

# LINGUAGEM ASSEMBLY

## INTRODUÇÃO AO PADRÃO INTEL 8086

```
=====
[LINE]   LOC: MACHINE CODE   SOURCE
=====

[  1]      :                      ORG 100H
[  2]      :
[  3]   0100: B8 0A 00          MOV AX, 000AH
[  4]   0103: BB 0B 00          MOV BX, 000BH
[  5]   0106: 50                PUSH AX
[  6]   0107: 53                PUSH BX
[  7]   0108: E8 01 14          CALL PROCEDIMENTO
[  8]   010B: 40                INC AX
[  9]   010C: 43                INC BX
[ 10]   010D: E8 01 14          CALL PROCEDIMENTO
[ 11]   0110: 5B                POP BX
[ 12]   0111: 58                POP AX
[ 13]   0112: CD 20            INT 20H
[ 14]      :
[ 15]   0114: B8 01 00          PROCEDIMENTO: MOV AX, 0001H
[ 16]   0117: BB 02 00          MOV BX, 0002H
[ 17]   011A: 40                INC AX
[ 18]   011B: 43                INC BX
[ 19]   011C: C3                RET
=====
```





**José Augusto N. G. Manzano**



ORCID: 0000-0001-9248-7765

# Linguagem Assembly

## INTRODUÇÃO AO PADRÃO INTEL 8086

**São Paulo**  
**2021 - Propes Vivens**



© Copyright 2021 by José Augusto N. G. Manzano / Propes Vivens.

**Todos os direitos reservados.** Proibida a reprodução total ou parcial, por qualquer meio ou processo, especialmente por sistemas gráficos, microfilmicos, fotográficos, reprográficos, fonográficos, videográficos, internet, e-books. Vedada a memorização e/ou recuperação total ou parcial em qualquer sistema de processamento de dados e a inclusão de qualquer parte da obra em qualquer programa jus cibernético existentes ou que a venham existir. Essas proibições aplicam-se também às características gráficas da obra e à sua editoração, exceto pelo exporto no próximo parágrafo. A violação dos direitos autorais é punível como crime (art. 184 e parágrafos, do Código Penal, conforme Lei nº 10.695, de 07.01.2003) com pena de reclusão, de dois a quatro anos, e multa, conjuntamente com busca e apreensão e indenizações diversas (artigos 102 e 103 parágrafo único, 104, 105, 106 e 107 itens 1, 2 e 3 da Lei nº 9.610, de 19.06.1998, Lei dos Direitos Autorais).

Esta obra é distribuída gratuitamente em formato digital (somente em PDF) apenas e tão somente no sítio do autor ([www.manzano.pro.br](http://www.manzano.pro.br)) e para o que o deseja de forma impressa junto as plataformas **Clube de Autores** e **Agbook**. Nenhum outro local da Internet ou fora dela está autorizado a distribuir este material. Não é permitido este compartilhamento em outros locais ou por qualquer meio exceto o exposto neste parágrafo. Os infratores estão sujeitos a processo judicial.

O Autor acredita que as informações aqui apresentadas estão corretas e podem ser utilizadas para qualquer fim legal. Entretanto, não existe qualquer garantia, explícita ou implícita, de que o uso de tais procedimentos conduzirá sempre ao resultado desejado. O autor e a editora não poderão ser responsabilizados civilmente ou criminalmente. Os nomes de sítios e empresas, mencionados, foram utilizados apenas como ilustração, não havendo nenhum vínculo com a obra, não garantindo a sua existência nem divulgação a posteriori.

Conteúdo de acordo com Acordo Ortográfico da Língua Portuguesa, desde 1 de janeiro de 2009.

**Dados Internacionais de Catalogação na Publicação (CIP)**  
**(Câmara Brasileira do Livro, SP, Brasil)**

Manzano, José Augusto N. G.  
Linguagem Assembly [livro eletrônico] : introdução  
ao padrão Intel 8086 / José Augusto N. G. Manzano. --  
São Paulo : Propes Vivens, 2021.

PDF

Bibliografia.

ISBN 978-65-00-24498-4

1. Intel 8086 (Microprocessador) 2. Linguagens de  
montagem (Computadores) 3. Microcomputadores IBM  
I. Título.

21-70472

CDD-005.136

**Índices para catálogo sistemático:**

1. Assembly : Linguagem de programação :  
Computadores : Processamento de dados 005.136

Cibele Maria Dias - Bibliotecária - CRB-8/9427

**Revisões**

1.00 - 13/05/2021 - lançamento

1.01 - 30/06/2022 - atualização e revisão após mudanças de versões de alguns dos softwares usados

**Produção e Editoração:**

**Imagens:**

José Augusto Navarro Garcia Manzano

[canva.com](https://canva.com) e [freepng.es](https://freepng.es) (ibm-pc)

<https://www.freepng.es/png-evckf6/>

**Edição:**

Propes Vivens



# FERRAMENTAS UTILIZADAS

Produto: **EMU8086 - MICROPROCESSOR EMULATOR [Versão 4.08r11]**

Sítio: <http://www.emu8086.com> (descontinuado)

Alternativa: <https://github.com/AhmadNaserTurnkeySolutions/emu8086>

Custo: Private Licence 4,95 USD

1 Year School Licence 52,00 USD

Permanent Size Licence 295,00 USD

Produto: **ENHANCED DEBUG [Versão 1.32b]**

Sítio: <https://pcdosretro.github.io/enhdebug.htm>

Produto: **VDOS [Versão 2022.05.01]**

Sítio: <https://vdos.info/>

Produto: **VDOSPLUS [Versão 2015.11.01]**

Sítio: <http://vdosplus.org/>







# REQUISITOS DE HARDWARE E SOFTWARE

## Requisitos Comuns

- ◆ 1 MB de memória RAM;
- ◆ 10 MB de espaço em disco para instalação do compilador;
- ◆ Sistema Operacional Windows XP, Vista, 7, 8, 8.1 ou 10;
- ◆ Monitor com 1024 x 768 pontos ou superior para melhor visualização;
- ◆ Mouse ou outro periférico de apontamento;
- ◆ Modem e acesso à Internet;
- ◆ Programa emu8086 adquirido com custo financeiro;
- ◆ Programa DOS Debug versão 1.25.

**OBS.:** No Microsoft Windows 7 de 32 bits após atualizações poderão ocorrer problemas na execução do programa emu8086. Neste caso é aconselhável trabalhar com o programa em modo XP Mode. Use o modo XP Mode para rodar os programas debug do próprio XP ou o emu8086 quando estiver em uso a versão de 64 bits do sistema operacional Microsoft Windows 7 ou 10. Outra dica é executar todos os programas como administrador para minimizar eventuais problemas.

Foram feitos testes de execução do programa emu8086 no Microsoft Windows 8.1 e 10 de 32 e 64 bits e aparentemente não ocorreram problemas. No entanto na versão 64 bits o programa Microsoft Debug não é encontrado e não poderá ser executado. No entanto, há uma alternativa para executar do programa no sistema operacional Windows 8, 8.1 e 10 de 64 bits por meio do programa de virtualização vDos retratado nesta obra.

Contato com o autor: *certificado@mail.com*

Colocar em assunto o título *Linguagem Assenbly*.

Somente serão respondidos e-mails com dúvidas relacionadas, estritamente, ao que é apresentado neste trabalho. Qualquer outro tipo de dúvida ou questão fora deste contexto, não será atendida.



# RESTRIÇÃO OPERACIONAL

Devido a estrutura operacional de uso dos microprocessadores Intel 8086/8088 e do foco deste trabalho devem ser consideradas as seguintes restrições operacionais:

- ◆ Os programas apresentados nesta obra são desenvolvidos numa esfera simplista, apresentam apenas os recursos em média discutidos nos capítulos que são focados;
- ◆ Não são tratadas as entradas de valores numéricos com ponto flutuante, apenas entradas de dados baseada em valores numéricos inteiros e dados alfabéticos;
- ◆ O objetivo deste trabalho é ser uma introdução didática, o mais simples possível, apenas e tão somente para leitores que não conhecem absolutamente nada em programação Assembly;
- ◆ Não são tratados nenhum aspecto considerado intermediário ou avançado no nível de desenvolvimento dos programas ou do conteúdo da obra;
- ◆ Não é apresentado nenhum programa contextualizando o uso da linguagem Assembly para componentes de automação ou de uso comercial.



# AGRADECIMENTOS

À minha esposa Sandra e à minha filha Audrey, motivos de constante inspiração ao meu trabalho. Pela paciência nos momentos de ausência que tenho, quando estou absorto em escrever, dedicando-me ao ensino;

A Peter Norton, bandeirante desbravador, que sempre direcionou seu trabalho literário ao conhecimento mais refinado no uso dos microcomputadores padrão IBM-PC e principalmente ao processo de entendimento do funcionamento dos microprocessadores (micro controladores) Intel, como o fez com John Socha em sua obra Peter Norton's Assembly Language Book for IBM-PC.

A Alexander Popov que, com seu brilhantismo profissional, iniciou o desenvolvimento da ferramenta de estudo da linguagem de programação Assembly emu8086, e também a Tomasz Grystar, desenvolvedor do programa Flat Assembler que tem sua contribuição registrada neste trabalho, por tornarem o estudo da programação em baixo nível muito agradável e de fácil compreensão.

Ao amigo Stephen P. Morse, engenheiro que desenvolveu o microprocessador 8086/8088 e quando o fez, não imaginou que sua criação seria utilizada para compor um dos microcomputadores mais utilizados no mundo. Obrigado amigo pelo auxílio que ofereceu a mim quando interagimos sobre seu trabalho.

Há ainda uma pessoa que merece grande atenção, você, leitor, que me prestigia adquirindo esta e outras obras de minha autoria, sempre de forma legal, na livraria de sua confiança. Devido à sua postura tenho incentivo para continuar escrevendo, mesmo num País em que está enraizada a cultura da pirataria, das famigeradas apostilas e da cópia ilegal do trabalho alheio. Muito obrigado por sua confiança e respeito.

A todos os alunos que passaram e passam por minhas mãos, que acreditaram e acreditam na minha pessoa e seguiram e seguem as orientações passadas; por me incentivarem continuamente quando me questionam sobre temas que ainda não conheço, por me levarem a um patamar maior por exigirem assim que eu pesquise mais.

Vida longa e próspera.



# SUMÁRIO

## Parte I - Noções preliminares

### Capítulo 1 - Introdução

1.1 - Assembly ou Assembler .....	21
1.2 - Por que aprender Assembly? .....	22
1.3 - Restrições deste trabalho.....	23
1.4 - Por que usar o padrão 8086 e não outro mais recente? .....	25
1.5 - Arquitetura: Princípios básicos .....	25
1.6 - Código ASCII.....	27

### Capítulo 2 - Conceitos fundamentais

2.1 - Sistema computacional .....	31
2.1.1 - Unidade central de processamento .....	32
2.1.2 - Unidade de memória.....	32
2.1.3 - Unidades de entrada e de saída .....	32
2.2 - Organização de dados .....	33
2.2.1 - Nibble .....	33
2.2.2 - Byte .....	34
2.2.3 - Word .....	34
2.2.4 - Double word .....	35
2.2.5 - Quad word.....	35
2.2.6 - Unidades de medidas computacionais.....	35
2.2.7 - Sistemas de numeração e potências computacionais.....	36
2.3 - Arquitetura 8086.....	40
2.3.1 - Registradores gerais .....	44
2.3.2 - Registradores de segmento.....	45
2.3.3 - Registradores de deslocamento .....	46
2.3.4 - Registradores de estado (status flags).....	47
2.4 - Interrupções .....	48
2.5 - Segmentos e deslocamentos .....	49
2.6 - Endereçamento de memória .....	50
2.7 - A linguagem Assembly 8086.....	52
2.8 - Os modos 32 e 64 bits .....	53

## Parte II - Programação com Enhanced DEBUG

### Capítulo 3 - Ferramentas de desenvolvimento

3.1 - Como começou, como está, como fica .....	57
3.2 - Os Programas de depuração DEBUG.....	58
3.3 - Programa emulador assembler emu8086 .....	59
3.4 - Preparação do ambiente de trabalho .....	59

## **Capítulo 4 - Cálculos matemáticos básicos**

4.1 - O Programa Enhanced DEBUG .....	61
4.2 - Aritmética em modo hexadecimal.....	64
4.3 - Representação de valores negativos .....	65
4.4 - Cálculos em códigos de máquina.....	67
4.4.1 - Adição de valores hexadecimais.....	68
4.4.2 - Subtração de valores hexadecimais .....	72
4.4.3 - Multiplicação de valores hexadecimais.....	74
4.4.4 - Divisão de valores hexadecimais .....	75

## **Capítulo 5 - Apresentação de dados**

5.1 - Apresentação de um caractere.....	79
5.2 - Movimentação de dados .....	83
5.3 - Apresentar sequência de caracteres .....	87
5.4 - Apresentar valores binários .....	89
5.5 - Registradores de estado .....	95
5.6 - Instruções de salto condicional.....	99
5.7 - Execução básica de desvios .....	101
5.8 - Apresentar valor hexadecimal .....	105

## **Capítulo 6 - Entrada de dados**

6.1 - Entrada de dados por teclado .....	111
6.2 - Utilização de procedimento .....	117
6.3 - Utilização da pilha .....	119
6.4 - Consistência da entrada de dados .....	125

## **Parte III - Programação com emu8086**

## **Capítulo 7 - Operações essenciais**

7.1 - Simulador e assembler 8086 .....	131
7.2 - Programa "Alo Mundo!" .....	132
7.3 - De volta ao programa Enhanced DEBUG .....	142
7.4 - Depuração com uso da pilha.....	144
7.5 - Instruções da linguagem de montagem 8086.....	152
7.6 - Uso do modo ajuda .....	159

## **Capítulo 8 - Programação básica**

8.1 - Manipulação de registradores e dados.....	161
8.1.1 - Endereçamento imediato.....	161
8.1.2 - Endereçamento por registrador.....	161
8.1.3 - Endereçamento por deslocamento (offset) .....	162
8.1.4 - Outras formas de deslocamento .....	163
8.2 - Tipos de dados em Assembly .....	163



8.3 - Cálculos matemáticos intermediários.....	168
8.3.1 - Adição .....	168
8.3.2 - Subtração .....	177
8.3.3 - Divisão.....	180
8.3.4 - Multiplicação.....	187
8.4 - Procedimento próximo x procedimento distante .....	196

## **Capítulo 9 - Saltos, decisões e repetições**

9.1 - Salto incondicional .....	199
9.2 - Salto condicional .....	201
9.3 - Desvios condicionais.....	205
9.3.1 - Desvio condicional simples.....	205
9.3.2 - Desvio condicional composto .....	207
9.4 - Operações lógicas.....	209
9.5 - Repetições .....	217
9.6 - Utilização de macros.....	225

## **Capítulo 10 - Detalhes complementares**

10.1 - Mais sobre segmentos e deslocamentos .....	229
10.2 - Programas executáveis.....	237
10.3 - Bibliotecas em Assembly .....	248
10.4 - Apresentação de valores negativos .....	254
10.5 - Biblioteca de funções externas .....	258
10.6 - Manipulação de cadeias.....	263

## **Capítulo 11 - Recursos específicos**

11.1 - Funcionalidades da biblioteca emu8086.inc.....	269
11.1.1 - Funções de macro .....	269
11.1.2 - Funções de procedimento .....	272
11.2 - Endereçamento e acesso à memória.....	275
11.2.1 - Acesso direto .....	276
11.2.2 - Acesso indireto com registradores .....	280
11.2.3 - Acesso indexado.....	285
11.2.4 - Acesso de base indexada.....	288
11.2.5 - Acesso de base indexada mais deslocamento .....	289
11.3 - Prefixos de anulação.....	291
11.4 - Operações com matrizes .....	291
11.5 - Programas exemplo .....	298

## **Capítulo 12 - Como em alto nível**

12.1 - Tomada de decisão.....	301
12.1.1 - Decisão simples .....	302
12.1.2 - Decisão composta .....	305
12.1.3 - Decisões com duas condições .....	307
12.1.4 - Decisões sequências .....	313
12.1.5 - Decisão seletiva .....	316
12.1.6 - Decisão encadeada .....	319

12.2 - Laços de repetição .....	322
12.2.1 - Laço condicional pré-teste.....	322
12.2.2 - Laço condicional pós-teste .....	325
12.2.3 - Laço incondicional .....	328
12.3 - Demonstrações Assembly .....	330
12.4 - Linguagem assembly versus código de máquina.....	338

## Parte IV - Apêndice

Apêndice A - Referência operacional .....	341
Apêndice B - Enhanced DEBUG .....	363
Referência Bibliográfica .....	383

# PREFÁCIO

O objetivo central deste trabalho é ser um veículo de apresentação dos princípios preliminares e básicos que norteiam o início do aprendizado da linguagem de programação Assembly 8086/8088 de microcomputadores IBM-PC em baixo nível. Não há pretensão de explorar o tema com profundidade. Este livro é direcionado tão somente a leitores iniciantes. É uma introdução a atividade de programação em baixo nível. Se o leitor possui noções de programação em linguagem Assembly ou programação em baixo nível com *opcodes*, este material não é para você, pois não lhe acrescentará novos conhecimentos.

A escolha do modelo Assembly 8086/8088 para este estudo e das funções de programação básica de um microprocessador padrão Intel ou AMD se justifica por dois motivos: primeiro pelo fato de ser a mais usada e difundida para o ensino e aprendizado de programação de computadores em baixo nível em várias escolas espalhadas pelo mundo; segundo pelo fato das ferramentas *debug* e *emu8086* fazerem uso do padrão Intel 8086/8088.

Nos vários cursos de linguagem de programação Assembly nas mais variadas instituições de ensino superior espalhadas pelo mundo, normalmente escolhe-se o modelo 8086/8088 de microprocessador para o início do estudo, pois a família de computadores IBM-PC se desenvolveu e tornou-se popular sobre esta arquitetura. Não adianta tentar apresentar os últimos recursos do último modelo de um microprocessador Intel ou AMD para um iniciante que não conhece as bases mínimas desse ambiente de desenvolvimento.

O livro está dividido em quatro partes lógicas, nas quais se procurou agrupar o tema em áreas de concentração operacional (o que não é algo simples em se tratando de programação em baixo nível, pois os temas facilmente se misturam), no sentido de formarem as partes mínimas do todo, para o desenvolvimento do entendimento mental voltado à programação de computadores padrão IBM-PC em baixo nível.

A parte I apresenta as noções preliminares do estudo. São estudados a diferença entre assembly e assembler; o contexto do sistema computacional, a organização básica dos dados, os registradores, as interrupções, os segmentos, os deslocamentos e endereçamento de memória. Essa parte dá a noção básica relacionada à arquitetura interna do modelo de microprocessador 8086/8088.

A parte II destaca o uso do programa DEBUGX. Aqui são apresentados conceitos de aritmética em modo hexadecimal, representação de valores negativos, cálculos com códigos de máquina, movimentação de dados, apresentação de caracteres, valores binários e hexadecimais, utilização básica dos registradores de estado, apresentação inicial de salto condicional, entrada de dados por teclado, utilização de procedimentos, de pilha e o controle da consistência da entrada de dados. É pertinente salientar que boa parte dos programas exemplo utilizados foram baseados ou são adaptações dos programas encontrados na obra dos autores NORTON & SOCHA de 1987, que melhor descreveram as bases iniciais para o aprendizado da programação em baixo nível. Em alguns trechos dos capítulos desta parte são usados elementos colorizados para permitir melhor visualização dos elementos técnicos abordados.

A parte III trata das informações de aperfeiçoamento em relação ao estudo da parte anterior. É feita uma descrição dos recursos básicos da ferramenta emu8086, e são apresentadas informações teóricas de manipulação de registradores e dados (uma abordagem mais prática encontrada no final da obra). Outros pontos apresentados são a definição de tipos de dados e de cálculos matemáticos mais elaborados, o uso de saltos com decisões e laços de repetição, o uso de instruções lógicas, utilização de macros em complementação ao procedimento, apresentação mais detalhada de segmentos e deslocamentos, criação de biblioteca externa, apresentação de valores negativos, utilização da biblioteca de recursos que acompanha a ferramenta emu8086, abordagem de endereçamento e acesso à memória.

A parte IV apresenta anexo com informações complementares.

Por se tratar de um trabalho de apresentação dos recursos básicos da linguagem de programação Assembly 8086/8088, não é feito um estudo aprofundado das instruções do microprocessador Intel 8086/8088. Apresenta-se em torno de 60% do conjunto de instruções, o que já dá muito material para um estudo bem demorado.

Como sempre, a todos um grande abraço e um bom aprendizado. Que este trabalho seja de grande proveito e o investimento financeiro, ora nele feito, seja válido e proveitoso no final do seu estudo desta obra.

*Augusto Manzano*

"O aprendizado é um tesouro que segue seu dono por todos os lugares." - Provérbio chinês

# SOBRE O AUTOR

**José Augusto Navarro Garcia Manzano** é professor com mestrado possuindo formação em Análise e Desenvolvimento de Sistemas, Ciências Econômicas e Licenciatura em Matemática atuando na área de Tecnologia da Informação, Computação e Informática (desenvolvimento de software, ensino e treinamento) desde 1986. Participou do desenvolvimento de aplicações computacionais para áreas de telecomunicações e comércio. Na carreira docente iniciou sua atividade docente em cursos livres, trabalhando posteriormente em empresas de treinamento e atuando desde então nos ensinos técnico e superior.

Atualmente é professor com dedicação exclusiva no IFSP (Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, antiga Escola Técnica Federal). Em sua carreira desenvolveu competências e habilidades para ministrar componentes curriculares de Lógica de Programação (Algoritmos), Estrutura de Dados, Microinformática, Informática, Linguagens de Programação Estruturada, Linguagens de Programação Orientada a Objetos, Engenharia de Software, Sistemas de Informação, Engenharia da Informação, Arquitetura de Computadores e Tecnologias Web. Possui conhecimento de uso de diversas linguagens de programação imperativas, descritivas e de marcação, tais como: BASIC (clássico e estruturado), COBOL, COMAL, Logo, Scratch, Assembly, Pascal, FORTRAN (77 e 90), C, C++, D, Java, F#, Modula-2, C#, Lua, HTML, XHTML, CSS, Javascript, VBA, Ada, Rust, Python, LISP, Haskell, Standard ML, OCaml, Miranda, Hope, Groovy, Julia e Elixir. Possui mais de uma centena de obras publicadas, além de diversos artigos no Brasil e exterior.

.

