma estimativa optimista, pois pressupõe que o ADC trabalha ao dobro da velocidade do PDS; o PDS consegue executar uma instrução por ciclo de relógio, sem estados de espera; e duas instruções são suficientes para ler o resultado de uma aquisição e avaliar se a condição de desencadeamento ocorre nesse ponto.

Dado que essa dificuldade constituiria uma importante falha do sistema, concluiu–se que seria forçoso alterar o circuito do analisador de sinal. Assim, a grande alteração a efectuar seria a introdução de um circuito de desencadeamento por hardware, que viesse substituir o circuito de desencadeamento simulado por software.

No essencial, o circuito de desencadeamento consistirá num detector de nível, cujo limiar de desencadeamento poderá ser programado pelo utilizador através da sua aplicação cliente e do PDS. Paralelamente, existirá ainda electrónica capaz de detectar se o nível em questão foi atingido num flanco ascendente ou num flanco descendente. Quando a condição especificada pelo utilizador, constituída pelo nível de desencadeamento e pelo flanco de interesse, for atingida, o circuito de desencadeamento irá pôr em marcha uma ordem de aquisição ao ADC, que o fará adquirir um número *n* de pontos, sendo *n* também definido pelo utilizador. Terminada que esteja essa aquisição, será efectuada uma chamada de

interrupção ao PDS, para que este tenha então a oportunidade de ir ler os valores da aquisição, dando–lhes o seguimento devido.

É claro que, mesmo assim, poderá haver situações em que a frequência de acontecimentos de desencadeamento seja tão grande que, mesmo assim, as memórias FIFO se encham por completo (especialmente quando o número *n* de pontos a adquirir for grande) e surjam, de novo, períodos em branco da aquisição. Nesse caso, poderão surgir acontecimentos de desencadeamento não detectados. Porém, a situação já não será tão grave. Não será tão grave porque, em primeiro lugar, trata–se de uma situação normal nos osciloscópios digitais, que sofrem de um tempo morto entre desencadeamentos, durante o qual não detectam novas ocorrências de um determinado acontecimento; em segundo lugar, sempre que o utilizador estiver interessado em discriminar um acontecimento específico, pressupõe-se que esse terá uma ocorrência única ou temporalmente muito espaçada. Em muitos casos, existe mesmo uma opção de aquisição única, para que o aparelho detecte o acontecimento, proceda à aquisição e se detenha à espera de novos comandos. Nos casos em que o acontecimento de interesse ocorre com elevada frequência, não interessará ao utilizador detectar cada um deles individualmente, mas sim detectar muitos e exibi-los, para ter uma ideia estatística do comportamento de certos impulsos.

Dado que, infelizmente, não existem ainda protótipos da placa de aquisição discutida neste projecto que já incluam o circuito de desencadeamento referido, não é possível prolongar muito mais a discussão deste módulo do sistema.

Quanto ao programa cliente do módulo, também ainda não foi escrito, pois a introdução do circuito de desencadeamento irá alterar em grande medida o conjunto de possibilidades

oferecido por este módulo, o que se repercutirá forçosamente na concepção do programa cliente.

No entanto, várias ideias podem ser já avançadas, sem um grande risco de alteração posterior.

O esqueleto de base da aplicação cliente será, em quase tudo, uma réplica dos clientes analisador multicanal e contador multicanal discutidos anteriormente. A estrutura de comunicação com o programa servidor que, por sua vez, comunica com a placa, não deverá sofrer alterações. Manter-se-á, portanto, o princípio do envio de comandos do programa cliente para o PDS, por via do programa servidor. Sempre que esses comandos impliquem um pedido de renovação de dados, o PDS, mais uma vez, irá ao seu armazém de informação mais recentemente adquirida e responderá ao pedido com os dados mais actualizados de que disponha, respeitantes à área de interesse da aplicação cliente.

Também a estrutura gráfica não sofrerá grandes alterações. Como em qualquer aplicação Windows, a aplicação cliente responderá às mensagens de actualização do ecrã enviadas pelo sistema operativo com uma rotina de exibição gráfica dos últimos dados recebidos do PDS. Periodicamente, a iniciativa dessa actualização poderá partir do próprio programa, a fim de não deixar desactualizar a informação contida no ecrã.

No caso de uma aplicação em modo de osciloscópio digital, porém, as necessidades de actualização dos dados do ecrã poderão ser bastante superiores. Nesse caso, poderá ser mesmo necessário recorrer à actualização contínua. Essa consiste, na criação de um novo fio de execução do programa, cuja responsabilidade será a de efectuar pedidos constantes de refrescamento de dados ao PDS e actualizar a todo o momento a janela do

osciloscópio. Para isso, poderá revelar–se útil recorrer às novas facilidades de acesso directo ao ecrã introduzidas pela Microsoft nos seus sistemas operativos, com o fim de melhorar o desempenho de jogos de acção em Windows. Essas facilidades encontram–se disponíveis num novo conjunto de interfaces de programação denominado WinG e permitem que, de um modo estruturado, mas sem perda de desempenho, uma aplicação crie gráficos muito mais rápidos, que não dependam da pesada estrutura de actualização do ecrã do ambiente Microsoft Windows.

Quanto a outras capacidades a introduzir na aplicação, elas rodam à volta do processamento digital de sinal. O programa cliente do analisador de sinal deverá ter comandos que permitam efectuar, entre outras, uma transformada de Fourier rápida, uma auto-correlação, uma análise espectral de potência etc. sobre qualquer sinal que entretanto seja adquirido.

Balanço da Carga entre Processadores

Coloca–se, entretanto, uma questão importante quanto ao balanço de carga entre o processador digital de sinal existente na placa e os microprocessadores dos computadores que correm as várias aplicações cliente. Com efeito, muitas das operações consideradas na discussão dos vários módulos, mas com especial relevância no caso do analisador de sinal, envolvem uma utilização intensiva do poder de cálculo de um processador. Estão nesse caso operações como a FFT, a auto–correlação, convoluções com diversas formas de onda, etc. Põe–se a questão de saber qual o melhor sítio para efectuar esse cálculo. Se, por um lado, parece óbvio que o PDS é a escolha mais adequada para efectuar cálculos ligados ao processamento digital de sinal, pois toda a sua arquitectura foi optimizada para esse tipo de cálculos, também é verdade que, em plena utilização, o PDS poderá estar já demasiado

sobrecarregado com outras tarefas de aquisição para poder estar a servir todas as aplicações cliente que pretendam ver efectuada uma determinada operação de processamento.

Suponhamos, por exemplo, que a placa de aquisição se encontra a adquirir um espectro no analisador multicanal, com uma frequência de entrada de impulsos bastante elevada. Digamos que, em média, logo após o termo do processamento de um impulso um outro impulso penetra no sistema, de modo que a percentagem de ocupação do analisador multicanal é muito alta. É verdade que o tempo que decorre durante a conversão do ADC700 não obriga à ocupação do PDS. Digamos, portanto, que em cada impulso que entra haverá um tempo útil de processamento correspondente à conversão do ADC e um tempo morto, correspondente ao serviço às interrupções do módulo. Poderemos admitir que, nessa situação extrema, o PDS passará cerca de 20% do seu tempo a servir interrupções do analisador multicanal. Dos 80% que restam, vamos admitir, mais uma vez, a ocorrência de uma situação extrema no contador multicanal, com um tempo relativamente curto entre avanços sucessivos de canal. Sendo certo que a rotina de serviço à interrupção deste módulo tem uma tarefa muito simples a realizar, não ocupando muito tempo de processamento, a verdade é que com um tempo bastante curto entre avanços de canal, a percentagem de ocupação poderá atingir, digamos, os 10%. Ficamos, então, com uma percentagem de ocupação de 30% entre os dois módulos multicanais. Por fim, há ainda a considerar a percentagem de ocupação do módulo analisador de sinal. Ora, numa aplicação hipotética em que se pretendesse uma aquisição contínua ou quase contínua da forma de um sinal, não seria difícil ocupar todo o tempo de processamento restante e, mesmo assim, nos casos de aquisição a uma frequência mais alta, esse tempo não seria sequer suficiente.

Assim sendo, o processador, levado ao extremo, poderia passar o tempo todo exclusivamente a atender interrupções, não lhe sobrando tempo nem para interpretar comandos, muito menos para fazer processamento matemático.

Felizmente, considerámos um caso altamente hipotético em que todos os módulos estariam simultaneamente a funcionar a um ritmo que não se espera que venha a ocorrer numa aplicação real. No entanto, não deixa de ser um facto que uma percentagem algo considerável do tempo de processamento de um PDS poderá passar-se no serviço às interrupções dos módulos. Numa utilização razoavelmente intensiva, poderemos supor que a percentagem de tempo disponível seria sempre superior aos 70%. Ora, esses 70% teriam que ser repartidos entre todas as aplicações clientes que efectuam pedidos ao PDS através do programa servidor no computador hospedeiro. Alguns desses pedidos, como os pedidos de dados adquiridos, serão sempre inevitáveis, pois só o próprio PDS os pode satisfazer. Outros pedidos, porém, como por exemplo a auto-correlação de uma onda adquirida, podem ser igualmente satisfeitos por outros processadores. Tanto é possível pedir ao PDS a FFT de um sinal que adquiriu, como os dados do sinal em bruto, para depois calcular a mesma FFT já na aplicação cliente. Assim, coloca-se a questão do balanço entre o cálculo no PDS e o cálculo na aplicação cliente. Deverá, por exemplo, uma convolução, pedida pelo utilizador na aplicação cliente do analisador de sinal, ser calculada na própria aplicação ou deverá ser pedida ao PDS?

Em favor da primeira hipótese, pesa o facto anteriormente descrito de o PDS já se poder encontrar algo sobrecarregado com o processamento normal da placa e, com um número superior de clientes, acabar por ficar algo emperrado, não dando vazão a todos os pedidos de processamento que lhe chegam.

Em favor da segunda hipótese, pesa o facto de o PDS ser o tipo de processador mais adequado a este género de processamento, a maior simplicidade de arquitectura associada ao processamento centralizado no PDS (constituindo a aplicação cliente uma espécie de terminal onde são emitidos os comandos do utilizador, com exibição gráfica no ecrã do resultado desses comandos).

A polémica entre estes dois pontos de vista, de resto, já há alguns meses que grassa na indústria informática, nomeadamente no campo das aplicações multimédia. De um dos lados, certas empresas apostam cada vez mais na inclusão, nas suas placas de periféricos ou nas próprias consolas do computador, de processadores digitais de sinal dedicados, os quais são usados com os mais diversos fins, desde o processamento audio e vídeo em tempo real até à própria modulação de sinais de linha telefónica, para criação de modems, sistemas de atendimento automático, etc. Existe mesmo uma tentativa, por parte da Microsoft, de criação de uma interface comum a esses processadores de sinal, o que abriria caminho à rápida expansão da indústria, com o aparecimento de aplicações multimédia que não dependeriam do hardware específico de cada fabricante. No campo oposto, outras empresas acreditam firmemente que o próprio processador central do computador tem perfeita capacidade para tratar ele próprio das tarefas que seriam delegadas num PDS, não havendo necessidade de encarecer a placa com um novo processador. É o caso da Apple, por exemplo, que em alguns dos seus modelos equipados com o processador PowerPC optou por não utilizar um PDS dedicado, confiando antes no alto desempenho daquele processador RISC para executar simultaneamente as tarefas normais de gestão do computador e as tarefas ditas multimédia, como a exibição de vídeo no ecrã, tratamento audio, modulação da linha telefónica, etc. Os críticos afirmam que esta última opção

constitui um desperdício, pois se trata da utilização de um processador bastante caro numa aplicação para a qual não está especialmente vocacionado, quando existem outros processadores mais baratos que o estão. A Apple contrapõe que os processadores normais, sendo cada vez mais poderosos e mais baratos, atingiram um tal ponto da relação custo / desempenho que justifica a sua utilização em aplicações que tradicionalmente cairiam no âmbito dos PDS.

No caso presente da nossa placa, a questão assume contornos um tanto diferentes, visto que quer o PDS quer o microprocessador se encontram presentes, não se pondo a questão de os inserir ou não, o debate gira em torno da carga suportada pelo PDS e do tempo disponível que este terá numa aplicação típica. Infelizmente, esses dados só serão conhecidos uma vez completo o módulo analisador de sinal e só então será possível tomar uma de quatro opções:

1. Atribuir ao PDS toda a responsabilidade pelo processamento digital dos dados adquiridos, deixando a aplicação cliente liberta de qualquer trabalho matemático sobre o conjunto dos dados que recebe.

2. Dividir as tarefas de processamento em tarefas cuja execução seja francamente vantajosa no PDS e tarefas que um processador normal possa executar sem grande dificuldade. As primeiras seriam implementadas como comandos dos respectivos módulos no PDS, as segundas seriam efectuadas localmente em cada aplicação cliente.

3. Implementar de ambos os lados os algoritmos de processamento, deixando ao utilizador a possibilidade de optar entre um processamento local na aplicação e um processamento

remoto no próprio PDS. Esta solução poderá também constituir uma forma de avaliar quais as operações cuja realização no PDS é mais vantajosa, com vista à implementação da solução 2.

4. Atribuir à aplicação local toda a responsabilidade do processamento, deixando entregue ao PDS apenas as tarefas já descritas de comando da aquisição e de fornecimento de dados em bruto às aplicações clientes.

Apêndice A

Descrição geral dos Comandos

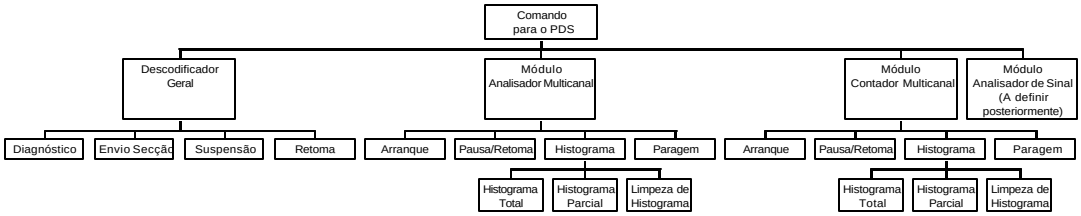


Figura 15 - Comandos destinados aos diferentes módulos da placa de aquisição.

A Figura 15 sumaria os diferentes comandos que poderão ser enviados ao processador da placa de aquisição, pelas diversas aplicações clientes do sistema e pelo próprio programa servidor que controla o funcionamento da placa e lhe retransmite os comandos oriundos dos clientes.

Na tabela seguinte são descritos os vários comandos finais (casas inferiores da figura) e alguns parâmetros úteis desses comandos.

Comando	Módulo Destino	Programa que Envia	Bytes de Parâmetros	Descrição Parâmetros	Tipo Resposta
Diagnóstico	PDS	Servidor	0		Informação geral do estado da placa
Envio Secção	PDS	Servidor	8 + bytes da secção	Endereço destino e tamanho	Não tem
Suspensão	PDS	Servidor	Não tem	Não tem	Suspende funcionamento
Retoma	PDS	Servidor	Não tem	Não tem	Retoma funcionamento
Arranque	Módulo AMC	Cliente AMC	8	Número canais Termo aquisição	Põe aquisição em marcha
Pausa / Retoma	Módulo AMC	Cliente AMC	Não tem	Não tem	Pausa/retoma aquisição
Histograma Total	Módulo AMC	Cliente AMC	Não tem	Não tem	Histograma do AMC
Histograma Parcial	Módulo AMC	Cliente AMC	8	Primeiro canal Número canais	Parte do histograma
Limpeza Histograma	Módulo AMC	Cliente AMC	Não tem	Não tem	Reinicializa histograma
Arranque	Módulo AMC	Cliente AMC	8	Número canais Termo aquisição	Põe aquisição em marcha
Pausa / Retoma	Módulo CMC	Cliente CMC	Não tem	Não tem	Pausa/retoma aquisição
Histograma Total	Módulo CMC	Cliente CMC	Não tem	Não tem	Histograma do CMC
Histograma Parcial	Módulo CMC	Cliente CMC	8	Primeiro canal Número canais	Parte do histograma
Limpeza Histograma	Módulo CMC	Cliente CMC	Não tem	Não tem	Reinicializa histograma

Tabela 3 - Comandos gerais do sistema.

Apêndice B

Correspondência Inglesa de Algumas Expressões Utilizadas

Ao longo desta tese foi feito um esforço para evitar a utilização abusiva de expressões inglesas, infelizmente tão frequente na literatura electrónica e informática portuguesa. Como é evidente, não se procurou substituir termos já tão consagrados e do conhecimento público como "hardware" ou "software". Porém, outros termos há que por serem tão recentes na própria literatura original ou por se prestarem a uma fácil tradução na nossa língua, pareceu despropositado estar a usar um termo estrangeiro.

É, porém, um facto que o hábito de recorrer ao termo original por simples preguiça de pensar na tradução está demasiado enraizado na comunidade técnica portuguesa. Já não nos apercebemos de que empregamos expressões tão cómicas como: "Passa-me daí o file com o query da spreadsheet..." Desse modo, tentar utilizar as traduções naturais de certas expressões pode criar alguma confusão em quem lê, por não estar habituado a encontrar as várias entidades designadas em português. A fim de obviar a essas confusões e de, ainda assim, manter as vantagens da utilização de um linguagem mais natural, decidiu-se deixar em apêndice a correspondência entre certas expressões utilizadas e as suas correspondentes na língua original.

- Ajuda em linha – *Help on–line*.

- Analisador multicanal – *Multichannel analyzer*.
- Analisador de sinal – *Signal analyzer*.
- Área de transferência – *Clipboard*.
- Bandeira – *Flag*.
- Barra de estado – *Status bar*.
- Barra de ferramentas – *Toolbar*.
- Circuito de desencadeamento – *Trigger*.
- Contador multicanal – *Multichannel scaler*.
- Controlador de Dispositivo – *Device driver*.
- Grau de ampliação – *Zoom level*
- Linguagens orientadas por objectos – *Object–oriented languages*.
- Memória tampão – *Buffer*.
- Menu de sobreposição – *Pop–up menu*.
- Tomadas de comunicação – *Communication sockets*.
- Vista – *View*.

Bibliografia

Todo o trabalho discutido ao longo desta tese foi efectuado numa das áreas mais dinâmicas da actual tecnologia humana: o software de computadores pessoais. Não é invulgar neste campo uma informação com mais de um mês estar desactualizada. Todas as semanas surgem novas componentes dos sistemas operativos, novas interfaces de programação, novos protocolos e novas normas. As mudanças são de tal forma dramáticas que, por duas vezes ao longo da escrita da tese, dois capítulos tiveram que ser rescritos por já não serem válidos alguns pressupostos de que partiam.

Escusado será dizer que, de um modo geral, o clássico livro impresso tem pouca utilidade nesta área, pois existe o sério risco de estar desactualizado antes de acabar o percurso que o leva desde o autor até à prateleira de uma livraria ou de uma biblioteca. Por esse motivo, serão muito escassos os livros em papel citados nesta bibliografia.

Por outro lado, apercebendo-se da dificuldade dos programadores em ter acesso a informação correcta e actualizada sobre os últimos desenvolvimentos, certas empresas começaram a implementar programas de distribuição documental sob a forma digital, que vêm resolver dois grandes problemas com que se confrontam os programadores: acesso rápido à informação à medida que ela se encontra disponível e procura automática de conhecimentos no meio da profusão de documentação que é produzida numa área tão dinâmica como a da programação moderna. Aqui, não se pode deixar de destacar o grande papel desempenhado pela Microsoft, produtora dos sistemas operativos mais divulgados no

mundo dos PC, pelos seus programas de suporte à programação. Tendo como hábito distribuir trimestralmente, sob a forma de discos compactos, toda a literatura de apoio à programação dos seus sistemas operativos, as versões mais actualizadas de cada um desses sistemas operativos e, mais recentemente, algumas versões dos seus mais populares produtos comerciais e ferramentas de desenvolvimento, por um preço pouco superior ao preço da própria produção dos discos, esta empresa contribuiu decisivamente para a opção pelos seus sistemas operativos em virtude das facilidades concedidas de acesso à informação.

Também a possibilidade de acesso interactivo à rede Internet constituiu um importante auxílio documental durante a realização do projecto agora descrito. Destacam–se, em particular, as páginas disponibilizadas na WWW pela Texas Instruments, fabricante do TMS 32031 usado na placa de aquisição e, mais uma vez, as páginas da Microsoft que complementavam a informação contida em disco compacto no período que decorria entre duas distribuições sucessivas.

- Microsoft Corporation, *Microsoft Development Library*, Redmond, Washington, April 1994 – April 1996.
- Microsoft Corporation, *Microsoft Developers Platform*, Redmond, Washington, April 1994 ‘ April 1996.
- Microsoft Corporation, *Visual C++ Books On-line*, v. 1.0 (32 bits) – v. 4.0 (32 bits), Redmond, Washington.
- Custer, Helen, *Inside Windows NT*, Microsoft Press, 1993.

- Murray, William H. III & Pappas, Chris H., *Windows 3.1 Programming*, McGraw–Hill, 1992
- Schildt, Herbert, *C The Complete Reference*, 2nd edition, McGraw–Hill, 1990.
- http://www.ti.com
- http://www.microsoft.com/devonly/
- http//www.microsoft.com/msdn/
- http://www.yahoo.com
- Vertes, Attila, L. Korecz, K. Burger, *Mossbauer Spectroscopy*, Amsterdam, Elsevier Scientific, 1979.

Índice de Figuras

Índice de Tabelas