

システム設計演習（前期分）

秋田純一

<http://j.mp/akita-class>

akita@ifdl.jp (@akita11)

この講義の目標

- ✓ コンピュータに代表される情報機器はハードウェアとソフトウェアが連携して機能するシステムが重要な役割を果たしている。前期ではソフトウェアによるデジタル回路の記述, 論理合成などのVLSI設計の基礎について知識を身につけ, 演習を通して課題の解決方法を具体的に学ぶ。後期では前期で学んだ知識を含め, デジタル回路, 電子回路(アナログ回路), コンピュータアーキテクチャなどのハードウェアとプログラミングやアルゴリズムなどのソフトウェアの知識を組み合わせたシステムを設計, 製作することにより総合的な創造力を養うとともに問題点を自分で解決できる力を身につける。

VLSIとは・・・？

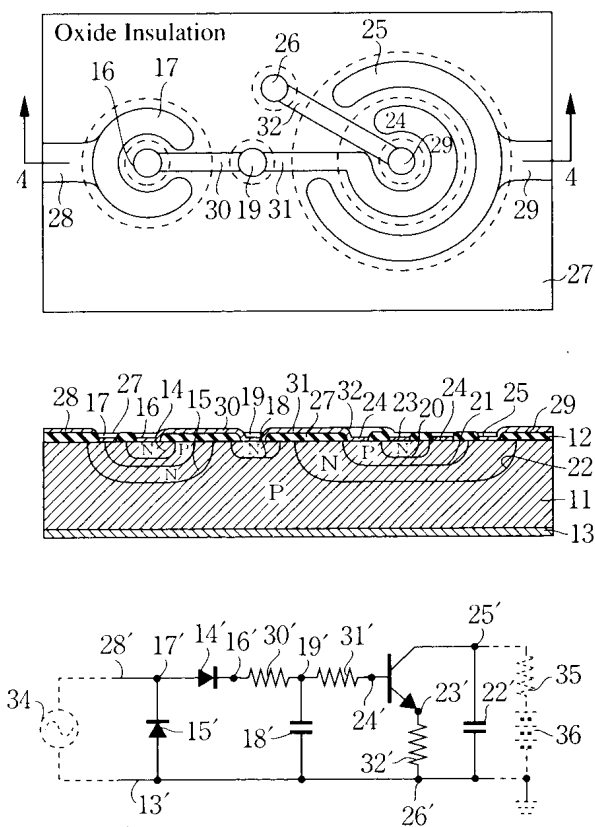
- ✓ VLSI = Very Large Scale Integration
(大規模集積回路)
- ✓ 集積回路 = 「集積」された「回路」
 - ✓ 素子 (トランジスタ、抵抗、・・・) + 配線
 - ✓ 通常はシリコンなどの中に作りこまれる

集積回路はどこにある？

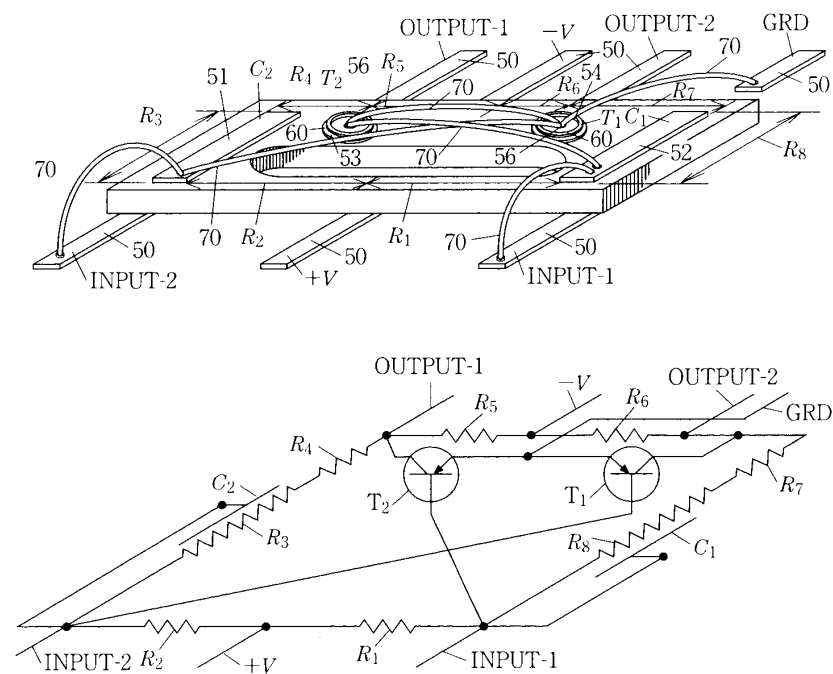
- ✓ ほとんどあらゆる電子機器・情報機器の中
 - ✓ 「黒いムカデ」のような形の部品
 - ✓ 中に「シリコンのチップ」が入っている



世界最初の集積回路



US Patent No. 2 981 877 (R. Noyce)
(1961)



US Patent No. 2 138 743 (J. Kilby)
(1959)

集積回路の歴史(プロセッサ)

☑️マイクロプロセッサ(MPU)

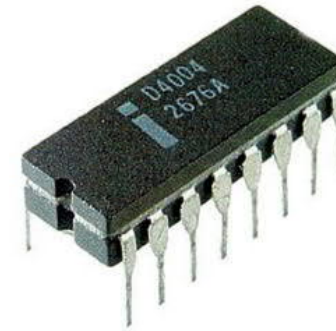
☑️i4004(1970, Intel)

☑️i8008(1972, Intel)

☑️i8086(1978, Intel)

☑️68000(1979, Motorola)

☑️Pentium(1993, Intel)



$f = 714\text{kHz}$, 2300Tr.



$f = 66\text{MHz}$, 3,100,000Tr.

集積回路の歴史(メモリ)

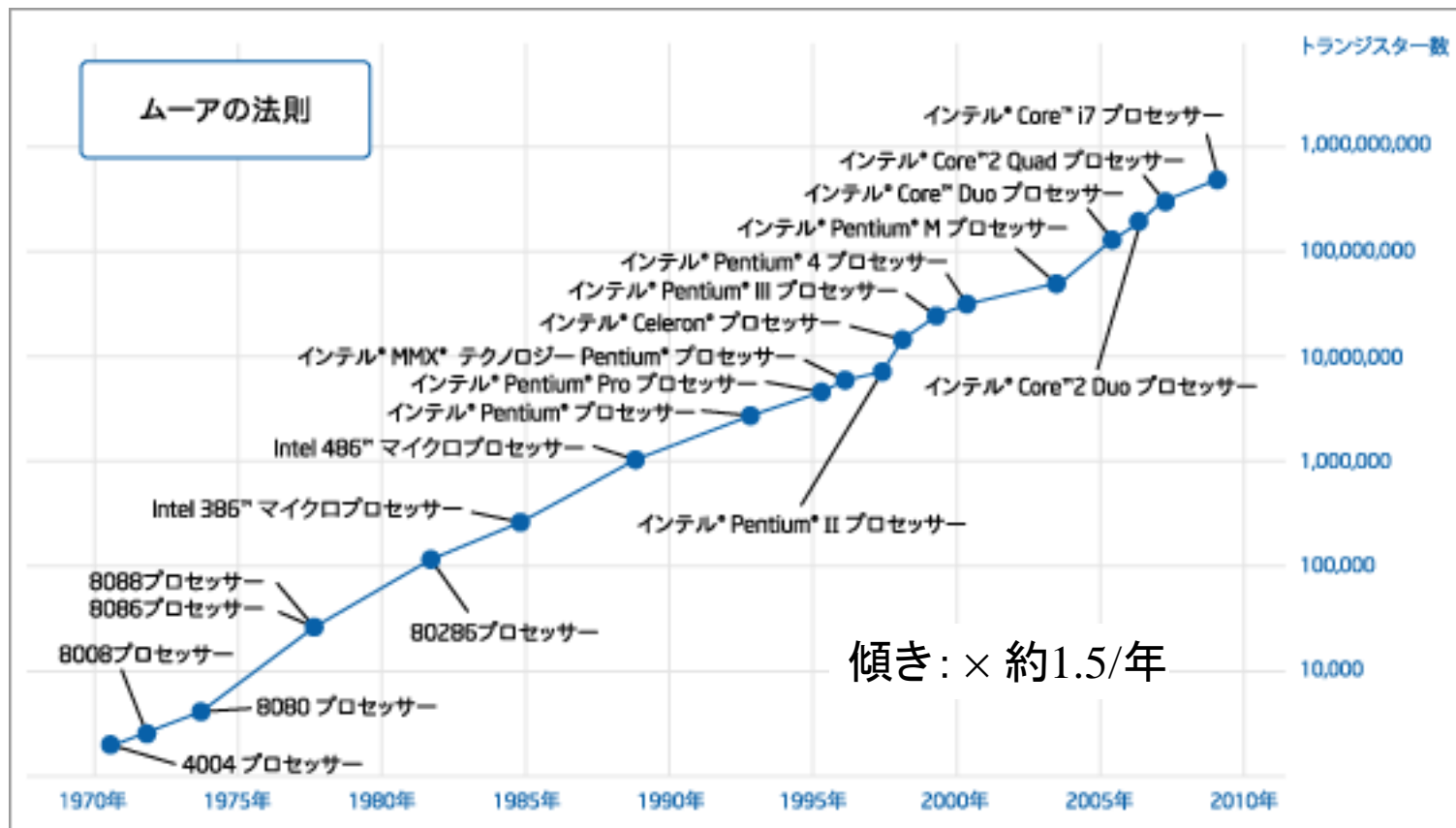
☑ メモリIC (DRAM)

- ☑ 1kb (1972, Intel)
- ☑ 4kb (1975, Texas Inst.)
- ☑ 16kb (1977, Mostek)
- ☑ 64kb (1980, Hitachi)
- ☑ 256kb (1983, Fujitsu)
- ☑ 1Mb (1986, Toshiba)
- ☑ 4Mb (1989, Hitachi)
- ☑ 16Mb (1991, Hitachi)
- ☑ 64Mb (1994, NEC, Samsung)
- ☑ 256Mb (1997, Samsung)
- ☑ 512Mb (2003, Samsung)
- ☑ 1Gb (2004, Samsung)



メモリ容量:
3年で4倍
32年で1,000,000倍

Mooreの法則



ref: <http://www.intel.com/jp/intel/museum/processor/index.htm>

Mooreの法則のクラクリ: スケーリング

☑️ MOSTランジスタを、より小さく作ると・・・？

☑️ 寸法: $1/\alpha$

☑️ 不純物濃度: α

☑️ 電源電圧: $1/\alpha$

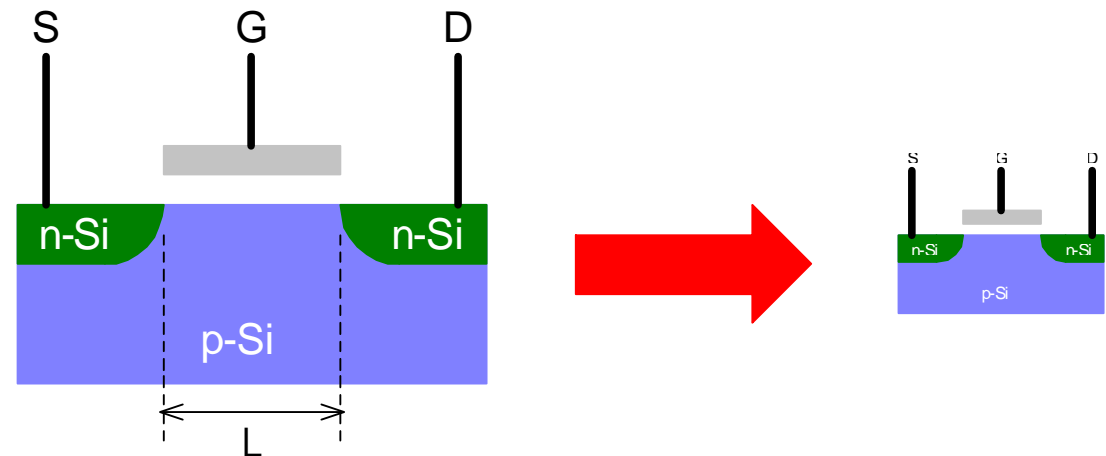
☑️ 結論: いいことばかり

☑️ 速度↑

☑️ 消費電力↓

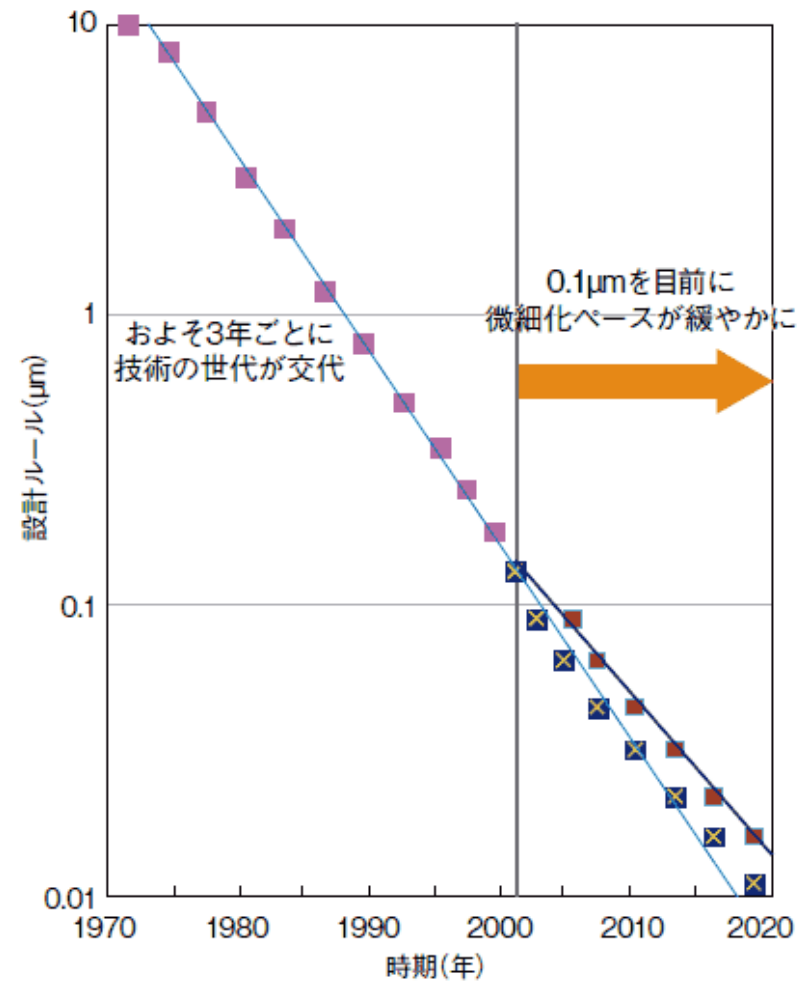
☑️ 集積度(機能)↑

☑️ 技術が進むべき方向性が極めて明確なまれなケース



MOSTランジスタの微細化の歴史

- ✓ 微細化するほど
メリットがある
= がんばって微細化



ref: 日経BP Tech-On! 2009/03/30の記事

スケーリング（微細化）でうれしいこと

✓速度↑

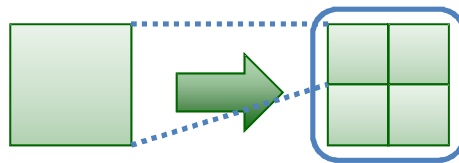
✓パソコンや携帯・スマホがサクサク動く

✓消費電力↓

✓バッテリーが長持ち

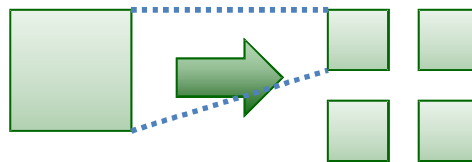
✓集積度↑: 2つの意味

✓機能↑



同一面積チップで4倍のMOS数
= 4倍の機能

✓コスト↓

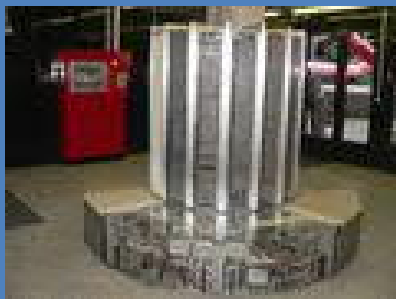


同一MOS数が1/4の面積
= 1/4のコスト

微細化によるコスト↓の別の側面



DEC VAX(1976)
1MIPS



Cray-1 (1978)
100MIPS

(世界最初のスーパーコンピュータ)



1000MIPS



100MIPS



300MIPS



20MIPS



10MIPS

※MIPS:Million Instruction Per Second (1秒間に実行できる命令数)

微細化によるコスト↓の意義

- ☑ コンピュータの低価格化＝普及
 - ☑ 昔は国で1台 → 会社に1台 → 一人1台
- ☑ もう一つの意義：
 - ☑ 「コンピュータ」が特殊なものではなくなった
 - ☑ コンピュータ＝パソコン、にとどまらない
 - ☑ 携帯、ゲーム機、家電、おもちゃ、・・・
 - ☑ →身の回りのあらゆるものに(ユビキタス化)

集積回路の今後に対する別の見方

✓ 集積回路の製造・設計技術の成熟

✓ 高性能な集積回路 (MPU, メモリ, ...)

✓ 低価格な集積回路 (性能はそこそこ)

✓ 高性能な集積回路:

✓ = 最先端の製造技術 (工場)

→ (~1000億円以上)

✓ 低価格な集積回路

✓ 性能はそこそこ。しかし...

これからの情報技術と集積回路

☑「道具」としての集積回路

- ☑設計技術・製造技術の成熟

- ☑誰でも使えるようになってきた

☑情報技術の実現方法としての集積回路

- ☑パソコンを使ってプログラム：
できることは、たかが知っている
(パソコンの枠の中だけの世界)

- ☑面白いもの・作りたいものを実現したいときに、
道具として活用する(実世界とのつながり)

集積回路を「道具」にするためには・・・

- ☑「作りたいもの」を設計しなければいけない
- ☑1億個のトランジスタが「使える」(available)
- ☑・・・どうやって設計するのか？

- ☑「作りたいもの」を「言語」で記述する
 - ☑Hardware Description Language: HDL
 - ☑言語で書いたものを、半自動で集積回路に変換
(トップダウン設計)

道具になる集積回路：CGMとニコ動

- ✓CGM(Consumer Generated Media)
 - ✓プロでない人がつくるコンテンツ
 - ✓(最近は特に音楽が多い)
 - ✓(ニコニコ技術部で電子工作も)

niconico | 動画 | 静画 | 生放送 | チャンネル | その他 ▼

ニコニコ動画(原宿)
NICO NICO DOUGA

あなたの Mac から

動画(21,857) | マイリスト(1,222) | 静画(140) | 生放送

ニコニコ技術部
ニコニコ技術部 を含む動画が 21,857 件見つかりました (1 - 32 件中)

関連タグ: ニコニコ技術部 ニコニコ技術部養成講座 ニコニコ技術部情報システム課 ニコニコ技術部キャ...
ニコニコ技術部ランキング wtkニコニコ技術部! ニコニコ技術部見習い ニコニコ技術部 (笑)

コメントが新しい順 | 検索オプション | 1 | 2 | 3 | 4 | 次へ

2012年10月28日 16:40 投稿
【痛グラス】『よつぱと』をグラスに影ってみた
■今回は『よつぱと』8巻の表紙をグラスに彫りました■製作時間は15分
間程で標準です■ニコ生のタイムシフトを使っているので時刻が写っている場合があ...

2012年10月28日 00:43 投稿
ペンストープ極太で煮込うどん
ペンストープ極太で煮込うどんを作りました。

2012年10月27日 22:02 投稿
カップヌードルを逆さから攻めたら旨さの工夫がわかっ
カップヌードルって麺と底にかなり間があるんだね、これで均等に麺が染
かくなるんだね!! 蛍光ペンを水に溶かしてブラックライト当てるとクワ
なく...

ifDL
Kanazawa Univ.

これらの動きが出てきた背景・・・？

☑️ 道具の進化

☑️ マイコン、初音ミク、レーザーカッター

☑️ 成果発信の手段

☑️ Web、ニコ動、SNS（Facebook/Twitter等）

☑️ 相互評価・尊重の文化

☑️ これまでは「自己満足」で終わっていた

未来の「ものづくり」・・・？

☑例：野尻抱介「南極点のピアピア動画」

日本の次期月探査計画に関わっていた大学院生・蓮見省一の夢は、彗星が月面に衝突した瞬間に潰え、恋人の奈美までが彼のもとを去った。省一はただ、奈美への愛をボーカロイドの小隅レイに歌わせ、ピアピア動画にアップロードするしかなかった。しかし、月からの放出物が地球に双極ジェットを形成することが判明、ピアピア技術部による“宇宙男プロジェクト”が開始される……ネットと宇宙開発の未来を描く4篇収録の連作集

・・・？？？



(早川書房・ISBN: 4150310580)

要約(ネタバレ)と「示唆」

- ☑️ ニコニコ技術部でロケットつくって宇宙に行ったり、潜水艦でクジラと会話する、というお話
- ☑️ この小説の示唆・・・？(私の解釈)
 - ☑️ 個々人の才能は尖っている(レベルが高い)
 - ☑️ 皆で力をあわせると、すごいことができる
 - ☑️ 現在は、皆が「趣味」の範囲でやっている
 - ☑️ はたしてこれが産業になるのか・・・？

(近)未来の「ものづくり」・・・？

- ☑ 「ニコ技で作る」ことを職業にできる人は、
(おそらく)ほとんどいない
 - ☑ 今の職業を放りだしてまで取り組む自信が(たぶん)ない
- ☑ しかし「趣味の時間」でならできるところ
 - ☑ (睡眠時間を減らすのは持続的ではない)
 - ☑ TV観たりマンガ読んだりジョギングするの代わりに「ものづくり」
 - ☑ 「道具」(集積回路を含む)はそろってきている
 - ☑ 人を束ねるベース(SNS等)は既にある
 - ☑ Kickstarterのような少額資金調達の素地もある
 - ☑ あとはこれを販売するルートがあれば現代の産業革命になるか・・・？
 - ☑ おそらく少量多品種&オープンソース。だからこそFabLabでつくれる
 - ☑ (「大量生産」は、現在の嗜好の多様化の時代にはそぐわない:たぶん)

講義のスケジュール

- ✓ 第1週(4/11) イントロダクション・HDL概要
- ✓ 第2週(4/18) 組み合わせ論理回路のHDL記述
- ✓ 第3週(4/25) 実習(1)
- ✓ 第4週(5/2) 順序回路のHDL記述
- ✓ 第5週(5/23) 実習(2)(カウンタを中心に)
- ✓ 第6週(5/30) 実習(3)(分周回路を中心に)
- ✓ (6/6頃) 前期中間試験
- ✓ 第7週(6/20) 順序回路とデータパス
- ✓ 第8週(6/27) 実習(4)
- ✓ 第9週(7/1月:補講) 実習(5)
- ✓ 第10週(7/8月:補講) マイクロプロセッサの動作
- ✓ 第11週(7/11) 実習(6)(レジスタ・ALUを中心に)
- ✓ 第12週(7/18) 実習(7)(メモリ・命令実行を中心に)
- ✓ 第13週(7/25) 実習(8)(条件分岐を中心に)
- ✓ (8/1頃) 前期末試験
- ✓ 9/26 答案返却
- ✓ ※休講: 5/9,6/13,7/4

HDL概要

☑「論理回路」を記述する「言語」

☑主なもの: VHDL, VerilogHDL

☑例 (NOTゲート=インバータ)

```
library ieee;
use ieee.std_logic_1164.all;

entity not is
  port (
    a: in std_logic;
    x: out std_logic);
end not;

architecture arch of not is
begin
  x <= not a;
end arch;
```

ヘッダなど(おまじない)

エンティティ記述(回路の入出力の定義)
※in=入力、out=出力。“std_logic”は、「0/1の値」
回路名(“not”)

アーキテクチャ記述(回路本体の機能)

HDL記述の例：半加算器

```
entity half_adder is
  port (
    a, b : in std_logic;
    s, co : out std_logic);
end half_adder;
```

```
architecture arch of half_adder is
  signal w0, w1, w2: std_logic;
begin
  w0 <= a and b;
  w1 <= not w0;
  w2 <= a or b;
  s <= w1 and w2;
  co <= w0;
end arch;
```

回路図

HDL記述の例：全加算器

```
entity full_adder is
  port (
    a, b, ci: in std_logic;
    s, co: out std_logic);
end full_adder;

architecture arch of full_adder is
  component half_adder
    port (
      a, b: in std_logic;
      s, co: out std_logic);
  end component;
  signal w0, w1, w2: std_logic;
begin
  co <= w1 or w2;
  i0: half_adder port map (co=>w1, s=>w0, a=>a, b=>b);
  i1: half_adder port map (co=>w2, s=>s, a=>w0, b=>ci);
end arch;
```

回路図

※このような書き方を「構造記述」を呼ぶ

HDL記述の例：4ビット加算器（1）

```
entity adder4 is
  port (
    a, b: in std_logic_vector(3 downto 0);
    ci : in std_logic;
    s: out std_logic_vector(3 downto 0); ← 4本の信号線(バス)をまとめて宣言
    co: out std_logic);
end adder4;
```

```
architecture arch of adder4 is
  component full_adder
    port (
      a, b, ci: in std_logic;
      s, co: out std_logic);
  end component;
  signal w0, w1, w2: std_logic; ← Signal宣言(使う信号線の宣言)
begin
  i0: full_adder port map(co=>w0, s=>s(0), a=>a(0), b=>b(0), ci=>ci);
  i1: full_adder port map(co=>w1, s=>s(1), a=>a(1), b=>b(1), ci=>w0);
  i2: full_adder port map(co=>w2, s=>s(2), a=>a(2), b=>b(2), ci=>w1);
  i3: full_adder port map(co=>co, s=>s(3), a=>a(3), b=>b(3), ci=>w2);
end arch;
```

※このような書き方を「構造記述」を呼ぶ

構造記述された4ビット加算器

回路図

HDL記述の例：4ビット加算器（2）

```
entity adder4 is
  port (
    a, b: in std_logic_vector(3 downto 0);
    ci : in std_logic;
    s: out std_logic_vector(3 downto 0);
    co: out std_logic);
end adder4;
```

```
architecture behv of adder4 is
begin
  process (a, b, ci)
    variable tmp: std_logic_vector(4 downto 0);
  begin
    tmp := ("0"&a) + ("0"&b) + ("0000"&ci);
    co <= tmp(4);
    s <= tmp(3 downto 0);
  end process
end behv;
```

} 4ビット加算器の動作
(=「加算する」)を記述

※このような書き方を「動作記述」を呼ぶ

4ビット加算器動作記述での細かいこと

- ☑️ 動作記述では"process"文を使う
 - ☑️ processの()内は、process文の動作を「行う」ための条件（センシティブティ・リスト(sensitivity list)）
 - ☑️ この()内の変数が変化したら、「動作」を行う
- ☑️ process文の中で代入する信号はvariableで宣言
- ☑️ 動作記述の「加算の動作」では、両辺のビット数をそろえる
 - ☑️ この例では両辺共に5ビットとなるように、"&"でビット数をそろえる（「"0000"&ci」で4+1=5ビット、など）