

# Sistemas Digitais e Microprocessadores

Roberto A. Hexsel

```
library IEEE;
use IEEE.std_logic_1164.all;
use work.p_WIRES.all;

entity mico is
  port ( m_reset:          in      std_logic;
         m_de, m_rw, m_ie, m_io: out  std_logic;
         m_mrdy:         in      std_logic;
         m_inst_addr, m_data_addr: out  reg16;
         m_inst:         in      reg16;
         m_data:         inout   reg16 );
end mico;
```

# Sistemas Digitais e Microprocessadores

Roberto A Hexsel<sup>1</sup>

Departamento de Informática  
Universidade Federal do Paraná

14 de junho de 2015

# Sumário

<b>Prefácio</b>	<b>11</b>
<b>1 Circuitos Combinacionais</b>	<b>16</b>
1.1 Abstração de Sinais como Bits	16
1.1.1 Operações sobre Bits	18
1.1.2 Expressões	21
1.1.3 Tipos e Especificações	21
1.1.4 Tabela Verdade e a Soma de Produtos	23
1.1.5 Mapa de Karnaugh	24
1.2 Portas Lógicas	27
1.2.1 Conjunto Mínimo de Operadores	30
1.3 Circuitos Combinacionais Básicos	31
1.3.1 Multiplexador	31
1.3.2 Demultiplexador	33
1.3.3 Seletor	34
1.4 Circuitos Implementados em TTL	36
1.4.1 Multiplexador 74151	37
1.4.2 Decodificador 74154	37
1.4.3 Seletor 74138	38
1.5 Deslocamentos	41
1.5.1 Deslocador Exponencial	42
1.5.2 Rotação	43
1.6 Unidade de Lógica e Aritmética	44
1.6.1 Soma	46

1.6.2	Subtração . . . . .	47
<b>2</b>	<b>A Tecnologia de Circuitos CMOS</b>	<b>50</b>
2.1	Semicondutores e Transistores CMOS . . . . .	50
2.1.1	Materiais e Fabricação . . . . .	51
2.1.2	Operação dos Transistores . . . . .	52
2.2	Implementação de Portas Lógicas . . . . .	54
2.2.1	Circuitos com chaves . . . . .	55
2.2.2	Portas Lógicas CMOS . . . . .	58
2.2.3	Portas Complexas . . . . .	62
2.2.4	Terceiro Estado . . . . .	64
2.2.5	Portas de Transmissão . . . . .	66
2.3	O Tempo de Propagação É Maior que Zero . . . . .	68
2.3.1	Um Rápido Passeio Pelo Reino da Física . . . . .	68
2.3.2	Tempo de Propagação em Circuitos CMOS . . . . .	71
2.4	ROM – <i>Read-Only Memory</i> . . . . .	75
<b>3</b>	<b>Circuitos Sequenciais</b>	<b>79</b>
3.1	Circuitos com Memória . . . . .	80
3.2	<i>Flip-Flops</i> . . . . .	81
3.3	Registradores . . . . .	83
3.4	Contadores . . . . .	84
3.4.1	<i>Ripple Counter</i> . . . . .	85
3.4.2	Contadores Síncronos . . . . .	86
3.4.3	Contador em Anel . . . . .	88
3.4.4	Contador Johnson . . . . .	89
3.4.5	74163 . . . . .	90
3.4.6	74191 . . . . .	93
3.5	Registradores de Deslocamento . . . . .	94
3.5.1	74194 . . . . .	95
3.5.2	Somador Serial . . . . .	96
3.6	Uma Rápida Olhada no Relógio . . . . .	97
3.6.1	Divisão de Frequência . . . . .	98
3.6.2	Ciclo de Trabalho . . . . .	98

3.6.3	Velocidade Máxima de Operação . . . . .	99
3.7	Projeto de Máquinas de Estados . . . . .	104
3.7.1	Diagrama de Estados . . . . .	104
3.7.2	Implementação de Máquinas de Estado . . . . .	108
3.7.3	Máquina de Vender Chocolates . . . . .	109
3.8	Microcontroladores . . . . .	113
3.8.1	Memória ROM . . . . .	113
3.8.2	Microcontrolador Baseado em ROM . . . . .	114
3.8.3	Controle de Fluxo . . . . .	115
3.8.4	Máquina de Vender Chocolates – Versão 2 . . . . .	118
3.9	Circuitos Complexos . . . . .	121
3.9.1	Bloco de Registradores . . . . .	121
3.9.2	Memória RAM . . . . .	123
3.9.3	Pilha . . . . .	125
3.9.4	Fila Circular . . . . .	128
<b>4</b>	<b>Memória</b>	<b>133</b>
4.1	Tipos de Memória . . . . .	133
4.2	Interface Processador-Memória . . . . .	135
4.2.1	Intertravamento dos Sinais na Interface . . . . .	136
4.2.2	Ciclo de Leitura . . . . .	137
4.2.3	Ciclo de Escrita . . . . .	137
4.3	Circuitos Integrados de Memória . . . . .	138
4.3.1	Um Bit . . . . .	139
4.3.2	Vários Bits . . . . .	140
4.3.3	Muitos Bits . . . . .	141
4.3.4	Memória com Largura Não Unitária . . . . .	145
4.3.5	Milhões de Bits . . . . .	146
4.3.6	Memória Rápida . . . . .	146
<b>5</b>	<b>O Microprocessador Mico</b>	<b>150</b>
5.1	Computador Construído com o Mico . . . . .	152
5.2	Organização do Processador . . . . .	152
5.3	Conjunto de Instruções . . . . .	154

5.3.1	Instruções de Lógica e Aritmética . . . . .	155
5.3.2	Acesso à Memória . . . . .	158
5.3.3	Saltos e Desvios . . . . .	159
5.3.4	Suporte a Funções . . . . .	162
5.3.5	Instruções de Entrada e Saída . . . . .	163
5.3.6	Instruções de Controle . . . . .	163
5.3.7	Modos de Endereçamento . . . . .	164
5.3.8	Codificação das Instruções . . . . .	164
5.4	Execução das Instruções . . . . .	167
5.4.1	Fases de Execução das Instruções . . . . .	168
5.4.2	Diagrama de Estados . . . . .	172
5.5	Interface com Memória . . . . .	173
5.5.1	Intertravamento dos Sinais na Interface . . . . .	174
5.5.2	Ciclo de Busca . . . . .	176
5.5.3	Ciclo de Leitura . . . . .	176
5.5.4	Ciclo de Escrita . . . . .	176
5.5.5	Ciclos de Entrada/Saída . . . . .	177
5.5.6	Circuito de Memória . . . . .	177
5.6	Circuito de Controle . . . . .	178
5.6.1	Sinais de Controle da Interface com Memória . . . . .	178
5.6.2	Sinais de Controle do Circuito de Dados . . . . .	180
5.6.3	Controle Microprogramado . . . . .	182
5.6.4	Busca Antecipada . . . . .	186
5.7	Interface com Periféricos . . . . .	189
<b>6</b>	<b>Programação</b> . . . . .	<b>196</b>
6.1	Acesso a Estruturas de Dados . . . . .	196
6.1.1	Cálculo de endereços . . . . .	196
6.1.2	Implementação em <i>Assembly</i> . . . . .	197
6.2	Funções . . . . .	199
6.2.1	Segmentos de Código, Dados e Pilha . . . . .	199
6.2.2	Variáveis Locais . . . . .	200
6.2.3	Avaliação de parâmetros . . . . .	200
6.2.4	Instruções de Chamada e de Retorno de Função . . . . .	201

6.2.5	Convenções . . . . .	203
6.2.6	Recursão . . . . .	205
<b>7</b>	<b>Sistemas de Memória</b>	<b>209</b>
7.1	Implementação de Sistemas de Memória . . . . .	209
7.1.1	Referências Fracionadas . . . . .	211
7.1.2	Capacidade Configurável . . . . .	213
7.2	Barramentos . . . . .	214
7.2.1	Barramento Multiplexado . . . . .	214
7.2.2	Barramento com Sobreposição de Fases . . . . .	216
7.2.3	Barramento Assíncrono . . . . .	217
7.2.4	Barramento Multimestre . . . . .	220
7.3	Acesso Direto à Memória . . . . .	222
7.4	Referências em Rajadas . . . . .	224
7.5	Referências Concorrentes . . . . .	226
7.6	Desempenho de Sistemas de Memória . . . . .	228
<b>8</b>	<b>Interfaces</b>	<b>230</b>
8.1	Interrupções . . . . .	230
8.1.1	Sinais de Interrupções . . . . .	231
8.1.2	Vetor de Interrupções . . . . .	232
8.1.3	Transação de Aceitação no Barramento . . . . .	234
8.1.4	Cadeia de Aceitação . . . . .	235
8.1.5	Salvamento do Contexto de Execução . . . . .	236
8.2	Interface Paralela . . . . .	239
8.2.1	Ligação ao mundo externo . . . . .	240
8.2.2	Ligação ao Processador . . . . .	242
8.2.3	Modos de Operação . . . . .	243
8.2.4	Programação da Porta Paralela . . . . .	244
8.2.5	Tratador de Interrupções da Porta Paralela . . . . .	246
8.3	Interface Serial . . . . .	247
8.3.1	Comunicação Serial . . . . .	247
8.3.2	Ligação ao Processador . . . . .	251
8.3.3	Programação da Interface Serial . . . . .	254

8.3.4	<i>Double Buffering</i> . . . . .	256
8.3.5	Tratador de Interrupções da Porta Serial . . . . .	257
8.4	Interfaces Analógicas . . . . .	258
8.4.1	Conversor Digital-Analógico . . . . .	259
8.4.2	Conversor Analógico-Digital . . . . .	262
8.5	Contadores e Temporizadores . . . . .	264
8.5.1	Modos de Operação . . . . .	265
<b>9</b>	<b>Modelagem de Sistemas Digitais com VHDL</b>	<b>267</b>
9.1	Somadores . . . . .	271
9.1.1	Modelo Funcional do Somador . . . . .	272
9.1.2	Modelo Estrutural do Somador . . . . .	274
9.2	Comandos Concorrentes e Sequenciais . . . . .	276
9.2.1	Comandos Concorrentes . . . . .	277
9.2.2	Processos e Comandos Sequenciais . . . . .	279
9.2.3	Programas de Teste . . . . .	281
9.3	Modelagem de Temporização . . . . .	284
9.3.1	Detecção de Bordas . . . . .	285
9.3.2	Modelo do Somador com Temporização . . . . .	289
9.4	Modelo Funcional do Mico . . . . .	299
9.4.1	Modelo em VHDL . . . . .	300
9.4.2	Modelo Funcional do PC . . . . .	301
<b>A</b>	<b>Código VHDL</b>	<b>309</b>
A.1	Programa de Teste dos Somadores . . . . .	309
A.2	Programa de Teste do Mico . . . . .	312
	<b>Referências Bibliográficas</b>	<b>316</b>
	<b>Índice Remissivo</b>	<b>319</b>



# Prefácio

Sistemas digitais e microprocessadores são, em larga medida, ubíquos em ambientes domésticos, profissionais e de lazer. Por exemplo, numa sala de estar típica de classe média (em 2012) há microprocessadores na televisão e em seu controle remoto, vários processadores digitais de sinais e microprocessadores no aparelho de DVD e em seu controle remoto, um ou mais no aparelho de *home theater* e em seu controle remoto. Dependendo do nível de sofisticação do automóvel, há ao menos um microprocessador em cada um dos freios ABS, painel de instrumentos, injeção eletrônica, um em cada *air bag*, rádio ou toca-CD/DVD/MP3. Na cozinha e área de serviço encontram-se microprocessadores no forno de micro-ondas, na geladeira, na lavadora de roupas, no condicionador de ar e em seu controle remoto. Quanto mais sofisticado é o aparelho de telefonia móvel, maior o número de microprocessadores: um para controlar o teclado, tela, e aplicativos ‘simples’ como agenda e calendário, um para controlar as funções de rádio, um ou mais para a câmera e tratamento de imagens, um para controlar o acesso à Internet e processar os aplicativos sofisticados.

O computador pessoal de mesa em que este texto está sendo editado contém uma unidade de processamento com dois processadores (*cores*), um microprocessador de 8 bits em cada uma das interfaces USB, no *mouse*, no teclado, ao menos um microprocessador de 32 bits nos dispositivos mais sofisticados como monitor, gravador de DVDs, interface de rede, interface de vídeo, disco rígido, e *modem* ADSL. Um único equipamento contém, ao menos, 16 microprocessadores de variadas capacidades de processamento e custo.

Exceto pela unidade de processamento do computador pessoal, todos os outros sistemas digitais mencionados são considerados “sistemas embarcados” ou “sistemas embutidos” (*embedded systems*). Para todos os sistemas mencionados, foi necessário desenvolver *software* do nível de aplicação e de sistema, e a maior parte do código é ‘novo’ para cada aplicação e/ou dispositivo. Portanto, comparando-se o número de linhas de código fonte ‘novo’ escritas para os sistemas mencionados acima, com o código fonte ‘novo’ escrito para os programas de aplicação do computador pessoal, a proporção é da ordem de 5-10 para 1, numa estimativa conservadora. Isso significa que as oportunidades abertas ao

projetista de *software* embarcado são algo maiores e mais interessantes do que aquelas no mercado de aplicativos para sistemas *desktop*.

Este texto contém material que é fundamental para quem pretende trabalhar no projeto e desenvolvimento de sistemas digitais e de sistemas embarcados, tanto no desenvolvimento de circuitos e sistemas de *hardware*, quanto no desenvolvimento de *software* de aplicação e de sistemas operacionais, embora não trate explicitamente destes sistemas.

O conteúdo e a apresentação deste texto são o resultado das experiências do autor ao lecionar a disciplina de Projetos Digitais e Microprocessadores no Bacharelado em Ciência da Computação da Universidade Federal do Paraná. O autor pressupõe que o leitor já teve contato com os fundamentos de lógica proposicional, circuitos combinacionais e sequenciais, em textos tais como Mano & Kime [Man02] e Katz & Borriello [KB04], bem como com uma linguagem de programação em alto nível como C. Para a compreensão do material não são necessários conhecimentos de eletromagnetismo nem de eletrônica para além daqueles adquiridos em um bom curso de Física no Ensino Médio.

Uma característica importante do texto é a ênfase na interface entre o *hardware* e o *software*, porque o texto é uma ferramenta para a formação de cientistas da computação, analistas e programadores, para quem o *hardware* é uma plataforma de trabalho, mais do que um fim em si mesmo. Portanto, o Capítulo 5 descreve cuidadosamente o conjunto de instruções do processador Mico, antes de descrever uma implementação daquele conjunto. O conjunto de instruções é a *application programming interface* (API) do processador, e este é uma implementação daquela API. O Capítulo 6 discute dois aspectos importantes na programação em C, que são o acesso a estruturas de dados e o suporte a funções, com ênfase no mapeamento entre as abstrações da linguagem C e suas traduções para o *assembly* do Mico. O Capítulo 8 introduz quatro periféricos simples como sendo interfaces entre o processador e o mundo externo àquele. O capítulo propõe um sistema de interrupções, tanto do ponto de vista do *hardware* como do *software*, e para as interfaces serial e paralela, apresenta os periféricos propriamente ditos, sua ligação ao processador, e mostra versões primitivas para os *drivers* destes dispositivos.

Esta forma de exposição do material enfatiza as APIs entre distintos níveis de abstração – sistema digital, código *assembly*, código C, componentes do sistema operacional – porque tal enfoque determina uma simetria entre distintas visões, aquela do projetista dos circuitos e sistemas digitais, e aquela do programador que se utiliza daqueles sistemas. Com esta apresentação, o estudante é colocado alternadamente na posição de quem implementa uma API, e na posição de quem a utiliza, e esta simetria é extremamente útil do ponto de vista didático [HC06].

Com a popularização das linguagens de descrição de *hardware* (LDH), tais como VHDL e Verilog, e de ferramentas para a simulação e síntese de circuitos descritos nestas linguagens, tornou-se viável estudar as disciplinas de projeto de

sistemas digitais inteiramente *in abstracto*, com os projetos desenvolvidos somente em termos de descrições em uma LDH, com todos os circuitos abstraídos para sinais e processos. Na experiência do autor, e do ponto de vista didático, esta abordagem só é efetiva se o estudante possui uma sólida compreensão do funcionamento concreto de circuitos digitais, e de seu comportamento temporal. Sem a compreensão do concreto, a visão abstrata pode tornar-se demasiado distante da realidade. Frequentemente, as consequências deste distanciamento são projetos irrealizáveis.

A organização dos conteúdos obedece a sequência usual no estudo de Sistemas Digitais: uma abstração do comportamento dos sinais elétricos baseada em Álgebra Booleana, circuitos combinacionais, e circuitos sequenciais. Circuitos de memória são introduzidos a seguir e então o projeto e a implementação do microprocessador, seguidos por uma introdução à interface entre linguagem de alto nível e o *hardware* que a executa. São estudados vários projetos de barramentos, em ordem crescente de desempenho e complexidade, e são apresentados quatro periféricos simples, frequentemente encontrados em sistemas embarcados. O último capítulo apresenta um pequeno subconjunto da linguagem VHDL como veículo para a modelagem do microprocessador Mico. Os conteúdos de cada capítulo são apresentados no que se segue.

O Capítulo 1 introduz a abstração de *sinais como bits*, introduz operações sobre bits, define suas propriedades, e introduz um formalismo para especificar circuitos, inspirado na Linguagem Z. Segue-se a descrição e especificação formal de três circuitos combinacionais básicos: multiplexadores, demultiplexadores e seletores. Exemplos de circuitos combinacionais complexos ilustram o uso de componentes da família TTL-74xx. O capítulo se encerra com a descrição, no nível funcional, de circuitos de tecnologia CMOS. Outros exemplos de circuitos combinacionais complexos – três projetos de somadores e uma Unidade de Lógica e Aritmética – são discutidos no Capítulo 9, no contexto da modelagem em VHDL de componentes de processadores.

Circuitos Sequenciais são investigados no Capítulo 3, com uma revisão sobre o comportamento de circuitos com memória, com ênfase em *flip-flops*. O comportamento de vários contadores e registradores de deslocamento é formalizado, empregando-se como exemplo componentes da família TTL-74xx. São então apresentados vários conceitos relacionados à temporização de circuitos sequenciais síncronos e discute-se o projeto de máquinas de estados finitas. Tomando como base o projeto de uma máquina de vender chocolates, são apresentadas duas implementações para o seu controlador, sendo que na segunda emprega-se a técnica de implementação com microcontroladores. Dentre os exemplos complexos discutidos no final do capítulo destaca-se o bloco de registradores que é empregado no processador.

O Capítulo 4 introduz circuitos de memória, apresentando de forma relativamente abstrata o projeto de circuitos de memória de grande capacidade. A

interface destes circuitos com o mundo externo a eles é discutida num nível suficiente de detalhe para facilitar a compreensão do material do Capítulo 5. Este tópico, a interface entre memória e processador, é retomado e expandido no Capítulo 7.

O microprocessador Mico é apresentado no Capítulo 5. Este processador é uma adaptação para 16 bits do MIPS R3000 descrito em Patterson & Hennessy [PH09]. O conjunto de instruções do Mico é precisamente definido, e uma implementação daquele conjunto é apresentada em detalhe. Esta implementação emprega uma disciplina de temporização na qual cada instrução é executada num certo número de ciclos (curtos) de relógio, que é chamada de “disciplina multiciclos”. O Capítulo 9 apresenta um modelo VHDL do processador projetado com a “disciplina de temporização ciclo longo”, sob a qual cada instrução é executada em um único ciclo (longo) de relógio. Este capítulo ainda introduz duas técnicas de projeto para as interfaces do processador com memória e periféricos.

O Capítulo 6 faz a ligação de dois níveis de abstração da programação, (i) entre programas escritos na linguagem de alto nível C, e (ii) sua tradução para a linguagem de montagem (*assembly*) do Mico. Discute-se a relação entre o endereçamento implícito de componentes de estruturas de dados em C, e o endereçamento que deve ser explicitado no código em *assembly*. Um protocolo para o código que permite a implementação de funções é apresentado. O capítulo se encerra com o estudo sobre o comportamento do programa recursivo que computa o fatorial.

Com base na descrição do processador, versões mais sofisticadas para a interface entre o sistema de memória e o processador são introduzidas no Capítulo 7. São apresentados projetos de barramentos com níveis crescentes de concorrência entre as transferências, portanto com melhor desempenho e maior complexidade.

O Capítulo 8 introduz os periféricos mais simples que são encontrados em sistemas de pequeno porte, tais como interfaces paralela e serial, contadores e conversores analógico-digitais. Um sistema realista de sinalização e tratamento de interrupções é apresentado, com exemplos simplificados de tratadores de interrupções das interfaces paralela e serial.

A modelagem de sistemas digitais em VHDL é introduzida no Capítulo 9. Um subconjunto relativamente pequeno das construções da linguagem é apresentado como um efeito colateral da discussão sobre modelos de circuitos realistas, tais como somadores rápidos com cadeia de adiantamento de vai-um, e do circuito de controle de fluxo de execução do Mico, que compreende o contador de programa e o circuito de decodificação das instruções de saltos e desvios. O Apêndice contém o código dos programas de teste dos circuitos detectores de bordas – que são o veículo para a discussão sobre modelagem de tempo –, e do modelo funcional do Mico.

Os exercícios não são agrupados ao final dos capítulos, mas estão distribuídos ao longo das seções porque eles são uma parte importante do texto e foram escolhidos para reforçar a compreensão do material bem como expandir os conteúdos e ideias apresentadas.

### **Agradecimentos**

O autor agradece aos alunos da disciplina Projetos Digitais de Microprocessadores, das turmas de 1997 a 2004, e após 2010, que foram, ao mesmo tempo, a inspiração para este texto, e as cobaias de sua aplicação em Projetos. Agradeço também aos alunos das turmas de Organização e Arquitetura de Computadores, de 1995 a 2011, por propiciarem a oportunidade, o aprendizado, e a experiência de tentar expandir o que aqueles haviam apreendido em Projetos. É o desejo do autor que eventuais danos que lhes possa ter infligido nestas disciplinas sejam perdoados com o passar dos anos.

As aulas espetacularmente claras de Anatólio Laschuk, na Escola de Engenharia da UFRGS, serviram de inspiração para este trabalho – espero lhes ter feito justiça. Com Jeff Sanders aprendi o valor de especificar clara e precisamente o que se tenta projetar e implementar.

Agradecimentos são devidos aos colegas do Departamento de Informática da UFPR, pelo ambiente de trabalho cordial e produtivo ao longo destes anos, em especial, aos colegas da área de Sistemas: Aldri L. dos Santos, André L. P. Guedes, Armando L. N. Delgado, Bruno Müller Jr, Cristina D. Murta, Eduardo Todt, Elias P. Duarte Jr, Fabiano Silva, Heinz A. Niederheitmann Jr, Hélio Pedrini, Heraldo F. Madeira, Luis Allan Kunzle, Luis Carlos E. de Bona, Luiz Carlos P. Albin, Michele N. Lima, Nelson Suga, Renato J. S. Carmo, e Wagner N. Zola.

A forma de apresentação da abstração de sinais como bits resultou de uma sugestão de Renato Carmo. Maria Cristina Périgo fez um excelente trabalho ao revisar o texto. Evidentemente, todos os erros remanescentes são de responsabilidade do autor.

# Referências Bibliográficas

- [Ash96] Peter J Ashenden. *The Designer's Guide to VHDL*. Morgan Kaufmann, 1996. ISBN 1558602704.
- [Bac86] Maurice J Bach. *The Design of the UNIX Operating System*. Prentice-Hall, 1986. ISBN 0132017997.
- [Bal03] Mark Balch. *Complete Digital Design*. McGraw-Hill, 2003. ISBN 0071409270.
- [BJ97] Gerrit A Blaauw and Frederick P Brooks Jr. *Computer Architecture: Concepts and Evolution*. Addison-Wesley, 1997. ISBN 0201105578.
- [Bob87] Leonard S Bobrow. *Elementary Linear Circuit Analysis*. Holt, Rinehart & Winston, 1987. ISBN 0030072980.
- [Bro04] Stuart Brorson. Circuit simulation using gEDA and SPICE – HOWTO. Páginas html, Electroniscript, inc., 2004. Disponível em <<http://www.brorson.com/gEDA/SPICE/intro.html>>. Acesso em 8/5/2012.
- [Byb12] Anthony Bybell. GTKWave – visualization tool for VCD, LXT, LXT2, VZT, FST, and GHW files, 2012. Disponível em <<http://gtkwave.sourceforge.net>>. Acesso em 11/7/2012.
- [CH05] Tadeu Carmona and Roberto A Hexsel. *Universidade Redes*. Editora Digerati, 2005. ISBN 8589535797.
- [Cla80] Wesley A Clark. From electron mobility to logical structure: A view of integrated circuits. *ACM Computing Surveys*, 12(3):325–356, Sep 1980.
- [CM06] Ney Calazans and Fernando Moraes. Simulação VHDL do processador MRStd. Páginas html, Faculdade de Informática, PUC-RS, 2006. Disponível em <[http://www.inf.pucrs.br/~calazans/undergrad/orgcomp\\_EC/mips\\_mono/mips\\_v0.vhd](http://www.inf.pucrs.br/~calazans/undergrad/orgcomp_EC/mips_mono/mips_v0.vhd)>. Acesso em 8/5/2012.
- [Fle97] William I Fletcher. *An Engineering Approach to Digital Design*. Prentice Hall, 1997. ISBN 9780132776998.
- [Gin12] Tristan Gingold. GHDL – G Hardware Design Language, 2012. Disponível em <<http://ghdl.free.fr>>. Acesso em 11/7/2012.
- [HC06] Roberto A Hexsel and Renato Carmo. Ensino de Arquitetura de Computadores com enfoque na interface hardware/software. In *WEAC'06: Workshop sobre Educação em Arquitetura de Computadores*, pages 9–16, 2006.

- [Hex01] Roberto A Hexsel. Redes de dados: Tecnologia e programação. Relatório Técnico RT-DINF 005/2001, Depto. de Informática, UFPR, 2001. Disponível em <[http://www.inf.ufpr.br/roberto/rt005\\_01.pdf](http://www.inf.ufpr.br/roberto/rt005_01.pdf)>. Acesso em 8/5/2012.
- [HP90] John L Hennessy and David A Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 1st edition, 1990. ISBN 1558600698.
- [HP12] John L Hennessy and David A Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 5th edition, 2012. ISBN 9780123838728.
- [HU79] John E Hopcroft and Jeffrey D Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979. ISBN 020102988X.
- [JNW08] B L Jacob, S W Ng, and D T Wang. *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2008. ISBN 0123797513.
- [Kat94] Randy H Katz. *Contemporary Logic Design*. Benjamin-Cummings, 1994. ISBN 0805327037.
- [KB04] Randy H Katz and Gaetano Borriello. *Contemporary Logic Design*. Prentice Hall, 2004. ISBN 978-0201308570.
- [KL96] Sung-Mo Kang and Yusuf Leblebici. *CMOS Digital Integrated Circuits: Analysis and Design*. McGraw-Hill, 1996. ISBN 0070380465.
- [Koh78] Zvi Kohavi. *Switching and Finite Automata Theory*. Tata McGraw-Hill, 2nd edition, 1978. ISBN 0070993874.
- [Kor01] Israel Koren. *Computer Arithmetic Algorithms*. A K Peters, 2nd edition, 2001. ISBN 1568811608.
- [KR88] Brian W Kernighan and Dennis M Ritchie. *The C Programming Language*. Prentice Hall, 2nd edition, 1988. ISBN 9780131103628.
- [Man02] M Morris Mano. *Digital Design*. Prentice Hall, 3rd edition, 2002. ISBN 01306211218.
- [MC80] Carver Mead and Lynn Conway. *Introduction to VLSI Systems*. Addison-Wesley, 1980. ISBN 0201043583.
- [MK00] M Morris Mano and Charles R Kime. *Logic and Computer Design Fundamentals*. Prentice Hall, 2nd edition, 2000. ISBN 0130124680.
- [Mou01] Arnaldo V Moura. *Especificações em Z*. Editora da Unicamp, 2001. ISBN 8526805754.
- [PD11] Larry L Peterson and Bruce S Davie. *Computer Networks: A Systems Approach*. Morgan Kaufmann, 5th edition, 2011. ISBN 9780123850591.
- [Per93] Douglas L Perry. *VHDL*. McGraw-Hill, 2nd edition, 1993. ISBN 0070494347.
- [Per02] Douglas L Perry. *VHDL: Programming By Example*. McGraw-Hill, 4th edition, 2002. ISBN 0071400701.
- [PH05] David A Patterson and John L Hennessy. *Computer Organization & Design: The Hardware/Software Interface*. Morgan Kaufmann, 3rd edition, 2005. ISBN 1558606041.

- [PH09] David A Patterson and John L Hennessy. *Computer Organization & Design: The Hardware/Software Interface*. Morgan Kaufmann, 4th edition, 2009. ISBN 9780123744937.
- [PT96] David Pellerin and Douglas Taylor. *VHDL Made Easy!* Prentice Hall, 1996. ISBN 0136507638.
- [RCN03] Jan M Rabaey, Anantha Chandrakasan, and Borivoje Nikolic. *Digital Integrated Circuits – A Design Perspective*. Prentice Hall, 2nd edition, 2003. ISBN 0130909963.
- [S<sup>+</sup>99] Richard Stallman et al. GCC, the GNU Compiler Collection. Páginas html, Free Software Foundation, 1999. Disponível em <<http://gcc.gnu.org>>. Acesso em 28/7/2013.
- [San90] Jeff W Sanders. Lectures on the foundations of hardware design. Lecture notes, Programming Research Group, Oxford University Computing Laboratory, 1990.
- [SCO96] The Santa Cruz Operation SCO. *System V Application Binary Interface – MIPS RISC Supplement*. 3rd edition, 1996.
- [Sha96] Tom Shanley. *Pentium Pro Processor System Architecture*. MindShare Addison-Wesley, 1996. ISBN 0201479532.
- [Spi89] J M Spivey. *The Z Notation*. Prentice Hall, 1989. ISBN 013983768X.
- [SS90] Adel S Sedra and Kenneth C Smith. *Microeletronic Circuits*. Holt, Rinehart & Winston, 3rd edition, 1990. ISBN 003051648X.
- [Swe07] Dominic Sweetman. *See MIPS Run – Linux*. Morgan Kaufmann, 2nd edition, 2007. ISBN 0120884216.
- [Tan92] Andrew S Tanenbaum. *Modern Operating Systems*. Prentice-Hall, 1992. ISBN 0135881870.
- [Tan99] Andrew S Tanenbaum. *Organização Estruturada de Computadores*. Livros Técnicos e Científicos, 4th edition, 1999. ISBN 8521612532.
- [TS89] Herbert Taub and Donald Schilling. *Digital Integrated Electronics*. McGraw-Hill, 1989. ISBN 007Y857881.
- [WE92] Neil H E Weste and Kamran Eshraghian. *Principles of CMOS VLSI Design: A Systems Perspective*. Addison-Wesley, 1992. ISBN 0201533766.
- [WH10] Neil Weste and David Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison-Wesley, 4th edition, 2010. ISBN 0321547748.



# Índice Remissivo

## Símbolos

$(p, q)$ , 145  
 $R^+$ , 145  
 $X(n..m)$ , 145  
 $[p, q]$ , 145  
 $\Leftarrow$ , 145  
 $\equiv$ , 21  
 $\gg$ , 41, 145  
 $\wedge$ , 19, 21  
 $\triangleleft \triangleright$ , 21, 22  
 $\leftarrow$ , 76, 145  
 $\Leftrightarrow$ , 21  
 $\Rightarrow$ , 21, 23  
 $\ll$ , 41, 145  
 $\neg$ , 19, 21, 47  
 $\vee$ , 19, 21  
 $\oplus$ , 21, 26, 30  
 $\mapsto$ , 21  
 $\bar{a}$ , veja  $\neg$   
*demux*, 33  
*ext*, 152, 155  
*mod*, 43, 84, 205  
*mux*, 31  
*sel*, 34  
 $\&$ , 145, 267  
**B**, 18  
 $[r]$ , 102, 104  
 $[v]$ , 256  
**N**, 21  
*num*, 22, 23, 27, 34, 125, 256  
 $num^{-1}$ , 27  
 $,$ , 145  
**R**, 21, 252, 254  
 $;$ , 145  
 $|N|$ , 83, 160  
 $\langle \rangle$ , 18, 21, 83, 95

## Números

6800, 294  
6805, 294  
7400, 36  
7402, 36  
7404, 36

7474, 36  
8085, 183, 294  
8086, 149, 183, 206, 207, 294  
16550, 251  
68000, 294  
74138, 36, 38, 39  
74151, 37, 40  
74154, 37, 39  
74163, 84, 110, 115, 123  
74191, 87, 120, 122, 123  
74194, 84, 89  
74374, 97  
80486, 149, 184

## A

ABI, 197  
abstração, 17, 18, 144, 262  
bits como sinais, 16–20  
Acesso Direto à Memória, veja ADM  
adição, 47  
adiantamento de vai-um, 288  
ADM, 217, 218  
arbitragem de barramento, 218  
transferência em rajada, 218  
Álgebra de Boole, 18  
API, 149  
apontador de pilha, 119, 156, 194  
*Application Binary Interface*, veja ABI  
*Application Programming Interface*, veja API  
arbitragem de barramento, 214, 218, 229  
arquitetura, 148  
Harvard, 182, 295  
Princeton, 182  
*assembly*, 148, 151, 184, 191–202  
associatividade, 20  
atraso,  
de transporte, 278  
inercial, 278  
atribuição,  
assíncrona, veja  $\Leftarrow$   
síncrona, veja  $\leftarrow$   
autômatos finitos, 100

## B

báscula, 74  
 mestre-escravo, 83  
 relógio multifase, 83  
 barramento, 63, 140, 203  
 assíncrono, 212  
 do Mico, 168, 210  
 do PentiumPro, 222  
 multimestre, 214  
 multiplexado, 208  
 relógio, 211  
 RESULT, 164, 296  
 segmentado, 220  
 sobreposição de fases, 210  
*barrel shifter*, 43  
*big endian*, 206  
 bit de final, 243  
 bit de início, 243  
*bitfields*, 238, 248  
 bits, 16–21, 30  
 definições da abstração, 18  
 expressões e fórmulas, 21  
 propriedades da abstração, 19  
 variável, 21  
 bits por segundo, 242  
 bloco de registradores, 115, 147, 174, 263  
*bootstrapping*, veja inicialização  
 bps, 242  
*buffer*,  
 de barramento, 210, 223  
 de escrita, 210  
 de recepção, 251  
 de transmissão, 250  
*buffer three-state*, 62  
*buffer tri-state*, 137, 174  
 busca, 170, 177, 179, 228, 296  
 antecipada, 181  
 busca binária, 126

C

C, linguagem, 107, 190  
 código,  
 ASCII, 233  
 Gray, 39, 83, 101  
 recursivo, 231  
 reentrante, 231  
 caminho crítico, 94  
 campo de bits, veja *bitfields*  
 capacitor, 68  
*chip select*, 117, 135, 168, 234, 236, 246  
 CI, 36  
 ciclo,  
 de ADM, 217  
 de barramento, 130  
 de busca, 170, 177

de entrada, 171  
 de escrita, 131, 138, 170  
 de interrupção, 226, 228  
 de leitura, 131, 137, 170  
 de memória, 131, 168  
 de saída, 171  
 de trabalho, 92  
 período mínimo, 94  
 ciclos por instrução, veja CPI  
 circuito,  
 combinacional, 31, 73  
 dual, 61  
 segmentado, 97  
 sequencial, 73  
 CMOS, 27, 30, 50–65  
*buffer three-state*, 62  
 inversor, 59  
 nand, 61  
 nor, 60  
 porta de transmissão, 63  
*column address strobe*, 135  
 comparação de magnitude, 49, 125  
 comparador, 256  
*Complementary Metal-Oxide Semicon-*  
*ductor*, veja CMOS  
 complemento, veja  $\neg$ , 20  
 complemento de dois, 48, 49  
 comunicação serial, 241  
 comutatividade, 20  
 concatenação, veja &  
 condicional, veja  $\triangleleft$   $\triangleright$   
 conjunção, veja  $\wedge$   
 conjunto de instruções, 148  
 conjunto mínimo de operadores, 30  
 contador, 78  
 74163, 84  
 74191, 87  
 assíncrono, 79  
 em anel, 82, 92, 97  
 Johnson, 83  
 módulo-8, 79  
 modelo VHDL, 298, 299  
 programável, 247, 258  
 síncrono, 80  
 contador de programa, veja PC  
 contexto de execução, 230  
 controlador, 103, 109  
 de ADM, 217  
 de memória, 140  
 controle de fluxo de execução, 298  
 convenções de programação, 197  
 conversor,  
 analógico-digital, 256  
 digital-analógico, 253  
 paralelo-série, 88, 143, 242

série-paralelo, 88, 242  
CPI, 180, 182

**D**

*daisy chain*, 215, 229  
DDR-SDRAM, 129  
decodificador, 37  
  de endereços, 204  
  de prioridades, 40, 226  
demultiplexador, 33, 37  
deslocamento, 41  
  aritmético, 41, 49  
  exponencial, 42, 65, 94  
  instruções, 150, 301  
  lógico, 41, 45  
  rotação, 43  
deslocamento à direita, *veja*  $\gg$   
deslocamento à esquerda, *veja*  $\ll$   
detecção de bordas, 279  
diagrama de estados, 98, 167, 187  
disciplina de temporização, 293  
disjunção, *veja*  $\vee$   
distributividade, 20, 25  
divisão inteira, 23, 126  
divisor de frequência, 92, 258  
DMA, *veja* ADM  
*don't care*, 265  
dopante, 51  
*double buffering*, 143, 250  
DRAM, 128, 135  
  controlador, 136, 140  
  DDR-SDRAM, 143  
  *extended data out*, 128  
  *fast page mode*, 128, 140  
  página, 140  
  *page mode*, 128  
  SDRAM, 143  
  VRAM, 143  
*driver*, *veja* tratador de interrupção  
  interface paralela, 239  
  interface serial, 249  
dual, 20, 57, 61  
*duty cycle*, 92

**E**

E/S, 183  
  espera ocupada, 251  
  por interrupção, 252  
  por programa, 252  
  portas, 157  
EEPROM, 128  
endereçamento, 158  
  a byte, 186  
  bancos, 218  
  base-deslocamento, 152, 164

  base-deslocamento escalado, 187  
  capacidade configurável, 207  
  de periféricos, 157, 183, 236  
  E/S como E/S, 157, 183, 296  
  E/S como memória, 183, 296  
  fração de palavra, 205  
  indireto à memória, 187  
  mapeamento híbrido de E/S, 184  
  memória, 204  
  pseudoabsoluto, 154  
  relativo ao PC, 154  
endereçamento com RAS e CAS, 135  
endereço,  
  alocação de vetores, 190  
  de retorno, 156, 194  
  efetivo, 131, 152, 170, 171, 191  
  segmentos lógicos, 193  
entrada e saída, *veja* E/S  
EPC, 227, 229  
EPROM, 128  
equação característica do FF, 75  
equivalência, *veja*  $\Leftrightarrow$   
espaço de endereçamento, 146, 183  
especificação, 22, 149, 262  
espera ocupada, 251  
exceção, 187  
*ExceptionPC*, *veja* EPC  
execução paralela, *veja* ,  
execução sequencial, *veja* ;  
*extended data out*, 141

**F**

*fan-out*, 70  
fases de execução, 162  
  acesso à memória, 164  
  busca, 163  
  decodificação, 163  
  execução na ULA/MULT, 163  
  máquina de estados, 166  
  mudança de fluxo, 165  
  NOP e HALT, 165  
  resultado, 164  
*fast page mode*, 140  
fechamento, 19  
FET, 54  
*Field Effect Transistors*, *veja* FET  
FIFO, 122  
fila, 122–125, 251  
  circular, 122  
FLASH, memória, 128  
*flip-flop*, 75  
  comportamento, 76  
  modelo VHDL, 275  
frequência máxima de operação, 94  
função, 193

endereço de retorno, 194  
 folha, 194  
 parâmetros, 194  
 pilha, 193, 194, 200  
 protocolo de entrada, 198, 231  
 protocolo de saída, 198  
 registro de ativação, 198  
 tratador de interrupção, 230  
 valor de retorno, 194  
 variáveis locais, 194  
 função de próximo estado, 100, 102  
 função de saída, 100, 102  
 função de transferência, 254, 256  
 função, aplicação bit a bit, 20  
 função, tipo (op. infixo), *veja*  $\mapsto$

**G**

gcc, 190  
 ghd1, 282  
 gtkwave, 282

**H**

*handshake*, *veja* protocolo  
*heap*, 193  
*hold time*, 93, 301

**I**

IA32, 149  
 idempotência, 19  
 identidade, 19  
 implementação, 22, 149  
 implicação, *veja*  $\Rightarrow$   
 inicialização, 169, 208, 228  
 instrução,  
   addm, 187  
   clrStatus, 188  
   cnst, 186  
   desvio, 154  
   di, 230  
   ei, 230  
   halt, 157  
   inp, 157, 167  
   j, 153  
   jal, 156, 195  
   jr, 156, 195  
   la, 191  
   lb, 186  
   ld, 131, 186, 191, 205, 294, 301  
   ldm, 187  
   lds, 187  
   leStatus, 188  
   madd, 188  
   nop, 157  
   otp, 157, 167  
   reti, 227, 229

salto, 153  
 sb, 186  
 setStatus, 188  
 st, 131, 186, 191, 205  
 sts, 187  
 instruções,  
   busca, 163  
   codificação, 159  
   decodificação, 163, 296  
   dinâmicas, 180  
   formato, 158  
 interface, 224  
   analógica, 252–258  
   controladores, 258–260  
   EIA232, 242, 244  
   intertravamento dos sinais, 130  
   paralela, 184, 232–240  
     interrupção, 240  
     programação, 238  
   processador-memória, 129, 167, 172, 203  
   processador-periféricos, 183  
   serial, 184, 240–252  
     interrupção, 252  
     programação, 248  
 interrupção, 187, 224–232, 240, 252  
   cadeia de aceitação, 229  
   programação, 230  
 intervalo,  
   aberto, *veja*  $[p, q]$   
   fechado, *veja*  $(p, q)$   
 intervalo de amostragem, 253  
 involução, 19, 29

**L**

largura do passo, 191  
 latência, 222  
*latch*, *veja* búscula  
 LIFO, 119  
 ligação,  
   barramento, 63  
   em paralelo, 56, 61  
   em série, 55, 61  
   endereço de retorno, 197  
 linguagem de máquina, 151  
 linha de memória, 219  
*little endian*, 206

**M**

máquina de estados, 98, 100, 166, 244  
   barramento assíncrono, 212  
   Mealy, 100, 102, 106, 122  
   Moore, 100, 102, 105, 120  
   projeto, 102  
 Máquina de Mealy, *veja* máq. de estados

Máquina de Moore, *veja* máq. de estados  
 máximo e mínimo, 19  
 módulo de contagem, 79  
 MADD, 95, 188, 302  
*malloc*, 193  
 Mapa de Karnaugh, 24, 41, 71, 105, 114  
 memória, 167, 168  
   bit, 74  
   circuito integrado, 132  
   DDR-SDRAM, 129  
   desempenho, 222  
   dinâmica, 136  
   EDO, 128, 141  
   FLASH, 128  
   FPM, 128, 140, 219  
   microROM, 176  
   não volátil, 128  
   RAM, 117, 127  
   ROM, 107, 127  
   rom8K, 171  
   SDRAM, 129  
   sram8K, 171  
   tipos de RAM, 128  
   tipos de ROM, 128  
   volátil, 128  
 metaestabilidade, 75  
 Mico, 144–190, 293–300  
   *assembly*, 188  
   codificação das instruções, 158  
   conjunto de instruções, 148  
   controlador, 172  
   diagrama de estados, 166  
   entidade VHDL, 296  
   execução das instruções, 161  
   interface com memória, 167  
   interface com periféricos, 183  
   modelo funcional, 262, 264, 293  
   modos de endereçamento, 158  
   tabela de controle, 296  
 microcontrolador, 107, 109, 112, 177  
   controle de fluxo, 109  
   VAX11-780, 111  
 microinstrução, 109, 176, 177, 179, 296  
 microPC, 114, 176  
 microprocessador, 145, 146, 184, 293  
 microprograma, 109, 178, 296  
   estreito, 179  
   largo, 179, 296  
 microprogramação, 106–115, 176–179  
    $\mu I$ , 176  
    $\mu PC$ , 109, 176  
 microROM, 114  
 microrrotina, 176, 179  
 MIPS, 197, 205, 206, 229, 293  
 MIPS R3000, 145

modelo, 262  
   *dataflow*, 263  
   estratificado, 144  
   estrutural, 263, 293  
   funcional, 23, 263, 293  
   RTL, 263  
   temporal, 265, 278, 293  
 modelo funcional,  
   *flip-flop*, 275  
   Mico, 293  
   multiplexador, 272  
   RAM, 306  
   somador, 267  
 modos de endereçamento, 148, 158  
 montador, *veja assembly*  
   addrHL, 155, 192  
   pseudoinstrução, 192  
 MOSFET, 54  
 multiciclo, 294  
 multiplexador, 22, 29, 31, 37, 63, 64  
   modelo funcional, 272  
 multiplicador,  
   paralelo, 301, 302  
   segmentado, 301  
   serial, 90  
   somadas repetidas, 125  
*multiply-add*, *veja* MADD

## N

níveis de abstração, *veja* abstração  
 negação, *veja*  $\neg$   
*net list*, 263  
 nível lógico,  
   0 e 1, 18  
   indeterminado, 17, 63, 210

## O

onda quadrada, 91  
*opcode*, 158, 159, 178, 296  
 operação,  
   binária, 18, 149  
   bit a bit, 20, 151  
   com *carry*, 150, 301  
   MADD, 94  
   unária, 18, 149  
 operações sobre bits, 18–20  
 operadores lógicos, 21  
 ou exclusivo, *veja*  $\oplus$   
 ou inclusivo, *veja*  $\vee$   
*output enable*, 117, 135, 168  
*overflow*, 49

## P

paridade, 143, 245  
 PC, 154, 162, 174, 229, 230

modelo VHDL, 298  
 Pentium, 149  
 período mínimo do relógio, 94  
 periféricos, 183, 224  
 pilha, 119–122, 193, 194  
*pipelining*, veja segmentação  
 piso, veja [v]  
*pop*, 119  
 porta lógica,  
   and, 27  
   modelo temporal, 279  
   de transmissão, 63, 74  
   nand, 28, 30, 36, 61  
   nor, 28, 36, 60  
   not, 27, 36, 59  
   modelo temporal, 279  
   or, 27  
   modelo temporal, 279  
   xor, 28, 30  
 porta paralela, 184, 233  
   reg. de controle, 238  
   reg. de dados, 239  
   reg. de *status*, 238  
 porta serial, 246  
   reg. de controle, 248  
   reg. de dados, 249  
   reg. de *status*, 248  
 precisão, 255  
 prioridade,  
   circular, 215, 216  
   de interrupções, 226  
   posicional, 215, 216, 229  
 processo, 230  
 programa de teste, 276, 277, 280  
   detector de bordas, 281  
   Mico, 306  
   somador, 303  
 PROM, 128  
 protocolo, 234  
   assíncrono, 212, 235  
   EIA232, 244  
   porta paralela, 234  
   serial assíncrono, 243  
   tratamento de interrupção, 230  
 pseudoinstrução, 192  
*pull-up*, 63  
*push*, 119

## R

RAM, 127  
   modelo funcional, 306  
 RamBus, 129  
 RamLink, 129  
 recursão, 199  
 redução por soma, 202

referência em rajada, 218  
 referências concorrentes, 220  
*Register Transfer Language*, veja RTL  
 registrador, 77, 97, 112, 115  
   base, 191  
   carga paralela, 77  
   contador de programa, veja PC  
   de entrada, 246  
   de instrução, veja RI  
   de segmento, 95  
   de *status*, veja STAT  
   *ExceptionPC*, veja EPC  
   interface com memória, 164  
 registrador de deslocamento, 88, 143  
   74194, 89  
   paralelo-série, 242, 251  
   série-paralelo, 242  
 registradores, 149  
   de controle, 233, 247, 258  
   de entrada, 233  
   de saída, 233, 246  
   de *status*, 233, 237, 247, 258  
   invisível, 162  
   ra, 157, 195  
   sp, 157, 196  
   visível, 148, 161, 230  
 registro de ativação, 198  
 relógio, 91  
   ciclo de trabalho, 92  
   de tempo-real, 260  
   frequência máxima, 94  
   multifase, 82  
*reset*, 76  
 resolução, 253, 254  
*return address*, veja registradores, ra  
 RI, 147, 162, 163, 174, 179, 181, 182  
 ROM, 127  
 rotação, 43, 49  
*row address strobe*, 135  
 RTL, 96, 263, 302

## S

salvamento de registradores, 230  
 SDRAM, 129  
 segmentação, 95, 301  
 seletor, 34, 38  
 semicondutores, 50  
 sequência de operações, veja ;  
 sequência de valores, veja [p, q]  
*setup time*, 93, 96, 301  
 sinal analógico, 252  
 síntese, veja VHDL, síntese  
 sistema de memória, 203  
*sizeof*, 190, 192  
 soma, veja somador

soma de produtos, 24, 25  
 somador, 46  
   adiantamento de vai-um, 97, 288, 291  
   completo, 46, 47, 268  
   entidade VHDL, 266  
   modelo com temporização, 283  
   modelo estrutural, 268  
   modelo funcional, 267  
   serial, 90  
   temporização, 96  
 spice, 17  
 SRAM, 128  
 stack pointer, *veja* registradores, sp  
 start bit, 243  
 STAT, 150, 176, 232, 295  
 stop bit, 243  
 string, 201  
 subtração, 47

## T

tabela de excitação do FF, 75  
 tabela verdade, 23  
 tamanho, *veja* |N|  
 tempo,  
   de bit, 242  
   de execução, 180  
   de propagação, 70, 74, 278  
 tempo médio de acesso à memória, 218  
 temporização,  
   ciclo curto, 294  
   ciclo longo, 293  
   somador, 278, 283, 291  
 temporizador, 259  
 Teorema,  
   DeMorgan, 29, 30, 57, 60  
   Dualidade, 61  
   Simplificação, 24  
 terceiro estado, *veja* three-state  
 testbench, *veja* programa de teste  
 teto, *veja* [r]  
 three-state, 62  
 tipo de sinal, 21  
 transação, 212  
   saída paralela, 235  
 transação no barramento,  
   interrupção, 228  
   referências concorrentes, 220  
   transferência em rajada, 218  
   transferência por ADM, 217  
 transistor, 52  
 Transistor-Transistor Logic, *veja* TTL  
 transistores CMOS, 57  
 transmissão,  
   dúplex, 245  
   semidúplex, 245

  serial, 241  
 tratador de interrupção, 224, 227, 240,  
   252  
 tri-state, 133, 135, 167, 168, 209, 265  
 TTL, 27, 30, 36, 91  
 tupla, *veja* ⟨⟩

## U

UART, 245  
   16550, 251  
   erro de paridade, 245  
   erro de sobrescrita, 251  
 ULA, 44–49, 115, 149–152, 176, 263, 294  
   status, 48  
 Unidade de Lógica e Aritmética, *veja*  
   ULA  
 unidades de projeto, 265  
 Universal Asynchronous Receiver-  
   Transmitter, *veja* UART  
 USB, *Universal Serial Bus*, 244

## V

vazão, 222  
 velocidade de transmissão, 241  
 velocidade máxima de operação, 93  
 vetor de bits, 18, *veja* ⟨⟩  
 vetor de interrupções, 227  
 VHDL, 145, 265–293  
   área concorrente, 271  
   **assert**, 276  
   atraso,  
     de transporte, 278  
     inercial, 278  
   atribuição, 271  
     condicional, 272  
     selecionada, 272  
   comandos concorrentes, 271  
   comandos sequenciais, 274  
   delta, 270  
   entidade, 266  
   evento, 270  
   **for**, 274  
   **generate**, 288  
   lista de sensibilidade, 273  
   **loop**, 274, 303, 306  
   modelo executável, 262  
   **package**, 265  
   **port map**, 269  
   processo, 273, 276  
   **record**, 291, 306  
   síntese, 262, 272, 278  
   std.logic, 265  
   testbench, *veja* programa de teste  
   tipos, 21, 265  
   unidade de projeto, 294

variável, 276  
**wait**, 273  
**while**, 274  
VRAM, 143

**W**

*write*, 135, 234

**Z**

Z, linguagem, 18