



# 基本・組合せ回路の実現

---

例題3: SW状態('0'、'1')表示回路の記述

練習3: 7セグメントLEDデコーダ回路の

記述・実装



# 組合せ回路とは

---

- AND(論理積)やOR(論理和)などの  
論理素子を組み合わせる  
目的の論理を実現する回路
- 一時的にデータを保持するような回路は  
含まれない  
データが入力されると、  
すぐに何らかのデータを出力する

# 例題3

## Logical\_01

スイッチのON/OFF状態を

7セグメントLEDの0/1表示であらわす回路

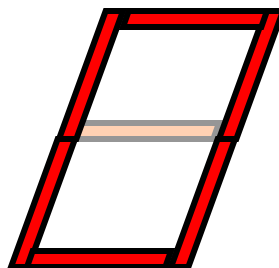
- std\_logic\_vector型

std\_logic型のデータ列

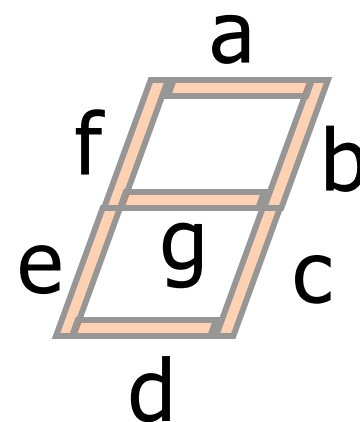
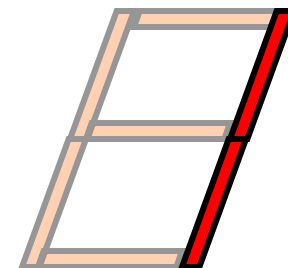
- 7セグメントLEDのLED配列

“abcdefg”、0で点灯、1で消灯

「0」を表示  
“0000001”



「1」を表示  
“1001111”



## 例題3

## architecture部: process文

```
SEG_CONTROL : process(SW)
```

```
begin
```

```
case SW is
```

※ボタン: 押→

```
when '0' => SEG7 <= "0000001";
```

```
when '1' => SEG7 <= "1001111";
```

```
when others => SEG7 <= "1111110";
```

```
end case;
```

```
end process;
```

「0」を表示

「1」を表示

「ー」を表示

# 練習3

## Pattern\_8\_slv

3つのボタンで表されるbitパターン(2進数)に  
相当する10進数を7セグメントLEDに

表示する回路

- 記述箇所は  
architecture部
- 例題3を参考に  
0～7までの  
パターンを作る

ボタン			bitパターン (2進数)			10進 数
放	放	放	0	0	0	0
放	放	押	0	0	1	1
放	押	放	0	1	0	2
放	押	押	0	1	1	3
押	放	放	1	0	0	4
押	放	押	1	0	1	5
押	押	放	1	1	0	6
押	押	押	1	1	1	7