演算子

●演算子

優先順位別					
not	*	+	=	and
abs	/	-	/=	or
**(冪)	mod	&	<	xor
	rem		>	nand
			<=	nor
			=>	

高 ←― 優先順位 ―→ 低

演算機能別	論理	関係	連接	算術
	not	=	&	+
	and	/=		-
	or	<		*
	xor	>		**(冪)
	nand	<=		/
	nor	=>		mod
				rem
				abs

本科目では扱わない

項数別	単項	2項
	not	and
	abs	or
	+	xor
	-	+
		など
		その他
	書式例	
	not A	B and C

簡単な例

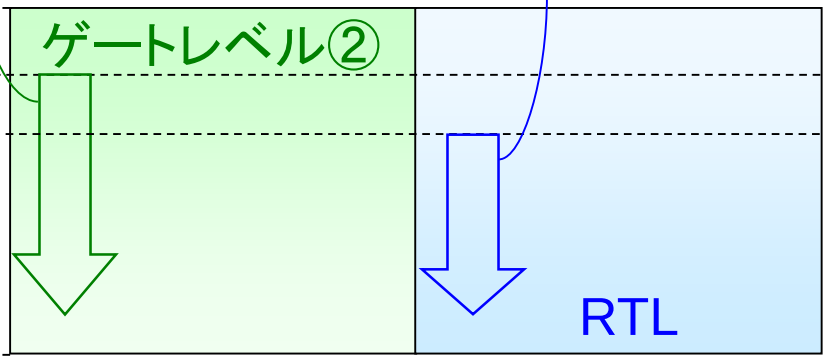
論理式記述-組み合わせ回路

機能記述-組み合わせ回路

機能記述-レジスタ

複合回路

高度な例



（例）組み合わせ回路の機能記述その1―A

●算術演算子のみを用いた組み合わせ回路の機能記述

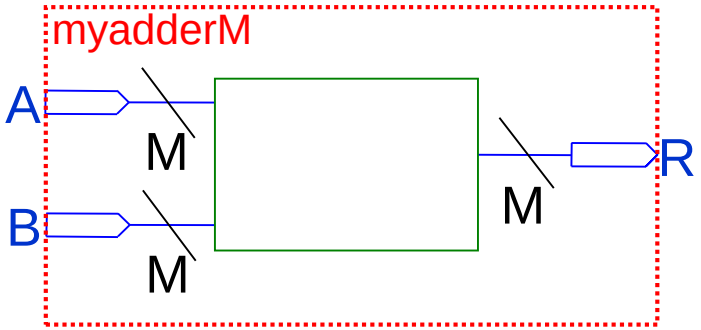
エンティティ名.vhd

```
entity エンティティ名 is
      ジェネリック宣言の記述
      入出力ポートの記述
end エンティティ名;

architecture RTL of エンティティ名 is
各種宣言の記述
begin
      回路機能の記述
end RTL;
```

キャリー入出力なしMビット加算回路

myadderM.vhd



ジェネリック宣言の記述

```
generic(M:integer:=4);
```

入出力ポートの記述

```
port(
  A : in std_logic_vector(M-1 downto 0);
  B : in std_logic_vector(M-1 downto 0);
  R : out std_logic_vector(M-1 downto 0)
);
```

回路機能の記述

```
R <= A + B;
```

【注】
本科目で用いている[パッケージ]により
算術演算子に対する、2つの被演算子の
ビット長は異なっていてもよい。

（例）組み合わせ回路の機能記述その1―B

●連接演算子のみを用いた組み合わせ回路の機能記述

ジェネリック宣言の記述

```
generic(M:integer:=4);
```

入出力ポートの記述

```
port(
  A : in std_logic_vector(M-1 downto 0);
  X : out std_logic_vector(M-1 downto 0)
);
```

回路機能の記述

① `X<='0'&A(M-1 downto 1);` 論理右シフト

② `X<=A(M-1)&A(M-1 downto 1);` 算術右シフト

③ `X<=A(0)&A(M-1 downto 1);` 右ローテート

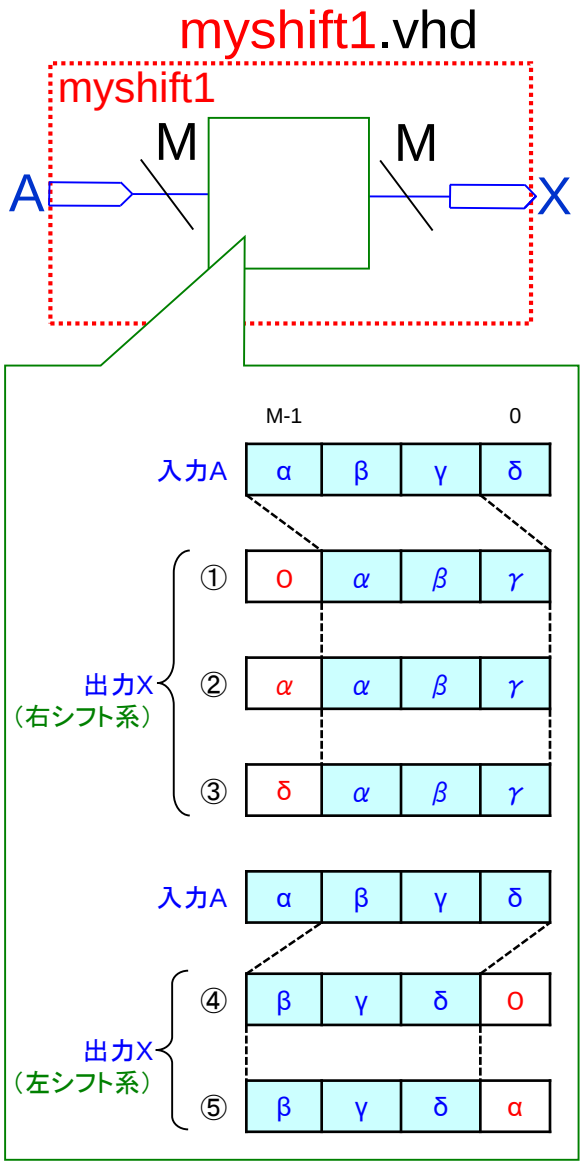
④ `X<=A(M-2 downto 0)&'0';` 左シフト

⑤ `X<=A(M-2 downto 0)&A(M-1);` 左ローテート

【注】連接演算子の解説 HW-5_VHDL-01.pdf p.7 も参考に

1ビットシフト回路

myshift1.vhd

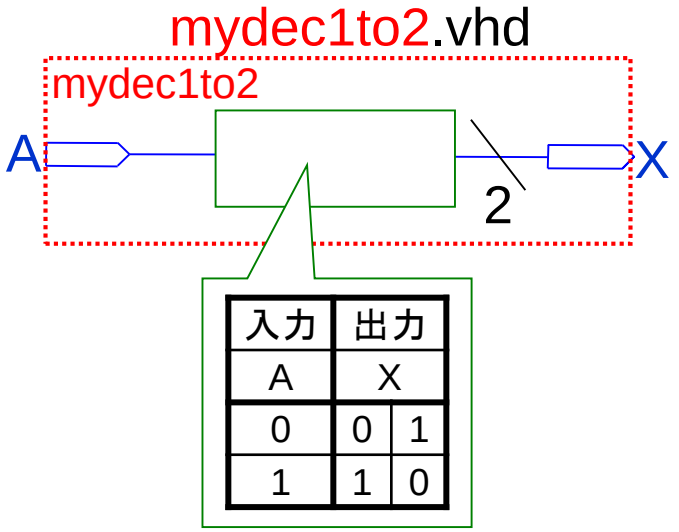


（例）組み合わせ回路の機能記述その2

●条件付信号代入文を用いた組み合わせ回路の機能記述

1入力2出力デコーダ回路

mydec1to2.vhd



入出力ポートの記述

```
port(
  A : in std_logic;
  X : out std_logic_vector(1 downto 0)
);
```

回路機能の記述

（ふつうの）信号代入文	条件付信号代入文
真理値表 → 論理式 ↓ 信号代入文	条件付信号代入文

入力	出力	
A	X	
0	0	1
1	1	0

```
X <= "01" when(A='0') else
     "10";
```

条件付信号代入文

●条件付信号代入文:
条件iに対応する、定数iまたは式iで決まる値を、信号に代入する。(i=1,…,n)
条件1～条件nのどれをも満たさない場合は、
定数Eまたは式Eで決まる値を、信号に代入する。

書式

```
信号名 <= 定数1または式1 when(条件1) else
          定数2または式2 when(条件2) else
                  ⋮
          定数Nまたは式n when(条件n) else
          定数Eまたは式E;
```

- 信号名(左辺とは別の)
- 演算式

関係演算子を除く

●演算子

【※】信号代入の記号ではない

優先順位別					
not	*	+	=	and
abs	/	-	/=	or
**(冪)	mod	&	<	xor
	rem		>	nand
			<= 【※】	nor
			=>	

高 ←優先順位→ 低

演算機能別	論理	関係	連接	算術
	not	=	&	+
	and	/=		-
	or	<		*
	xor	>		**(冪)
	nand	<= 【※】		/
	nor	=>		mod
				rem
				abs

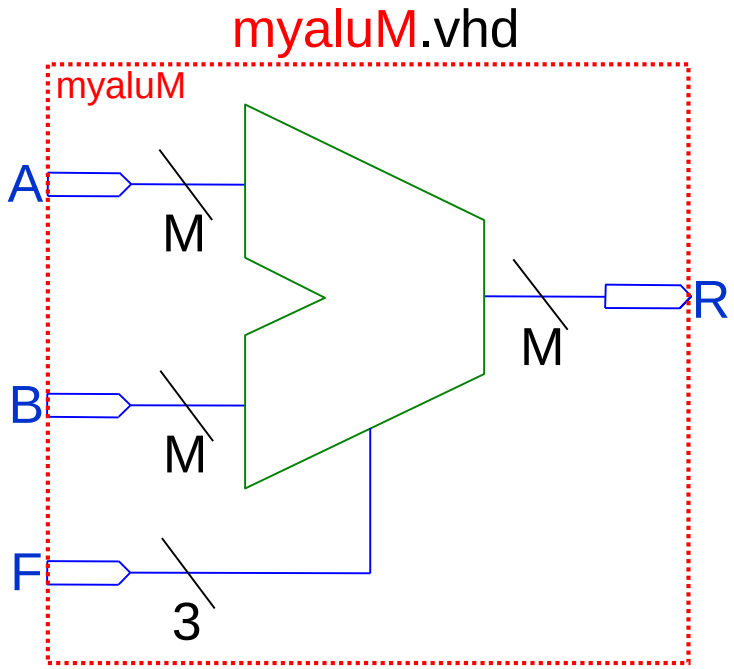
項数別	単項	2項
	not	and
	abs	or
	+	xor
	-	+
		など
		その他
書式例	not A	B and C

■必須課題■

■プロジェクト「myaluM」
下の表は、MビットALUの動作表である。
これについて、条件付信号代入文を用いて回路を設計せよ。

なお、
・Mはジェネリック宣言で指定すること。
・算術加算の記述には、算術演算子「+」を使用すること。

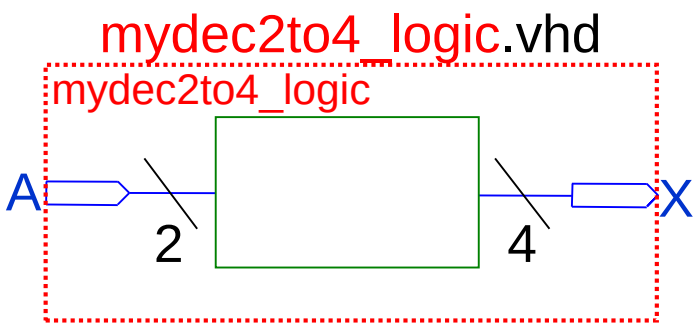
入力			出力
F			R
0	0	0	AとBの算術加算値（キャリー出力なし）
0	0	1	AとBの論理積の値
0	1	0	AとBの論理和の値
0	1	1	AとBの排他的論理和の値
その他			Aの値



▲追加課題▲

▲プロジェクト「mydec2to4」
下の表は、2入力4出力デコーダの真理値表である。
これについて、条件付信号代入文を用いて回路を設計せよ。

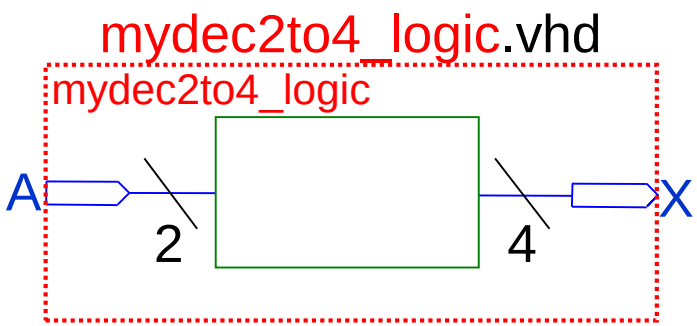
入力		出力			
A		X			
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0



▲追加課題▲

▲プロジェクト「mydec2to4_logic」
下の表は、2入力4出力デコーダの真理値表である。
これについて、「真理値表→論理式→信号代入文」の手順で回路を設計せよ。

入力		出力			
A		X			
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0



これまでの例の整理

Top-Level Entity名	スライド	論理回路種別	代入文	演算子	内部信号	generic宣言		モジュール化デザイン
mynand2	VHDL-01	組み合わせ	信号代入	論理	―	―		―
myor3	VHDL-02	組み合わせ	信号代入	論理	○	―		―
mynand2xM	VHDL-02	組み合わせ	信号代入	論理	―	○	値渡しなし	―
mynand2_cmp_inext	VHDL-02	組み合わせ	信号代入 （下位Level ）	論理	―	―		○（その1）
mynand2xM_cmp_or3	VHDL-02	組み合わせ	信号代入 （Top/下位Level ）	論理	○	○	値渡しあり ・Top-Level (signal宣言文) ・下位Level (generic map)	○（その2）
mynand2_cmp_fg	VHDL-02	組み合わせ	信号代入 （下位Level ）	論理	―	○	値渡しあり ・Top-Level (for-generate文)	○（その3）
myadderM	VHDL-03	組み合わせ	信号代入	算術(+)	―	―		―
myshift1	VHDL-03	組み合わせ	信号代入	連接(&)	―	―		―
mydec1to2	VHDL-03	組み合わせ	条件付信号代入	関係	―	―		―

バリエーション
なし

先のステップ
で学習

バリエーション
がやや増えた

次のステップ
でさらに学習

さまざまなバリエーション
で学習済

適宜応用する