Professor
Edson Pedro FERLIN

Arquitetura de Computador

Linguagem de Montagem

(Assembly)

Prof. Edson Pedro Ferlin

Professor
Edson Pedro FERLIN

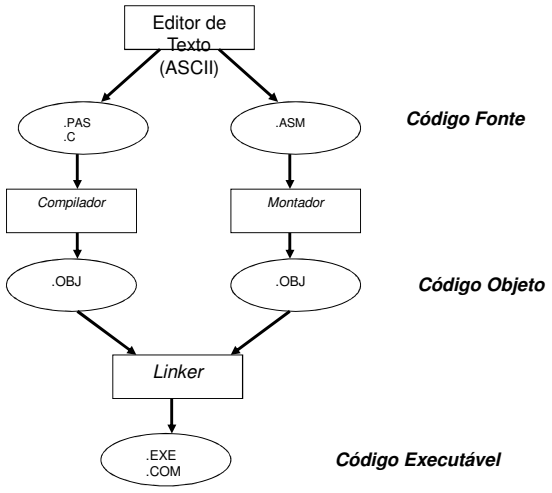
Arquitetura de Computador

- Objetivos
  - Apresentar a Linguagem de Montagem
- Conteúdos
  - Introdução à Linguagem de Montagem
  - Processo de Montagem

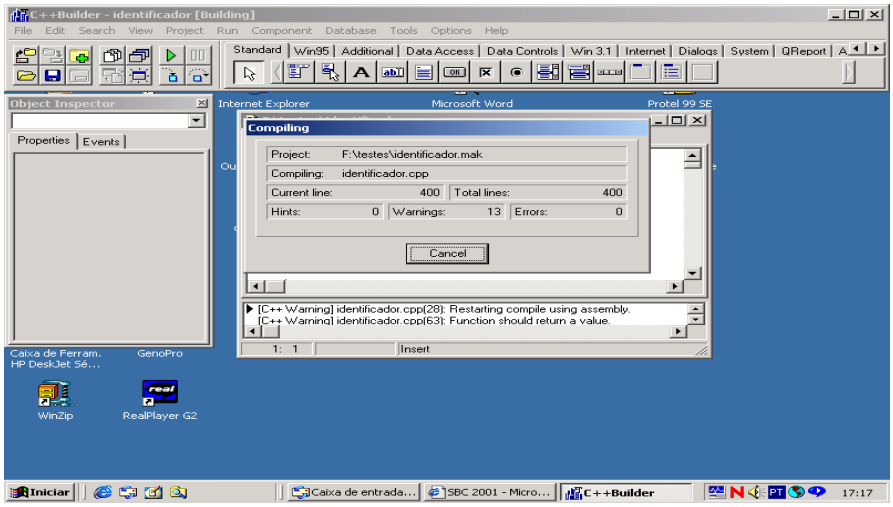
Linguagens de Programação

- Alto Nível – *Pascal, Lisp, Prolog, ...*
- Nível Médio – *C*
- Baixo Nível – *Assembler (Assembly)*

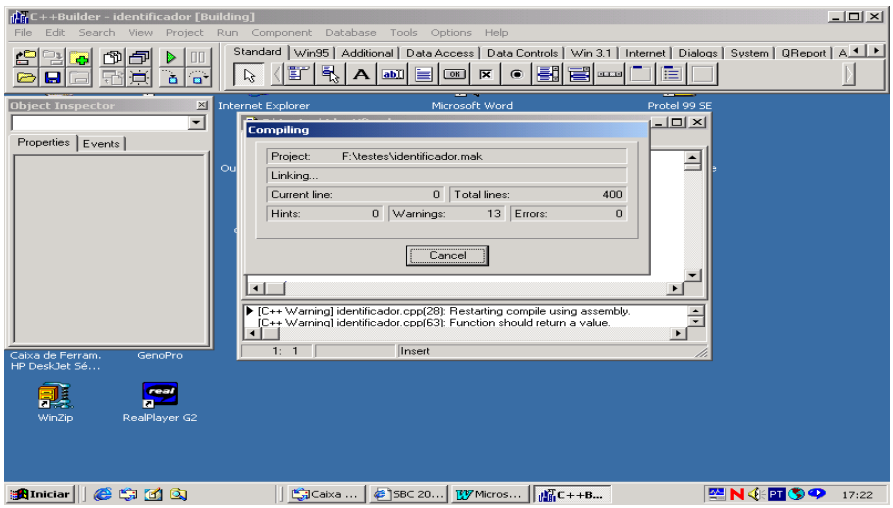
Processo de Compilação



Etapa de Compilação



Etapa de Linkedição



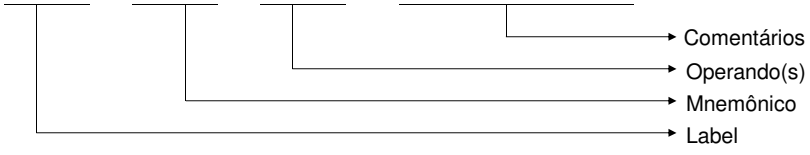
Linguagen de Montagem

É uma representação simbólica para uma linguagem numérica de máquina.

- Cada comando (mnemônico) produz exatamente uma instrução de máquina.
- É fácil de programar (linguagem de máquina).
- O programador tem acesso a todos os recursos e instruções disponíveis na máquina.

Formato da Linguagen de Montagem

```
        mov     a,#1d       ; a recebe 1 decimal
        mov     b,#2d       ; b recebe 2 decimal
Loop:   add     a,b         ; a := a + b
        inc     b           ; incrementa b
        sjmp    Loop        ; Loop infinito
```



Processo de Montagem

Montadores de Dois Passos:

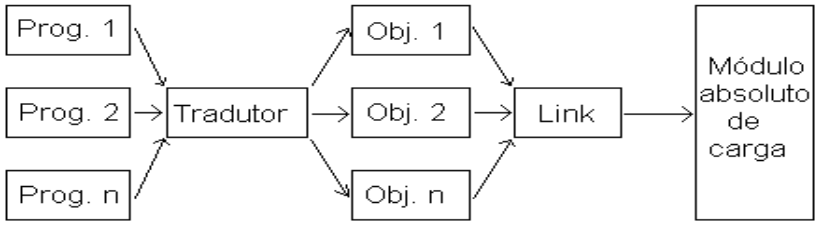
- Passo 1 – *Criação das tabelas (símbolos e de instruções)*
- Passo 2 – *Gerar o código objeto e as informações para o linker*

Erros – *Símbolo indefinido, duplicação de símbolos, operação não válida*

Módulo Objeto

Identificação
Tabela de *Entry Points* *(símbolos internos)*
Tabela de Referência Externas *(Símbolos externos)*
Instruções de Máquina *e Constantes*
Dicionário de Relocação
Fim do Módulo

Ligação e Carga



Exemplo de Compilação

PRINTF ("a");

MOV DX, #41h
MOV AH, #2h
INT 21h

1011-1010-0100-0001-0000-0000
1011-0100-0000-0010
1100-1101-0010-0001

BA 41 00
B4 02
CD 21

Exemplo de Código em Assembly

(x=x+3)+4

```
; TINY
;
LDA     x
LOD     x
LDC     3
ADI
STN
LDC     4
ADI
```

```
; Intel 80x86
; Borland C 3.0
MOV     AX,word ptr [bp-2]
ADD     AX,3
MOV     word ptr[bp-2],AX
ADD     AX,4
```

```
; SparcStations
; Sun C 2.0
LD      [%fp+-0x4],%o1
ADD     %o1,0x3,%o1
ST      %o1,[%fp+-0x4]
LD      [%fp+-0x4],%o2
ADD     %o2,0x4,%o3
```

```
; 80x51/80x31
;
MOV     A,x
ADD     A,#3h
MOV     x,A
ADD     A,#4h
```


Afinação de Programas

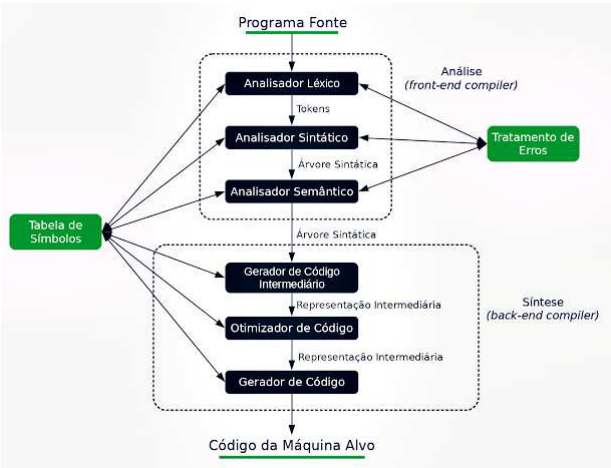
```
timeStart = timeGetTime();                  //GET TICK EDGE
for(;;)
{
        timeStop = timeGetTime();
        if ( (timeStop-timeStart) > 1 )     // ROLLOVER PAST 1
        {
        __asm{
                xor     eax, eax
                xor     ebx, ebx
                xor     ecx, ecx
                xor     edx, edx
                _emit   0x0f                // CPUID
                _emit   0xa2
                _emit   0x0f                // RDTSC
                _emit   0x31
                mov     [StartTicks], eax   // TICK COUNTER STARTS HERE
              }
        break;
        }
}
```

Compiladores

Elemento-chave para a obtenção do desempenho

Fonte **Executável**

Módulos do Compilador



Otimização - Eliminação de Dependências

- Renomeação

```
A = B + C          A1 = B + C
D = A + E    →     D = A1 + E
A = A + D          A = A1 + D
```

- Substituição a frente

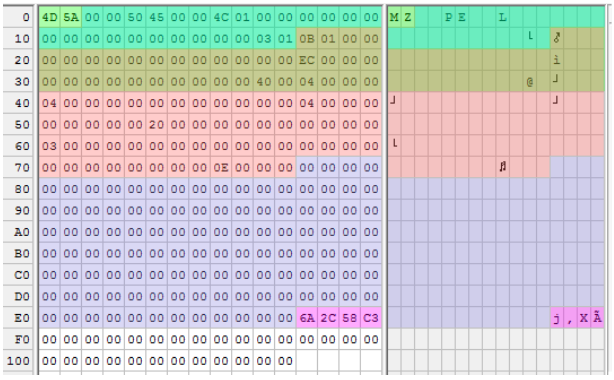
```
A = B + C          D = B + C + E
D = A + E    →     A = B + C + B + C + E
A = A + D
```

- Expansão Escalar

```
DO I = 1, N                DO I = 1, N
S:    X = C(I)      →      S':   X(I) = C(I)
T:    D(I) = X + 1         T':   D(I) = X(I) + 1
END_DO                     END_DO
```

- Distribuição do Laço

Estrutura – Arquivo EXE




```
0x6A   // PUSH
0x2C   // value to push
0x58   // POP EAX
0xC3   // RETN
```

268 bytes is the absolute minimum size for a working executable under Windows 7 64-bit edition

Escolha do Compilador

Sugestões para escolher um Compilador

Fornecidas pela Intel

(Capítulo 6 – *Suggestions for Choosing a Compiler*)

- *Important Features for a Compiler*
- *Compiler Switches Recommendation*

Contato

eferlin@live.com

(BLOG) professorferlin.blogspot.com

(SITE) professorferlin.webnode.com.br

(YOUTUBE) ProfEdsonPedroFerlin