"O PROJETO PEGASUS-32X/PLURIX"

Newton calle:
Manuel Luís Anido
Pedro Seliebeuch
Nivalde Ribeiro Figueira
Gonzo Calo
José Luiz Ribeiro Filho
Carlos Mendes de Azevedo
Luiz Fernando Huet de Bacellar
Elaine Barros Oliveira Paiva
Serafim Brandão Pinto
Chun Yin Hsu
José Antonio Borges
Gabriel Pereira da Silva

Núcleo de Computação Eletrônica da UFRJ
Cidade Universitária – Ilha do Fundão
CCMN – Bloco C – Rio de Janeiro/RJ
Tel: 290-2212, ramal 290

Resumo:

O sistema PEGASUS-32X é uma família de
supermicrocomputadores de 32 bits com múltiplos processadores
baseado na linha 68000 da Motorola. O PLURIX é um sistema
operacional compatível com o UNIX da AT&T, especialmente
projeto para o máximo aproveitamento dos recursos
computacionais do PEGASUS-32X (múltiplos processadores,
espaco de endereçamento de 16MB, processadores de 8/16MB, etc.).

Este trabalho apresenta as características de ambos os sistemas
sob a forma especial aos aspectos relativos à sua concepção e
espansão. Em particular, a importância de integração entre
hardware e software em projetos desta envergadura é também um
tópico deste trabalho.

Palavras-Chave: Superrmicro, Microcomputadores, Micro de 32 bits, Sistemas Operacionais, UNIX, Multiprocessamento, VME, Processadores Periféricos

1. Introdução

O sistema PEGASUS-32X é uma família de
supermicrocomputadores de 32 bits com múltiplos processadores
baseado na linha 68000 da Motorola. O PLURIX é um sistema
operacional compatível com o UNIX da AT&T, especialmente
projeto para o máximo aproveitamento dos recursos
computacionais do PEGASUS-32X (múltiplos processadores, espaço
de endereçamento de 16MB, processadores de 8/16MB, etc.).

Este trabalho apresenta as características de ambos os sistemas
sob a forma especial aos aspectos relativos à sua concepção, teste, homologação e expansão. Em particular, a importância de
integração entre hardware e software em projetos desta envergadura é também um tópico deste trabalho.

2. O PEGASUS-32X

2.1. Descrição Geral

O PEGASUS-32X representa uma família de
supermicrocomputadores, homogêneas, simétricas, de 32 bits,
construídas com diversas unidades de processamento (UCP's) da
familia MC68000 operando em paralelo (multiprocessamento),
Unidades de Processamento Periférico (UPP's) inteligentes para
entrada e saída (E/S), memória global e barramento de
intercomunicação VME bus.

Na fase de concepção levamos em consideração que a construção
de sistemas de multiprocessamento homogêneos quejam em que os
módulos básicos como UCP's, Unidades de Memória (UM) e
Unidades de Processamento Periférico (UPP's) podem ser
compartilhados indistintamente, apresentam uma série de vantagens
sobre outros, tais como:

- O aumento do desempenho (porte) do sistema é imediato, já que
  ele é composto de módulos homogêneos e, portanto, basta
  replicá-los;
- quanto mais homogêneo, mais robusto (menos sujeito a falhas),
  pois ainda restam recursos remelhantes, quando um falha;
- a utilização de um módulo básico em cada parte do sistema
  multiprocessador diminui o custo de projeto, fabricação e
  programação;
- a homogeneidade propicia uma só versão do "software",
  independente do porte do sistema;
- o tempo de desenvolvimento é muito menor, pois o desempenho
  não está relacionado à complexidade dos módulos, mas à sua
  quantidade.

A simetria dos módulos do sistema permite uma melhor utilização
dos recursos computacionais, tanto em operações de cálculo,
como em E/S.

Todas as UCP's, Unidades de Processamento Periférico (UPP's) e
Unidades de Memória (UM's) são interconectadas pela barra global
do sistema (VME bus).

Todas as UCP's e UM's são endereçadas de forma idêntica para todas
as UCP's. Isso significa que um programa pode ser executado em
qualquer um dos processadores, sem qualquer modificação no modo
de endereçar a memória ou as Unidades de Processamento Periférico.

Da mesma forma, todas as UPP's possuem uma visão simétrica das
UCP's e da memória.

Esta simetria é também explorada para melhorar a confiabilidade
do sistema. Quando o mesmo é ligado, cada módulo realiza um
auto-teste e reporta o resultado a um dos processadores, que é
designado como mestre.

Esta UCP mestre inicia a execução do código do Sistema
Operacional, que determina o número de componentes do sistema
que passaram nos auto-testes e configura o sistema, baseado nos
componentes operacionais.

Desde o início do projeto, deu-se uma atenção especial à capacidade
e facilidade de auto-teste de cada um dos módulos do sistema. Isso
provou ser de enorme valia, pois os testes dos módulos foram feitos
em paralelo e independentemente, concorrendo para o rápido
desenvolvimento do sistema.

2.2. Características Gerais do "Hardware"

O PEGASUS-32X consiste de um sistema multiprocessador
homogêneo que permite migrar de sistemas simples para sistemas
complexos, através de reconfiguração e que possui as seguintes
características:

- suporte para memória virtual;
- gerência de memória para relocação e proteção de programas em
  ambientes multi-usuários e multitarefas;
- UCP's da família MC68000, com "cache", para controle do
  Sistema Operacional e programação de usuários;
- instruções de 8, 16 ou 32 bits;
- barramento "VME bus", com caminhos de 32 bits para dados e 24
  bits para endereços;
- taxa de transferência de dados na barra de 6,7 MBps;
- memória com até 16MB para o usuário e sistema;
- "cache" de 4K bytes por UCP;
- velocidade de 1,2 Mips (1 UCP) a 3 Mips (4 UCP's);
- unidades de processamento periférico (UPP's) inteligentes,
  utilizando o microprocessador 250-A;
- as placas são auto-testáveis;
- rápida reconfiguração em caso de falhas, devido à existência de
  diversos processadores e a modularidade do projeto;
- UPP's para discos com "cache" de trilhas, para reduzir o tempo
de acesso e consequentemente aumentar o desempenho.
2.3. Arquitetura do Sistema

Os sistemas da família PEGASUS-32X são formados a partir de múltiplos módulos escolhidos de cada um dos 7 tipos básicos. que são:

- a) Módulo UCP com saída para duas consórias;
- b) Módulo Unidade de Memória (UM) de até 1M bytes;
- c) Módulo UPP para 16 terminais e 2 impressoras (UPPTI);
- d) Módulo UPP para "SASI bus" - permite ligar discos "Winchester", discos flexíveis e fitas do tipo "Streaming";
- e) Módulo UPP para 4 discos SMD de alta capacidade;
- f) Módulo UPP para fitas magnéticas;
- g) Módulo UPP para interfaces especiais, como: "timer", calendário, Processador de Ponto Fluente, etc.

Deve-se observar que uma configuração simples posturária apenas 4 módulos, ou seja: um módulo UCP com console, um módulo de memória com até 1M bytes, um módulo UPP para até 16 terminais e duas impressoras e um módulo UPP para "SASI bus".

Devido à alta homogeneidade do sistema, a evolução da família para a faixa de desempenho dos supermíncos se dá pela replicação dos módulos básicos.

O sistema utiliza como meio de comunicação entre os diversos módulos o barramento "VME bus", o qual possui as seguintes vantagens:

- largura de 32 bits para dados;
- protocolo de comunicação assíncrono, que em conjunto com a largura de barra de dados propiciam alta taxa de transferência;
- é um padrão bastante utilizado, possuindo diversas interfaces disponíveis;
- os conectores apresentam as vantagens de serem mais resistentes a vibrações, serem fabricados no Brasil, serem largamente utilizados em telecomunicações e em controle de processos.

A mesma filosofia de utilizar padrões tipo "VME bus" e "SASI bus" foi seguida nos controladores de disco de alta capacidade, onde se utiliza o barramento SMD.

A Figura 1 ilustra a arquitetura básica do sistema.

2.4. As UCP's

Cada UCP do PEGASUS-32X consiste dos seguintes elementos:

- microprocessador da família MC68000;
- gerência de memória;
- memória "cache" de 4K bytes;
- interface para até 2 consórias;
- relógio do sistema;
- ROM de auto-teste e carga;
- interface com VME bus.

2.4.1. A Gerência de Memória

A gerência de memória realiza a alocação automática de endereços, permitindo a proteção de programas em ambientes multi-usuários e multi-tarefas, sem que isso implique em gasto adicional de tempo.

Características Gerais:
- a gerência de memória tem 3 conjuntos separados de 8 segmentos, um para o modo supervisor e outro para o modo usuário;
- o tamanho de cada segmento pode variar em múltiplos de 256 bytes até 2M bytes;
- permite as seguintes proteções:
  - segmento residente ou não;
  - segmento protegido contra escrita;
  - segmento do tipo STACK ou não;
- suporta memória virtual, operando no modo segmentado para programas até 64K bytes e no modo paginado, para programas maiores que 64K bytes.

2.4.2. Memória "Cache"

O uso de uma memória "cache" em um sistema com múltiplos processadores, possui as seguintes vantagens:

- descongestiona o barramento;
- acelera a execução de programas, pois o microprocessador não precisa de ciclos de "Wait".

O "cache" do PEGASUS-32X possui 4K bytes e é do tipo mapeamento direto, com atualização "Write-Through", ou seja, as operações de escrita, os dados são escritos no "cache" e na memória principal, simultaneamente.

2.5. UPP-SASI – Unidades de Processamento Periférico para SASI-bus

Uma UPP-SASI é composta dos seguintes elementos:

- microprocessador Z80A;
- interface serial;
- controlador de DMA;
- registros de comunicação;
- interface com SASI bus;
- interface com VME bus;
- "cache" de trilhas;
- software residente em ROM.

A UPP-SASI tem como objetivo controlar as transferências de blocos de dados entre a memória principal do sistema e unidades de disco flexível e Winchester. Os comandos dados à UPP-SASI pelo sistema operacional são de alto nível, abstratando-se da característica de hardware das unidades físicas. Um só comando pode transferir até 16M bytes.

2.5.1. "Cache" de Trilhas

É uma memória de até 256K bytes que é usada para armazenar as trilhas mais utilizadas pelo sistema operacional em determinado instante de tempo; com o objetivo de diminuir o tempo médio de acesso aos setores (blocos) dos discos.

Características Gerais:
- pode ser lida e escrita pelo Z80A com acesso direto (permitindo a realização de testes);
- utiliza geração e verificação de paridade por byte.

2.5.2. Software Residente e Interface Serial

A UPP-SASI contém rotinas de teste para todos os módulos. Estas rotinas são automaticamente executadas no início de operação da placa e podem ser-executadas a pedido do sistema operacional ou através do seu console.

Pela console é possível:
- observar os comandos recebidos do sistema;
- observar o resultado do teste em um módulo específico;
- disparar a execução de um comando (como se fosse recebido do sistema);
- executar passo-a-passo os comandos.

2.5.3. Interface SASI-bus

O SASI-bus é uma barra de comunicação de 8 bits, assíncrona, de propósito geral, que permite a comunicação com controladores de periféricos a uma taxa de até 1,5M Bytes/segundo. Além do controle de discos, o SASI-bus pode também ser utilizado para transferências de informação via DMA entre dois sistemas PEGASUS-32X através de suas UPP-SASI.

2.6. UPPTI’s – Unidades de Processamento Paralelo para Terminais e Impressoras

Características Gerais:
- controla 16 linhas de comunicação serial, síncronas/assíncronas;
- admite protocolos orientados a byte ou bit;
- admite interfaces RS232C ou e1o de corrente;
. conta com dois impressores com interface padrão CENTRONIC.
. cada UPT1 é controlada por um microprocessador 2808.
. cada UPT1 apresenta um "front-end" para o PEGASUS-32X,
. pois pré-processa a informação dos canais e a avisa para o UCP.
. pode operar no modo linha (só envia para os UCP's do PEGASUS-
. 32X, após receber um <CR>) ou no modo caracter.

2.7. Unidades de Memória ("UM's")

2.7.1. Descrição Geral

O PEGASUS-32X admite até 16M bytes de endereçamento. Esta
. capacidade é dividida em unidades de memória (placas) que
. até 1M bytes (ou 256K palavras de 32 bits) cada.
. O último mega (15M-16M) é reservado para endereços de
. processadores periféricos, ROM, etc. Portanto, podem existir no
. máximo 15M bytes de memória física.
. Cada unidade de memória possui controle próprio, não existindo
. um controle central, o que propicia ao sistema maior confiabilidade,
. pois mesmo que uma unidade apresente erro, as outras podem
. continuar operando.
. As UM's podem ser configuradas em qualquer endereço físico
. inicial múltiplo de 1M byte, dentro do espaço de endereçamento.
. Para programar uma unidade de memória é preciso escrever um
. endereço válido (diferente do último Mb) no registro de endereço
. de memória desta. Para isto, é preciso selecioná-la, escrevendo no
. registro de seleção das UM's o número de unidade que se quer
. selecionar e/ou seguir, escrever no registro de endereço de
. memória.

3. O Sistema Operacional Plurix

3.1. Descrição Geral

O sistema PLURIX baseia-se na versão 7 do sistema UNIX,
. possuindo diversas melhorias encontradas em outras versões, assim
. como características próprias, onde podemos citar principalmente o
. multiprocessamento. O sistema UNIX foi desenvolvido por K.
. THOMPSON e D. RITCHIE nos laboratórios de BELL
. nos EUA.
. O UNIX é um pequeno grande sistema operacional. É
. suficientemente pequeno para ser entendido por uma única pessoa e
. ao mesmo tempo tem características ainda não encontradas em muitos
. sistemas operacionais de grande porte. É um sistema multi-
. usuário desenvolvido originalmente para a linha PDP-11 da DEC e
. depois transportado para o VAX-11 também da DEC.
. Entre as principais características do PLURIX podemos citar:
. sistema de arquivos hierárquico, incluindo volumes montáveis;
. entrada/saída em arquivos, em dispositivos, e entre processos
. compatíveis entre si;
. ativação de processos assíncronos;
. seleção do interpretador de comands do sistema a nível de
. usuário/aplicação;
. alto grau de portabilidade, sendo a grande maioria do sistema
. escrito numa linguagem de alto nível (C);
. multiplicamento;
. chamadas às funções do sistema ("System Calls") padronizadas.

2.2. O Sistema de Arquivos

Do ponto de vista do usuário, são implementados quatro tipos de
. arquivos: arquivos regulares, diretórios, arquivos especiais e
. arquivos fio.
. Um arquivo regular contém qualquer informação colocada nele pelo
. usuário. Para o sistema estes arquivos não assumem nenhuma
. estruturação. São simplesmente uma sequência contínua de bytes. O
. acesso a estes arquivos pode ser sequencial ou direto e qualquer byte
. do arquivo.
. Os diretórios implementam o mapeamento entre o nome do arquivo
. e o arquivo propriamente dito, e assim descrevem a estrutura do
. sistema de arquivos como um todo. Um diretório pode ter um
. ponteiro ("Link") para qualquer tipo de arquivo. Qualquer arquivo
. que não seja um diretório pode ser apontado por mais de um
. diretório. Existe um diretório especial, reconhecido pelo sistema,
. o diretório "Root", a partir do qual pode ser encontrado qualquer
. arquivo do sistema de arquivos, bastando para tal especificar a
. sequência de diretórios a ser procurada.
. A cada dispositivo físico de entrada/saida (disco, fita, linhas de
. comunicação, impressora, memória física, etc.) está associado um
. ou mais arquivos especiais. Acessando estes arquivos para uma
. operação de E/S, causam na realidade ativação física do dispositivo
. associado. No caso de dispositivos tipo disco, é possível dividir-los em
. várias unidades lógicas diferentes, e ter um ou mais arquivos
. especiais associados a cada uma delas.
. Os arquivos fifo são arquivos os quais têm nenhuma informação
. associada. São utilizados para a comunicação entre quaisquer dos
. processos. Para tal, o processo consumidor deve abrir um arquivo
. fifo para leitura, enquanto que o processo produtor deve abrir o
. mesmo arquivo para escrita. Assim qualquer informação escrita
. no arquivo pelo processo produtor será passada ao processo
. consumidor.

3.3. Processos

Um processo é a execução de uma imagem, o ambiente de execução
. do computador. A imagem consiste basicamente de duas partes: a
. primeira contém várias informações necessárias para a execução do
. processo, tais como os valores dos registros do processador,
. registros de gerência, os arquivos abertos, a identificação do usuário,
. etc. A segunda contém a área de texto do programa sendo
. executado (o código), a área de dados e a "stack" do processo.
. Um processo pode criar um outro processo assíncrono (System
. Call "Fork"). Este novo processo será uma cópia idêntica do
. processo antigo, isto é, imagem dos dois processos será a mesma.
. A diferença entre os dois será o valor retornado pelo System Call.
. Para o processo original (PAI) será retornado um número único
. identificando o processo criado (processo FILHO). Para o processo
. filho será retornado o valor zero.
. Um processo pode substituir sua área de texto, dados e "stack"
. através do System Call "Exec". Este System Call recebe como
. parâmetros o nome do programa a ser executado e os seus
. parâmetros. Assim, todo o texto e dados do processo são
. substituídos pelo conteúdo do arquivo, mas a primeira parte da
. imagem não é alterada. É bem observar que o processo nunca
. volta a executar o código que deu o "Exec", a não ser em caso de
. erro na chamada ao System Call.
. Um processo pode esperar a fim de execução de um dos seus
. processos filho através do System Call "Wait". Este System Call
. retorna a identificação do processo filho que terminou, assim como
. o status de terminação. Este status é passado pelo processo filho ao
. processo pai como parâmetro do System Call "Exit" que tem como
. função principal terminar um processo.
. Para a comunicação entre dois processos "parentes" existe também
. o mecanismo chamado "pipe". Um pipe é um canal de
. comunicação, que, como um arquivo fifo, permite a um processo
. escrever informação de um lado para ser lida por um outro processo.
. Para utilizar um pipe o processo deve criá-lo através do System Call
. "pipe", que será passado ao processo filho através do System Call
. Fork, já que um pipe é tratado como um arquivo comum.

3.4. Controle de Entrada/Saída

Os pedidos de E/S dos usuários passam por uma interface usuário-
. sistema operacional, sendo analisados e distribuídos a rotinas
. específicas de E/S (drivers) para os diversos periféricos do
. sistema.
. Existem dois tipos básicos de rotinas de E/S: os que tratam
. periféricos tipo bloco, como discos, e os que tratam periféricos tipo
. caracter, como terminais.
. As primeiras maneiram um conjunto comum de buffers, de
. tamanho fixo, que contém os dados. Através de "cabecas de
buffere' encadeadas, são mantidas listas de buffers associados a cada periférico e uma de buffers disponíveis.

As que tratam periféricos tipo carater também manuseiam filas formadas pelo encadeamento de pequenos blocos, alocados e desalocados quando necessário, que contêm os caracteres.

Os periféricos são identificados por dois obérgos: um seleciona seu drive, através da entrada correspondente numa tabela de configuração do sistema. O outro é passado ao driver, para selecionar a unidade de um controlador, por exemplo.

3.4.1. Driver de Disco do Plurix

Essa roota foi elaborada de modo a atender a discos de características semelhantes, mas não obrigatoriamente idênticas.

Assim, Winchester e discos flexíveis, de várias dimensões, inclusive, são tratados por ele como várias unidades. Através de uma tabela, cada unidade é identificada (para o SASI-SUS) e dimensionada (setores/cilindro, setores/trilha).

Cada unidade, por sua vez, pode ser logicamente dividida em pseudo-devices, também inicializados por outra tabela (início e tamanho dos discos lógicos).

A roota mantém para cada unidade uma lista de pedidos pendentes.

As várias tarefas desempenhadas por ela são: a validação do pedido, sua colocação na respectiva fila (ordenada por cilindro) o início real da operação pedida, a retirada do pedido da fila (após a interrupção da operação), a verificação de erro na operação e a informação do resultado à interface do sistema.

3.4.2. Driver de Consolé do Plurix

A roota da consolé atende até duas unidades.

Opera sobre uma estrutura rígida definida para cada terminal, onde estão alocadas as filas de entrada e saída para os caracteres, o endereço físico do periférico, um ponteiro para a roota de início de operação física e os caracteres de controle default.

A função básica dessa roota é o recebimento ou envio do caracter a uma outra roota do sistema, comum para qualquer tipo de terminal, que então tratará os caracteres especiais ou armazenará os comuns.

3.5. Multiprocessamento

Cada UCP do sistema recebe uma identificação única (através de chaves). A UCP marcada como 0 tem funções diferenciadas, tanto a nível de software como de hardware. A nível de software, é a UCP0 que inicializa o sistema, executa o Kernel do sistema e atribui as funções às demais UCPS.

É função da UCP0: tratar os periféricos, ativando-os e atendendo suas interrupções; gerenciar os recursos do sistema, isto é, o processador, memória, área de Swap, etc.; executar os Systems Calls dos processos usuários, ou seja, a UCP0 realiza todas as funções do Kernel do PLURIX e também pode executar programas de usuários, caso seja configurado da forma.

Todas as outras UCPS têm como função a execução dos processos usuários. Para tal, a UCP0 passa para outras UCPS as informações necessárias para a execução de um processo, informações que basicamente são a ponteira para a imagem do processo. Ao terminar sua função, a UCP0 passa para a UCP0 informações indicando quando a execução terminou, isto é, System Call, erro no process, etc.

Toda a comunicação entre a UCP0 e as outras UCPS, incluindo o controle de um processo pelo UCP0, é feita via interrupções e pelos registradores de comunicação existentes nas placas das UCPS.

3.6. Interface entre o Sistema e o Usuário

O PLURIX apresenta os mesmos comandos e utilitários básicos do UNIX, além de suas principais opções. O principal interpretador de comandos ("Shell"), possui as mesmas características básicas, que são: execução de programas com parâmetros; redirecionamento de entrada e saída; execução de arquivo de comandos.

3.7. Inicialização do Sistema

Após o término do auto-teste, a UCP0 entra na rotina de carga, que se encontra em ROM, enquanto que as outras UCPS entram numa rotina de espera de um comando da UCP0.

A rotina de carga é bastante inteligente, no sentido de poder efetuar a carga de qualquer dispositivo do sistema (Disquettes, Winchester, etc.). Esta rotina também reconhece o sistema de arquivos do SO e pode carregar qualquer programa que se encontre nos dispositivos do sistema. Para tal, a rotina espera que o operador tecle no console o nome do arquivo e o periférico onde este se encontra.

Carregado o SO., este inicializa suas tarefas e força a criação de um processo, o qual executará o programa "init". Este processo, que já é um processo normal, disparará vários processos filhos, um para cada terminal do sistema. Cada um destes processos ficará esperando o usuário teclar um código de acesso ao sistema válido. Recebendo este, o processo executará o interpretador de comandos associado ao usuário. Terminando a sessão do usuário, o processo inicial será informado e automaticamente disparará outro processo de identificação para este terminal.

3.8. Ferramentas de Depuração/Teste

3.8.1. Introdução

Na fase inicial do desenvolvimento de um sistema baseado em microprocessador é de extrema importância a existência de um conjunto de ferramentas de diagnósticos e de depuração baseados em software à disposição do equipe dos projetistas do hardware do sistema.

O esquema de diagnosticar e depurar o hardware do sistema adotando o software como o seu instrumento principal apresenta uma série de vantagens, sendo a mais importante o fator tempo de depuração ser reduzido.

Tendo em vista não tornar o desenvolvimento das ferramentas de diagnósticos e de depuração parte do caminho crítico do projeto, definiu-se um programa de diagnósticos e um depurador contendo os elementos essenciais para assistir à fase inicial da depuração da UCP do PEGASUS-32X.

3.8.2. Programa de Diagnósticos

Este programa tem por objetivo testar os elementos vitais da UCP do PEGASUS-32X, parte por parte, e de forma autônoma. Cada roota que compõe o programa, tem por função diagnosticar um dos elementos vitais da UCP: os testes realizados verificam o mínimo necessário para certificar o funcionamento do elemento, não tendo a finalidade de explorar todas as combinações e variações para submeter o elemento testado. Os testes mais sofisticados são carregados de disco flexível.

Na ocorrência de algum erro durante a execução de um teste, um loop específico para o teste em execução é acionado, permitindo a equipe dos projetistas de hardware acompanhar o evento através de medições apropriadas.

São as seguintes as principais rotinas que compõem o programa:
1) Rotina de teste do terminal;
2) Rotina de teste dos registros de relocação;
3) Rotina de teste do "cache" como memória;
4) Rotina de teste do "cache";
5) Rotina de teste de gerência de memória;
6) Rotina de teste de timer e da interrupção.

3.8.3. Depurador da UCP do PEGASUS-32X

O Depurador tem por objetivo tornar mais dinâmico e flexível o teste da UCP como um todo. Usando o cache como memória, programas simples podem ser carregados e executados para acompanhar a depuração do hardware.

São as seguintes as funções disponíveis no Depurador:
1) Função para alterar byte, word e long-word;
2) Função "offset";

3) Função loop leitura;
4) Função loop escrita;
5) Função "dump";
6) Função "fill";
7) Função "move";
8) Função "go";
9) Função aritmética hexa.

4. Integração "Hardware-Software"

Desde o início do projeto, foi de importância fundamental a interação entre as equipes de "hardware" e de "software" para realizar uma tarefa desta envergadura. Desta interação podemos citar alguns itens, tais como:
- escolha das famílias de microprocessadores a utilizar;
- arquitetura do sistema;
- análise do conjunto de instruções para poder suportar as características do PLURIX;
- tratamento de erros do sistema;
- definição do sistema de gerenciamento da memória;
- definição dos "drivers" de entrada/saída.

5. Dificuldades e Perspectivas

As principais dificuldades foram, e são, como geralmente ocorre nas universidades, de ordem material e financeira. No entanto, foi possível implementar, em tempo muito curto, um primeiro protótipo e provar que é possível construir no Brasil supermicrocomputadores muito potentes, dispensando qualquer importação de tecnologia.

As perspectivas para o PEGASUS-32X são no sentido de se obter um desempenho maior ainda, desenvolvendo módulos que permitam, entre outras coisas:
- interligação de diversos PEGASUS-32X numa rede de alta velocidade (sistemas "loosely-coupled");
- interligação de diversos PEGASUS-32X pela expansão do barramento paralelo de cada um (sistemas "tightly-coupled");
- aceleração da velocidade de comunicação no barramento VME.

Uma vez implementados os módulos citados, esperamos atingir um desempenho equiparável aos atuais sistemas de grande porte estrangeiros.

Bibliografia

DIGITAL EQUIPMENT CORPORATION – PDP 11/70 Processor Handbook, 1V, Massachusetts, USA, 200 p., 1976;
MOTOROLA SEMICONDUCTOR PRODUCTS INC. – Motorola Microprocessors Data Manual, 1V, Austin, USA, 450 p., 1981;
<table>
<thead>
<tr>
<th>SISTEMA</th>
<th>MIPS</th>
<th>BUS</th>
<th>MEMÓRIA MIN/MÁX</th>
<th>Nº DE UCP'S</th>
<th>CAPACID. CACHE</th>
<th>MÁXIMA TRANSF. DE DADOS</th>
<th>SISTEMA OPERAC.</th>
<th>ACESSO AO CACHE</th>
<th>CICLO DE CPU</th>
<th>SISTEMA BÁSICO/ PREÇO</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM 4331-M32</td>
<td>-</td>
<td>32</td>
<td>1M/4MB</td>
<td>1</td>
<td>8KB</td>
<td>-</td>
<td>ECPS:VSE</td>
<td>200ns</td>
<td>900ns</td>
<td>UCP + 0,5MB MEM + CONTROL - 8 TERMIN + PFF - US$ - 60,000</td>
</tr>
<tr>
<td>IBM 4341-2</td>
<td>1.1</td>
<td>32</td>
<td>2M/16M</td>
<td>1</td>
<td>16K</td>
<td>-</td>
<td>ECPS:VSE</td>
<td>-</td>
<td>240ns</td>
<td>UCP + 2MB MEM + 8KB CACHE + CONTROL 4 TERMIN + PFF - US$ - 205,000</td>
</tr>
<tr>
<td>DEC VAX-11/750</td>
<td>.75</td>
<td>32</td>
<td>256K/2M</td>
<td>1</td>
<td>4K</td>
<td>13.3MB/S</td>
<td>VAX/VMS</td>
<td>400ns</td>
<td>400ns</td>
<td>UCP + 0,5MB MEM + 4KB CACHE + 2 DISCOS 28MB + CONTROL - 8 TERM + CONSOLE + FITA TUS8 - US$ - 90,000</td>
</tr>
<tr>
<td>DEC VAX-11/780</td>
<td>1.06</td>
<td>32</td>
<td>512K/8M</td>
<td>1</td>
<td>8K</td>
<td>13.3MB/S</td>
<td>VAX/VMS</td>
<td>290ns</td>
<td>600ns</td>
<td>UCP + 1,5MB MEM + 8KB CACHE + DISCO 176MB + FITA 125ips + CONTROL 8 TERM. - US$ - 260,000</td>
</tr>
<tr>
<td>NCE PEGASUS-32X</td>
<td>1 (UCP)</td>
<td>32</td>
<td>256K/16M</td>
<td>MIN-1 MAX-4</td>
<td>FOR UCP</td>
<td>4KB</td>
<td>PIURIX</td>
<td>150ns</td>
<td>320ns</td>
<td>-</td>
</tr>
</tbody>
</table>

TABELA 01 - COMPARAÇÃO DAS CARACTERÍSTICAS DO PEGASUS - 32X
COM OUTROS SUPERMINIS. FONTE - AUERBACH

* - ESTIMADO.
FIG. 1 - ARQUITETURA BÁSICA DO PEGASUS-32X