

情報工学実験1

課題 2 : アセンブリ言語

第 2 回

実際のプログラミングの流れ

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 ah, 0x4c
```

```
int 0x21
```

← 前回はHLT命令だったが
今回からはこの2行を書く

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

アセンブル

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

実行方法

```
hello.com
```

前回との違い

前回

シミュレータでアセンブリ言語の練習をした。シミュレータでは画面に文字列を出力したり、アセンブリ言語のプログラムを使ってキーボードから入力を受け付けることはできなかった。

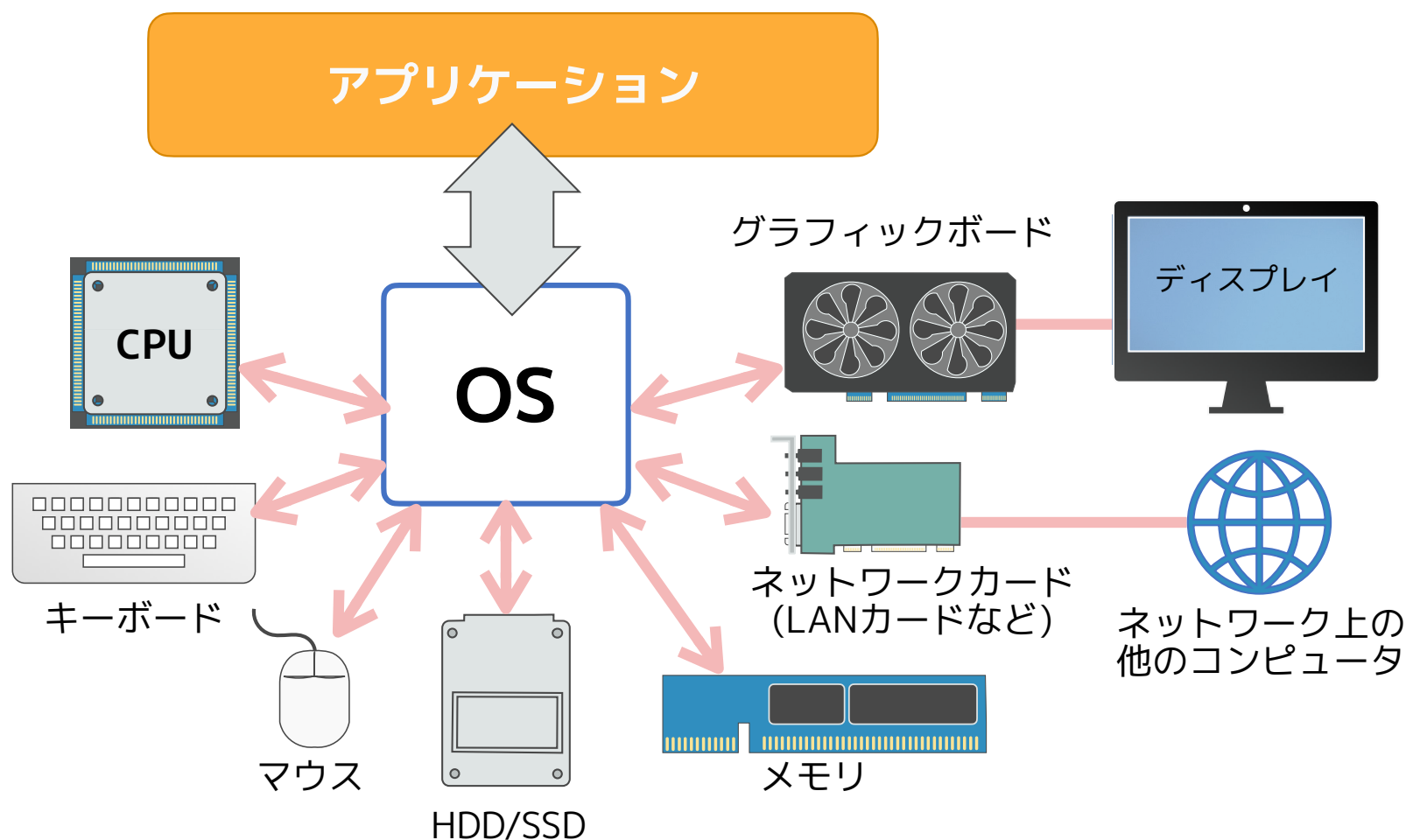
今回

MS-DOSと呼ばれるOS上で、アセンブリ言語のプログラムを動かす。OSの機能をアセンブリ言語のプログラムから呼び出すことで、画面の制御やキーボードからの入力を受け取ることができる。

※アセンブリ言語には、画面に文字を出力する、文字を入力する、などの命令が無い。画面への出力、キー入力にはOSにお願いするしか無い。

補足：オペレーティングシステム (OS)

コンピュータを動作させるために必要な基本ソフトウェア。ハードウェア(HDD, メモリなど)やアプリケーションの動作を管理する。



補足：OSの役割の例

共通の機能の提供

外付けのハードディスクやUSBメモリなどを接続したときにOSの使い方さえ知っていれば、ハードウェアの違いを気にせずに共通の操作でこれらのハードウェアを使える。

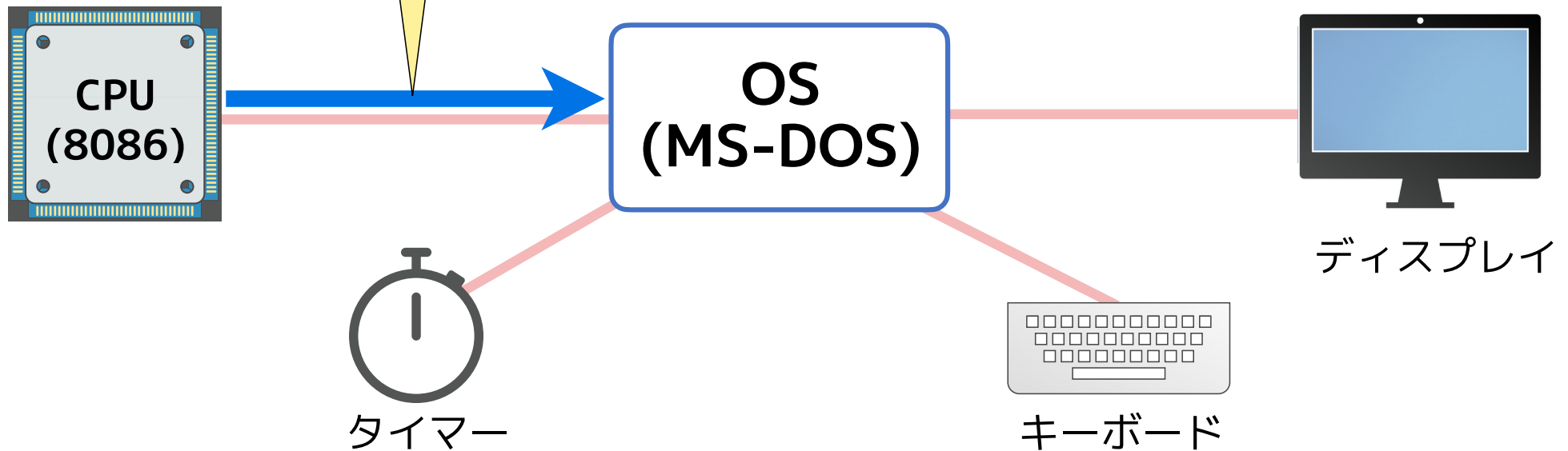
資源の管理

例えば、PCで友人とテレビ通話をしながら、ブラウザで調べ物をしてつつワープロソフトで文章を作成する状況を考える。このとき複数のアプリケーションが同時に滞りなく動作するために、どのアプリケーションが画面に描画し、音を鳴らすのか、音を鳴らすときはイヤホンを使うのか内臓スピーカーを使うのか、どのアプリケーションにどの程度のメモリ領域を割り当ててるのかを管理する。

システムコール (OSの機能の呼び出し)

INT 0x21 (←ソフトウェア割り込み)

⇔ サービス(仕事)を実行してとOSに連絡. OSは他の仕事の途中でもINT 0x21で依頼された作業を先に処理.



Q. OSに「お願いします」と伝えただけでは
OSは何をしていいのか, わからないのでは?

A. あらかじめセットした AH レジスタの内容(ファンクション番号)に応じて各種サービスが実行される.

ソフトウェア割り込み

ソフトウェア割込する(OSに仕事を頼む)ための命令

INT 0x21 ; 0x21番でシステムコール

MS-DOS が提供する各種サービス(仕事)の実行を依頼する

AHレジスタの値とサービス(OSに依頼する仕事)の内容

0x06 直接コンソール I/O (キー入力)

0x09 文字列出力 (出力する文字列をDXにセット)

0x0a 文字列入力 (入力文字列の保存先をDXにセット)

0x4c プロセス終了

※ 詳細は p. 104 -- 105 を参照

実際のプログラミングの流れ

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 ah, 0x4c
```

```
int 0x21
```

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

アセンブル

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

実行方法

```
hello.com
```

DB命令

1バイトの変数を宣言

変数の名前 **DB** 変数の値 ; 初期化された変数の宣言

MSG DB 'Hello! World.', 0x0d, 0x0a, '\$'

2バイトの変数を宣言

変数の名前 **DW** 変数の値 ; 初期化された変数の宣言

変数の名前 **TIMES 11 DB 0**

11回 DB 0 を繰り返す.
0 が入った 1 バイトの変数が11個用意される.

補足：文字と数値

コンピュータの内部では、文字を文字コードと呼ばれる数値で扱う

ASCIIコードと呼ばれる文字コードでは
'A'は0x41, 'a'は0x61という数値(16進数)になる

0x61は、10進数では $16 \times 6 + 1 = 97$

0x41は、10進数では $16 \times 4 + 1 = 65$

ASCIIコード表の一部

下位 \ 上位	3	4	5	6	7
0	0	@	P	`	p
1	1	A	Q	a	q
2	2	B	R	b	r
3	3	C	S	c	s
4	4	D	T	d	t
5	5	E	U	e	u
6	6	F	V	f	v
7	7	G	W	g	w
8	8	H	X	h	x
9	9	I	Y	i	y
a	:	J	Z	j	z
b	;	K	[	k	{
c	<	L	¥	l	
d	=	M	]	m	}
e	>	N	^	n	~
f	?	O	_	o	DEL

ASCIIコード表の一部

下位 \ 上位	3	4	5	6	7
0	0	@	P	,	p
1	1	A	Q	a	q
2	2	B	R	b	r
3	3	C			
4	4	D			
5	5	E			
6	6	F			
7	7	G			
8	8	H	X	h	x
9	9	I	Y	i	y
a	:	J	Z	j	z
b	;	K	[	k	{
c	<	L	¥	l	
d	=	M	]	m	}
e	>	N	^	n	~
f	?	O	_	o	DEL

‘a’ は 0x61

上位

下位

lst ファイルについて

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

コイントスのコード読解のヒント

CMP AX, BX ; AX と BX の比較

	ZF	CF
AX > BX	0	0
AX = BX	1	0
AX < BX	0	1

JZ label ; ZFフラグがセットされている場合,
labelにジャンプする.

キー入力が**無い**場合, ZF フラグがセットされる.

間接参照

MOV AX, DATA ; 10が置かれているアドレス

DATA DB 10

C言語のポインタと似ている. char *DATA;

MOV AX, [DATA] ; 10そのもの(DBの中身)

DATA DB 10

[]はC言語の間接参照演算子と同じ. *DATA

MOV byte [DATA], '\$'

DATA ... DB 10

DATA が byte なのか word なのかわからないので, 明記する.

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

- ・ 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.