Software Básico

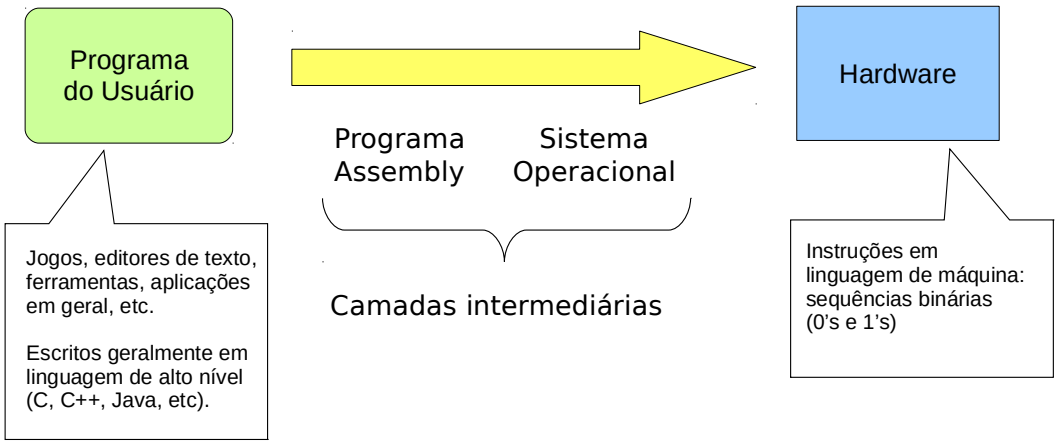
Introdução

INSTITUTO DE INFORMÁTICA - UFG

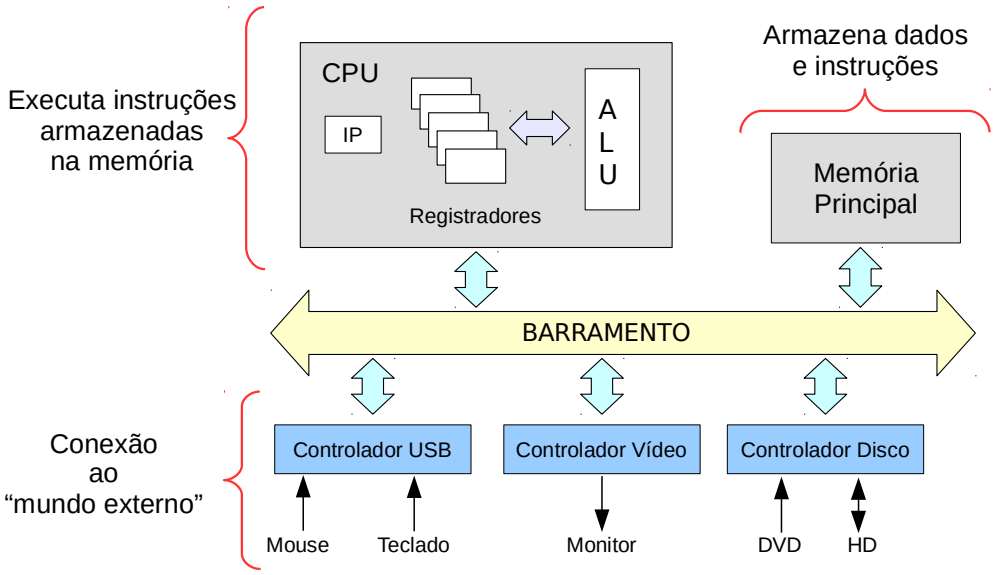
Reconhecimento

- Material produzido por:
  - Noemi Rodriguez – PUC-Rio
  - Ana Lúcia de Moura – PUC-Rio
- Adaptação
  - Bruno Silvestre – UFG

Hierarquia de Abstrações



Arquitetura Típica



Objetivo

- Entender como funciona um computador típico, como visto no nível de Linguagem de Montagem
- Perspectiva de software (foco no programador)
  - Suporte para abstrações de uma linguagem de programação (por exemplo, linguagem C)

Objetivo

- Programadores precisam de um entendimento sólido da hierarquia de abstrações de um Sistema de Computação
  - Otimizar o uso de recursos/desempenho de programas
  - Entender e saber evitar bugs (representação de dados, manipulação de memória, estouro da pilha, etc.)

Porque as abstrações vazam!!!

Geração de um Executável

- O programa fonte (arquivo texto) deve ser "traduzido" para uma sequência de instruções de linguagem de máquina, que é então armazenada em um arquivo executável (binário)
  - Essa tradução é realizada em 4 passos

hello.c

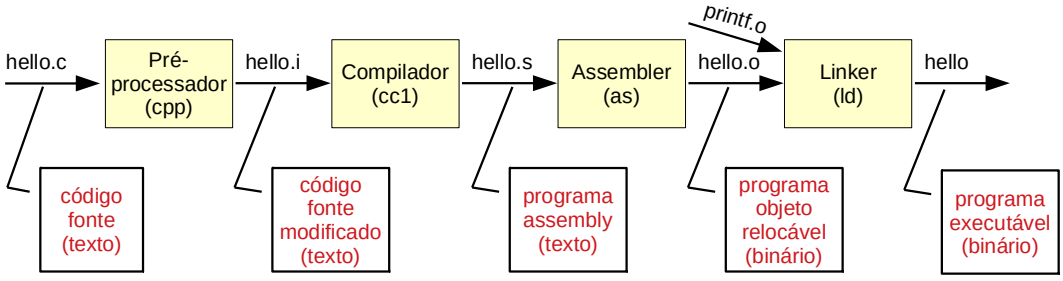
```
#include <stdio.h>

int main() {
  printf("hello, world\n");
  return 0;
}
```

```
$ gcc  hello.c   -o hello
```

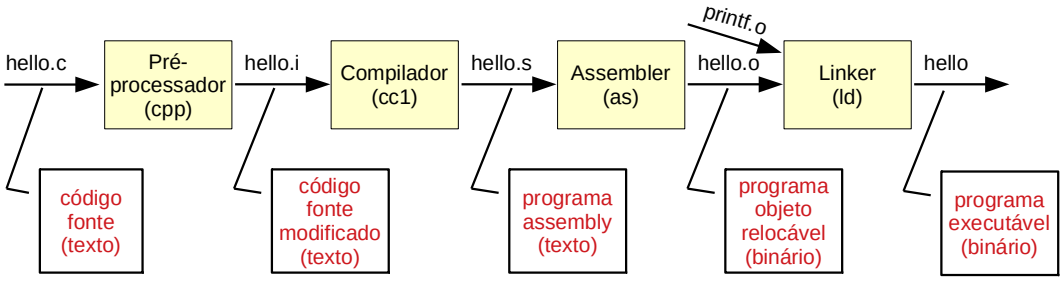
Geração de um Executável

- O programa fonte (arquivo texto) deve ser "traduzido" para uma sequência de instruções de linguagem de máquina, que é então armazenada em um arquivo executável (binário)
  - Essa tradução é realizada em 4 passos



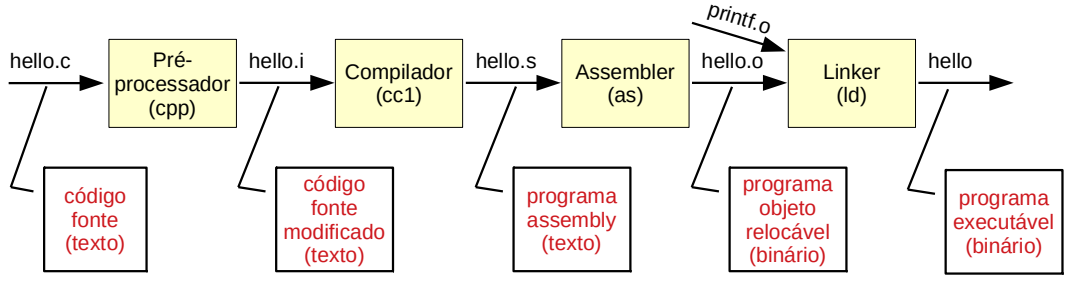
Passo 1: Pré-processamento

- Modifica o programa fonte C de acordo com as diretivas começadas por “#”
  - “#include <stdio.h>” faz com que o pré-processador leia o arquivo “stdio.h” e o insira no código fonte



Passo 2: Compilação

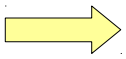
- Traduz o programa fonte modificado para um programa em linguagem de montagem (assembly)
  - É um formato de saída comum para os compiladores de linguagens de programação de alto nível



Linguagem de Montagem

- É um mapeamento bem direto para a linguagem de máquina
  - A linguagem de montagem (e de máquina) é específica para uma plataforma (CPU)
  - Cada linha de código fonte corresponde a uma instrução do processador
- Tem várias facilidades para um programador
  - Tipos básicos de dados (inteiros, endereços)
  - Uso de *mneumônicos* (nomes) para representar instruções e registradores

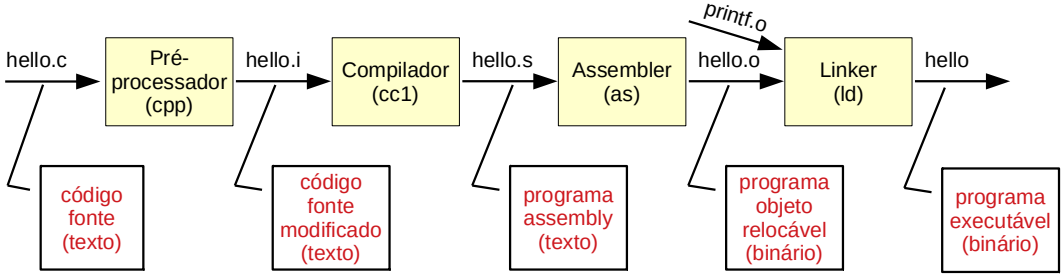
```
movl  $1024, %eax
```



```
b8 00 04 00 00
```

Passo 3: Montagem

- Traduz o programa fonte *assembly* para instruções da linguagem de máquina (objeto)
  - Armazenado como um arquivo binário com extensão ".o"



Passo 4: Ligação (Amarração)

- Gera um executável a partir do(s) módulo(s) objeto(s)
  - Uma aplicação pode ser composta por vários arquivos fonte, cada um gerando um módulo objeto diferente
  - Alguns módulos objeto podem estar armazenados em bibliotecas
  - O ligador faz a união dos módulos necessários para gerar o executável