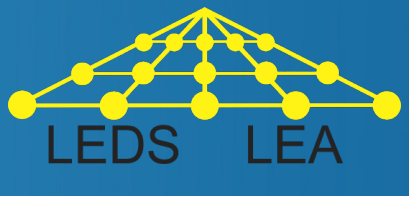
Arquitetura e Organização de Computadores

Conjunto de Instruções da Arquitetura – CompSim



Agenda

- Tipos de Instruções
- Manipulando a Pilha do Programa

Tipos de Instruções

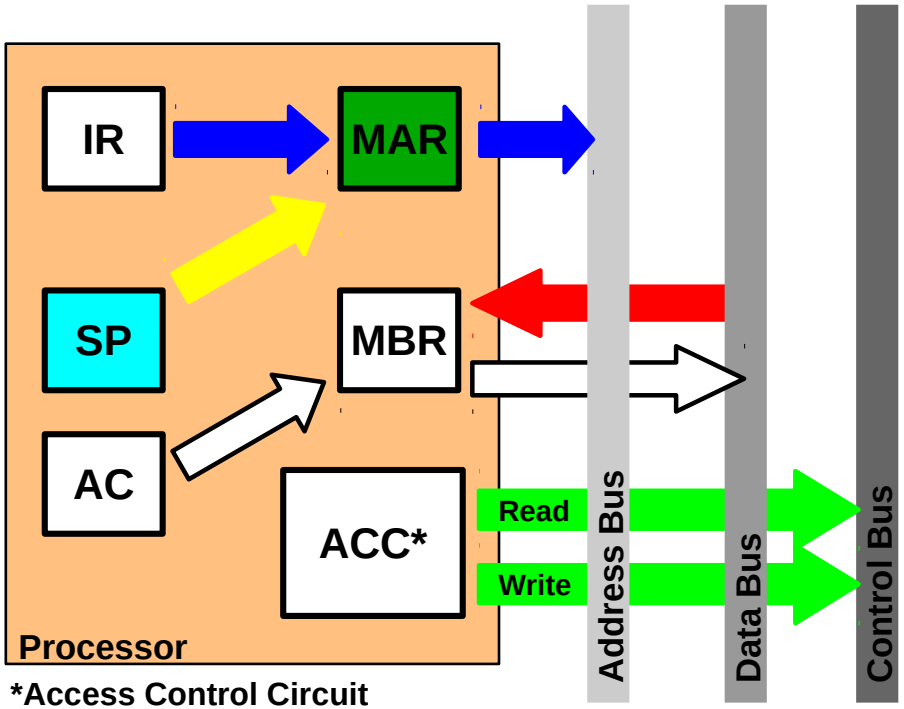
- Pseudo-Instruções do Montador (*Assembler*)
  - Segmento
    - .code, .data, .bss, .stack
  - Rótulo ou Nome
    - :
  - Delimitador de comentário
    - ;
  - Definição/Declaração de variáveis
    - DD, DB, RESD, RESB
- Conjunto de Instruções da Arquitetura (ISA)
  - Aritméticas
    - ADD, SUB
  - Lógicas
    - NAND, SHIFT
  - Transferência de dados
    - MOV, LDA, STA, LDI, STI, SOP
  - Transferência de controle
    - JMP, JN, JZ, CALL, RET, INT
  - Entrada/Saída
    - INT

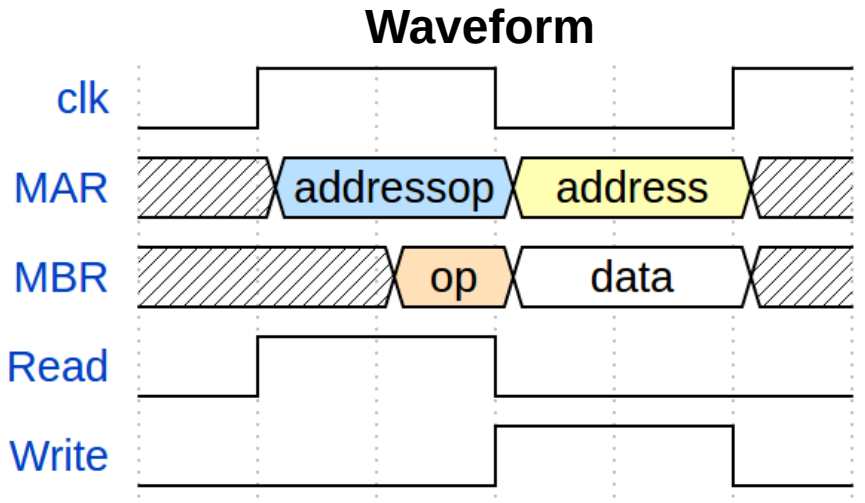
Manipulando a Pilha do Programa

- Vimos que todo programa possui um segmento de memória denominado “Pilha” (do inglês “*Stack*”), o qual implementa a estrutura de dados Pilha (LIFO ou *Last In, First Out*).
- Para tanto, utiliza-se os registradores SS (*Stack Segment*), que aponta para o final da memória reservada para a Pilha e SP (*Stack Pointer*), que aponta para o “topo” da Pilha.
- A instrução a seguir é utilizada para manipular a Pilha do programa:
  - SOP *(Stack OPeration)* – Adiciona ou remove uma palavra na pilha do programa.
- Sintaxe:
  - Push/Pop: [<rotulo>] SOP <endereco-memoria>
    - Onde o valor lido da memória deve informar a operação a ser realizada
      - “0” = Push
      - “1” = Pop

Manipulando a Pilha do Programa

- SOP (*Stack Operation*) - PUSH





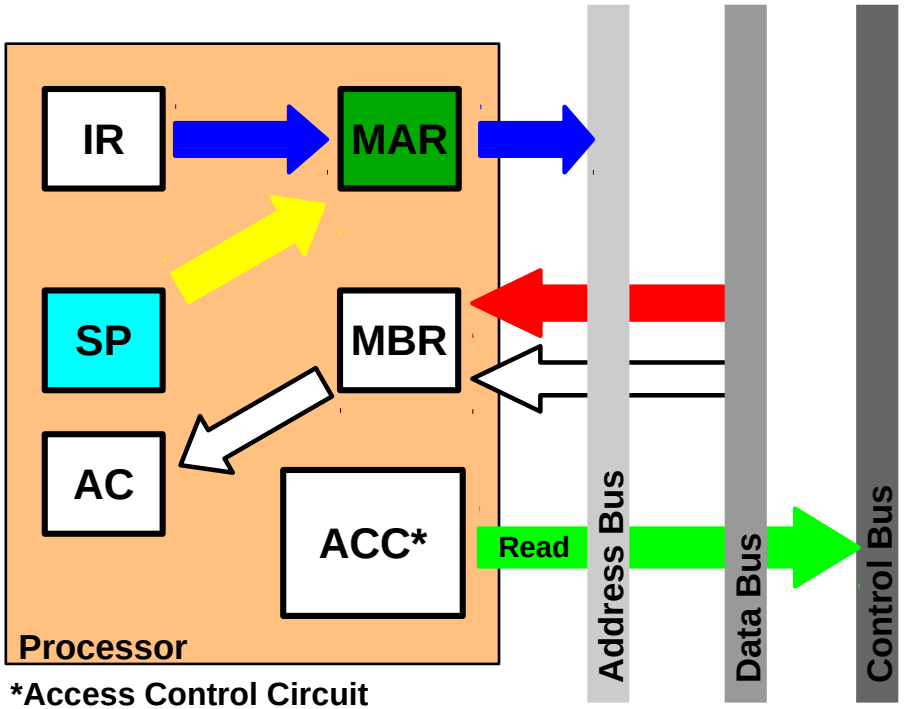
Microcode (RTN*)

```
MAR ← IR[11–0]
MBR ← M[MAR]
If MBR = 0 then
  SP ← SP – 1
  MAR ← SP
  MBR ← AC
  M[MAR] ← MBR
```

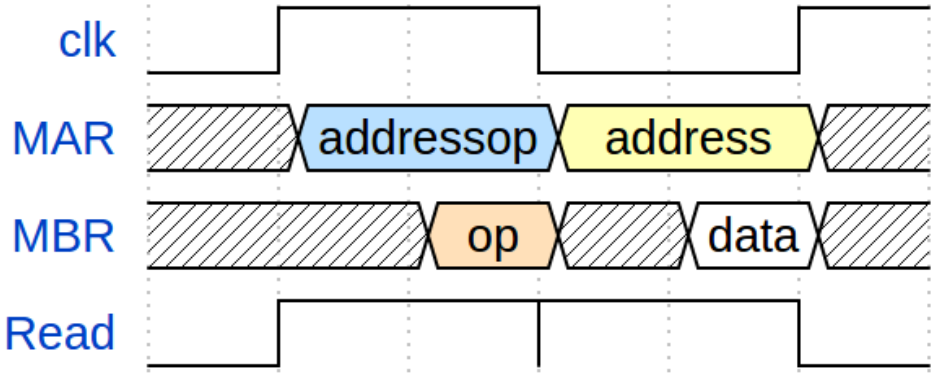
*Register Transfer Notation

Manipulando a Pilha do Programa

- SOP (*Stack Operation*) - POP



Waveform



Microcode (RTN*)

```
MAR ← IR[11–0]
MBR ← M[MAR]
If MBR = 1 then
   MAR ← SP
   MBR ← M[MAR]
   AC ← MBR
   SP ← SP + 1
```

*Register Transfer Notation

Manipulando a Pilha do Programa

- Exemplos práticos: Utilizando a Pilha do programa.
- Procedimento:
  - Baixar e extrair o pacote:
    - 7.handling_stack.zip
  - Menu “File” → “Open”
    - Ou Teclas “Ctrl+o”
  - Arquivo:
    - “stack_operations.asm”

```
1       .code
2           ; carrega o valor de a no AC
3           LDA  a
4           ; guarda AC na pilha
5           SOP push
6           ; retira da pilha para AC
7           SOP pop
8           ; finaliza o programa
9           INT exit
10
11      .data
12          a: DD 10
13
14          push: DD 0
15          pop: DD 1
16
17          exit: DD 25
18
19      .stack 10       ; Stack size
20
21
22
```

Atividade Prática

- Criar um programa que utiliza Pilha do Programa para inverter a ordem dos caracteres de uma string.
  - Exemplo:
    - “Hello!” → “!olleH”