Anales PANEL'81/12 JAIIO
Sociedad Argentina de informática
e Investigación Operativa. Buenos Aires, 1981

"LDSC - UMA LINGUAGEM PARA A DESCRIÇÃO E SIMULAÇÃO DE COMPUTADORES"

Caetano Traina Junior
Edson Luiz Recchia
Fernao Stella De Rodrigues Germano

Instituto de Ciências Matemáticas de Sao Carlos - USP
Departamento de Ciências de Computaçao e Estatística
Av. Dr. Carlos Botelho 1465, Sao Carlos - Brasil.

RESUMO

Foi desenvolvido um sistema de auxílio ao estudo de organização de Computadores, apoiado por computador. Esse sistema aceita a descrição da arquitetura de um computador, de seu conjunto de instruções e programas para serem executados nesse computador, e simula a execução desses programas. Estão disponíveis nove micro-instruções para a definição do conjunto de instruções que pode ter até 32 instruções. Para a descrição da arquitetura, podem ser usados 4 diferentes tipos de circuitos, num máximo de 40 circuitos. Esse sistema foi inicialmente desenvolvido num computador IBM 1130 e posteriormente transladado para um PDP 11/45. Usando esse sistema, alunos do curso de organização de computadores se tornam aptos a verificar praticamente as consequências das diferentes alternativas de projeto, tanto na arquitetura quanto no conjunto de instruções.

I. INTRODUÇÃO

O objetivo da Linguagem para a Descrição e Simulação de Computadores (LDSC) é fornecer a oportunidade de se verificar as alterações que ocorrem no software quando é alterado o hardware da máquina.

Foi desenvolvido um sistema capaz de receber do usuário a descrição da máquina que ele quer estudar, de um conjunto de instruções que essa máquina pode ter, e de executar por simu

lação programas escritos nessa linguagem de máquina.

A Linguagem LDSC é estruturada em 4 seções, cada uma destinada a descrever ou controlar as várias atividades do sistema:

Descrever a arquitetura do computador (Computador);

Descrever as instruções a serem simuladas (Instruções);

Especificar o programa a ser executado (Montagem);

Executar o programa (Run).

Essas seções serão consecutivas, no sentido de que não é possível passar a uma dada seção sem antes haver especificado as anteriores. Porém, pode-se voltar a seções anteriores a qualquer instante, e recomeçar as especificações a partir da seção à qual se voltou. Assim, por exemplo, pode-se descrever a arquitetura de um computador (seção Computador), estabelecer para ele um conjunto de instruções (seção Instruções), especificar um programa com essas instruções (seção Montagem) e executar esse programa (seção Run) para vários conjuntos de dados (seção Run repetida várias vezes); a seguir pode-se modificar o programa (seção Montagem) e executá-lo novamente (seção Run); pode-se então desejar modificar alguma das instruções (seção Instruções), reescrever o programa (seção Montagem) e testá-lo (seção Run).

II. CARACTERÍSTICAS DOS COMPUTADORES SIMULADOS PELO SISTEMA

São detalhadas a seguir, as restrições que são impostas pelo LDSC às características dos computadores que podem ser simulados pelo sistema.

MEMÓRIA: A memória é sempre endereçada através do conteúdo de um registrador específico, definido na seção Computador como Registrador de Endereço, e os dados são sempre transferidos de/para um Registrador de Dados, também definido na seção Computador. Seu tamanho pode variar em incrementos de 256 palavras, atingindo um máximo de 1024 palavras. Cada palavra tem sempre o comprimento de um byte (8 bits).

REGISTRADORES: Todos os registradores devem ser especificados na seção Computador, compostos de blocos de 8 bits cada bloco. Um registrador pode ser composto de um ou vários blocos concatenados.

OPERAÇÕES: Todas as operações simuladas são especificadas através de micro-instruções a nível de registradores e desempenham as seguintes funções: transferência de dados entre registradores via uma barra de dados, operações na unidade-lógico-aritmética, operações nos registradores (deslocamentos) e acesso à memória. Os operandos são sempre registradores, e cada operação é definida para um bloco de 8 bits, podendo ser extendida para os blocos subsequentes dos registradores envolvidos. Assim é possível manipular individualmente partes (1 bloco) de um dado registrador. Se uma operação envolve registradores de com

primentos diferentes, ela é continuada apenas até o número de blocos do menor dos registradores envolvidos. Os blocos a mais no registrador maior não são alterados.

UNIDADE LÓGICO-ARITMÉTICA (ULA): A ULA simula as operações soma, subtração E-Lógico e OU-Lógico. Todas as operações são sempre efetuadas com 2 operandos. Assim, caso se queira simular por exemplo a operação Complemento-2 em um operando, outro operando deve ser um zero fornecido à ULA por uma outra unidade da arquitetura. O sistema simulador restringe à existência de exatamente uma ULA em cada computador Simulado.

REGISTRADOR DE INSTRUÇÕES: Todo computador simulado deve ter um registrador especial chamado RI (Registrador de Instruções) que o simulador usa para executar o programa, especificado na seção Montagem. Esse registrador deve ser constituído de apenas 1 bloco, sendo portanto sempre de 8 bits dos quais se utiliza apenas os 5 bits de mais baixa ordem, para reconhecer a instrução que deve ser executada. Isso significa que apenas 32 instruções podem ser simuladas de cada vez. Os três bits mais significativos não são utilizados e assim não é possível simular instruções que indicam operandos na mesma palavra de instrução (apesar de ser possível simular instruções cuja indicação de operandos esteja em palavras consecutivas na memória).

MICRO-INSTRUÇÕES: O simulador pode executar apenas uma micro-instrução por vez, impedindo assim simulação de micro-instruções micro-concorrentes. Apesar disso, durante a especificação das instruções é feita uma verificação das disponibilidades de circuitos da arquitetura, segundo as especificações feitas na seção Computador. A execução das micro-instruções de uma instrução é sempre sequencial, somente sendo alterada a sequência quando da mudança do conteúdo do registrador RI. Micro-instruções podem ter sua execução inibida, dependendo de condições descritas adiante.

ENTRADA/SAÍDA: Nenhuma operação de entrada e saída pode ser especificada diretamente em forma de uma micro-instrução. Para isso existem 2 registradores especiais que quando ativados transferem dados de registradores especificados na seção Computador para dispositivos de entrada ou saída. Existe 1 registrador especial para entrada e outro para saída, limitando assim a 2 o número de dispositivos periféricos que se pode considerar na simulação. O usuário da Linguagem pode escolher como dispositivo de entrada uma leitora de cartões ou um terminal e como dispositivo de saída uma impressora ou um terminal.

Alguns aspectos dos computadores reais não podem ser simulados, na LDSC. Esses são aspectos que em geral envolvem a variável tempo:

TEMPORIZAÇÃO: Não se pode, no modelo simulado, especificar nem medir o tempo requerido pela execução de qualquer operação. Em todas as micro-instruções é considerado que se leva o menor tempo para processá-las. Parâmetros como sincronismo, e tempo de acesso não são considerados pelo simulador.

INTERRUPÇÃO: Não foi previsto nenhum tipo de alteração

na sequência normal de execução de programa simulado, através da ocorrência de eventos assíncronos externos do sistema. A única forma de comunicação desse programa com os dispositivos periféricos considerados na simulação é através dos registradores especiais de controle de Entrada/Saída. Armadilhas (traps) para detectar eventos assíncronos internos podem contudo ser simulados.

III. DESCRIÇÃO DAS SEÇÕES DA LINGUAGEM

Descreve-se aqui o que deve ser especificado e o que pode ser obtido em cada uma das seções da LDSC. O exemplo das figuras 1, 2, 3 e 4 ilustra essas várias seções.

III.1. SEÇÃO COMPUTADOR

Nesta seção é descrita a arquitetura do computador que será simulado. Para isso considera-se que todo computador é constituído de vários circuitos padrões interligados de diferentes maneiras. A figura 1a mostra uma arquitetura simples, e a figura 1b a forma como ela pode ser descrita na LDSC.

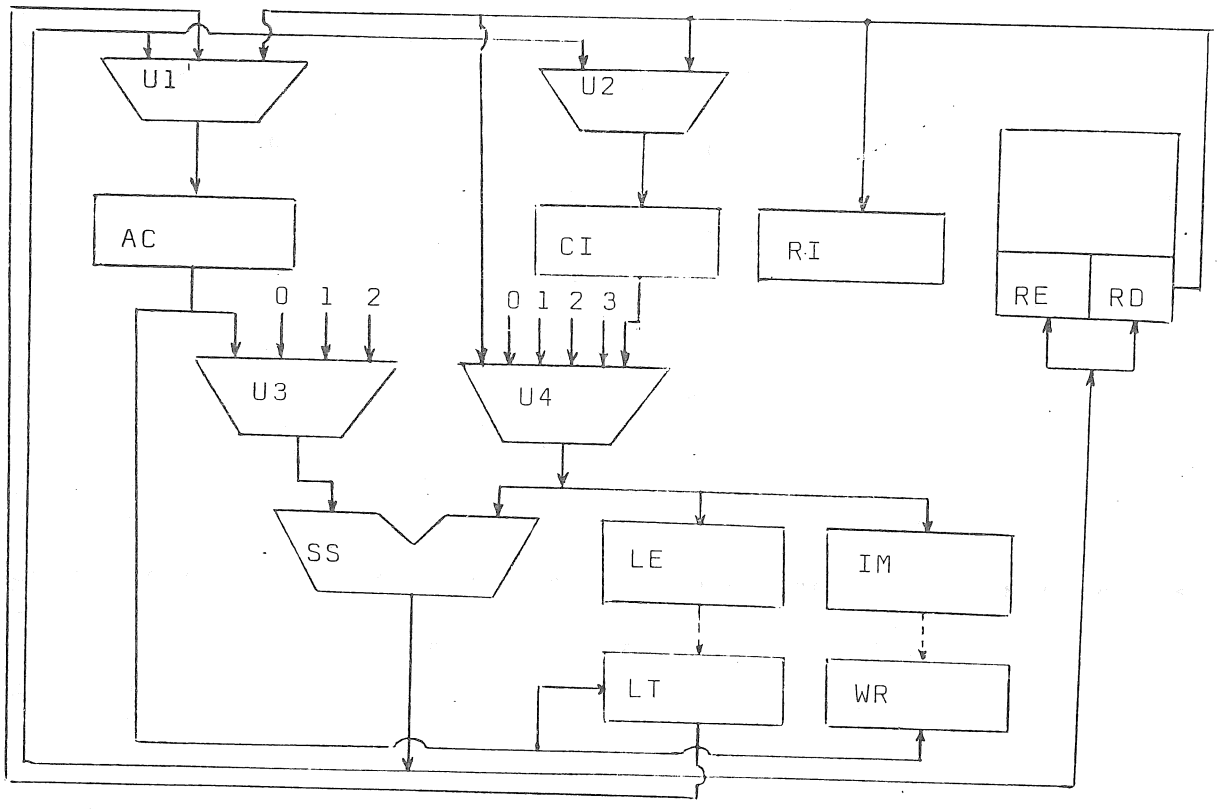

Figura 1a

```
/*COMP

 1   MEMORIA       RE RD
 2   S/S           SS U3 U4
 3   RI            RI RD
 4   CONSTANTE      0
 5   CONSTANTE      1
 6   CONSTANTE      2
 7   CONSTANTE      3
 8   REGISTRADOR RE SS
 9   REGISTRADOR RD SS
10   REGISTRADOR LT AC
11   REGISTRADOR WR AC
12   REGISTRADOR AC U1
13   REGISTRADOR CI U2
14   REG-CONTR-E LE U4 LT
15   REG-CONTR-S TM U4 WR
16   ULM           U1    3
          000 LT
          001 SS
          010 RD
17   ULM           U2    2
          000 RD
          001 SS
18   ULM           U3    4
          000 AC
          001  0
          010  1
          011  2
19   ULM           U4    6
          000 RD
          001 CI
          010  0
          011  1
          100  2
          101  3

/*FIM
```

Figura 1b

Os circuitos que são reconhecidos pelo simulador são os seguintes:

MEMÓRIA: É representada por dois registradores que sempre são associados a ela, chamados Registrador de Endereços e Registrador de Dados. É sempre o primeiro circuito que deve ser definido nesta seção, e pode haver apenas uma Memória em cada computador simulado.

UNIDADE LÓGICO-ARITMÉTICA (S/S): Circuito que executa as operações Soma, Subtração, E-Lógico e OU-Lógico. Recebe duas barras de dados e gera uma barra de dados. Deve existir um circuito desses em cada computador simulado.

REGISTRADOR DE INSTRUÇÕES (RI): Representado por um registrador, é no entanto um registrador especial na seção Montagem, pois indica qual a instrução que está sendo executada.

REGISTRADOR: É um circuito que pode memorizar um dado com um número de bits múltiplo de oito. Recebe uma barra de dados e gera uma barra de dados, por exemplo, na linha 8 da figura 1b, RE é especificado como um registrador que recebe dados do circuito SS.

MULTIPLEXADOR: É um circuito que pode escolher uma dentre várias barras de dados. Recebe um número de barras de dados que pode variar de 2 a 8, e gera uma barra de dados.

CONSTANTE: É um circuito que gera uma barra de dados com um dado sempre constante.

REGISTRADOR DE CONTROLE DE ENTRADA: É representado por um registrador. Sempre que este registrador receber um dado, o conteúdo de um registrador a ele associado será o dado lido do dispositivo de entrada. A forma com que o dado fornecido é interpretado depende do dado recebido pelo Registrador de Controle de Entrada:

1. É lido um caracter;
2. É lido um dado em octal;
3. Passar para um novo registro de entrada (novo cartão ou linha).

O dado presente na barra de saída deste registrador é sempre zero.

REGISTRADOR DE CONTROLE DE SAÍDA: É representado por um registrador. Sempre que este registrador receber um dado, o conteúdo de um outro registrador a ele associado, será enviado ao dispositivo de saída. A forma com que o dado será impresso, depende do dado recebido pelo Registrador de Controle de Saída:

1. O conteúdo do registrador associado é tomado como Caracter;
2. O conteúdo do registrador associado é impresso em octal;
3. Avançar uma linha.

A arquitetura de um computador deve ter no máximo 40 circuitos. A interligação dos vários circuitos é feita através de barras de dados. Toda barra de dados é gerada por apenas um circuito, mas pode ser recebida por qualquer número de circuitos.

Todo circuito é identificado por um código de duas letras. A interligação dos vários circuitos é especificada através da indicação, para cada circuito, de qual é o circuito que gera cada barra de dados que ele recebe.

O controle dos circuitos é simulado estabelecendo-se li

nhas de controle em cada circuito. Essas linhas, de um bit cada, simulam as que são geradas pela Unidade de Controle nos computadores reais.

III.2. SEÇÃO INSTRUÇÕES

Nesta seção são descritas as instruções do computador. A figura 2a mostra um exemplo das instruções mencionadas na figura 2b.

```
INSTRUCOES DEFINIDAS

1   SM
2   SR
3   CA
4   AA
5   DI
6   DC
7   ES
8   LE

/*FIM
```

Figura 2b

Uma Micro-instrução pode especificar uma das seguintes operações:

Leitura na memória: exemplo na linha 2

Armazenagem na memória: exemplo na linha 35

Transferência de um registrador para outro: exemplo na linha 3

Somar o conteúdo de 2 registradores e armazenar o resultado em outro registrador: exemplo na linha 1

Subtrair o conteúdo de 2 registradores e armazenar o resultado em outro registrador: exemplo na linha 17

Fazer a operação E-Lógico de 2 registradores e armazenar o resultado em outro registrador: exemplo análogo à linha 1 onde o sinal "+" é substituído pelo "E"

Fazer a operação OU-Lógico de 2 registradores e armazenar o resultado em outro registrador: exemplo análogo à linha 1 onde o sinal "+" é substituído pelo "O"

Deslocar o conteúdo de um registrador de um bit para a direita: por exemplo (AC) D

Deslocar o conteúdo de um registrado de um bit para a esquerda: por exemplo (AC) E.

Para cada instrução especificada, deve ser especificado um código de duas letras que passa a ser o mnemônico da instrução, usado na seção Montagem. A cada instrução é então asso

```
/*INSTRUCOES QUE DEFINEM A LINGUAGEM DE MAQUINA DO COMPUTADOR SIMULADO
*
FARRELMACI.IUUUU
MSIFUTHCIPM1214

      INSTRUCAO NUM.  1                                    .     V     .     V     .     V     .     V
  1   1  DEFINE SM  (RE) = / 0/ +(CI)                    0001000000000011
  2   2             (RD) =((RE))                         2000100000000011
  3   3             (RI) = (RD)                          0010000000000011
  4   4             (RE) = / 1/ +(CI)                    0001000000000021
  5   5             (RD) =((RE))                         2000100000000021
  6   6             (RE) = / 0/ +(RD)                    0001000000000010
  7   7             (RD) =((RE))                         2000100000000010
  8   8             (AC) = (AC) +(RD)                    0000000100010000
  9   9             (CI) = / 2/ +(CI)                    0000000010001131

      INSTRUCAO NUM.  2                                    .     V     .     V     .     V     .     V
 10   1  DEFINE SB  (RE) = / 0/ +(CI)                    0001000000000011
 11   2             (RD) =((RE))                         2000100000000011
 12   3             (RI) = (RD)                          0010000000000011
 13   4             (RE) = / 1/ +(CI)                    0001000000000021
 14   5             (RD) =((RE))                         2000100000000021
 15   6             (RE) = / 0/ +(RD)                    0001000000000010
 16   7             (RD) =((RE))                         2000100000000010
 17   8             (AC) = (AC) -(RD)                    0100000100010000
 18   9             (CI) = / 2/ +(CI)                    0000000010001131

      INSTRUCAO NUM.  3                                    .     V     .     V     .     V     .     V
 19   1  DEFINE CA  (RE) = / 0/ +(CI)                    0001000000000011
 20   2             (RD) =((RE))                         2000100000000011
 21   3             (RI) = (RD)                          0010000000000011
 22   4             (RE) = / 1/ +(CI)                    0001000000000021
 23   5             (RD) =((RE))                         2000100000000021
 24   6             (RE) = / 0/ +(RD)                    0001000000000010
 25   7             (RD) =((RE))                         2000100000000010
 26   8             (AC) = (RD)                          0000000010002010
 27   9             (CI) = / 2/ +(CI)                    0000000001002131

      INSTRUCAO NUM.  4                                    .     V     .     V     .     V     .     V
 28   1  DEFINE AA  (RE) = / 0/ +(CI)                    0001000000000011
 29   2             (RD) =((RE))                         2000100000000011
 30   3             (RI) = (RD)                          0010000000000011
 31   4             (RE) = / 1/ +(CI)                    0001000000000021
 32   5             (RD) =((RE))                         2000100000000021
 33   6             (RE) = / 0/ +(RD)                    0001000000000010
 34   7             (RD) = (AC) +/ 0/                    0000100000000002
 35   8            ((RE))= (RD)                          1000000000000002
 36   9             (CI) = / 2/ +(CI)                    0000000001000131

      INSTRUCAO NUM.  5                                    .     V     .     V     .     V     .     V
 37   1  DEFINE OI  (RE) = / 0/ +(CI)                    0001000000000011
 38   2             (RD) =((RE))                         2000100000000011
 39   3             (RI) = (RD)                          0010000000000011
 40   4             (RE) = / 1/ +(CI)                    0001000000000021
 41   5             (RD) =((RE))                         2000100000000021
 42   6             (CI) = (RD)                          0000000001000021
 43   7             (CI) = / 0/ +(CI)                    0000000001000111

      INSTRUCAO NUM.  6                                    .     V     .     V     .     V     .     V
 44   1  DEFINE DC  (RE) = / 0/ +(CI)                    0001000000000011
 45   2             (RD) =((RE))                         2000100000000011
 46   3             (RI) = (RD)                          0010000000000011
 47   4             (RE) = / 1/ +(CI)                    0001000000000021
 48   5             (RD) =((RE))                         2000100000000021
 49   6             (CI) = (RD)          2 (AC)          0000000001000021
 50   7             (CI) = / 2/ +(CI) 6 (AC)             0000000001001131

      INSTRUCAO NUM.  7                                    .     V     .     V     .     V     .     V
 51   1  DEFINE ES  (RE) = / 0/ +(CI)                    0001000000000011
 52   2             (RD) =((RE))                         2000100000000011
 53   3             (RI) = (RD)                          0010000000000011
 54   4             (RE) = / 1/ +(CI)                    0001000000000021
 55   5             (RD) =((RE))                         2000100000000021
 56   6             (RE) = / 0/ +(RD)                    0001000000000010
 57   7             (RD) =((RE))                         2000100000000010
 58   8             (AC) = (RD)                          0000000010002010
 59   9             (RE) = (AC)                          0000000100002010
 60  10             (IM) = / 2/                          0000000000012014
 61  11             (CI) = / 2/ +(CI)                    0000000001002131

      INSTRUCAO NUM.  8                                    .     V     .     V     .     V     .     V
 62   1  DEFINE LE  (RE) = / 0/ +(CI)                    0001000000000011
 63   2             (RD) =((RE))                         2000100000000011
 64   3             (RI) = (RD)                          0010000000000011
 65   4             (LT) = / 2/                          0000001000100014
 66   5             (RE) = / 1/ +(CI)                    0001000000000021
 67   6             (RD) =((RE))                         2000100000000021
 68   7             (RE) = / 0/ +(RD)                    0001000000000010
 69   8             (AC) = (LT)                          0000000010000010
 70   9             (RD) = (AC) +/ 0/                    0000010000000002
 71  10            ((RE))= (RD)                          1000000000000002
 72  11             (CI) = / 2/ +(CI)                    0000000001000131
```

Figura 2a

ciado um código de 5 bits, através do qual a instrução será reconhecida na seção Run. Esse código é o número binário correspondente à ordem em que cada instrução foi especificada. Assim por exemplo, a primeira instrução recebe o código 00001, a segunda o código 00010, e a trigésima segunda o código 00000.

Independente da descrição da arquitetura do computador efetuada na seção Computador, pode-se considerar que existe um Circuito Biestável associado à Unidade Lógico-Aritmética, que armazena uma indicação sobre a última operação executada por esse circuito: houve um Transporte de Vai-Um do bit mais significativo ou não. Além desse biestável, considera-se que associados a cada registrador existam mais 3 biestáveis, indicando respectivamente se o conteúdo do registrador é: negativo; par; zero.

Qualquer um desses biestáveis pode ser testado para permitir ou inibir a execução de qualquer micro-instrução. Sempre que uma micro-instrução estiver associada a uma condição, antes de sua execução a condição é testada, e caso ela não seja satisfeita, a micro-instrução é saltada.

Nesta seção o usuário pode requisitar do sistema o cordão de bits que deve ser gerado para controlar os circuitos especificados na seção Computador. O cordão de bits gerado contém uma especificação binária para cada uma das linhas de controle de cada circuito. Não é feita nenhuma multiplexagem das linhas, a não ser das linhas de controle de um mesmo multiplexador. As linhas de controle que não são usadas em uma dada micro-instrução são mantidas no mesmo nível do cordão de bits da micro-instrução anterior.

III.3. SEÇÃO MONTAGEM

Nesta seção, usando as instruções com os mnemônicos especificados na seção Instruções; o usuário pode especificar um programa para ser executado pelo computador descrito nas 2 seções anteriores. A figura 3 mostra um exemplo de um programa para a determinação do mínimo múltiplo comum de 2 números que pode ser executado no computador especificado nas figuras 1 e 2.

Qualquer valor numérico especificado nesta seção é interpretado como um número em octal. As instruções lidas são convertidas para o código de 5 bits estabelecido na seção anterior, e armazenadas na memória, uma instrução ou um número por byte de memória, começando pelo endereço um. Assim, dependendo do número de operandos fornecidos em cada instrução, o número de palavras gastas em cada instrução é variável.

III.4. SEÇÃO RUN

Nesta seção o simulador simplesmente executa o programa especificado na seção Montagem. Como exemplo, o resultado da simulação da execução do programa da figura 3 é mostrado na figura 4.

```
/*MONT

0001    LE 100
0003    ES 100
0005    LF 101
0007    ES 101
0011    CA 100
0013    SB 101
0015    DC 31
0017    CA 101
0021    SB 100
0023    DC 41
0025    ES 100
0027    DI 51
0031    CA 101
0033    SB 100
0035    AA 101
0037    DI 11
0041    CA 100
0043    SB 101
0045    AA 100
0047    DI 11
0051

/*FIM
```

Figura 3

```
/*RUN

** LR **  0030
** WR **  0004
** WR **  0004

/*FIM
```

Figura 4

O usuário toma contato com o resultado dessa execução de duas maneiras: através dos dados enviados pelo programa para o dispositivo de saída e, através dos valores assumidos por registradores escolhidos, quando tais registradores têm seu conteúdo modificado.

V. CONCLUSÕES

A LDSC foi preparada para ser usada na disciplina Organização de Computadores ministrada pelo Departamento de Ciências de Computação e Estatística do Instituto de Ciências Matemáticas de São Carlos - USP. Trata-se de disciplina obrigatória do Curso de Bacharelado em Computação e optativa para os demais cursos ministrados no campus de São Carlos da USP. Muitos alunos de Engenharia Elétrica/Eletrônica cursam essa disciplina com bastante interêsse.

Anteriormente à adoção da LDSC, as aulas dessa disciplina eram puramente teóricas, e os alunos tinham dificuldade em visualizar todas as operações e circuitos envolvidos na execução de uma única instrução em um computador.

Depois de adotada a LDSC, os alunos passaram a contar com um instrumento com o qual podem expressar suas idéias a respeito da arquitetura de computadores, verificar como a máquina por eles projetada se comporta e verificar como alterações nos vários níveis de especificação da máquina se refletem no resultado final. O aproveitamento durante o curso aumentou bastante, e muitos alunos inclusive têm desenvolvido trabalhos baseados nessa linguagem mesmo depois de terem sido aprovados na disciplina.

BIBLIOGRAFIA:

HAYS, G. - Computer-aided design - Simulation of Digital Design Logic. IEEE Trans. on Computers, Vol. C18, nº 1, Janeiro 1969.

JAYAKUMAR, M.S. & Mc CALLA, I.M. - Simulation of Microprocessor Emulation Using GASP-PL/I. IEEE Computer. Abril 1977.

RECCHIA, E.L. - Manual de Uso da Linguagem LDSC - SCE Documento do Trabalho nº 4, Março 1979.

STEWART, S.H. - LOGAL - A CHDL for Logic Design and Synthesis of Computers. IEEE Computer. June 1977.

Esse trabalho foi em parte desenvolvido dentro de um plano de trabalho de bolsa de iniciação científica concedida pelo CNPq ao estudante Edson Luiz Recchia, sob a orientação do professor Fernão Stella de Rodrigues Germano.

A concepção da LDSC, o projeto do Sistema de Simulação e sua aplicação ao ensino da disciplina Organização de Computadores foram feitos pelo engenheiro Caetano Traina Junior e contaram com o auxílio da FINEP e da FAPESP.