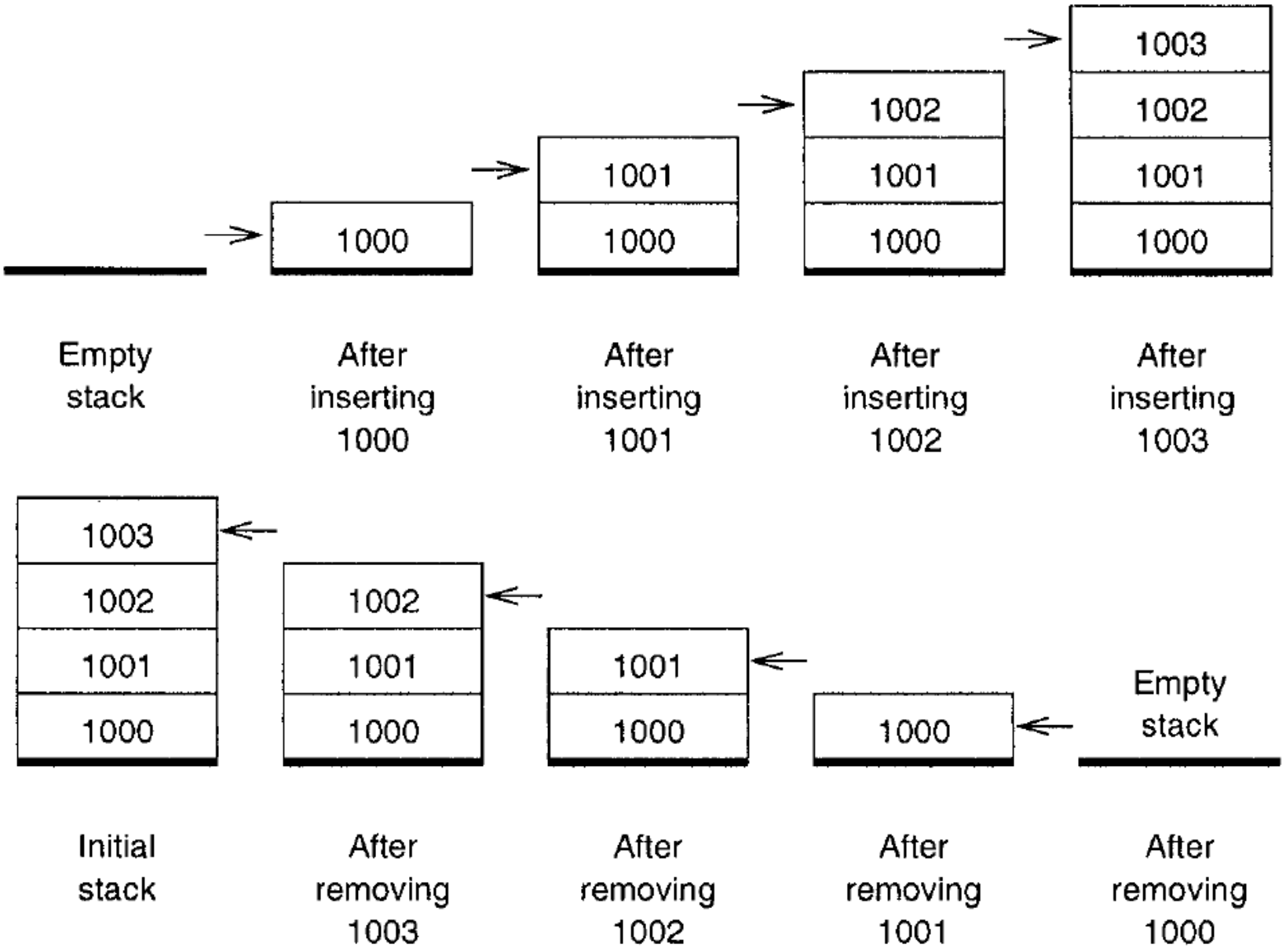
Pilhas

Pilha

- É uma estrutura de dados LIFO (Last in First out).
- Funciona como o empilhamento de livros.
- Existem duas operações associadas com a pilha: inserção (**push**) e deleção (**pop**).
- O único elemento que é acessível é o elemento no topo da pilha (**TOS** – Top of stack).

Inserção e deleção



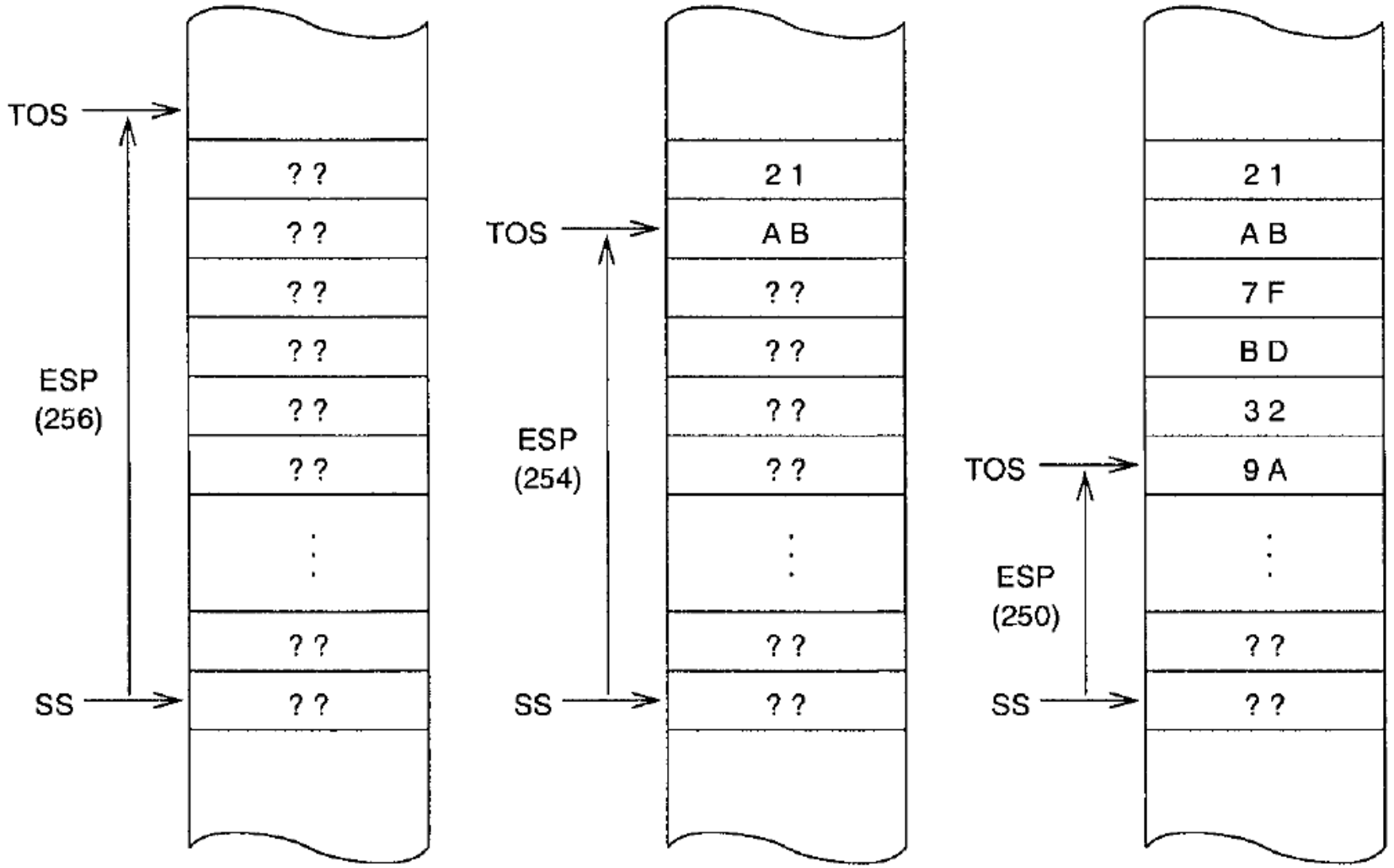
Implementando pilha

- O espaço de memória reservado no segmento de pilha (SS) é utilizado para implementar a pilha.
- Os registradores SS e ESP são utilizados para implementar a pilha.
- O topo da pilha (TOS) é indicado por SS:ESP com SS apontando para o início da pilha e ESP dando o offset do último item inserido.

Características da pilha

- Somente palavras (WORD) ou palavras duplas (DWord) são inseridas na pilha.
- A pilha cresce do maior endereço de memória para o menor.
- O topo da pilha (TOS) sempre aponta para o endereço mais baixo do último item colocado na pilha.

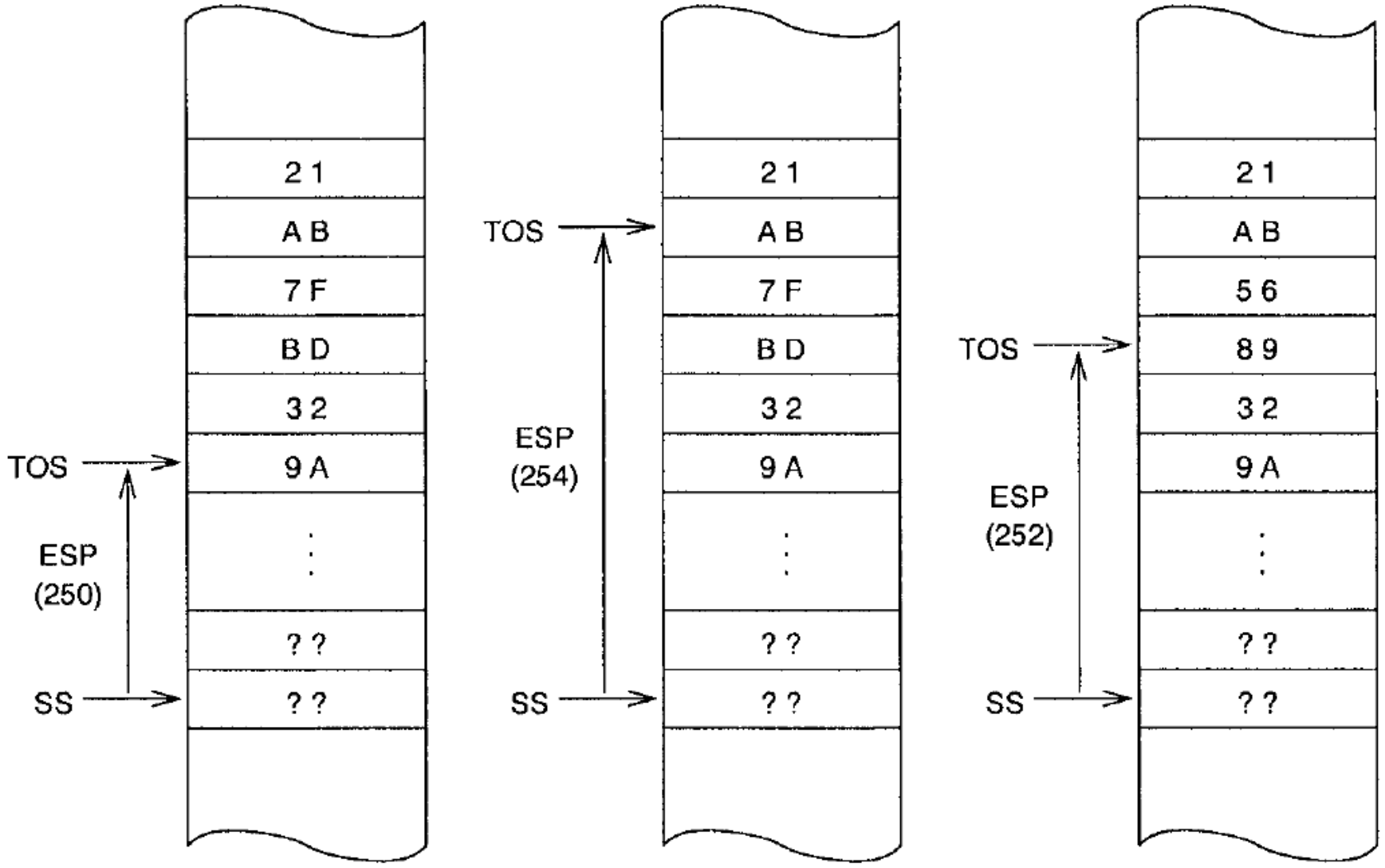
O endereço diminui



Características da pilha

- Quando uma palavra é incluida na pilha ESP é decrementado de 2 e a palavra é armazenada em SS:ESP.
- A pilha está cheia quando ESP = 0
- Se tentarmos inserir um item em uma pilha cheia dará erro de stack overflow.

Operações na pilha



Operações básicas

- A sintaxe:

```
push    source
pop     destination
```

  - O Exemplo abaixo:

```
push    21ABH
push    7FBD329AH
pop     EBX
```

Coloca 7FBD329AH em EBX

Operações

`push`	`source16`	ESP = ESP − 2 SS:ESP = `source16`	ESP is first decremented by 2 to modify TOS. Then the 16-bit data from `source16` is copied onto the stack at the new TOS. The stack expands by 2 bytes.
`push`	`source32`	ESP = ESP − 4 SS:ESP = `source32`	ESP is first decremented by 4 to modify TOS. Then the 32-bit data from `source32` is copied onto the stack at the new TOS. The stack expands by 4 bytes.
`pop`	`dest16`	`dest16` = SS:ESP ESP = ESP + 2	The data item located at TOS is copied to `dest16`. Then ESP is incremented by 2 to update TOS. The stack shrinks by 2 bytes.
`pop`	`dest32`	`dest32` = SS:ESP ESP = ESP + 4	The data item located at TOS is copied to `dest32`. Then ESP is incremented by 4 to update TOS. The stack shrinks by 4 bytes.

Uso da pilha

- Armazenamento temporário de dados
- Transferência de controle – quando uma procedure é chamada, o endereço de retorno é armazenado na pilha
- Passagem de parâmetros - meio para passar parâmetros para uma procedure chamada.

Armazenamento temporário de dados

- Troca de conteúdo de variáveis na memória:

```
mov     EAX,value1
mov     EBX,value2
mov     value1,EBX
mov     value2,EAX
```

- Os registradores EAX e EBX no entanto podem ter valores anteriores armazenados:

```
push    EAX
push    EBX
mov     EAX,value1
mov     EBX,value2
mov     value1,EBX
mov     value2,EAX
pop     EBX
pop     EAX
```

Exercícios

- Utilizando unicamente os comandos push e pop troque os conteúdos entre os registradores eax e ebx
- Utilizando os comandos push e pop e a instrução mov troque o conteúdo dos registradores eax e ebx
- Faça um programa em assembly que dado uma lista de números inteiros imprima o maior deles
- Faça um programa em assembly que dado uma lista de números inteiros os coloque na ordem crescente.

Exercícios

- Implemente uma calculadora pós-fixada, contendo as operações básicas (+, -, *, /)

- Faça um programa que para cada número inteiro de uma seqüência terminada por zero determine o equivalente em binário utilizando pilha.

Exercicios

- Faça um programa que leia uma seqüência de caracteres terminada por <Enter> e a exiba criptograficamente segundo as seguintes regras:

  Toda não consoante é exibida diretamente

  Toda seqüência de consoantes é exibida na ordem inversa obtida