

# 情報工学実験1

## 課題2：アセンブリ言語

### 第3回

# Hello World(前回)

ファイル名：hello.asm      \* 大文字・小文字はどちらでも良い

```
org 0x100
```

```
; ハローワールドの表示
```

```
mov dx, msg
```

```
mov ah, 0x09
```

```
int 0x21
```

```
; プログラムの終了
```

```
mov ax, 0x4c00
```

```
int 0x21
```

```
msg db 'Hello! World.', 0x0d, 0x0a, '$'
```

# Hello World(前回)

## アセンブル

```
nasm -fbin hello.asm -l hello.lst -o hello.com
```

ファイル名：hello.asm

ファイル名：hello.lst

**org** 0x100

;表示

**mov** dx, msg

**mov** ah, 0x09

**int** 0x21

;プログラムの終了

**mov** ax, 0x4c00

**int** 0x21

**msg db** 'Hello! World.', '\$'

ASCII コード表を確認しよう

00000000 B409 ...

00000002 BA[0C00] ...

00000005 CD21 ...

00000007 B8004C ...

0000000A CD21 ...

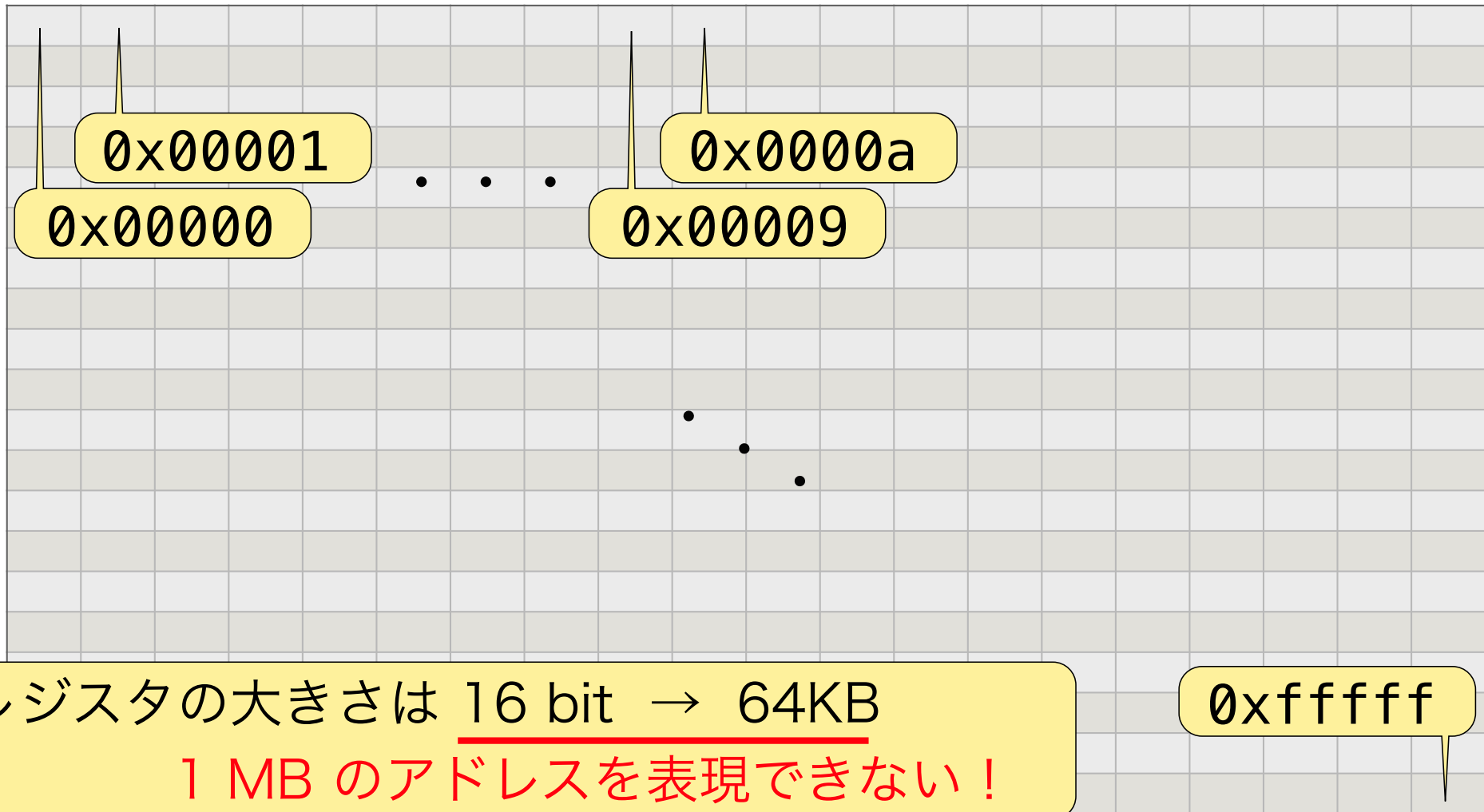
0000000C 48656C6C6F2120576F-

00000015 726C643F24

# 8086のメモリ空間

8086 では 1 MB のメモリを扱うことができる。

❖ メモリ空間と物理アドレスのイメージ



# 8086のセグメント方式

16ビットレジスタを2つ組み合わせて  
20ビットアドレスを表す



セグメント・レジスタ



セグメント・アドレス

オフセット・アドレス



物理アドレス

セグメント・アドレス  
**0x20000**

オフセット・アドレス  
**0x2345**

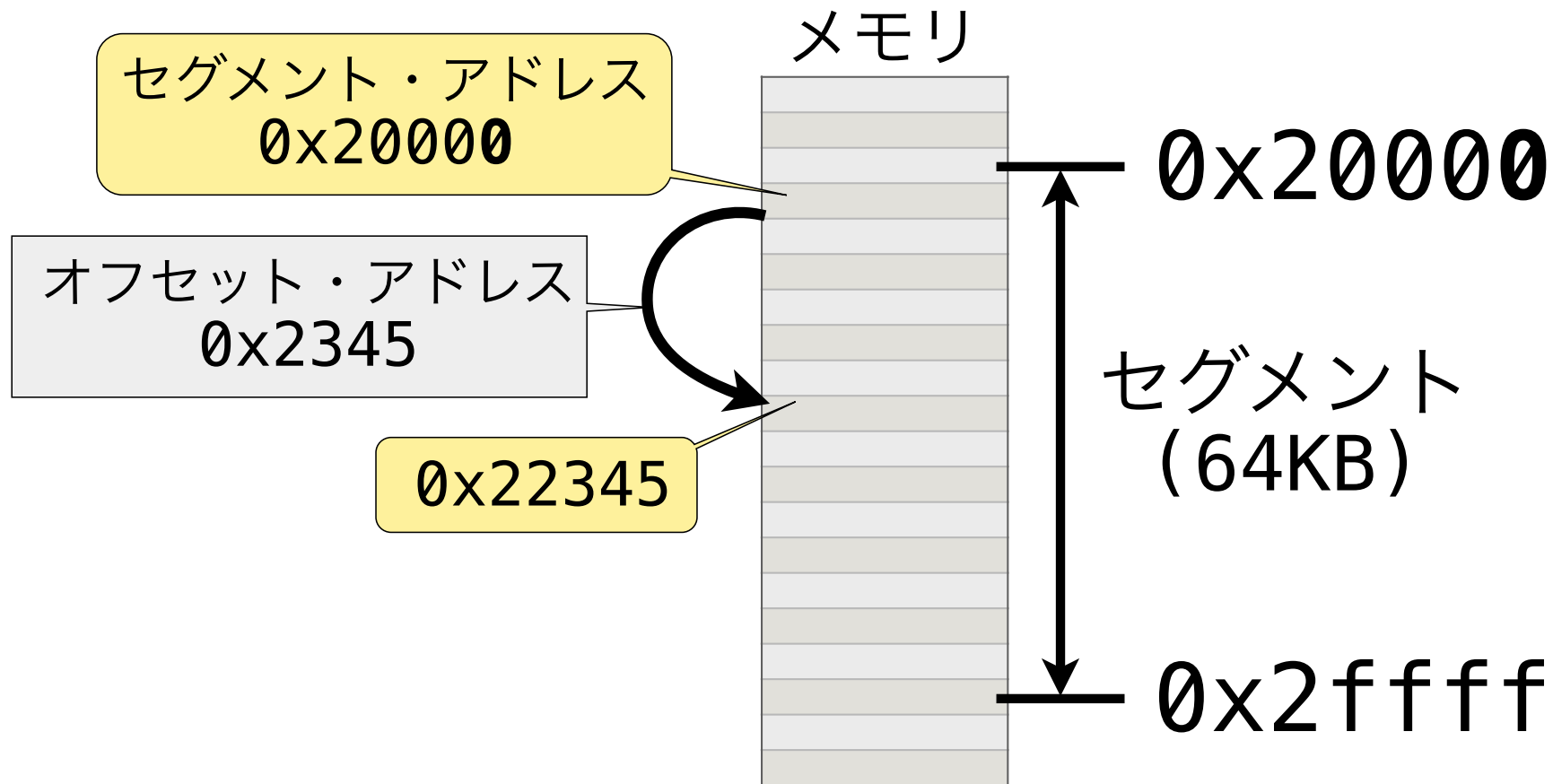
**0x22345**

メモリ



# セグメントとは

セグメント・レジスタによって指定される  
セグメント・アドレスを基準とした64KBの  
範囲のことをセグメントと呼ぶ



# セグメント・レジスタの種類

オフセット・アドレスが入るレジスタ

|    |                                |      |
|----|--------------------------------|------|
| CS | CPUが実行すべき命令が格納されるセグメントの始めを指す   | IP   |
| DS | データの格納されているセグメントの始めを指す         | 命令依存 |
| ES | データ転送命令で転送先などに用いられるセグメントの始めを指す | 命令依存 |
| SS | スタック動作の行われるセグメントの始めを指す         | SP   |

# Hello World(前回)

ファイル名：hello.asm      \* 大文字・小文字はどちらでも良い

```
org 0x100
```

```
; ハローワールドの表示
```

```
mov dx, msg
```

```
mov ah, 0x09
```

```
int 0x21
```

```
; プログラムの終了
```

```
mov ax, 0x4c00
```

```
int 0x21
```

```
msg db 'Hello! World.', 0x0d, 0x0a, '$'
```

# ストリング命令

## データをまとめて転送する命令

**MOVSB** ;操作するデータがバイト(1バイト)の場合

**MOVSW** ;操作するデータがワード(2バイト)の場合

\*使用するレジスタが決められている(→オペランドが不要)

DS . . . 転送**元**のセグメント・アドレス用レジスタ

SI . . . 転送**元**のオフセット・アドレス用レジスタ

ES . . . 転送**先**のセグメント・アドレス用レジスタ

DI . . . 転送**先**のセグメント・アドレス用レジスタ

CX . . . 転送**回数**の指定に用いるレジスタ

# ストリング命令

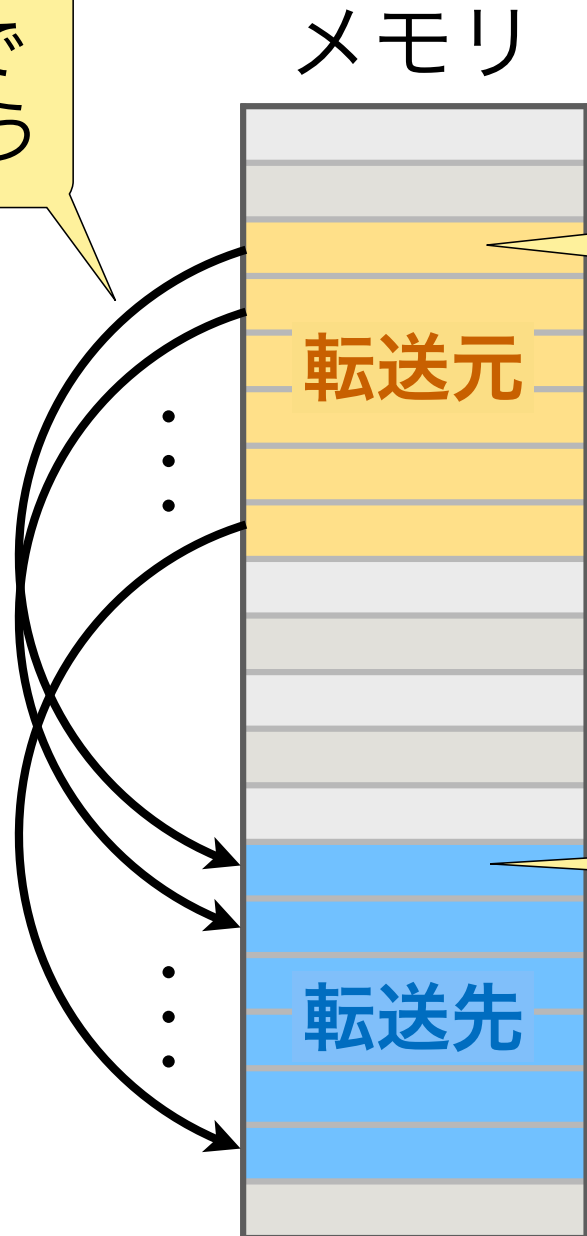
ブロック単位で  
CX回転送を行う

```
MOV AX, 0x2000
MOV DS, AX
MOV AX, 0x2345
MOV SI, AX

MOV AX, 0xb800
MOV ES, ax
MOV AX, 0x0123
MOV DI, ax

MOV CX, 0x0006

CLD
REP MOVSB
```



DS → 0x2000

SI → 0x2345

物理アドレス:

DS:SI → 0x22345

ES → 0xb800

DI → 0x0123

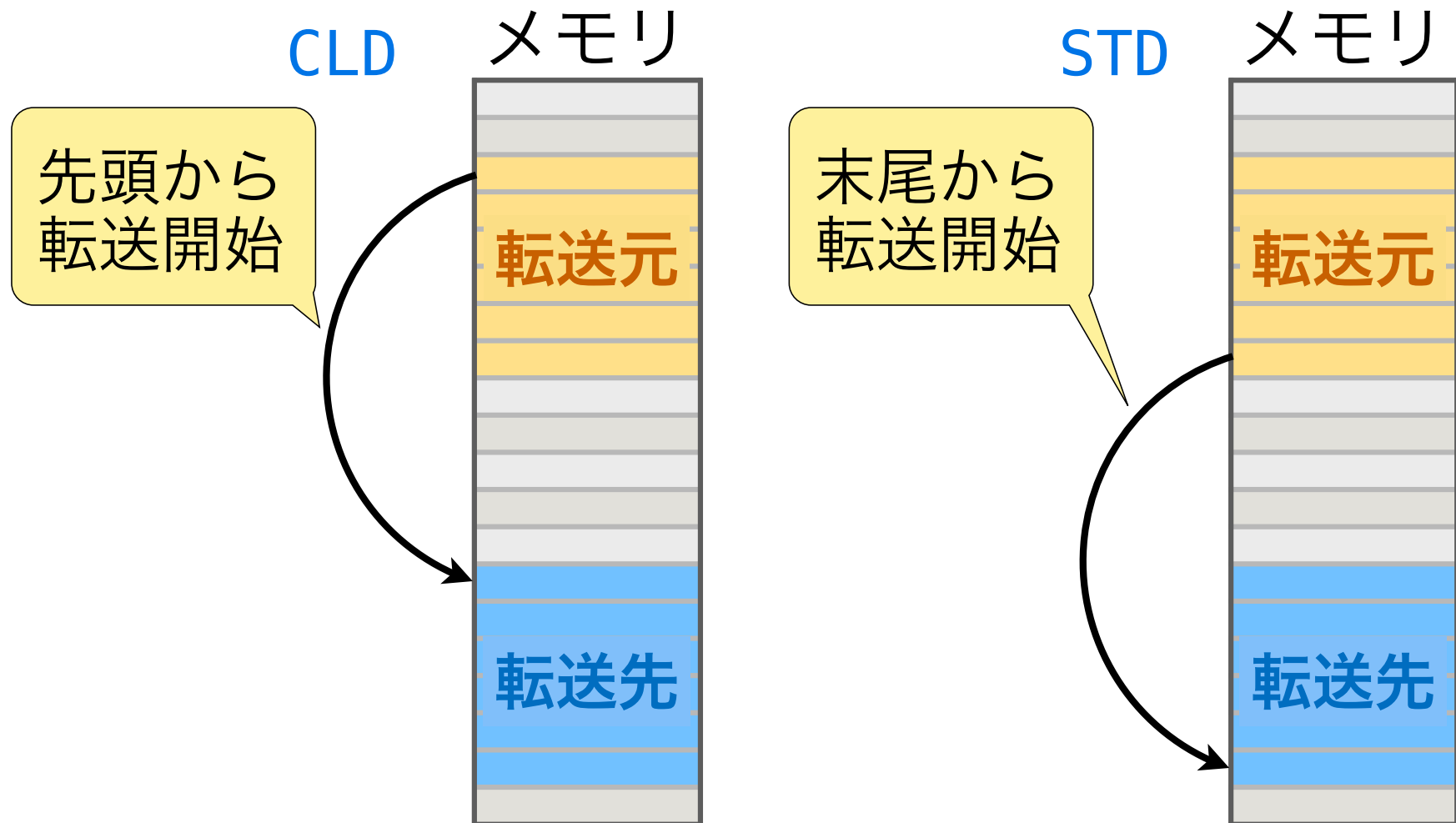
物理アドレス:

ES:DI → 0xb8123

# ストリング命令:転送方向

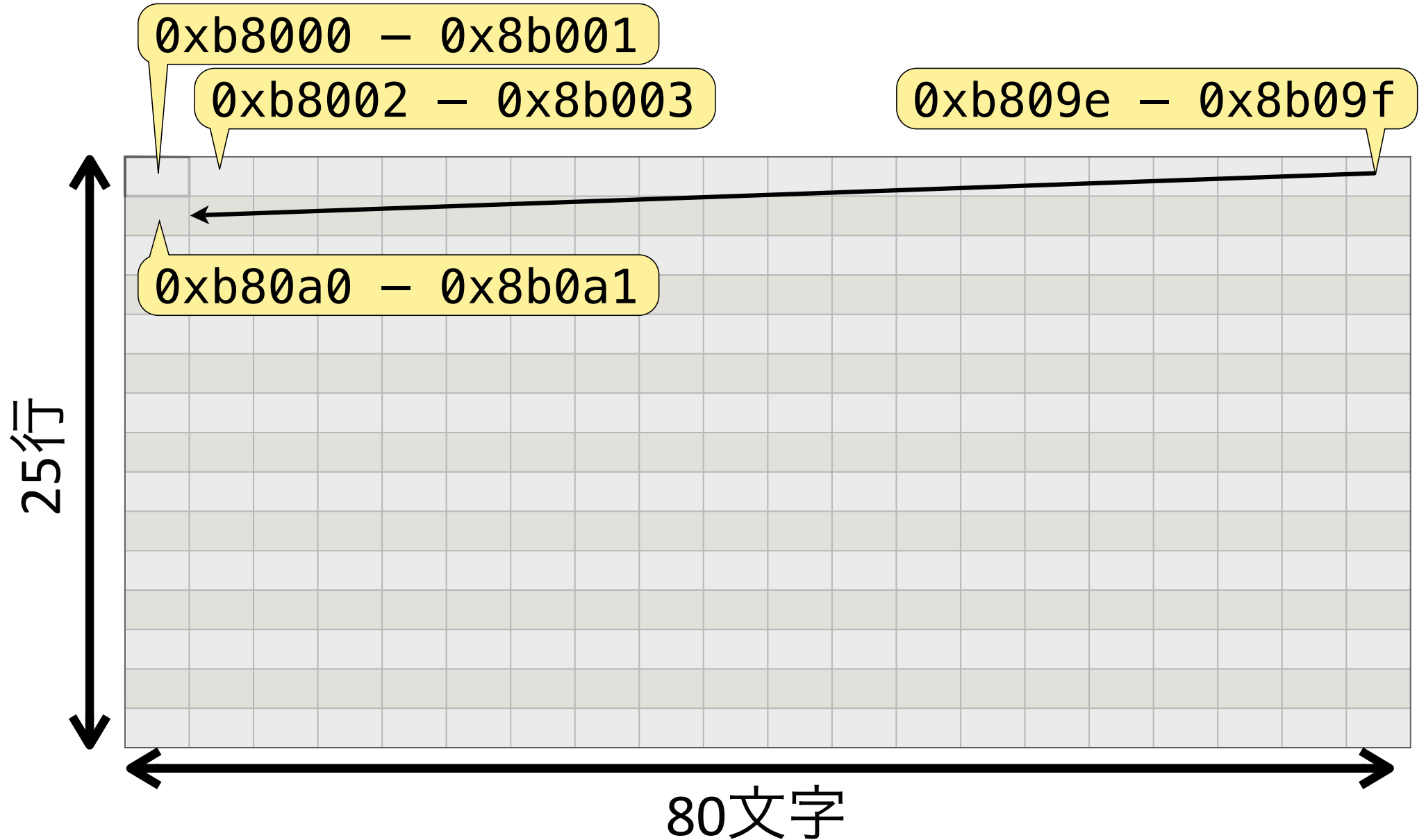
**CLD** . . . 順方向にデータを転送

**STD** . . . 逆方向にデータを転送



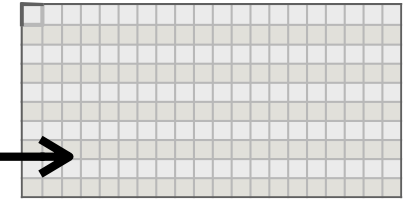
# テキストVRAM

表示データを保持する専用のメモリ領域

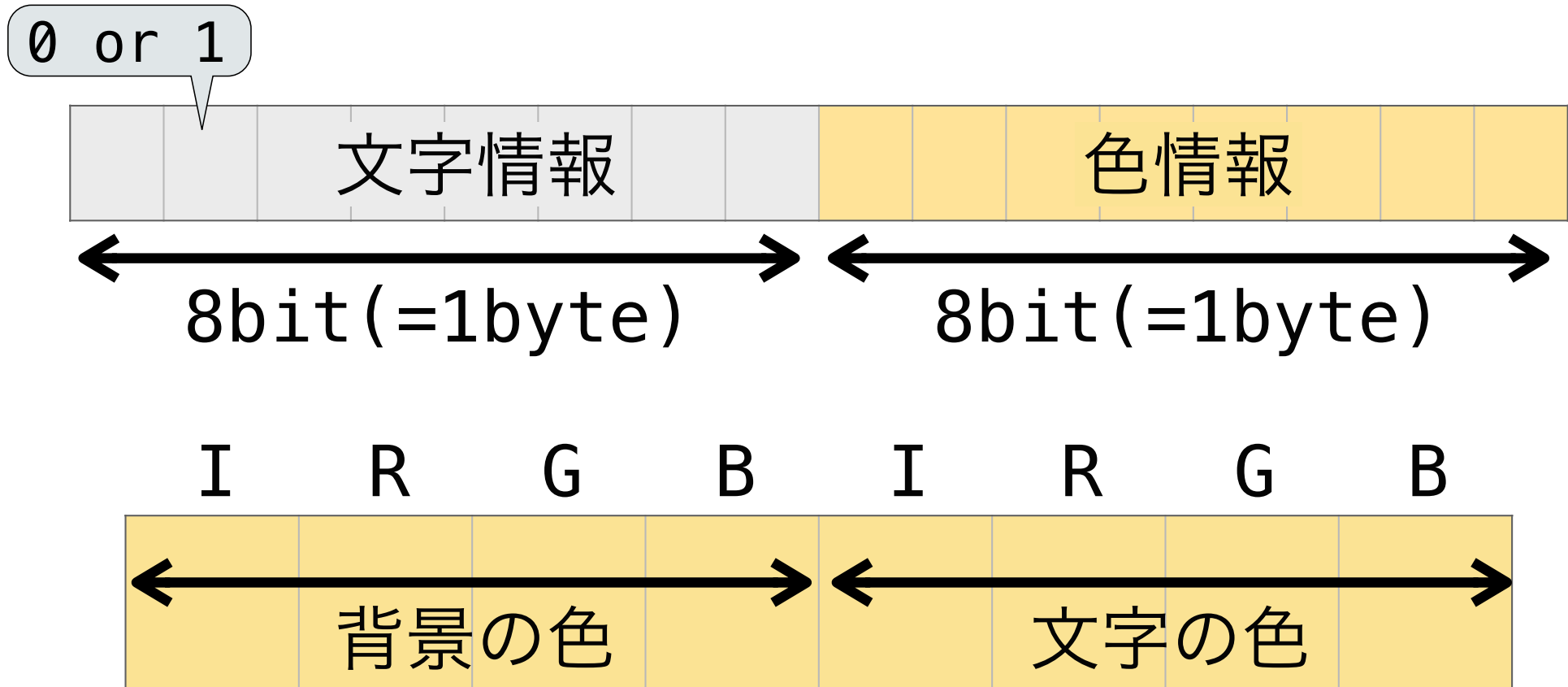


# テキストVRAM

1マスあたり2バイト



テキストVRAM



(例) 背景赤, 文字薄赤

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|

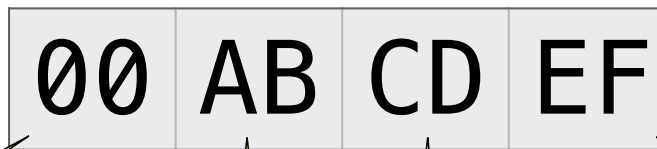
(これは16進数で0x4c)

# バイトオーダ

データが格納される順番

(例) 0x00ABCDEF を格納する場合

ビッグエンディアン:



1バイト



アドレス

0x1001番地

0x1002番地

0x1003番地

0x1004番地

リトルエンディアン:



テキストVRAM

色指定の例: 文字"A" (背景濃(暗)赤, 文字薄(明)赤)

ASCIIコードで"A"は0x41, 色は

0 1 0 0 1 1 0 0

文字情報

色情報

→ 0x4c41

# 本日の課題の提出について

提出先: LETUS上のレポートボックス

〆切: 来週 月曜日 10:20

詳細はサポートサイトを参照:

[www.hisenkei.net/~sima/lecture/itex1](http://www.hisenkei.net/~sima/lecture/itex1)

- ・ asm ファイルだけ提出すること
- ・ どこで何を実行しているのか, 適切なコメント文を入れること.