Descrição de uma Unidade Processadora Discreta Microprogramável

EDNA MIE KANAZAWA
WILIAN SOARES LACERDA

UFLA – Universidade Federal de Lavras
DCC – Departamento de Ciência da Computação
Cx. Postal 37 – CEP 37.200-000 Lavras (MG)

kanazawa@comp.ufla.br
lacerda@ufla.br

Resumo: Neste artigo é descrito o desenvolvimento de um protótipo de um processador utilizando-se circuitos integrados comerciais. O princípio da arquitetura de von Neumann é empregado no projeto do processador, usando um barramento de dados e outro de endereço, ambos de 8 bits. Alguns registradores são empregados para endereçamento e armazenamento de dados, sem aumentar a complexidade do projeto que é bastante simples. As instruções do processador são executadas via microprograma gravado em EPROM's, o que torna o sistema bem flexível para implementação de novas instruções.

Palavras Chaves: processador**,** microprograma, microprocessador, memória, CPU, barramento

1 Introdução

A grande maioria das arquiteturas de processadores digitais atuais ainda segue a arquitetura dos primeiros computadores eletrônicos, ou seja, a arquitetura de von Neumann. Este artigo descreve o desenvolvimento de um protótipo de um processador com uma arquitetura simples e de fácil entendimento, com finalidade de estudo e o aprimoramento de novas técnicas de projeto de processadores.

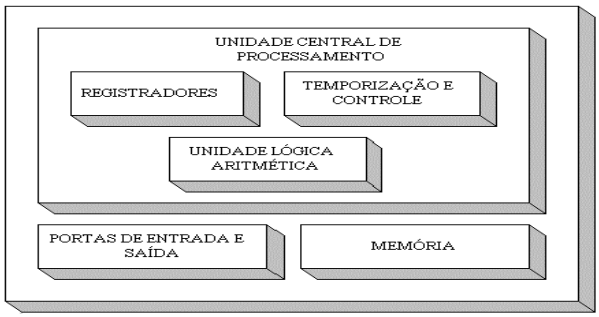

Figura 1: Componentes básicos de um computador

Na Figura 1 são mostrados em blocos as partes básicas que constituem um computador. Em geral, o computador constitui-se de 3 partes: Unidade Central de Processamento (CPU), Portas de Entrada/Saída e Memórias [4]. A CPU gerencia todos os componentes auxiliares do computador (portas de entrada/saída e memória).

A CPU contém elementos de armazenamento (registradores), Unidade Lógica Aritmética (ULA), e Circuitos de Controle e Temporização.

As funções primárias da CPU são:

- Buscar, decodificar, e executar instruções do programa armazenado na memória;
- Transferir dados entre a memória, e entre portas de entrada e saída;
- Prover os sinais de controle e temporização para todo o sistema.

A Unidade Lógica Aritmética (ULA) executa operações com os dados como: somar, deslocar, rodar, comparar, incrementar, decrementar, negar, AND, OR, XOR, complementar e zerar.

Dentre os registradores, existe o registrador Acumulador (ACC) e o registrador Temporário (TEMP), responsáveis pelo armazenamento dos dados manipulados pela ULA. Durante uma operação da ULA o conteúdo do Acumulador e do registrador Temporário

são tomados como operandos, e o resultado é armazenado no Acumulador.

De grande importância para o programador é o registrador de código de condição dos Flags. Os Flags são sinalizadores do resultado fornecido pela ULA. Incluem indicadores de resultado nulo, de resultado negativo, de vai-um, e de igual. Os Flags são usados para tomadas de decisão quando se usam instruções de desvio.

A Unidade de Controle e Temporização é a parte mais complexa e afeta todos os eventos da CPU no computador [4]. As atividades da CPU podem ser divididas em estágios de busca, decodificação e execução [2]. Durante o ciclo de busca, a CPU lê um endereço da memória onde está armazenado uma instrução para ser executada. Esta instrução é armazenada em um Registrador de Instruções. No ciclo de decodificação, a instrução é reconhecida dentre as diversas instruções existentes. E finalmente, no ciclo de execução, a unidade de controle se encarrega de gerar os sinais necessários para execução da instrução. A execução da instrução envolve a chamada de um microprograma que contém os passos seqüenciais da instrução. A seqüência de busca-decodificação-execução de instrução é fundamental para a operação do computador [2].

Existe um conjunto de instruções que o processador pode executar [4]:

- Instruções aritméticas: somar, subtrair, incrementar, decrementar, comparar e negar;
- Instruções lógicas: AND, OR, XOR, NOT;
- Instruções de transferência de dados: carregar, armazenar, mover, zerar;
- Instruções de desvio: desvio incondicional, desvio de zero, desvio se não zero, desvio se igual, desvio se desigual, desvio se positivo, desvio se negativo;
- Instruções de chamada e retorno de sub-rotina;
- Instruções de pilha: empilhar, desempilhar.

2 Funcionamento

Na Figura 2 é mostrado o diagrama em blocos do protótipo do processador, objeto deste artigo. Em seguida é descrito o funcionamento de cada parte do processador.

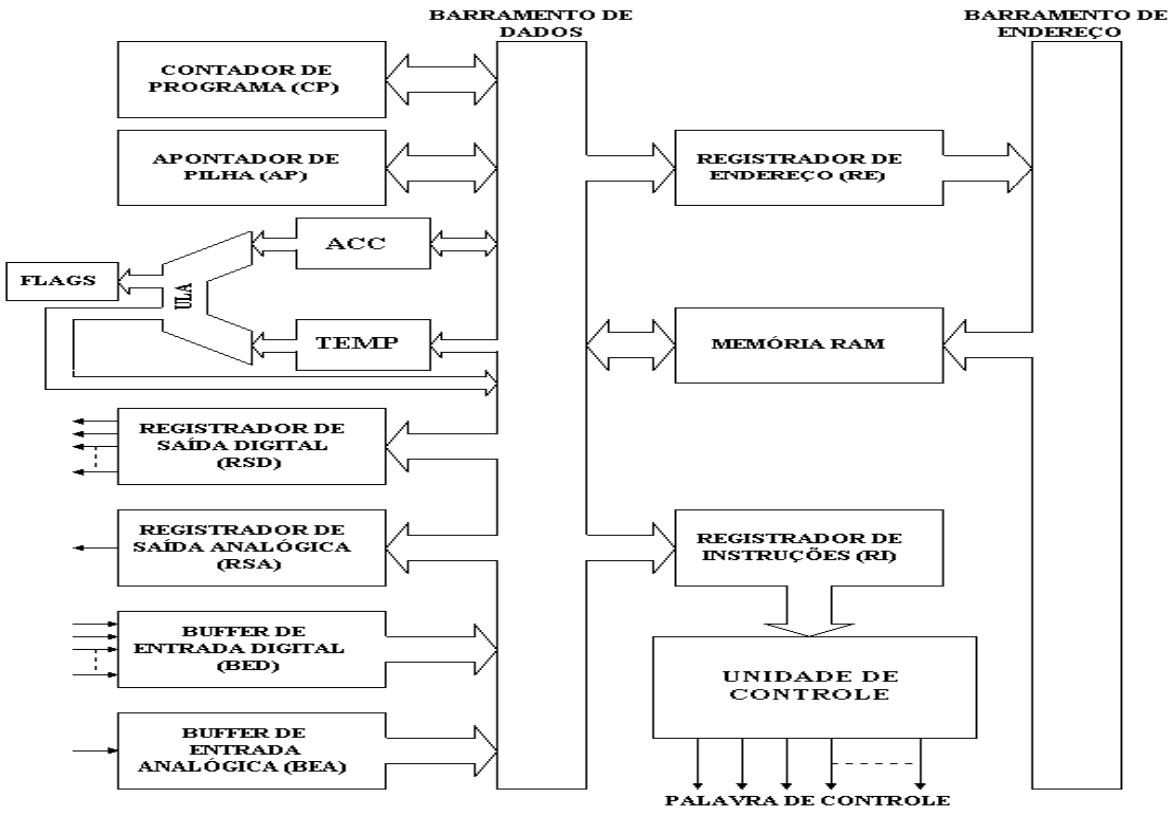

Figura 2: Diagrama em blocos da Unidade Processadora Discreta Microprogramável

2.1 Circuito de sincronismo [2]

Os sinais gerados pelo circuito de sincronismo (não mostrado na Figura 2) afetam todos os eventos da CPU. O Gerador de Clock e Contador em Anel fazem parte do circuito de sincronismo. Na Figura 3 é mostrado o diagrama em blocos do circuito de sincronismo, e também os sinais gerados em cada bloco.

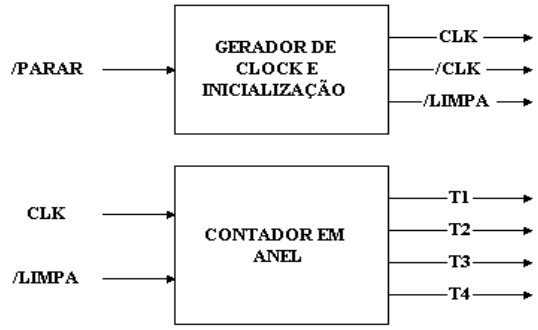

Figura 3: Circuito de inicialização e sincronismo

O Gerador de Clock possui duas opções controladas pelo usuário: clock automático, onde é gerado uma onda quadrada de 1kHz; e o clock manual, onde pulsos são gerados através de uma chave manual. Com o clock manual é possível operar a CPU passo a passo.

No Contador em Anel, as saídas T1, T2, T3, T4 determinam o ciclo de busca, decodificação e execução da instrução. As saídas do Gerador de Clock e do Contador em Anel podem ser visualizadas na Figura 4.

Em T1 o endereço apontado pelo Contador de Programa é armazenado no Registrador de Endereço. Em T2 o Contador de Programa é incrementado, ou seja aponta para o endereço da próxima instrução. Em T3 o endereço armazenado no Registrador de Endereço é transferido para o Barramento de Endereço e a memória RAM é habilitada para leitura. O conteúdo da memória RAM deve corresponder a um código de uma instrução, cujo valor é armazenado no Registrador de Instruções.

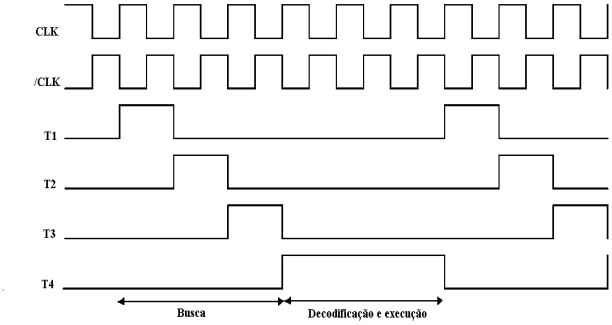

Figura 4: Formas de onda dos sinais de sincronismo

T4 corresponde ao ciclo de decodificação e execução da instrução. O ciclo de execução pode durar até 13 ciclos de clock, controlados por um contador, ou seja, em T4 são executadas as microinstruções de uma instrução. A seqüência de microinstruções é executada até que o sinal de controle FIM seja ativado pelo circuito de controle, e um novo ciclo de busca e execução é iniciado.

2.2 Unidade de Controle [1, 5]

A Unidade de Controle é composta pelo Registrador de Instruções, Contador de Controle e por memórias EPROM's. Na Figura 5 é mostrado o diagrama em blocos da Unidade de Controle.

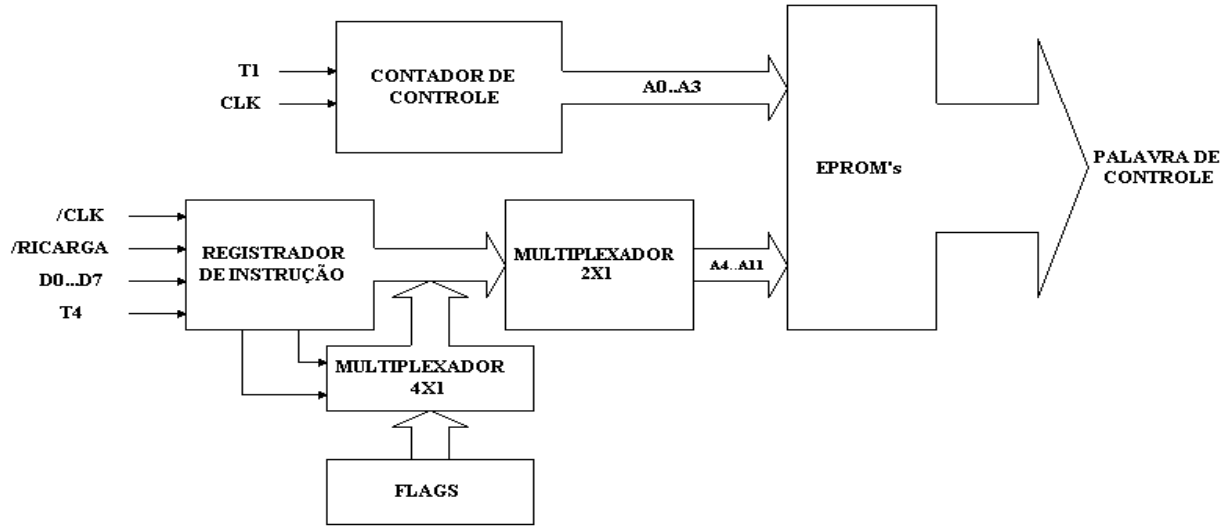

Figura 5: Diagrama em blocos da Unidade de Controle

Quando T1 é ativado, o Contador de Controle é zerado. Este contador é o responsável pela geração dos 4 bits menos significativos do endereço das memórias de controle (EPROM's). Os 8 bits restantes do endereço das EPROM's são zerados pelo multiplexador durante ativação de T1, T2 e T3. Portanto, inicialmente o endereço acessado nas EPROM's corresponde a zero, e os dados armazenados nas EPROM's neste endereço correspondem a geração dos sinais de controle para transferência do valor do Contador de Programa para o Registrador de Endereço (T1).

No próximo ciclo de clock (T2), o Contador de Controle é incrementado, passando para 001 o endereço acessado nas EPROM's. Nesta posição de memória das EPROM's, são gerados os sinais de incremento do Contador de Programa.

No ciclo de clock seguinte (T3), o contador de controle é incrementado acessando a posição 002 das memórias EPROM's. São gerados os sinais para transferência do código da instrução de programa armazenada na memória RAM para o Registrador de Instrução.

Em T4, clock seguinte, o código da instrução é utilizado como linhas de endereço para as memórias EPROM's, em conjunto com os 4 bits do Contador de Controle. Então os sinais de controle da primeira microinstrução da instrução carregada no Registrador de Instrução são gerado pelas memórias EPROM's. Nos ciclos de clock seguintes, o Contador de Controle é incrementado, acessando as microinstruções em sequência até que seja executada toda a instrução.

No circuito de microprogramação projetado é possível obter mais de 127 instruções diferentes. Se o bit mais significativo do código da instrução estiver ativo, ou seja em 1 (um) teremos instruções de desvio condicional.

O sinal de controle /RICARGA, quando ativado, armazena o conteúdo do barramento de dados no registrador de instruções, na transição positiva do /CLOCK e quando T4 estiver ativo.

2.3 Memória RAM

O circuito da Memória RAM é utilizado para armazenamento de dados e programas por meio de chaves. O circuito permite o acesso a 8 páginas de memória que podem ser selecionadas individualmente também por meio de chaves. Na Figura 6 é mostrado o diagrama em blocos e os sinais de controle desta unidade.

São os seguintes sinais de controle da Unidade de Memória RAM:

- /MEMHAB – Habilita a memória;
- MEML/E – Em nível alto, ativa a leitura, onde o conteúdo do endereço selecionado é transferido para o barramento de dados. Em nível baixo, ativa escrita em memória, onde o conteúdo do barramento de dados é armazenado no endereço selecionado;
- Os endereços de A8 a A10 determinam as páginas de memória.

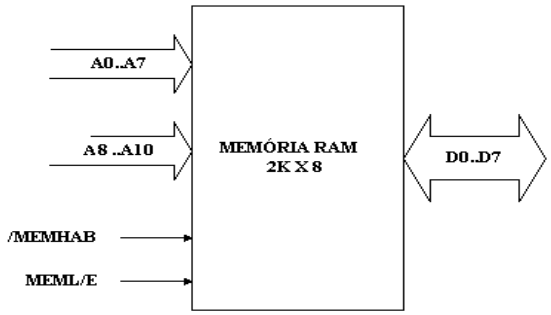

Figura 6: Diagrama em blocos da Memória

2.4 Unidade Lógica Aritmética

A Unidade Lógica Aritmética (ULA) é responsável em realizar as operações lógicas e aritméticas do processador [2]. Na Figura 7 é mostrado o diagrama em blocos da ULA. Os seus sinais de controle são:

- /RACARGA – Habilita a transferência do conteúdo do barramento de dados para o Acumulador, quando houver uma transição positiva do /CLOCK;
- /RBCARGA - Habilita a transferência do conteúdo do barramento de dados para o Registrador Temporário, quando houver uma transição positiva do /CLOCK;
- MODOULA – Define o modo (aritmético = 1 ou lógico = 0) de operação da ULA;
- S0, S1, S2, S3 – define a operação a ser realizada;
- /FLAG – Armazena o estado dos Flags no Registrador de Flags;
- /UM – Define o estado do carry de entrada;
- /ULALER – Habilita a transferência do resultado para barramento de dados;

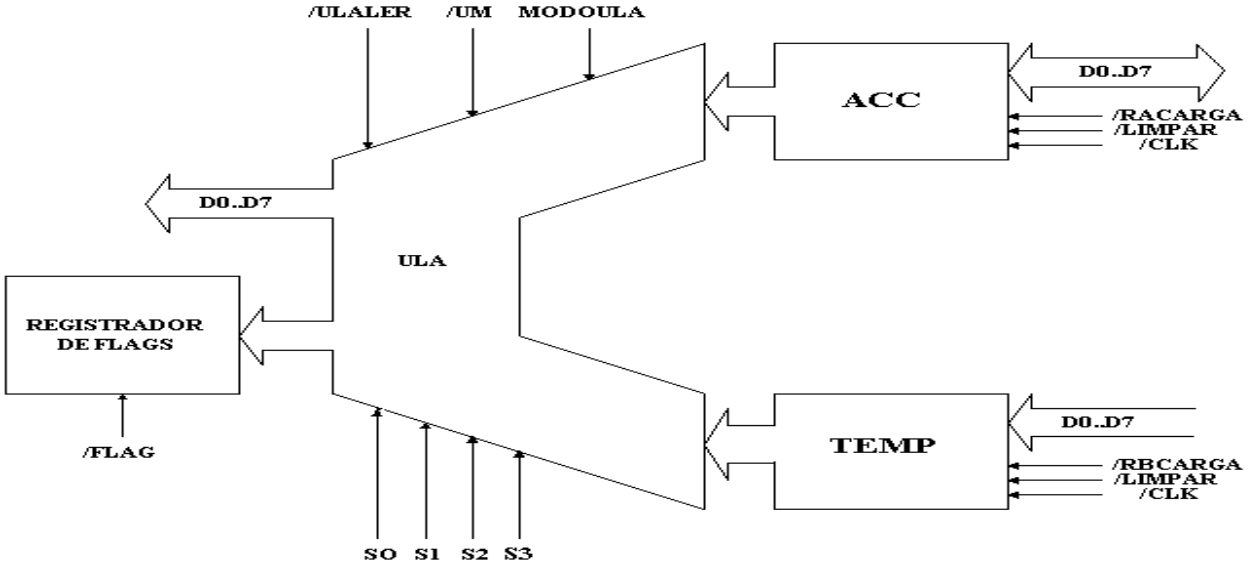

Figura 7: Diagrama em blocos da ULA

2.5 Registrador de Endereço

No Registrador de Endereço é armazenado o conteúdo do Contador de Programa ou do Apontador de Pilha. Na Figura 8 é mostrado o diagrama em blocos e os sinais de controle que ativam esta unidade.

São os seguintes sinais de controle do Registrador de Endereço:

- /RECARGA – Habilita o armazenamento do dado do barramento de dados no Registrador de Endereço;
- /RELER – Habilita a transferência do conteúdo armazenado no Registrador de Endereço para o barramento de endereço.

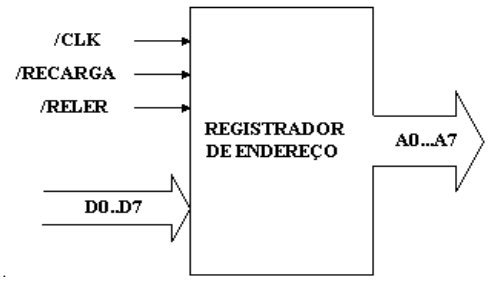

Figura 8: Registrador de Endereço

2.6 Contador de Programa e Apontador de Pilha

O Contador de Programa indica o endereço da próxima instrução a ser executada.

O Apontador de Pilha indica o endereço da pilha onde foi armazenado o valor do Contador de Programa antes de iniciar a execução de uma subrotina. Ele também é utilizado nas instruções de PUSH e POP, onde os dados são armazenados e/ou retirados da pilha. Na Figura 9 são mostrados os sinais de controle que ativam esta unidade.

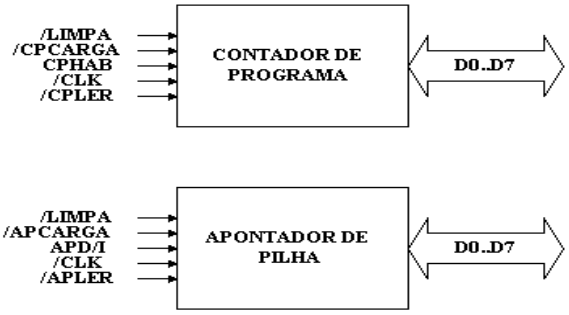

Figura 9: Diagrama em blocos do Contador de Programa e Apontador de Pilha

São os seguintes sinais de controle desta unidade:

- /CPCARGA – Carrega o Contador de Programa com o dado presente no barramento de dados;
- CPHAB – Habilita a contagem do Contador de Programa;
- /CPLER – Habilita a transferência do dado do Contador de Programa para o barramento de dados;
- /LIMPA – Zera o Contador de Programa;
- /APCARGA - Carrega o Apontador de Pilha com dado do barramento de dados;

- APD/I – Habilita contagem crescente ou decrescente do Apontador de Pilha;
- /APLER – Habilita a transferência do dado do Apontador de Pilha para o barramento de dados.

2.7 Registradores e Buffers de Entrada/Saída

O registrador de saída digital armazena um dado proveniente do barramento de dados e os disponibiliza externamente ao protótipo. Este registrador é usado para acionar LED's.

O registrador de saída analógica utiliza um conversor digital para analógico para disponibilizar um sinal analógico externamente.

O buffer de entrada digital permite que um dado digital possa ser inserido no barramento de dados durante a execução de um programa.

O buffer de entrada analógica utiliza um conversor analógico para digital para inserir um sinal analógico no barramento de dados durante a execução de programa.

3 Instruções de programa [3]

O processador possui o seguinte conjunto de instruções mostrado na Tabela 1:

Tabela 1: Conjunto de instruções

Instrução	Código	Descrição
ADI <dado>	05	ACC ← ACC + <dado>
ADD <endereço>	20	ACC ←ACC + [endereço]
SUBI <dado>	06	ACC ←ACC - <dado>
SUB <endereço>	21	ACC ←ACC – [endereço]
CMP <dado>	19	compara ACC com <dado> e seta Flag de igual se ACC = dado
INC	07	ACC ←ACC + 1
DCC	08	ACC ←ACC – 1
LDI <dado>	04	ACC ←<dado>
LDA <endereço>	09	ACC ←[endereço]
STA <endereço>	10	[endereço] ←ACC
AND <dado>	11	ACC ←ACC AND <dado>
OR <dado>	12	ACC ←ACC OR <dado>
XOR<dado>	13	ACC ←ACC XOR <dado>
CLR	14	ACC ← 0
NOT	15	ACC ←/ACC
JMP <endereço>	16	PC ←endereço
JZ <endereço>	82	se /ZERO = 0 CP ←endereço
JC <endereço>	91	se /VAI-UM = 0 CP ←endereço
JS <endereço>	A7	se SINAL = 1 CP ←endereço
JEQ <endereço>	E5	se IGUAL =1 CP ←endereço
JNZ <endereço>	B3	se /ZERO =1 CP ←endereço
JNC <endereço>	C1	se /VAI-UM = 1 CP ←endereço
JNS <endereço>	D6	se SINAL = 0 CP ←endereço
JNEQ <endereço>	F4	se IGUAL = 0 CP ←endereço
CALL endereço	17	[AP] ←CP; CP ← endereço; AP ← AP – 1
RET	18	CP ←[AP]; AP← AP + 1
INA	22	ACC ←entrada analógica
IND	23	ACC ←entrada digital
OUTA	24	saída analógica ←ACC
OUTD	25	saída digital ←ACC
HALT	26	Parada

As instruções acima possuem, cada uma, um conjunto de microinstruções armazenado na memória de controle EPROM, iniciando no endereço correspondente ao código da instrução. Cada microinstrucão é responsável pela geração dos sinais de controle para execução da instrução. Cada instrução termina com a microinstrução que gera o sinal de FIM.

4 Resultados

O protótipo de processador descrito neste artigo foi simulado e montado utilizando circuitos integrados em placas de wire-wrap.

Como os circuitos podem ser modificados e testados facilmente, é possível implementar novas técnicas de projeto de processadores. O microprograma armazenado em memória EPROM permite que as instruções sejam modificadas e que novas instruções possam ser implementadas.

5 Referências

[1] LANGDON Jr, G. G. e **FREGNI**, E. *Projeto de Computadores Digitais*. São Paulo: Edgard Blucher Ltda, 1990, 357p.

[2] MALVINO, Albert Paul. *Microcomputadores e Microprocessadores*. São Paulo: McGraw-Hill, 1986, 577p.

[3] STALLINGS, WILLIAM. *Computer Organization and Architecture – Designing for Performance*. Prentice Hall, Fourth Edition, 1996, 682p.

[4] TOKHEIM, Roger L. *Introdução aos Microprocessadores*. São Paulo: MacGraw-Hill do Brasil, 1985, 431p.

[5] ZUFFO, João Antônio. *Fundamentos da Arquitetura e Organização dos Microprocessadores*. São Paulo: Editora Edgard Blucher Ltda, 1978, 419p.