

体験大学授業

**電子回路でストップ・ウォッチとスロットマシンを
作ってみよう！！
信頼性情報システム工学科**

コンピュータの歴史

- 機械式計算機 (歯車など)

失敗

バベジ「解析機関」1834年

理由: 10進数をベースに
していたため構造が複雑
(掛算・割算・微分・積分...)



コンピュータの歴史

第2次世界大戦後(1946) **アメリカ**
砲弾の飛び方の研究のために世界最初の
電子計算機(コンピュータ)を開発



「ENIAC」(エニアック)

(**E**lectronic **N**umerical **I**ntegrator **a**nd **C**omputer)

【長さ45m, 幅1m, 高さ3m, 重さ30t, 18,000 の真空管】

エッカート, モークリー, フォン・ノイマン(米)

他に アラン・チューリング(英)



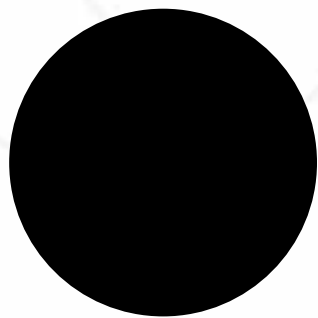
コンピュータの発達

- 基本的構造は1940年代にほぼ完成
「フォン・ノイマン型」コンピュータ
何が違うか？ **【その形・値段・使われ方】**
- 1948年 「電子回路の革命」
真空管 トランジスタ
- 1959年 IC (Integrated Circuit: 集積回路)
- 1971年 「マイクロプロセッサ」
CPU回路を1つのチップ (LSI化)

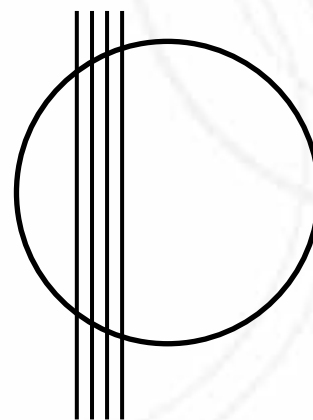


コンピュータの発達

- 最新のプロセッサ (Intel Core i7-980X)
トランジスタ数 11億7000万个
プロセスルール(配線の幅) 32nm



人間の髪の毛の
直径: $90 \mu\text{m}$



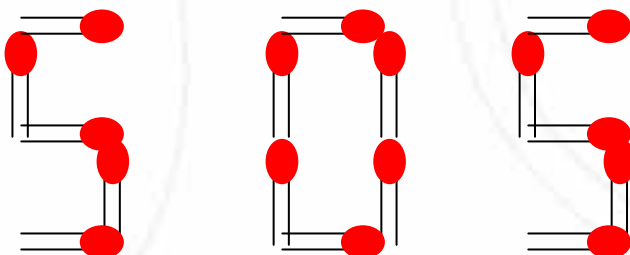
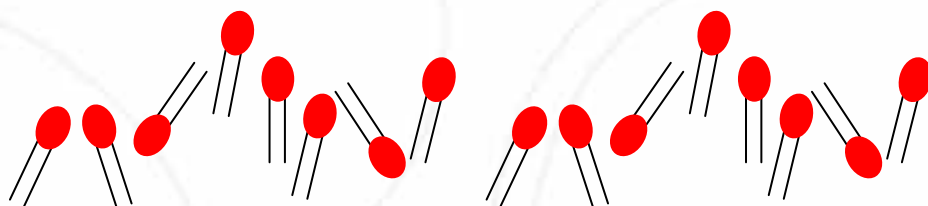
髪の毛を輪切り
2800本の線



情報とは？

- 物理学(物質とエネルギー)だけでは解き明かせないもの **情報**

次の図の違いは何か？



共にマッチが16本

従来の物理学では区別できない！！



情報理論

コード化 名前 出席番号
情報量の最小単位 ビット 0 or 1 (2進数)
情報を科学的に定義. 数式や方程式で
情報を扱えるようにした.



情報理論

コード化 名前 出席番号

情報量の最小単位 ビット 0 or 1 (2進数)

情報を科学的に定義. 数式や方程式で
情報を扱えるようにした.



情報理論

コード化 名前 出席番号

情報量の最小単位 ビット 0 or 1 (2進数)

情報を科学的に定義. 数式や方程式で
情報を扱えるようにした.



2進数

0と1だけで全ての数を表す。

(10進数は, 0から9で全ての数を表す)

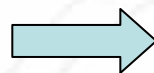
10進数	2進数
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111



2進数（画像）



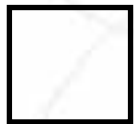
拡大



小さな四角形であるピクセルの集まり、各ピクセルはRGB値をもつ。



2進数（画像）



1ピクセルにつき，赤・緑・青の値を
0～255で表す。
(00000000 ~ 11111111)



赤：255，緑：0，青：0



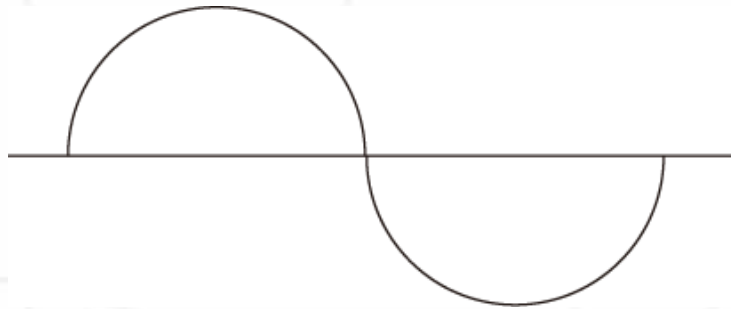
赤：255，緑：255，青：0



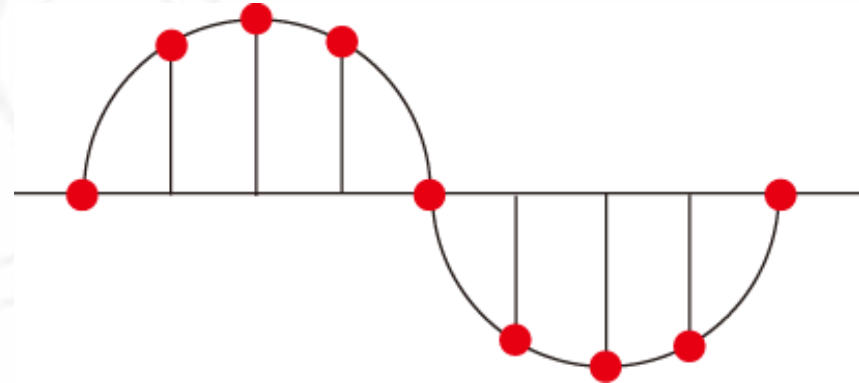
赤：100，緑：30，青：120



2進数（音楽）



アナログ波形



1/44100秒【44.1KHz】
でサンプリング(標本化)

2進数へ(量子化)

デジタル情報は劣化しにくい

伝言する場合：口頭(アナログ)と文字(デジタル)
テープとDVD



情報理論

コード化 名前 出席番号

情報量の最小単位 ビット 0 or 1 (2進数)

情報を科学的に定義. 数式や方程式で
情報を扱えるようにした.



ブール代数

出発点: 「論理や推論を数学に置き換える」

論理が正しければ1, 間違っていれば0
論理の数学化に成功

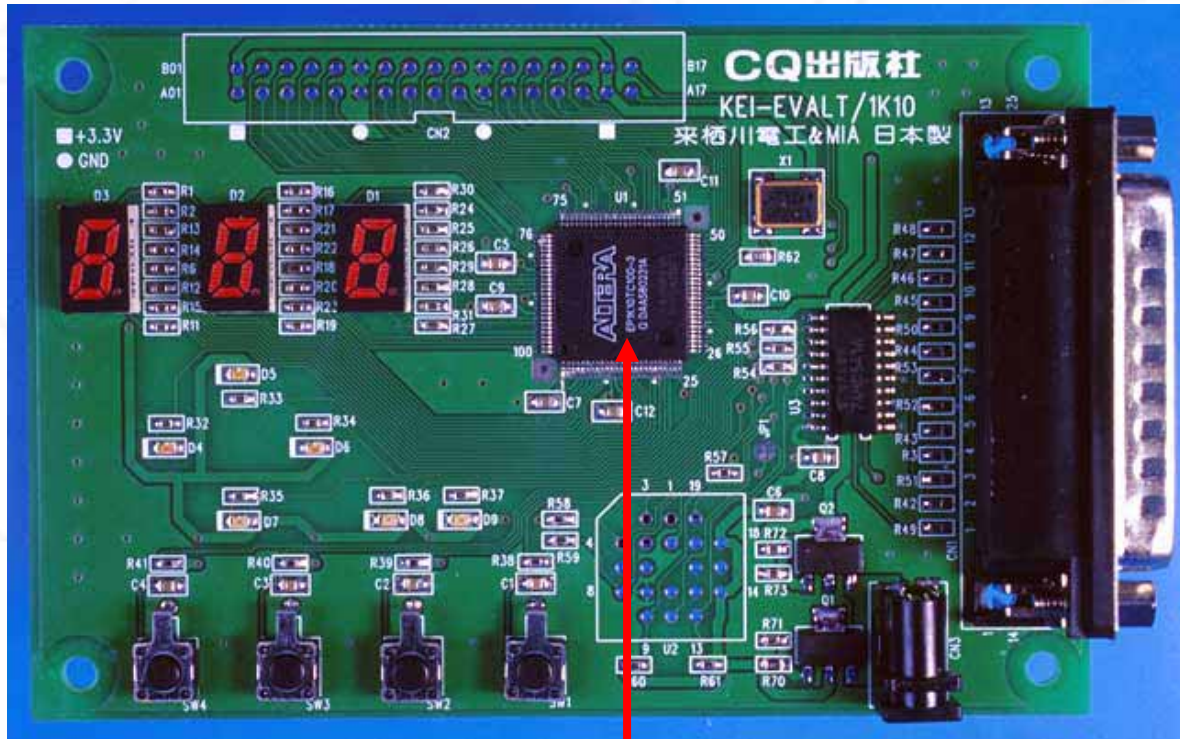
例えば, 「雨が降っている」 (1), × (0)

2進数 電流が流れている(1), 流れていない(0)

10進数 電圧1V, 2V, 3V ... 制御が非常に複雑
ノイズの影響で1.2Vになったら...不安定



デジタル回路設計へ



FPGA

FPGA (Field Programmable Gate Array)

LSI (Large Scale Integration) の一種。
専用LSIとは違い、中身の書き換え可能。

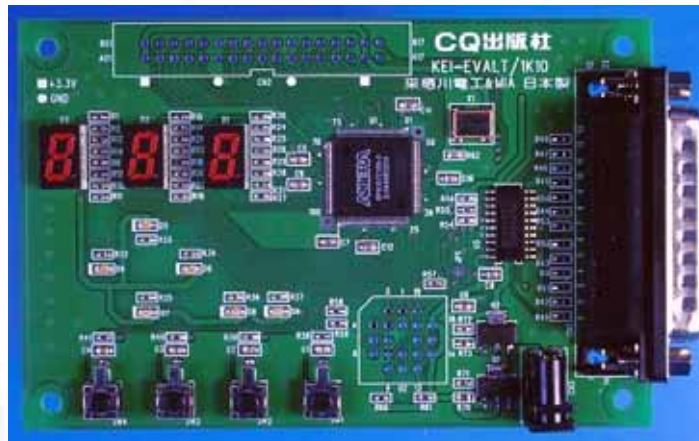


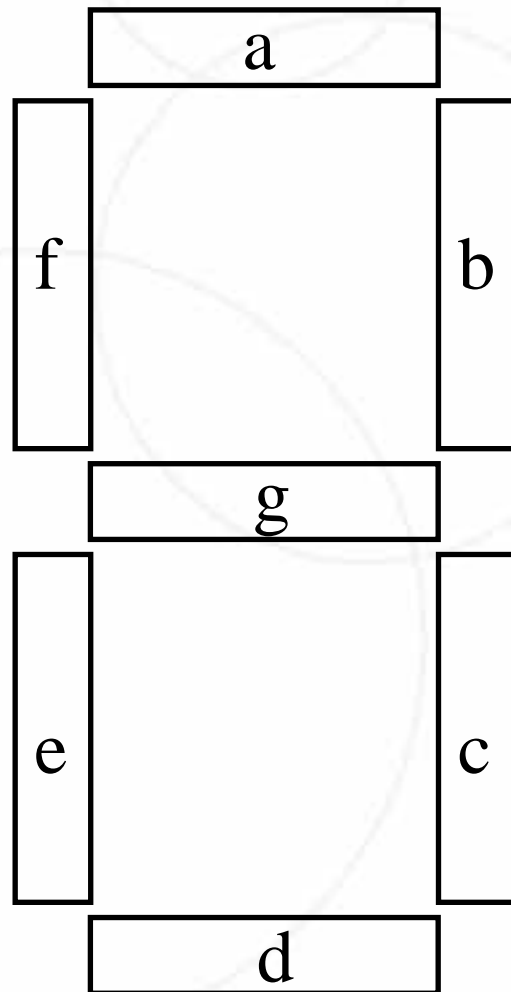
図1：今回使用するFPGAボード

開発期間とコスト削減，改善のしやすさから，
現在は様々な用途に普及が進んでいる。

動かしてみよう



演習 1 好きな数字を光らせよう！

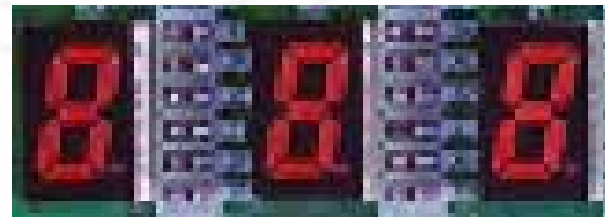


フォルダ「ex1」の下に「ex1.vhd」があります。

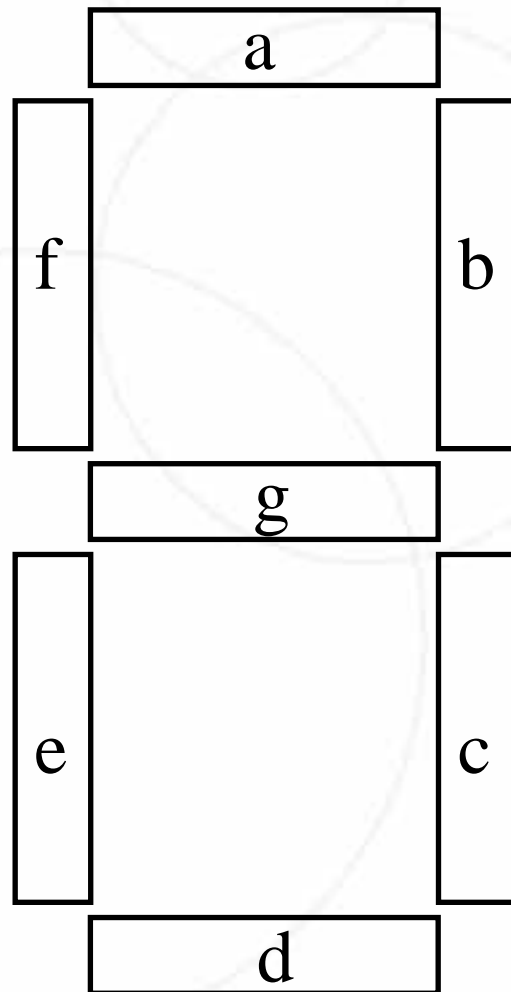
配布したマニュアルに従って

- ・プロジェクトの作成
- ・コンパイル
- ・ピンの配置
- ・コンパイル
- ・回路にデータを転送を行ってください。

dp



演習 1 好きな数字を光らせよう！

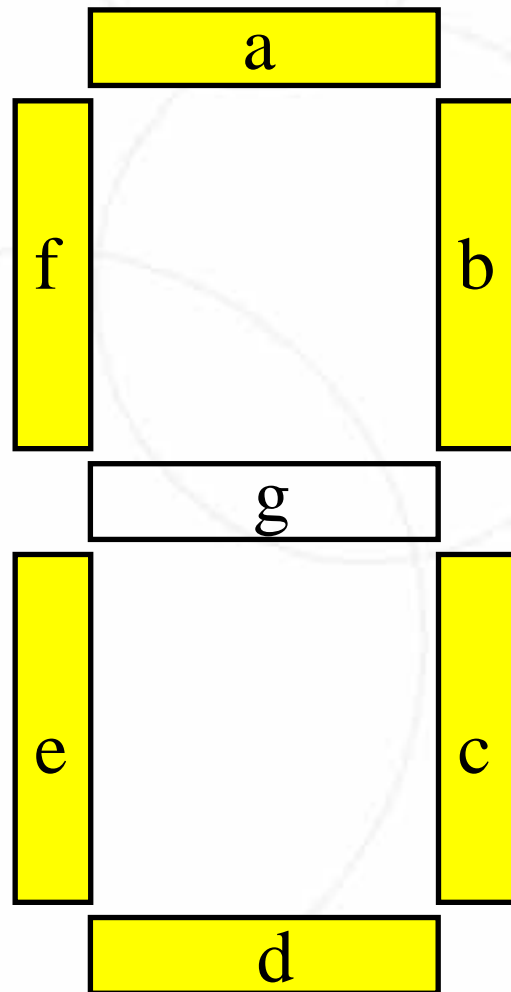


フォルダ「ex1」の下に「ex1.vhd」があります。

led_out の右の8桁の数字は
“a b c d e f g dp” の値が
入っています。各値が0であれば
対応する場所が光ります。

dp

演習 1 好きな数字を光らせよう！



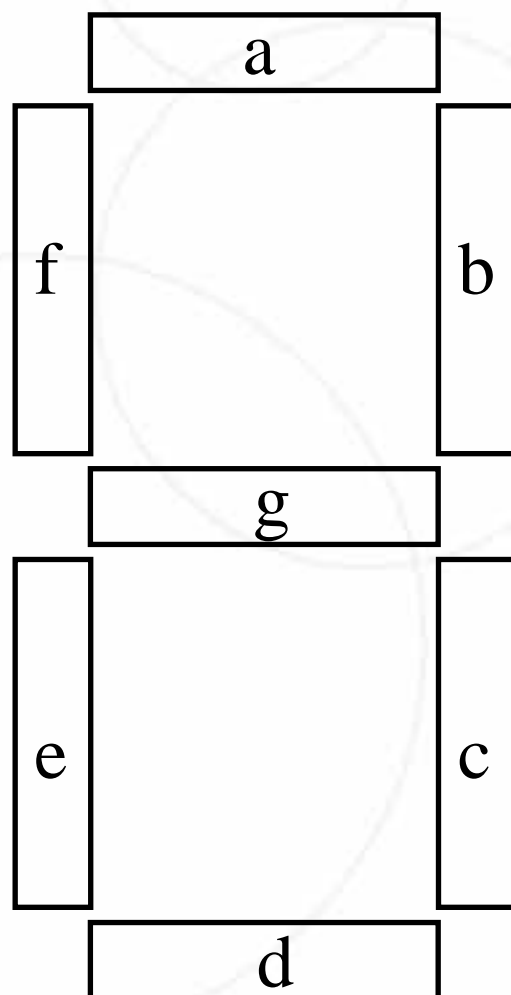
フォルダ「ex1」の下に「ex1.vhd」があります。

今, led_out の右の8桁の数字は
“00000011” なので
左の図のように 0 が表示されます。

値を変えて好きな数字を光らせよう！

dp

演習2 好きな数字を3つ光らせよう！

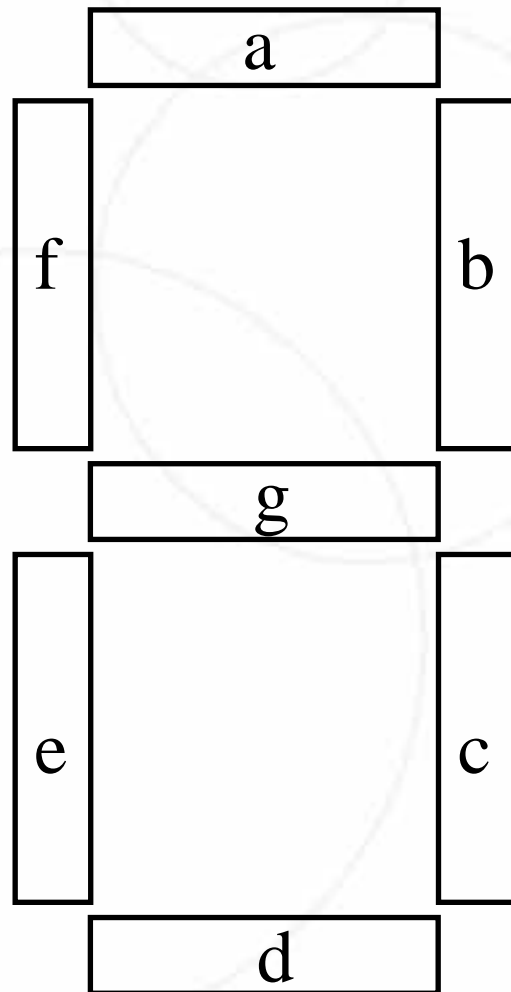


フォルダ「ex2」の下に「ex2.vhd」があります。

led_out の他に **led_out1**, **led_out2** を作成し, それぞれに8桁の0または1を代入し数字を3桁光らせよう。

dp

演習3 ストップウォッチを作ろう！



フォルダ「ex3」の下に「ex3.vhd」があります。

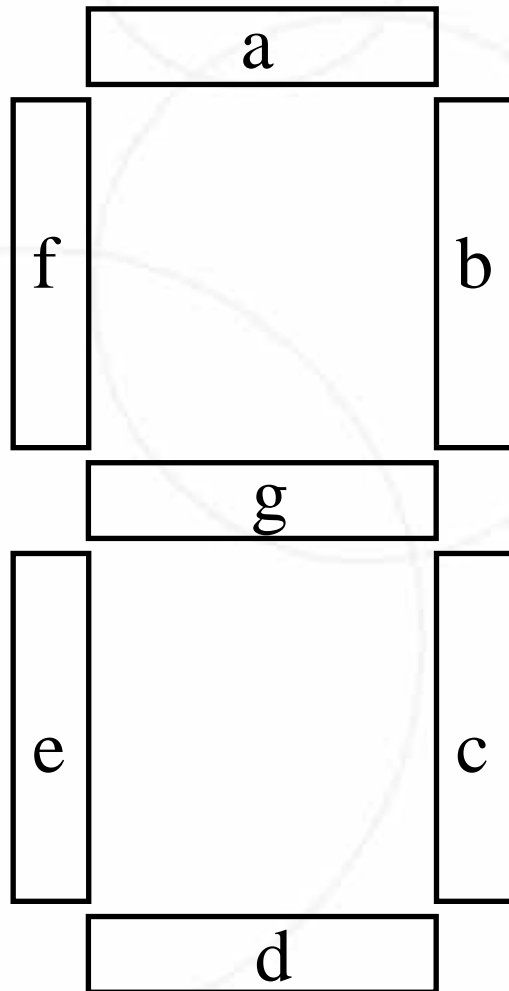
70, 71行目:

```
when "0000" =>  
    led_out <= "00000011";
```

2進数で0000は10進数の0
0が光るようにled_outに8桁を代入

dp

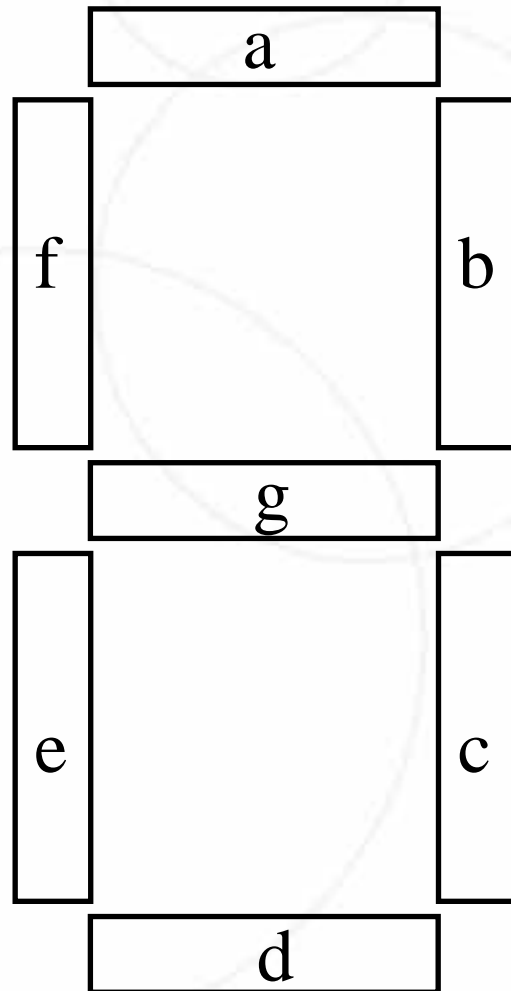
演習3 ストップウォッチを作ろう！



dp

2進数	10進数	led_out
0000	0	00000011
0001	1	?
0010	2	?
0011	3	?
0100	4	?
0101	5	?
0110	6	?
0111	7	?
1001	8	?
1010	9	?

演習3 ストップウォッチを作ろう！



73 行目の「--1」の右横の「」の中に1が光るように0または1を8桁記入しよう！

以下「--n」がある行でnが光るよう「」を埋めていこう！

73, 102, 131行目は同じものが入ります。

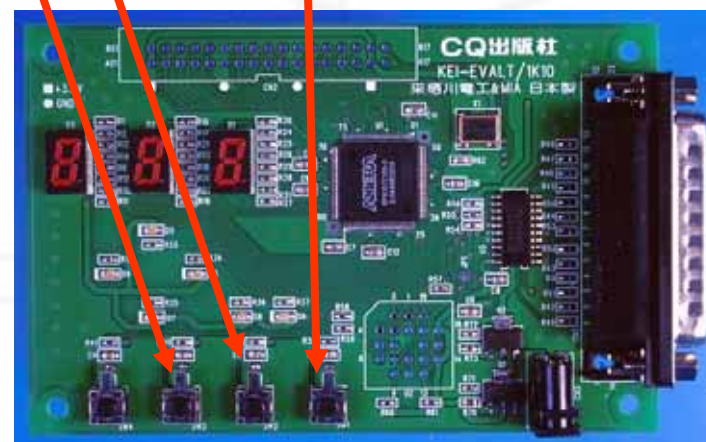
dp

演習3 ストップウォッチを作ろう！

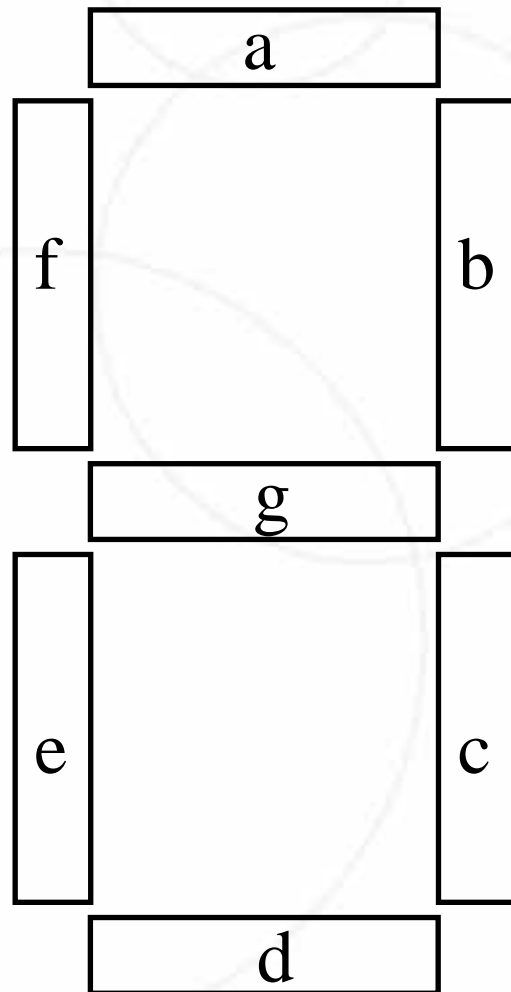
入力の中で

- ・ clk は39番ピン
- ・ sw1 は34番ピン (一番右のスイッチ SW1)
- ・ sw2 は33番ピン (SW2)
- ・ reset は32番ピン (SW3)

に割り当てよう。



演習4 ルーレットを作ろう！



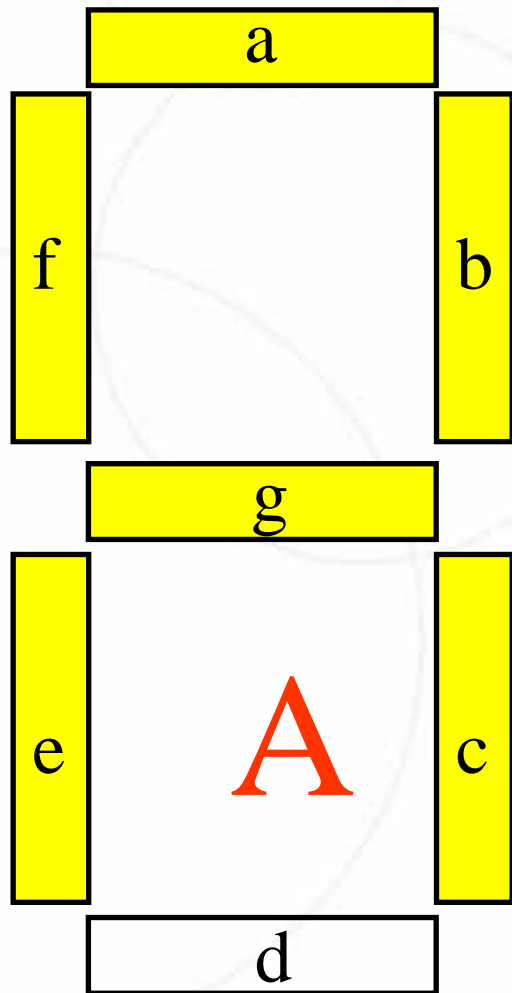
フォルダ「ex4」の下に「ex4.vhd」と「ex41.vhd」があります。

「ex41.vhd」の56行目から演習3と同じ要領で埋めていこう。

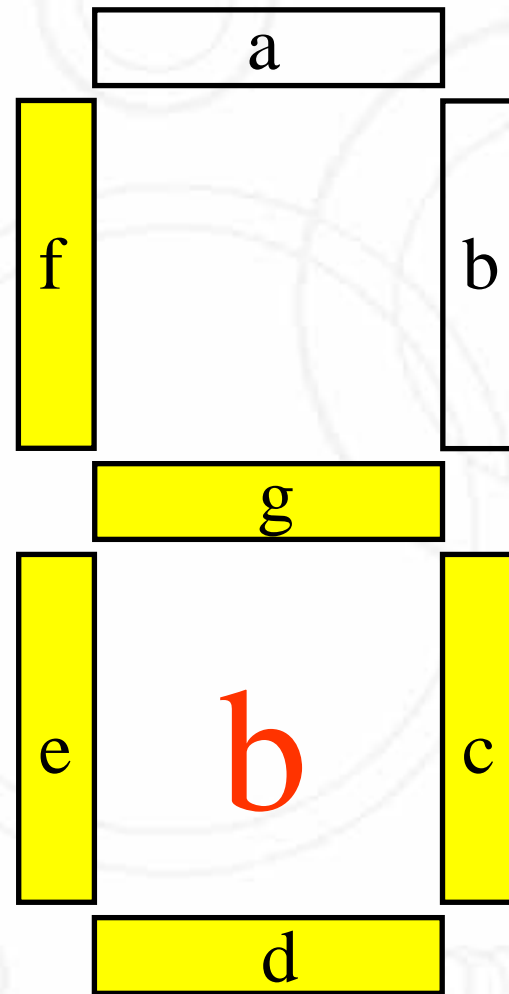
ここでは、a～fが追加されています。
(0～9, a～fを使って数字を表す手法を16進数といいます)

□ dp

演習4 ルーレットを作ろう！

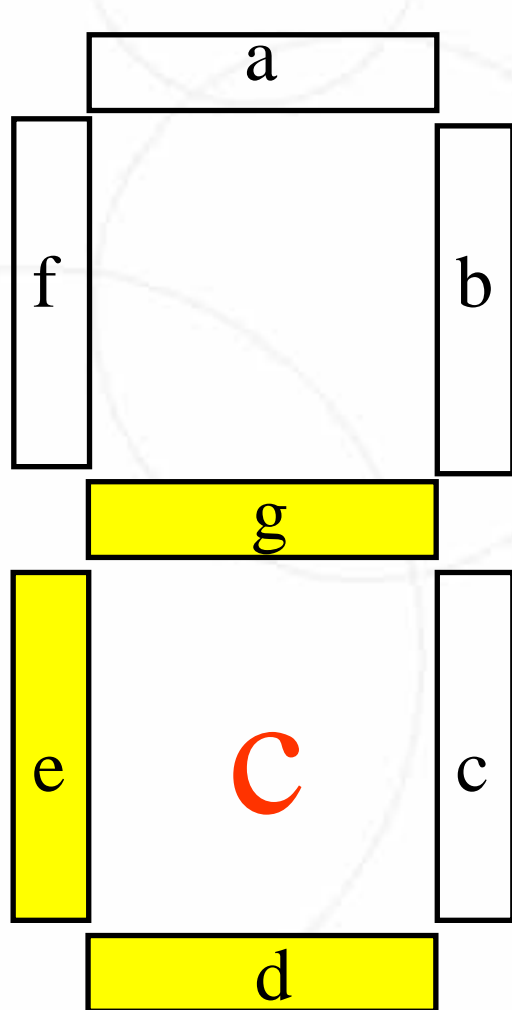


dp

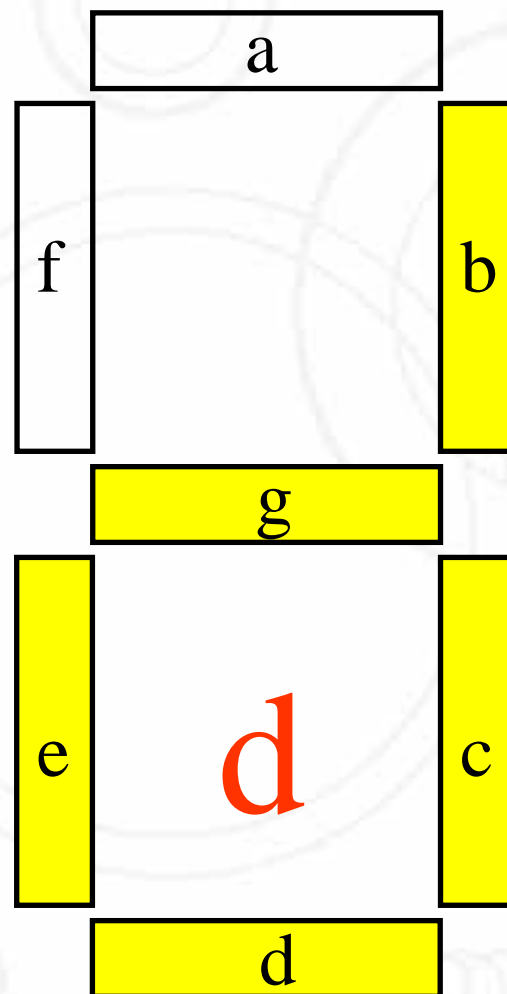


dp

演習4 ルーレットを作ろう！

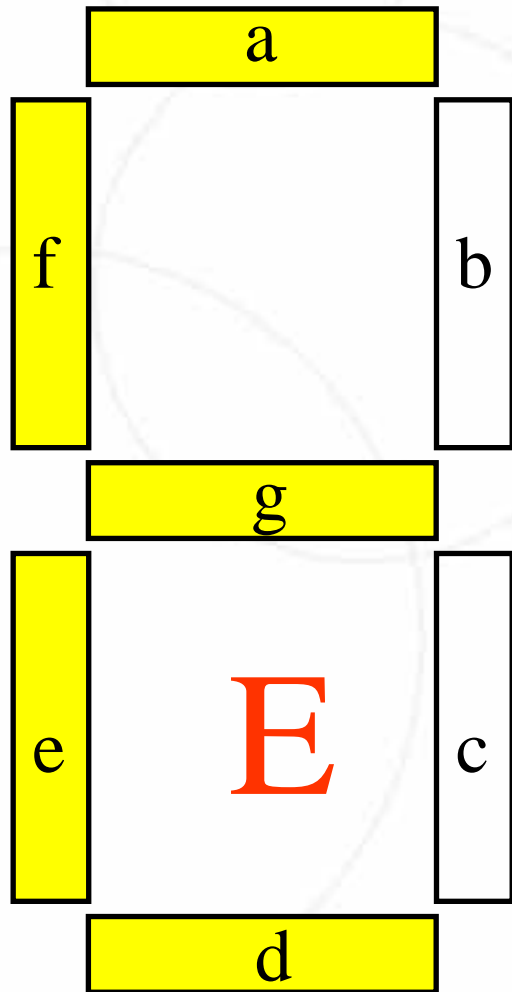


dp

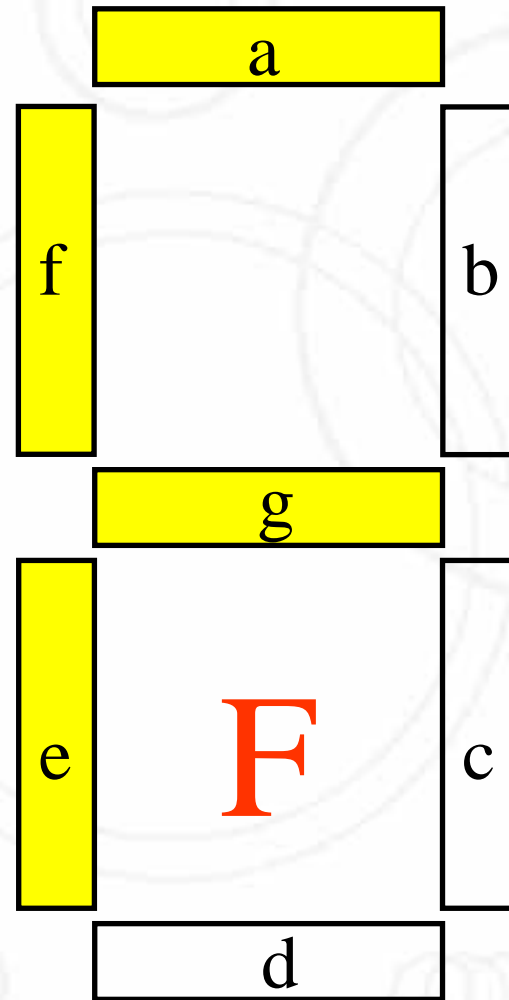


dp

a, e と f は大文字 (b, c, d は小文字)



dp



dp

演習4 ルーレットを作ろう！

入力の中で

- ・ clk は39番ピン
- ・ sw_in は34番ピン (SW1)
- ・ resetn は33番ピン (SW2)

に割り当てよう。