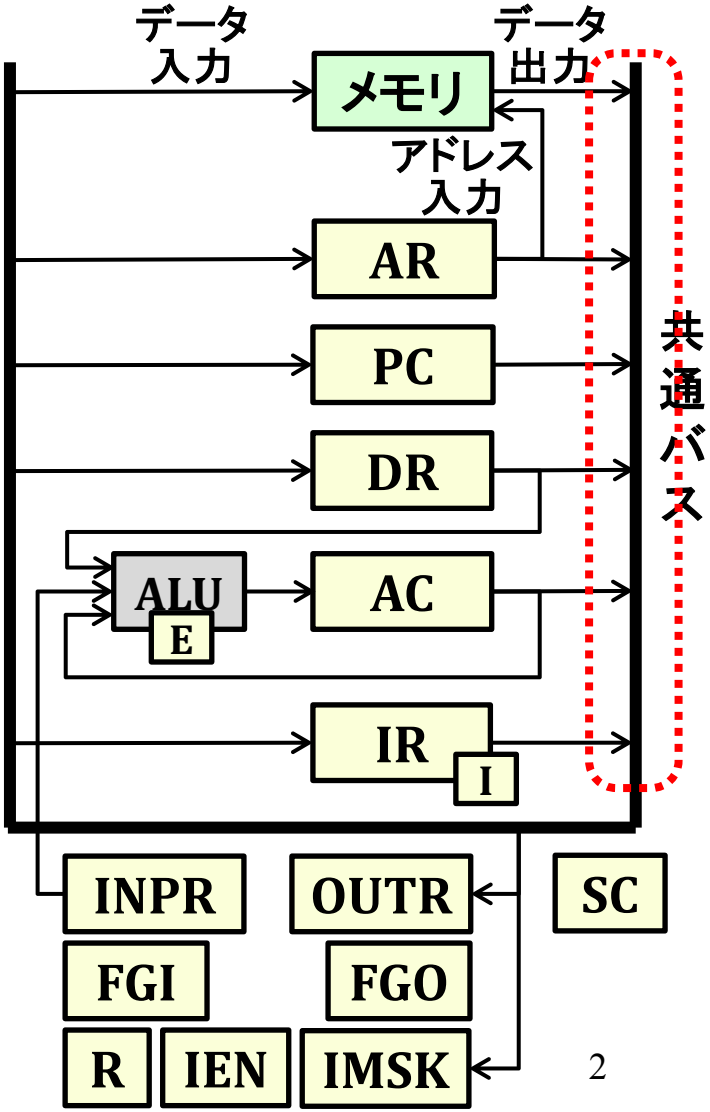
# 計算機論理設計
# 8. バス回路設計

一色 剛
工学院情報通信系
isshiki@ict.e.titech.ac.jp

# 命令実行サイクル

講義資料5：P9 参照

- ■ 計算機構成要素（メモリ、レジスタ、ALU、共通バス）間のデータ転送を順序回路で制御し、以下の動作を実現
- 命令フェッチ：命令読出し
- 命令デコード：計算機の状態遷移の決定
- 命令実行：メモリアクセス、演算、レジスタ更新
- PC更新：次命令のアドレス計算
- ■ データ転送の制約
- メモリアドレス：AR出力が直接接続
- 共通バス出力：共通バスに出力するメモリ（データ出力）, AR, PC, DR, AC, IRから一つ選択し、その信号をバスに転送する
- 共通バス入力：共通バスから入力するメモリ（データ入力）, AR, PC, DR, IRはバス信号を常に受け取ることができる
- AC入力：ACは共通バスからの直接の入力はなく、ALUの出力信号が入力に接続される

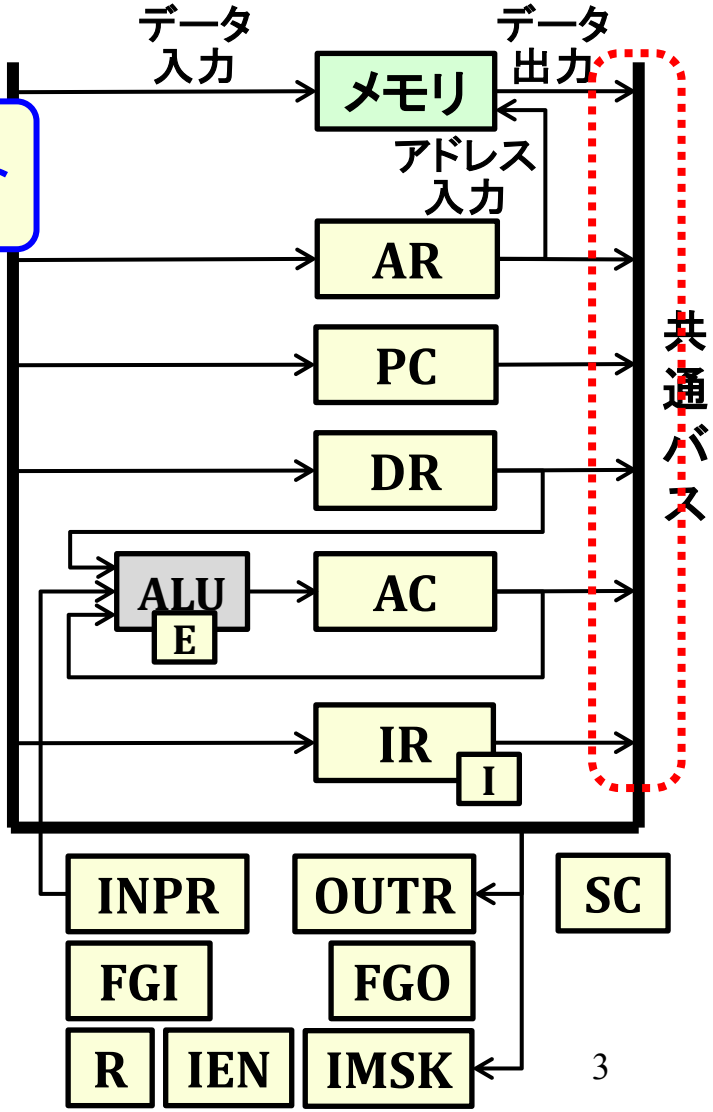


# 計算機ハードウェアアーキテクチャ
## （入出力レジスタ･タイミング生成器含む）

| | |
|---|---|
| メモリ | アドレス : 12 bits、データ : 16 bits |
| AR (11:0) | アドレスレジスタ |
| PC (11:0) | プログラムカウンタ |
| DR (15:0) | データレジスタ |
| AC (15:0) | アキュムレータ |
| IR (15:0) | 命令レジスタ |
| E | 状態レジスタ |
| I | 命令フラグレジスタ |
| SC (2:0) | タイミング生成器 |
| INPR (7:0) | 入力レジスタ |
| OUTR (7:0) | 出力レジスタ |
| FGI | 入力フラグレジスタ |
| FGO | 出力フラグレジスタ |
| IEN | 割込み制御レジスタ |
| IMSK(1:0) | 割込みマスクレジスタ |
| R | 割込み状態レジスタ |

ARとPCの(15:12)ビット出力は0000

共通バス制御に関わる出力



# 命令実行サイクル中のデータ転送動作

**データ転送動作のみを抜粋**

## 割込みサイクル

| | |
|---|---|
| $R \cdot T(1):$ | $M[AR] \leftarrow PC$ |

## メモリ参照命令実行サイクル

| | | |
|---|---|---|
| 間接アドレス | $\overline{D(7)} \cdot I \cdot T(3):$ | $AR \leftarrow M[AR]$ |
| AND | $D(0) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| ADD | $D(1) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| LDA | $D(2) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| STA | $D(3) \cdot T(4):$ | $M[AR] \leftarrow AC$ |
| BUN | $D(4) \cdot T(4):$ | $PC \leftarrow AR$ |
| BSA | $D(5) \cdot T(4):$ | $M[AR] \leftarrow PC$ |
| | $D(5) \cdot T(5):$ | $PC \leftarrow AR$ |
| ISZ | $D(6) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| | $D(6) \cdot T(6):$ | $M[AR] \leftarrow DR$ |

## 命令フェッチサイクル

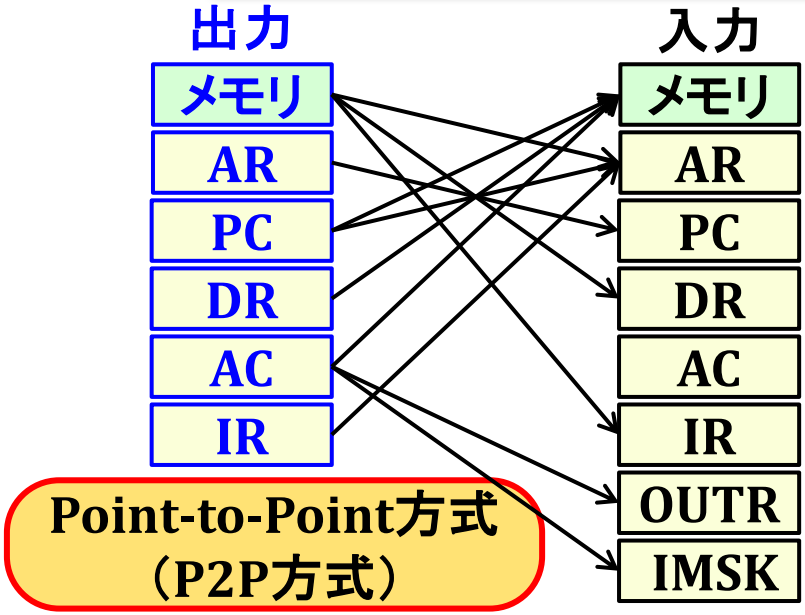
| | |
|---|---|
| $\overline{R} \cdot T(0):$ | $AR \leftarrow PC$ |
| $\overline{R} \cdot T(1):$ | $IR \leftarrow M[AR]$ |
| $\overline{R} \cdot T(2):$ | $AR \leftarrow IR(11:0)$ |

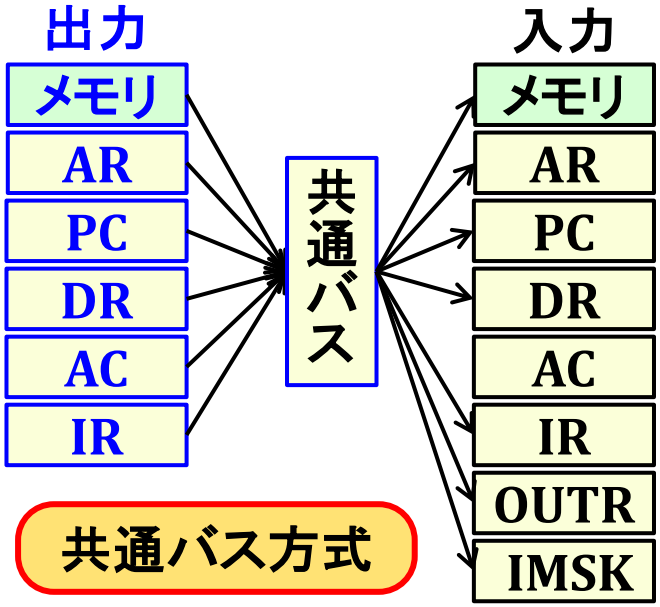
## 入出力命令実行サイクル

| | $p = D(7) \cdot I \cdot T(3)$ | |
|---|---|---|
| OUT | $p \cdot OP2(10):$ | $OUTR \leftarrow AC(7:0)$ |
| IMK | $p \cdot OP2(5):$ | $IMSK \leftarrow AC(1:0)$ |

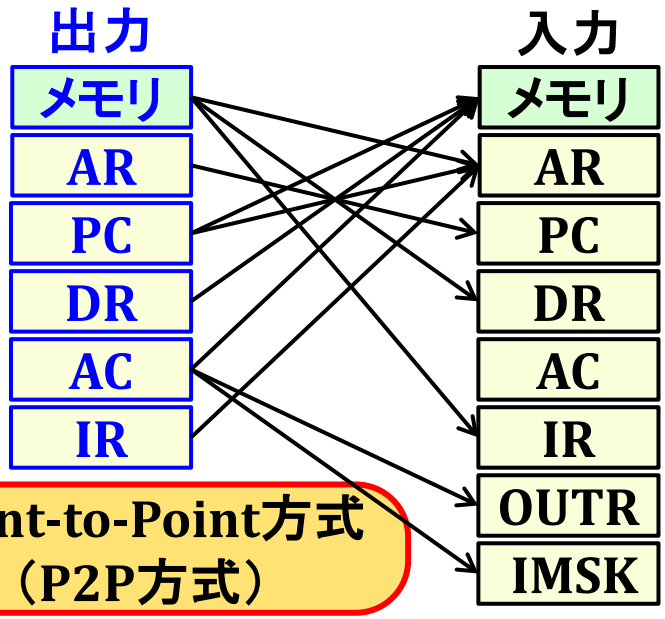
# データ転送の実現方法

| 出力＼入力 | メモリ | AR | PC | DR | AC | IR | OUTR | IMSK |
|---|---|---|---|---|---|---|---|---|
| メモリ | | ○ | | ○ | | ○ | | |
| AR | | | ○ | | | | | |
| PC | ○ | ○ | | | | | | |
| DR | ○ | | | | | | | |
| AC | ○ | | | | | | ○ | ○ |
| IR | | ○ | | | | | | |

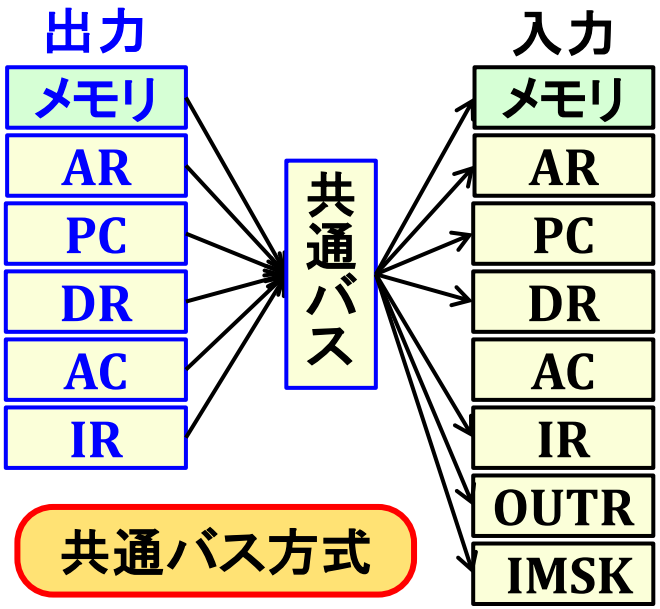


# データ転送の実現方法




Point-to-Point方式
（P2P方式）




共通バス方式

- 設計が複雑：複雑なデータの流れ、個別のデータ転送の制御論理設計が必要
- 高速転送：複数の同時データ転送可能、論理遅延小

- 設計が単純：単純なデータの流れ、共通バス制御とレジスタロード制御の論理設計だけで済む
- 転送速度が制限：複数の同時データ転送不可、論理遅延大

両者の具体的比較は後述

# 共通バス出力選択回路（MUX-Tree構成）

## 共通バス出力選択動作

- 共通バスに出力するメモリ（データ出力）, AR, PC, DR, AC, IRから一つ選択し、バス BD(15:0)に転送

## バス出力ID

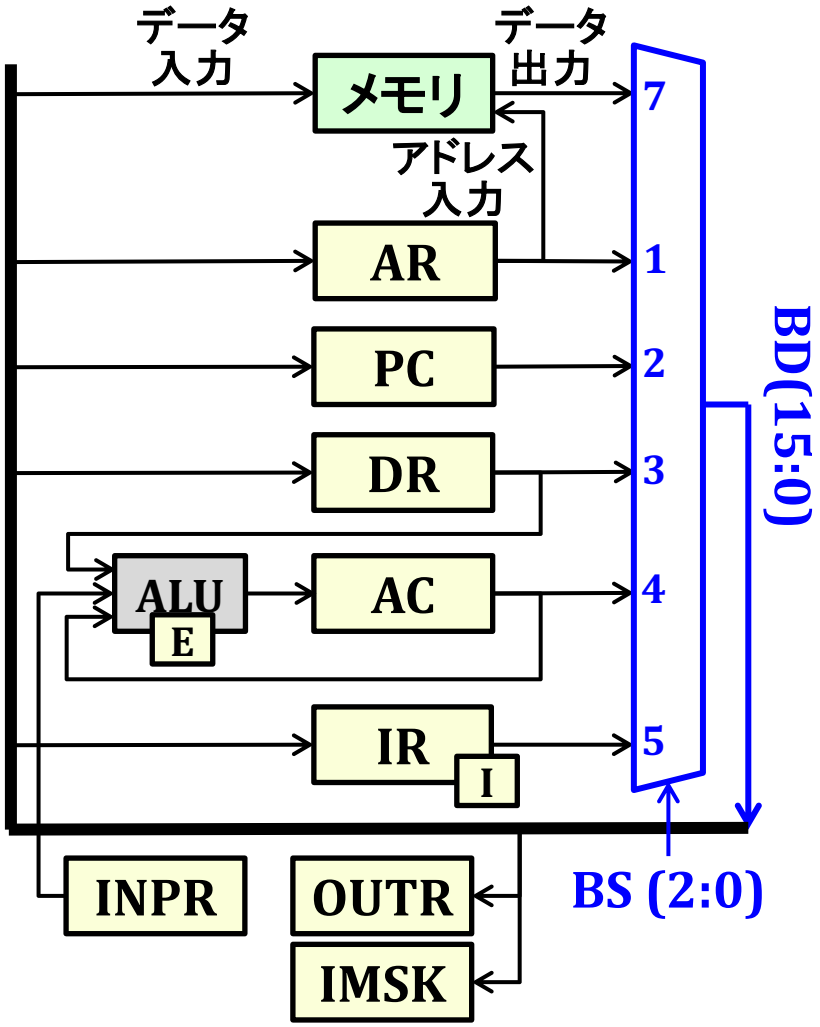
- 各バス出力に固有IDを割当てる（割当て方法は任意）

## バス制御信号 : BS(2:0)

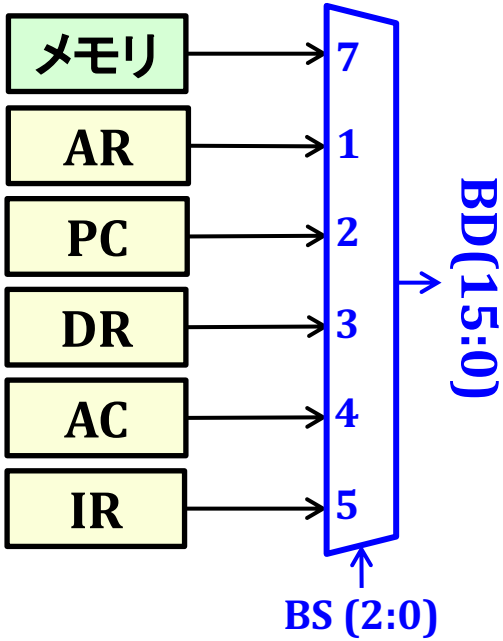
- 選択するバス出力IDを設定

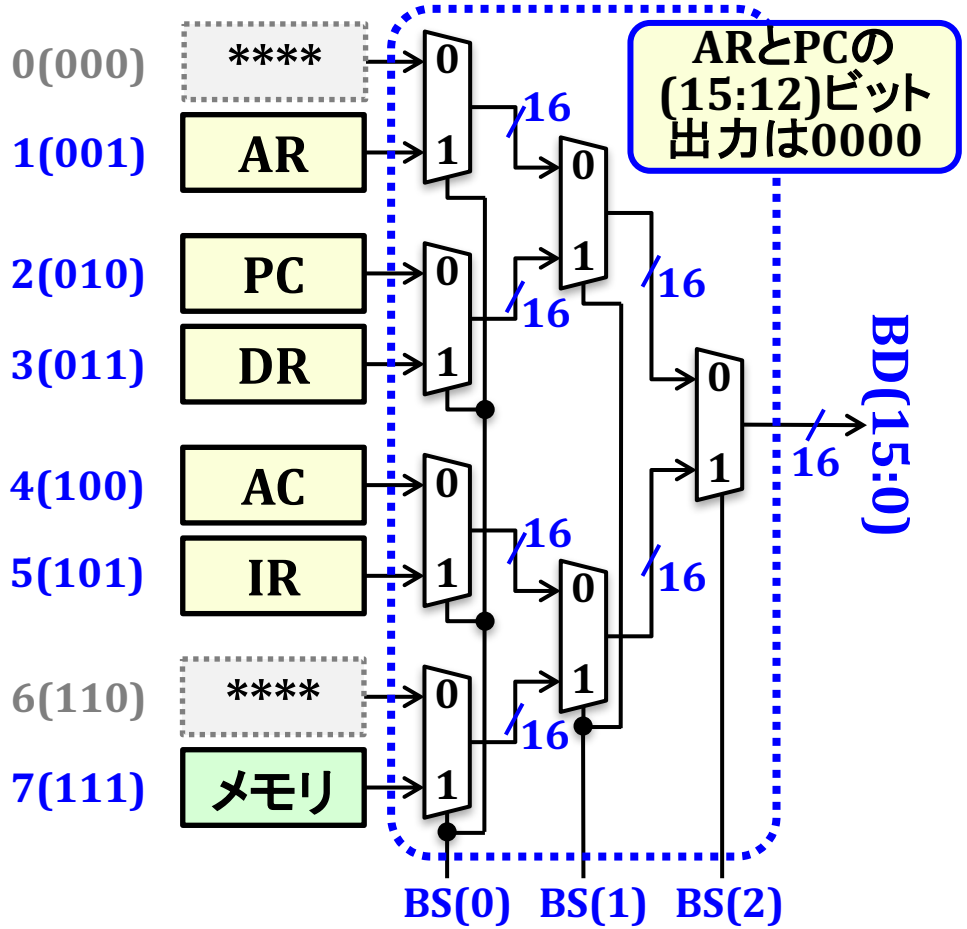
## N入力マルチプレクサ回路 :

- 2入力マルチプレクサの木構造で構成（MUX-Tree）→ バス制御信号 BS(2:0)の値に対応するバス出力を選択

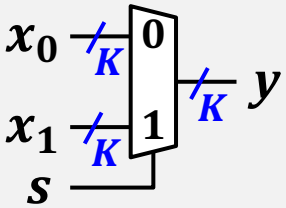


# MUX-Tree型N入力マルチプレクサ




## $K$ビット2入力マルチプレクサ

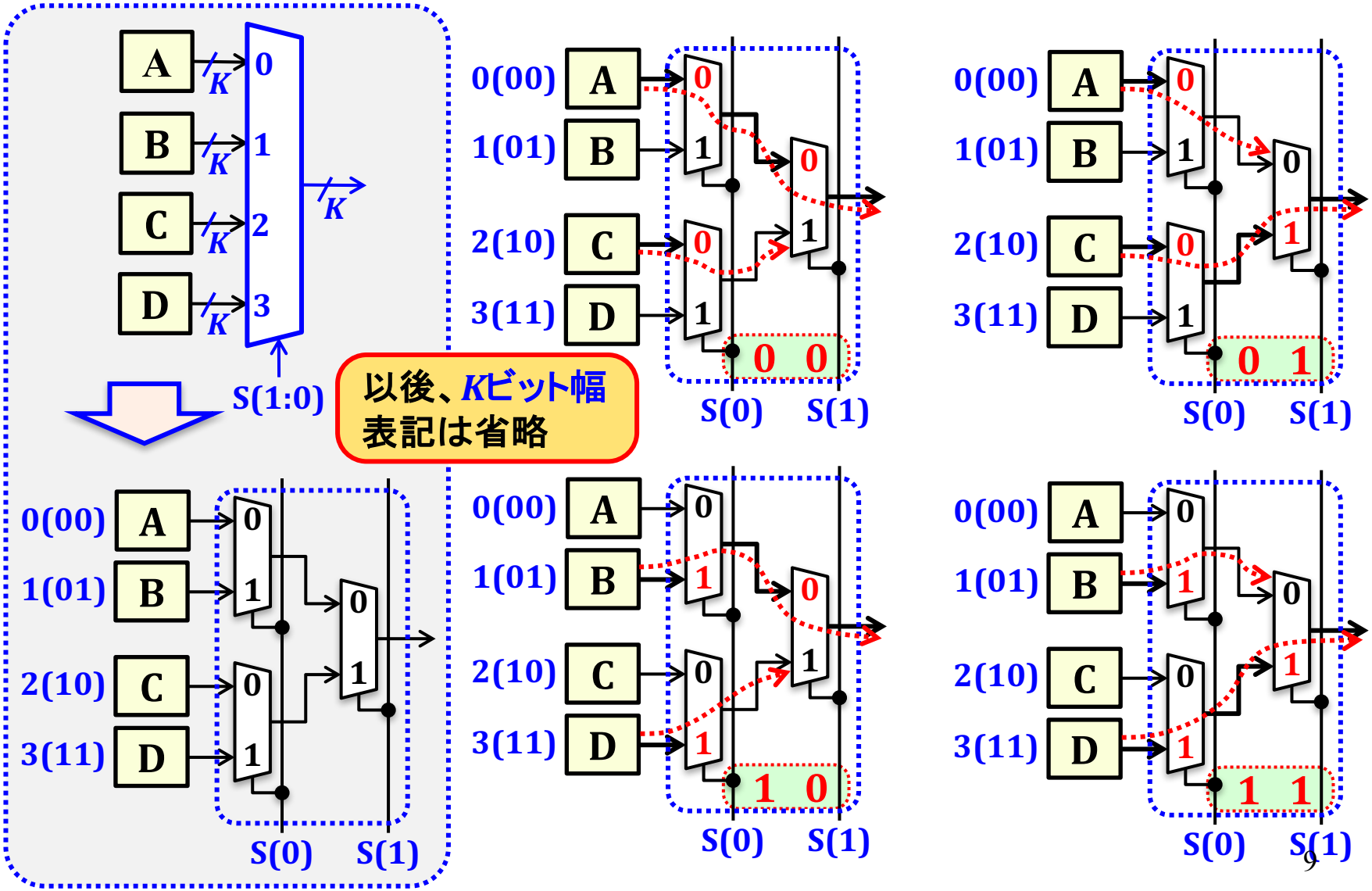



$$y = \begin{cases} x_0 & (s = 0) \\ x_1 & (s = 1) \end{cases}$$

$$y(k) = \bar{s} \cdot x_0(k) + s \cdot x_1(k)$$
$$(k = 0, 1, \cdots, K-1)$$

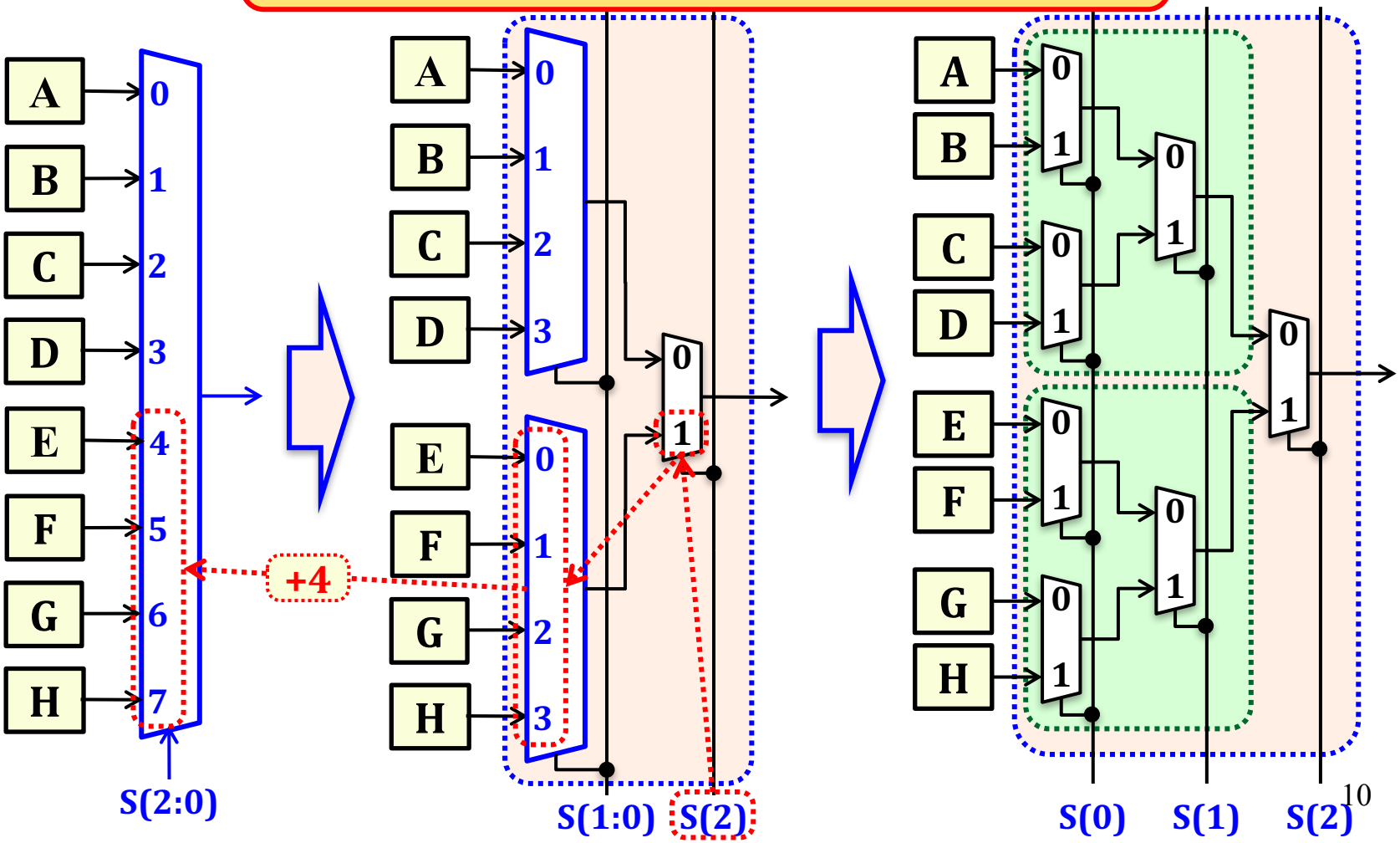
N入力マルチプレクサ：深さ$M$の
2分木構造によって$2^M$入力を選択

# $2^2$入力MUX-Tree回路構成



# $2^3$入力MUX-Tree回路構成

$2^M$入力MUX-Tree構成法：
2つの $2^{M-1}$ 入力Mux-Treeを2入力MUXで接続



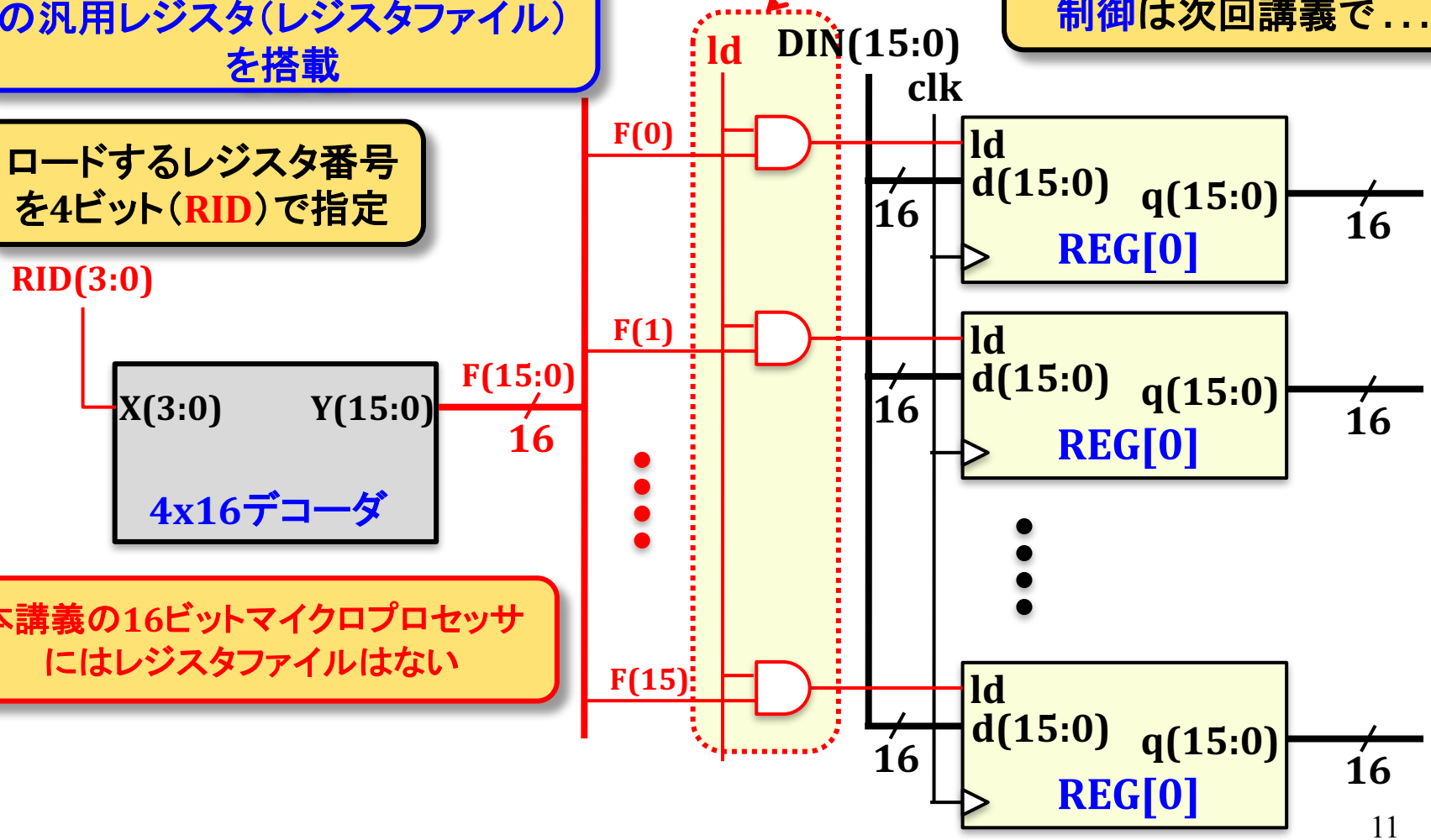
講義資料7 P11：訂正

# レジスタファイルの書込み制御

標準マイクロプロセッサでは多数の汎用レジスタ（レジスタファイル）を搭載

レジスタファイルの読出し制御は次回講義で...

ロードするレジスタ番号を4ビット（RID）で指定

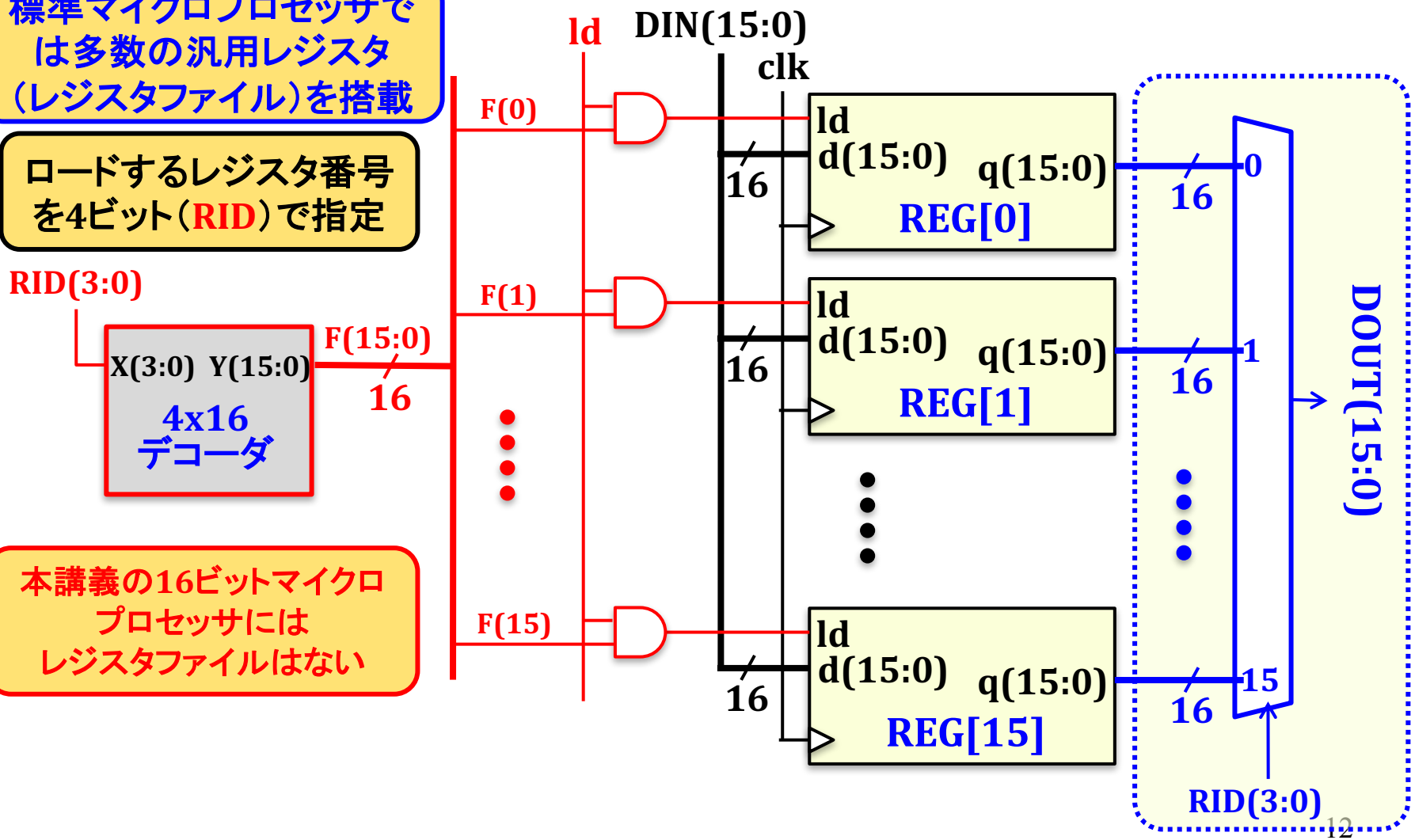



本講義の16ビットマイクロプロセッサにはレジスタファイルはない

# レジスタファイルの読出し制御

標準マイクロプロセッサでは多数の汎用レジスタ（レジスタファイル）を搭載

ロードするレジスタ番号を4ビット（RID）で指定




本講義の16ビットマイクロプロセッサにはレジスタファイルはない

# 共通バス出力制御論理回路

## 共通バス出力選択動作

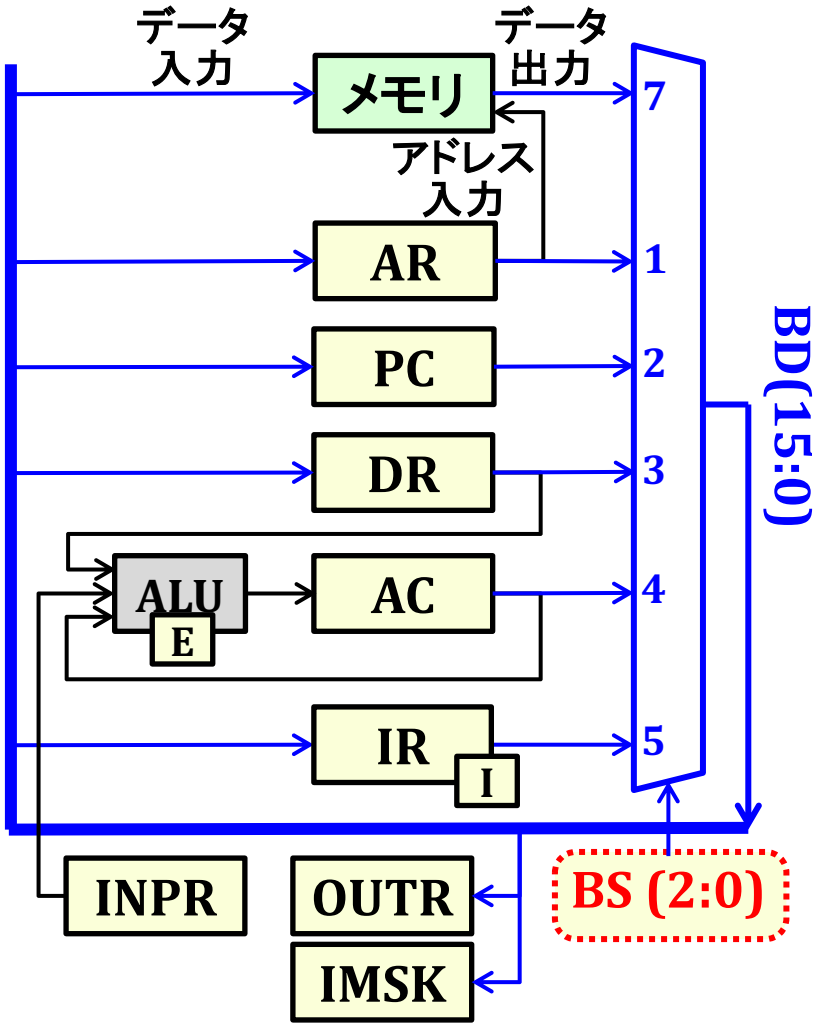
- 共通バスに出力するメモリ(データ出力), AR, PC, DR, AC, IRから一つ選択し、バス BD(15:0)に転送

## バス制御信号 : BS(2:0)

- 選択するバス出力IDを設定

## バス制御論理回路 :

- 命令実行サイクル動作記述から、共通バスを介したデータ転送を抽出し、該当するバス出力選択を実現する



# 共通バスデータ転送動作

**共通バスデータ転送動作のみを抜粋**

### 割込みサイクル

| | |
|---|---|
| $R \cdot T(1):$ | $M[AR] \leftarrow PC$ |

### メモリ参照命令実行サイクル

| | | |
|---|---|---|
| 間接アドレス | $\overline{D(7)} \cdot I \cdot T(3):$ | $AR \leftarrow M[AR]$ |
| AND | $D(0) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| ADD | $D(1) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| LDA | $D(2) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| STA | $D(3) \cdot T(4):$ | $M[AR] \leftarrow AC$ |
| BUN | $D(4) \cdot T(4):$ | $PC \leftarrow AR$ |
| BSA | $D(5) \cdot T(4):$ | $M[AR] \leftarrow PC$ |
| | $D(5) \cdot T(5):$ | $PC \leftarrow AR$ |
| ISZ | $D(6) \cdot T(4):$ | $DR \leftarrow M[AR]$ |
| | $D(6) \cdot T(6):$ | $M[AR] \leftarrow DR$ |

### 命令フェッチサイクル

| | |
|---|---|
| $\overline{R} \cdot T(0):$ | $AR \leftarrow PC$ |
| $\overline{R} \cdot T(1):$ | $IR \leftarrow M[AR]$ |
| $\overline{R} \cdot T(2):$ | $AR \leftarrow IR(11:0)$ |

### 入出力命令実行サイクル

| | $p = D(7) \cdot I \cdot T(3)$ | |
|---|---|---|
| OUT | $p \cdot OP2(10):$ | $OUTR \leftarrow AC(7:0)$ |
| IMK | $p \cdot OP2(5):$ | $IMSK \leftarrow AC(1:0)$ |

共通バスを介したデータ転送の実行条件は全て排他的
→ 同時に複数の実行条件が1になることはない

# バス入力転送動作（レジスタ・メモリ制御論理設計）

レジスタ・ロード制御論理：講義資料7 P22～P25参照

## メモリ書込み制御：

$$write_MEM = R \cdot T(1) + (D(3) + D(5)) \cdot T(4) + D(6) \cdot T(6)$$

| | | |
|---|---|---|
| 割込みサイクル | $R \cdot T(1)$： | $M[AR] \leftarrow PC$ |
| STA | $D(3) \cdot T(4)$： | $M[AR] \leftarrow AC$ |
| BSA | $D(5) \cdot T(4)$： | $M[AR] \leftarrow PC$ |
| ISZ | $D(6) \cdot T(6)$： | $M[AR] \leftarrow DR$ |

## IR入力制御：$IR_LD = \overline{R} \cdot T(1)$

| | | |
|---|---|---|
| 命令フェッチ | $\overline{R} \cdot T(1)$： | $IR \leftarrow M[AR]$ |

## DR入力制御：

$$DR_LD = (D(0) + D(1) + D(2) + D(6)) \cdot T(4)$$

| | | |
|---|---|---|
| AND | $D(0) \cdot T(4)$： | $DR \leftarrow M[AR]$ |
| ADD | $D(1) \cdot T(4)$： | $DR \leftarrow M[AR]$ |
| LDA | $D(2) \cdot T(4)$： | $DR \leftarrow M[AR]$ |
| ISZ | $D(6) \cdot T(4)$： | $DR \leftarrow M[AR]$ |

## PC入力制御：

$$PC_LD = D(4) \cdot T(4) + D(5) \cdot T(5)$$

| | | |
|---|---|---|
| BUN | $D(4) \cdot T(4)$： | $PC \leftarrow AR$ |
| BSA | $D(5) \cdot T(5)$： | $PC \leftarrow AR$ |

## AR入力制御：

$$AR_LD = \overline{R} \cdot (T(0) + T(2)) + \overline{D(7)} \cdot I \cdot T(3)$$

| | | |
|---|---|---|
| 命令フェッチ | $\overline{R} \cdot T(0)$： | $AR \leftarrow PC$ |
| | $\overline{R} \cdot T(2)$： | $AR \leftarrow IR(11:0)$ |
| 間接アドレス | $\overline{D(7)} \cdot I \cdot T(3)$： | $AR \leftarrow M[AR]$ |

## OUTR入力制御：$OUTR_LD = p \cdot OP2(10)$

| | | |
|---|---|---|
| OUT | $p \cdot OP2(10)$： | $OUTR \leftarrow AC(7:0)$ |

## IMSK入力制御：$IMSK_LD = p \cdot OP2(5)$

| | | |
|---|---|---|
| IMK | $p \cdot OP2(5)$： | $IMSK \leftarrow AC(1:0)$ |

# バス出力転送動作（バス制御論理設計）

## メモリ出力バス転送：

$$bus_MEM = \overline{R} \cdot T(1) + \overline{D(7)} \cdot I \cdot T(3) + (D(0) + D(1) + D(2) + D(6)) \cdot T(4)$$

| | | |
|---|---|---|
| 命令フェッチ | $\overline{R} \cdot T(1)$： | $IR \leftarrow M[AR]$ |
| 間接アドレス | $\overline{D(7)} \cdot I \cdot T(3)$： | $AR \leftarrow M[AR]$ |
| AND | $D(0) \cdot T(4)$： | $DR \leftarrow M[AR]$ |
| ADD | $D(1) \cdot T(4)$： | $DR \leftarrow M[AR]$ |
| LDA | $D(2) \cdot T(4)$： | $DR \leftarrow M[AR]$ |
| ISZ | $D(6) \cdot T(4)$： | $DR \leftarrow M[AR]$ |

## IR出力バス転送：$bus_IR = \overline{R} \cdot T(2)$

| | | |
|---|---|---|
| 命令フェッチ | $\overline{R} \cdot T(2)$： | $AR \leftarrow IR(11:0)$ |

## DR出力バス転送：$bus_DR = D(6) \cdot T(6)$

| | | |
|---|---|---|
| ISZ | $D(6) \cdot T(6)$： | $M[AR] \leftarrow DR$ |

## PC出力バス転送：

$$bus_PC = R \cdot T(1) + \overline{R} \cdot T(0) + D(5) \cdot T(4)$$

| | | |
|---|---|---|
| 割込みサイクル | $R \cdot T(1)$： | $M[AR] \leftarrow PC$ |
| 命令フェッチ | $\overline{R} \cdot T(0)$： | $AR \leftarrow PC$ |
| BSA | $D(5) \cdot T(4)$： | $M[AR] \leftarrow PC$ |

## AR出力バス転送：

$$bus_AR = D(4) \cdot T(4) + D(5) \cdot T(5)$$

| | | |
|---|---|---|
| BUN | $D(4) \cdot T(4)$： | $PC \leftarrow AR$ |
| BSA | $D(5) \cdot T(5)$： | $PC \leftarrow AR$ |

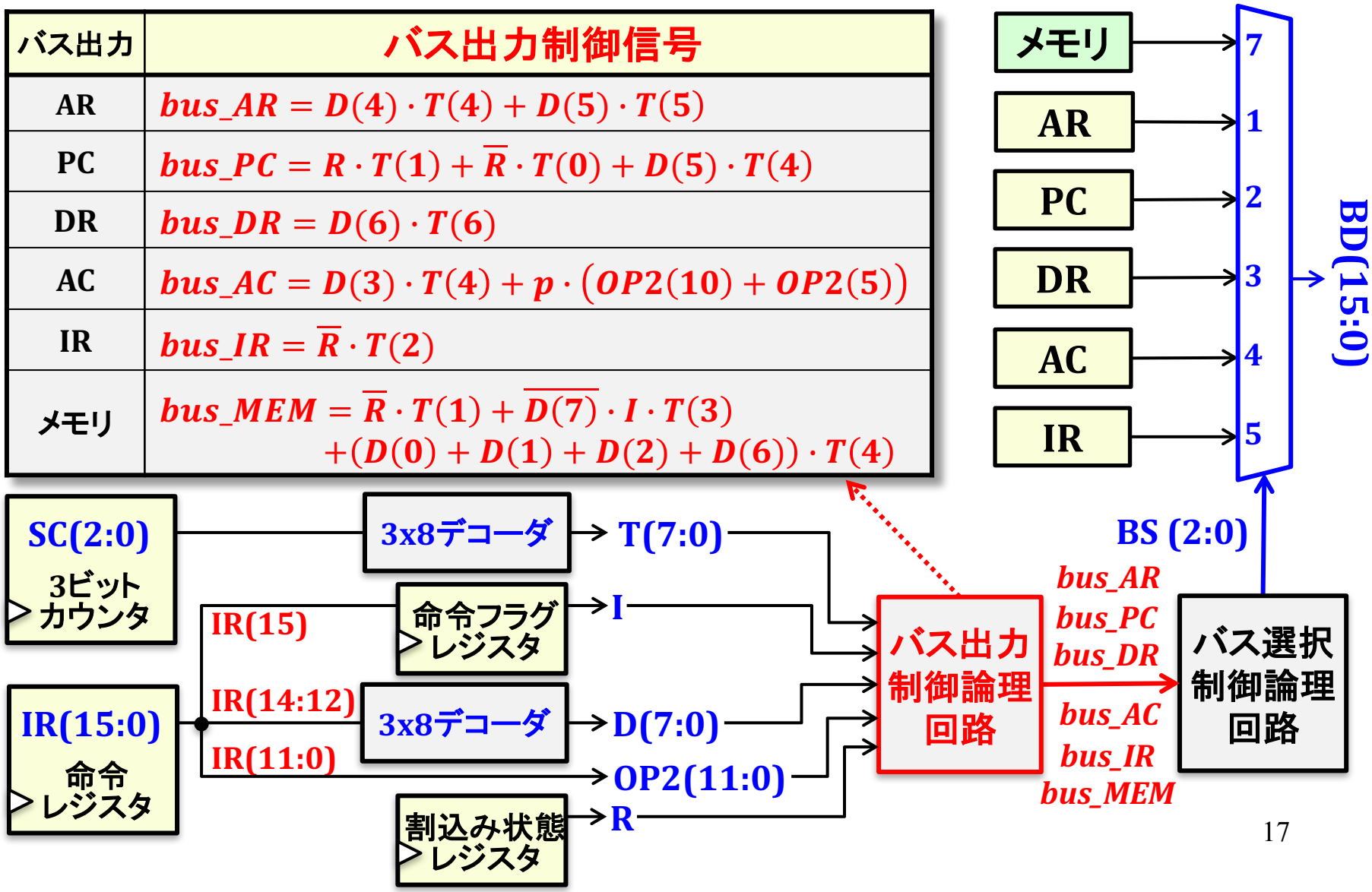
## AC出力バス転送：

$$bus_AC = D(3) \cdot T(4) + p \cdot (OP2(10) + OP2(5))$$

| | | |
|---|---|---|
| STA | $D(3) \cdot T(4)$： | $M[AR] \leftarrow AC$ |
| OUT | $p \cdot OP2(10)$： | $OUTR \leftarrow AC(7:0)$ |
| IMK | $p \cdot OP2(5)$： | $IMSK \leftarrow AC(1:0)$ |

# バス出力制御論理設計

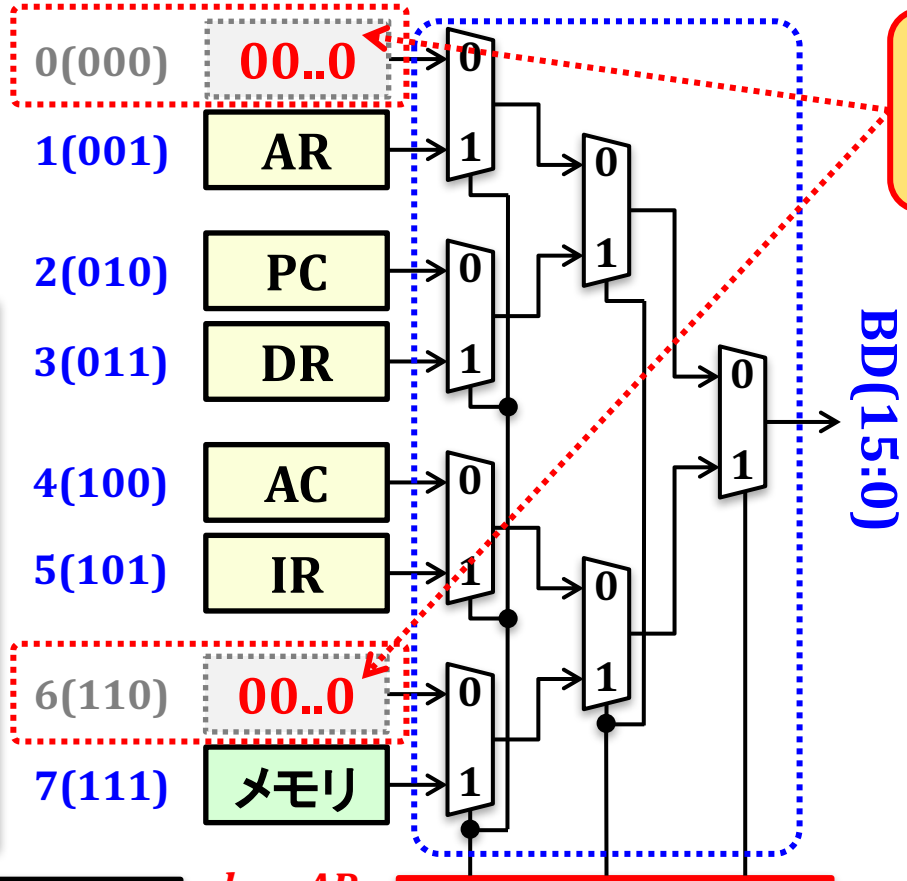
| バス出力 | バス出力制御信号 |
|---|---|
| AR | $bus_AR = D(4) \cdot T(4) + D(5) \cdot T(5)$ |
| PC | $bus_PC = R \cdot T(1) + \overline{R} \cdot T(0) + D(5) \cdot T(4)$ |
| DR | $bus_DR = D(6) \cdot T(6)$ |
| AC | $bus_AC = D(3) \cdot T(4) + p \cdot \left(OP2(10) + OP2(5)\right)$ |
| IR | $bus_IR = \overline{R} \cdot T(2)$ |
| メモリ | $bus_MEM = \overline{R} \cdot T(1) + \overline{D(7)} \cdot I \cdot T(3) + (D(0) + D(1) + D(2) + D(6)) \cdot T(4)$ |



# バス選択制御論理設計

ARとPCの(15:12)ビット出力は0000

| BC(5:0) | バス出力論理信号 |
|---|---|
| BC(0) = | bus_AR |
| BC(1) = | bus_PC |
| BC(2) = | bus_DR |
| BC(3) = | bus_AC |
| BC(4) = | bus_IR |
| BC(5) = | bus_MEM |




(方法1)未接続出力に任意の定数(例えば「00..0」)を接続する

| 出力 | BC(5:0) | BS(2:0) |
|---|---|---|
| AR | 000001 | 001 |
| PC | 000010 | 010 |
| DR | 000100 | 011 |
| AC | 001000 | 100 |
| IR | 010000 | 101 |
| メモリ | 100000 | 111 |
| なし | 上記以外 | *** |

共通バスを介したデータ転送の実行条件は全て排他的

T(7:0)
I
D(7:0)
OP2(11:0)
R

バス出力制御論理回路

bus_AR
bus_PC
bus_DR
bus_AC
bus_IR
bus_MEM

BS(0) BS(1) BS(2)

バス選択制御論理回路

# バス選択制御論理設計

| | BC(5:0) | BS(2:0) |
|---|---|---|
| $BC(0)$ = *bus_AR* | 000001 | 001 |
| $BC(1)$ = *bus_PC* | 000010 | 010 |
| $BC(2)$ = *bus_DR* | 000100 | 011 |
| $BC(3)$ = *bus_AC* | 001000 | 100 |
| $BC(4)$ = *bus_IR* | 010000 | 101 |
| $BC(5)$ = *bus_MEM* | 100000 | 111 |
| | 上記以外 | * * * |

$BS(0) = bus_AR + bus_DR + bus_IR + bus_MEM$

| BC(5:3) \ BC(2:0) | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|---|---|---|---|---|---|---|---|---|
| 000 | * | 1 | * | 0 | * | * | * | 1 |
| 001 | 0 | * | * | * | * | * | * | * |
| 011 | * | * | * | * | * | * | * | * |
| 010 | 1 | * | * | * | * | * | * | * |
| 110 | * | * | * | * | * | * | * | * |
| 111 | * | * | * | * | * | * | * | * |
| 101 | * | * | * | * | * | * | * | * |
| 100 | 1 | * | * | * | * | * | * | * |

$BS(1) = bus_PC + bus_DR + bus_MEM$

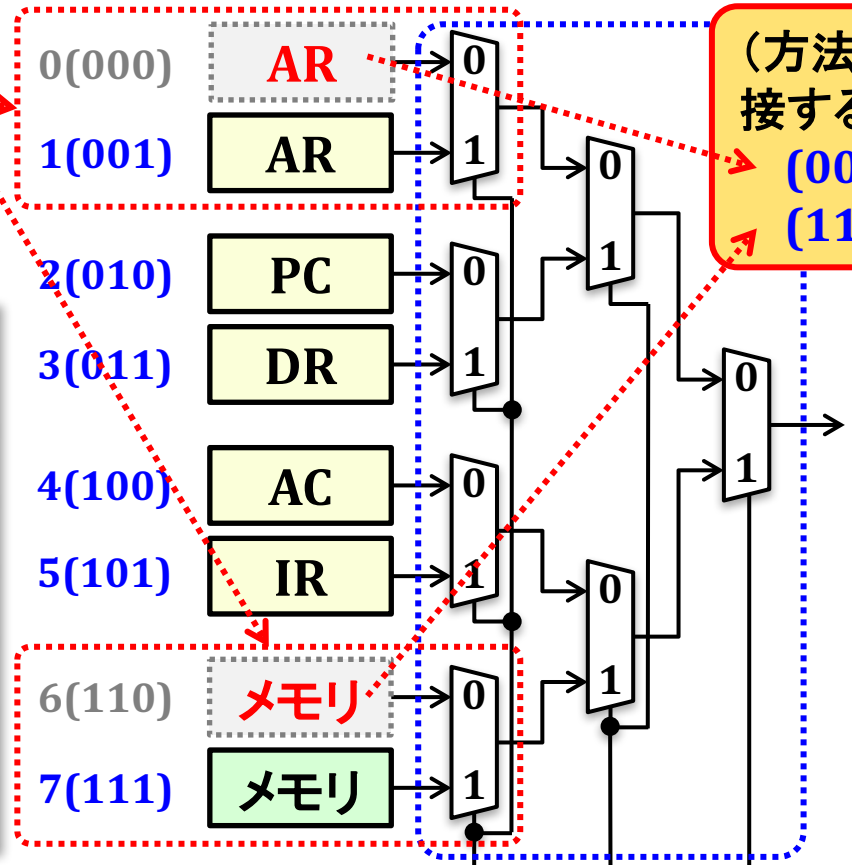
| BC(5:3) \ BC(2:0) | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|---|---|---|---|---|---|---|---|---|
| 000 | * | 0 | * | 1 | * | * | * | 1 |
| 001 | 0 | * | * | * | * | * | * | * |
| 011 | * | * | * | * | * | * | * | * |
| 010 | 0 | * | * | * | * | * | * | * |
| 110 | * | * | * | * | * | * | * | * |
| 111 | * | * | * | * | * | * | * | * |
| 101 | * | * | * | * | * | * | * | * |
| 100 | 1 | * | * | * | * | * | * | * |

$BS(2) = bus_AC + bus_IR + bus_MEM$

| BC(5:3) \ BC(2:0) | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|---|---|---|---|---|---|---|---|---|
| 000 | * | 0 | * | 0 | * | * | * | 0 |
| 001 | 1 | * | * | * | * | * | * | * |
| 011 | * | * | * | * | * | * | * | * |
| 010 | 1 | * | * | * | * | * | * | * |
| 110 | * | * | * | * | * | * | * | * |
| 111 | * | * | * | * | * | * | * | * |
| 101 | * | * | * | * | * | * | * | * |
| 100 | 1 | * | * | * | * | * | * | * |

# バス選択制御論理設計

2入力MUXが不要になる




（方法2）未接続出力IDに隣接する出力を接続する

(000) + (001) → (00*)

(110) + (111) → (11*)

| BC(5:0) | バス出力論理信号 |
|---|---|
| BC(0) = | bus_AR |
| BC(1) = | bus_PC |
| BC(2) = | bus_DR |
| BC(3) = | bus_AC |
| BC(4) = | bus_IR |
| BC(5) = | bus_MEM |

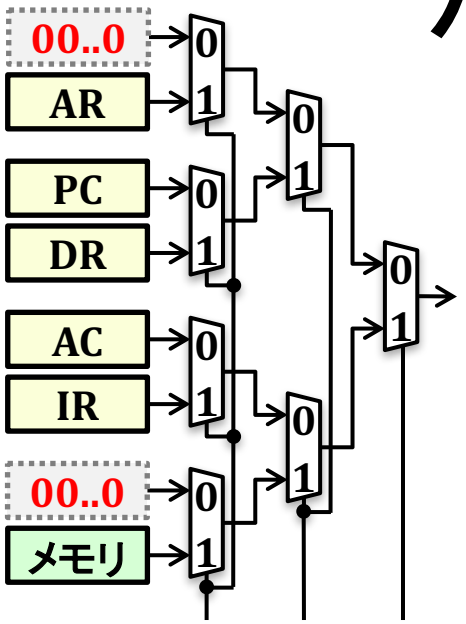
| 出力 | BC(5:0) | BS(2:0) |
|---|---|---|
| AR | 000001 | 00 * |
| PC | 000010 | 010 |
| DR | 000100 | 011 |
| AC | 001000 | 100 |
| IR | 010000 | 101 |
| メモリ | 100000 | 11 * |
| なし | 上記以外 | * * * |

T(7:0)
I
D(7:0)
OP2(11:0)
R

→ バス出力制御論理回路 → bus_AR, bus_PC, bus_DR, bus_AC, bus_IR, bus_MEM → BS(0) BS(1) BS(2) バス選択制御論理回路

共通バスを介したデータ転送の実行条件は全て排他的

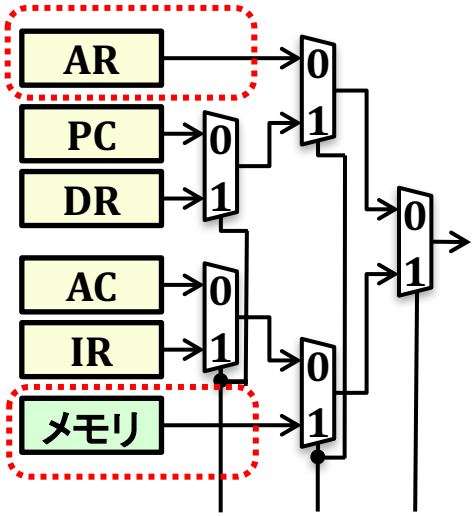
# バス選択制御論理設計




| BC(5:0) | BS(2:0) |
|---|---|
| 000001 | 001 |
| 000010 | 010 |
| 000100 | 011 |
| 001000 | 100 |
| 010000 | 101 |
| 100000 | 111 |
| 上記以外 | * * * |

$BS(0) = bus_AR + bus_DR + bus_IR + bus_MEM$

| BC(5:3) \ BC(2:0) | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|---|---|---|---|---|---|---|---|---|
| 000 | * | 1 | * | 0 | * | * | * | 1 |
| 001 | 0 | * | * | * | * | * | * | * |
| 011 | * | * | * | * | * | * | * | * |
| 010 | 1 | * | * | * | * | * | * | * |
| 110 | * | * | * | * | * | * | * | * |
| 111 | * | * | * | * | * | * | * | * |
| 101 | * | * | * | * | * | * | * | * |
| 100 | 1 | * | * | * | * | * | * | * |



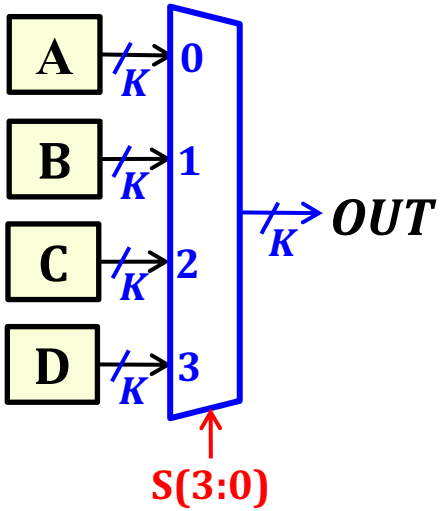

| BC(5:0) | BS(2:0) |
|---|---|
| 000001 | 00 * |
| 000010 | 010 |
| 000100 | 011 |
| 001000 | 100 |
| 010000 | 101 |
| 100000 | 11 * |
| 上記以外 | * * * |

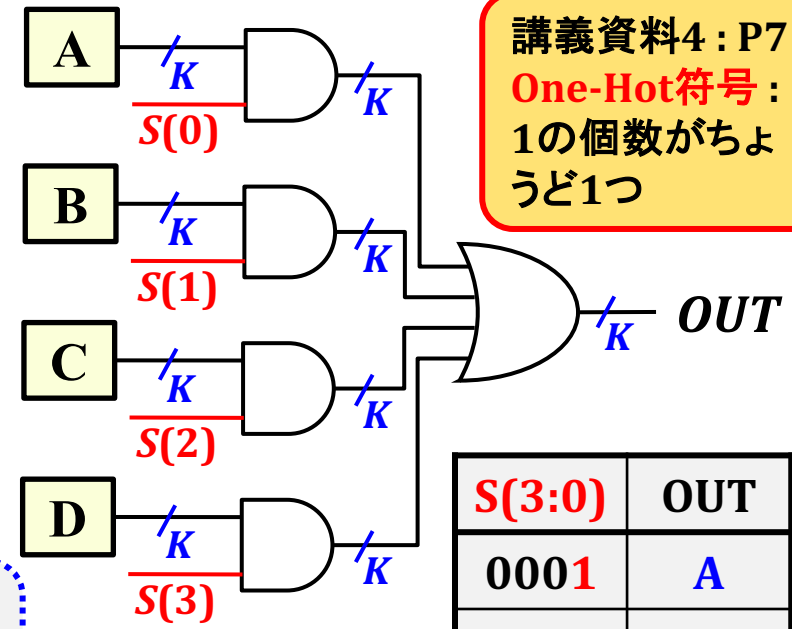
$BS(0) = bus_DR + bus_IR$

| BC(5:3) \ BC(2:0) | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|---|---|---|---|---|---|---|---|---|
| 000 | * | * | * | 0 | * | * | * | 1 |
| 001 | 0 | * | * | * | * | * | * | * |
| 011 | * | * | * | * | * | * | * | * |
| 010 | 1 | * | * | * | * | * | * | * |
| 110 | * | * | * | * | * | * | * | * |
| 111 | * | * | * | * | * | * | * | * |
| 101 | * | * | * | * | * | * | * | * |
| 100 | * | * | * | * | * | * | * | * |

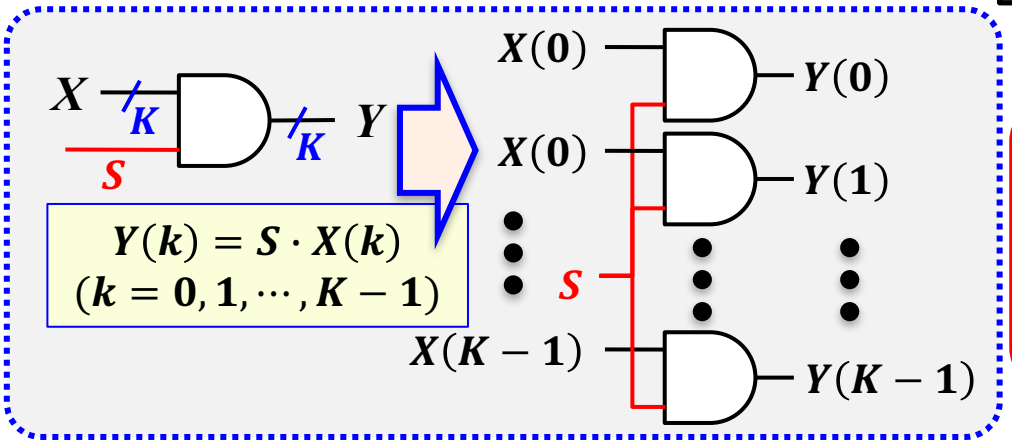
# ANDマスク型N入力マルチプレクサ

**N入力の選択信号がNビットOne-Hot符号の場合：**




講義資料4：P7
One-Hot符号：
1の個数がちょうど1つ

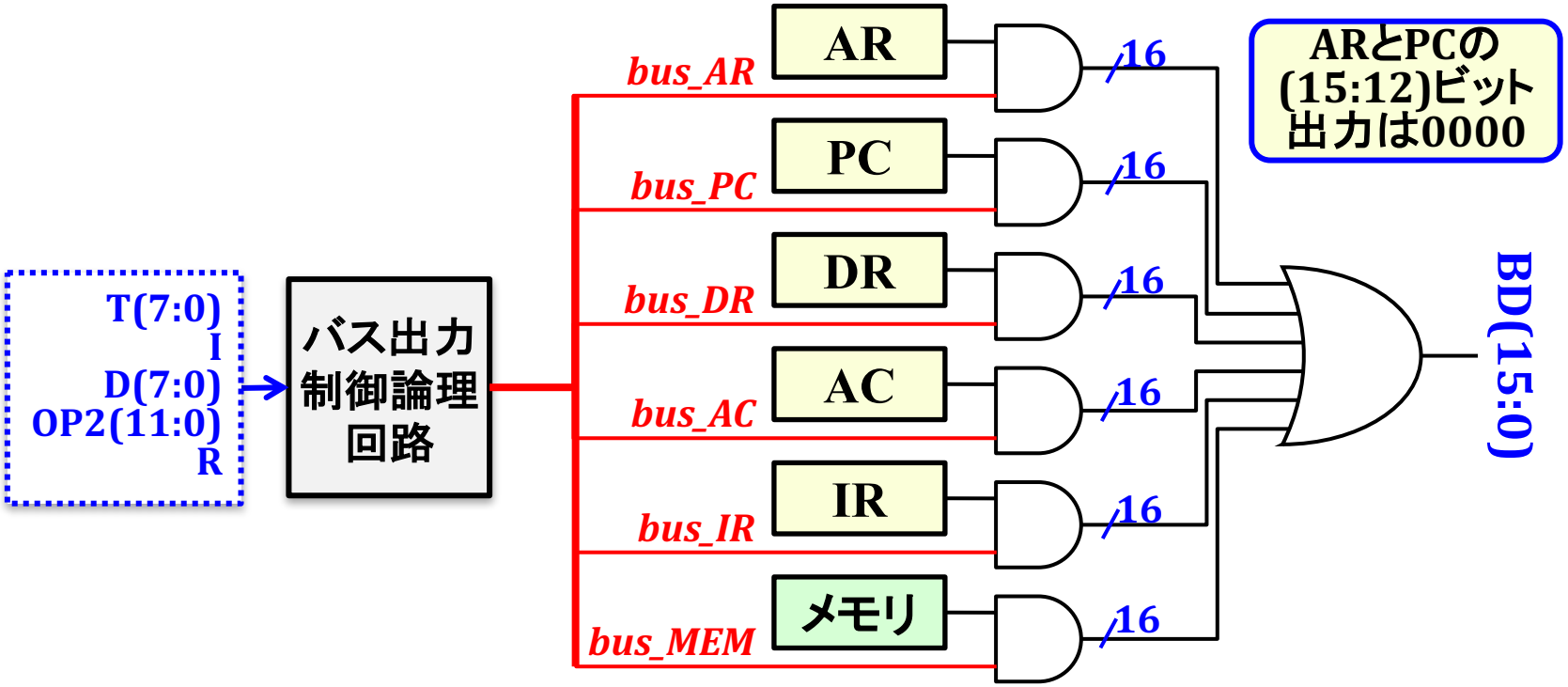



$$Y(k) = S \cdot X(k)$$
$$(k = 0, 1, \cdots, K-1)$$

選択信号S(3:0)は同時に2つ以上が「1」になってはならない

| S(3:0) | OUT |
|---|---|
| 0001 | A |
| 0010 | B |
| 0100 | C |
| 1000 | D |
| 0000 | 00..0 |

# バス制御論理設計（ANDマスク型）



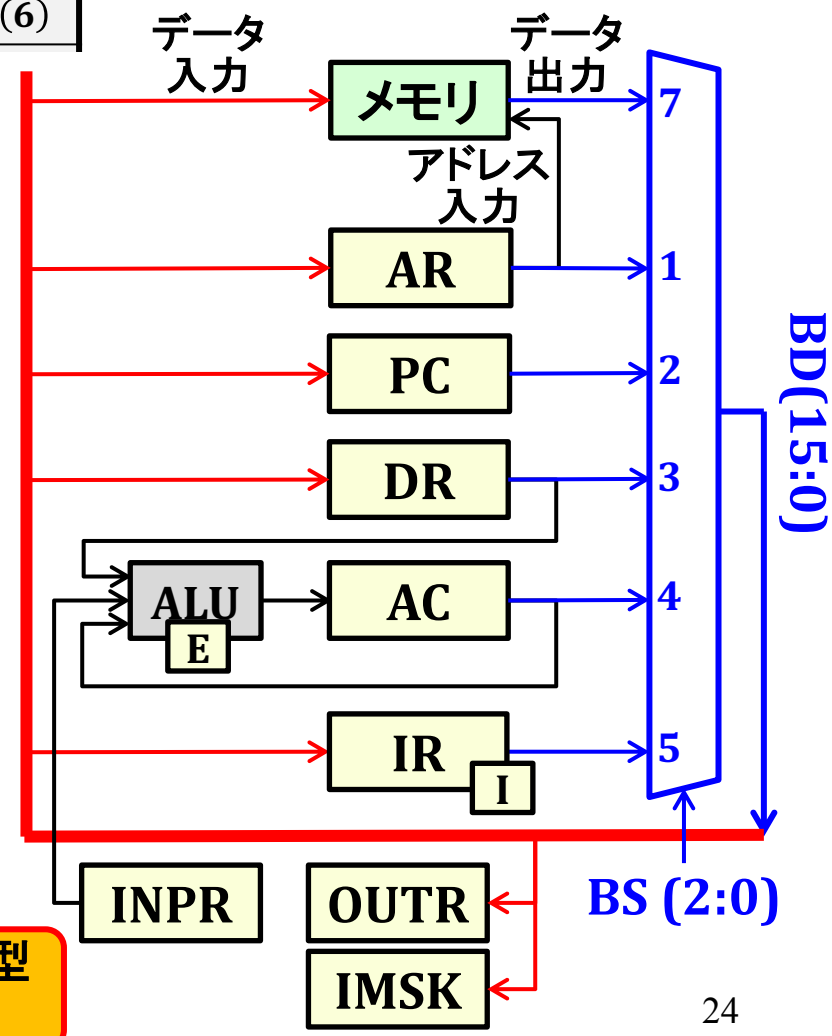

2段目のバス選択制御回路（BS(2:0)の生成）が不要
→ バス出力制御信号（bus_XXX）が互いに排他的である性質を利用

# 共通バスデータ転送制御論理回路

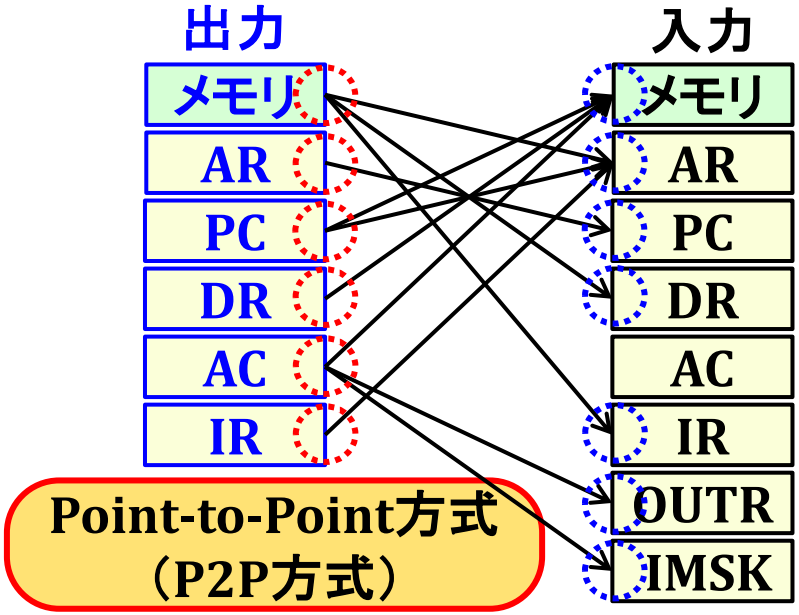
| 入力制御信号 |
|---|
| $write_MEM = R \cdot T(1) + (D(3) + D(5)) \cdot T(4) + D(6) \cdot T(6)$ |
| $IR_LD = \overline{R} \cdot T(1)$ |
| $DR_LD = (D(0) + D(1) + D(2) + D(6)) \cdot T(4)$ |
| $PC_LD = D(4) \cdot T(4) + D(5) \cdot T(5)$ |
| $AR_LD = \overline{R} \cdot \left(T(0) + T(2)\right) + \overline{D(7)} \cdot I \cdot T(3)$ |
| $OUTR_LD = p \cdot OP2(10)$ |
| $IMSK_LD = p \cdot OP2(5)$ |

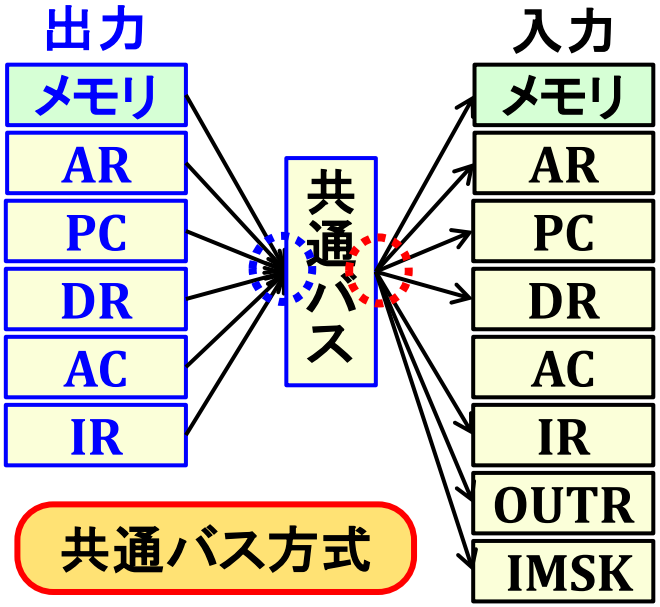
| バス出力制御信号 |
|---|
| $bus_AR = D(4) \cdot T(4) + D(5) \cdot T(5)$ |
| $bus_PC = R \cdot T(1) + \overline{R} \cdot T(0) + D(5) \cdot T(4)$ |
| $bus_DR = D(6) \cdot T(6)$ |
| $bus_AC = D(3) \cdot T(4) + p \cdot \left(OP2(10) + OP2(5)\right)$ |
| $bus_IR = \overline{R} \cdot T(2)$ |
| $bus_MEM = \overline{R} \cdot T(1) + \overline{D(7)} \cdot I \cdot T(3) + (D(0) + D(1) + D(2) + D(6)) \cdot T(4)$ |
| $BS(0) = bus_DR + bus_IR\ (+bus_AR + bus_MEM)$ |
| $BS(1) = bus_PC + bus_DR + bus_MEM$ |
| $BS(2) = bus_AC + bus_IR + bus_MEM$ |

AND-Mask型では不要



# データ転送回路の論理遅延



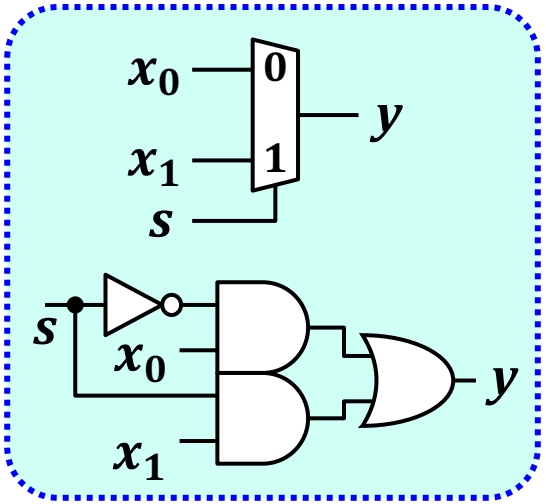



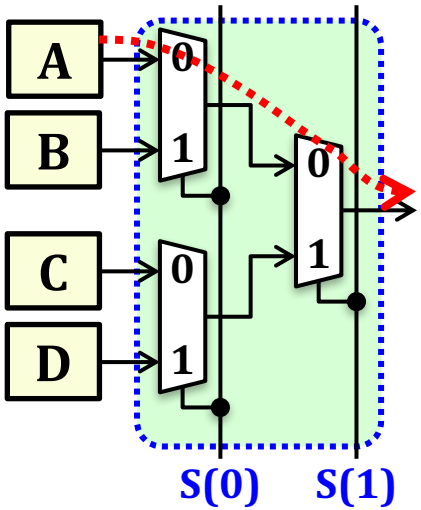

**P2P方式 : 論理遅延小**
- 個別の入力接続数 : 小
- 個別の出力接続数 : 小

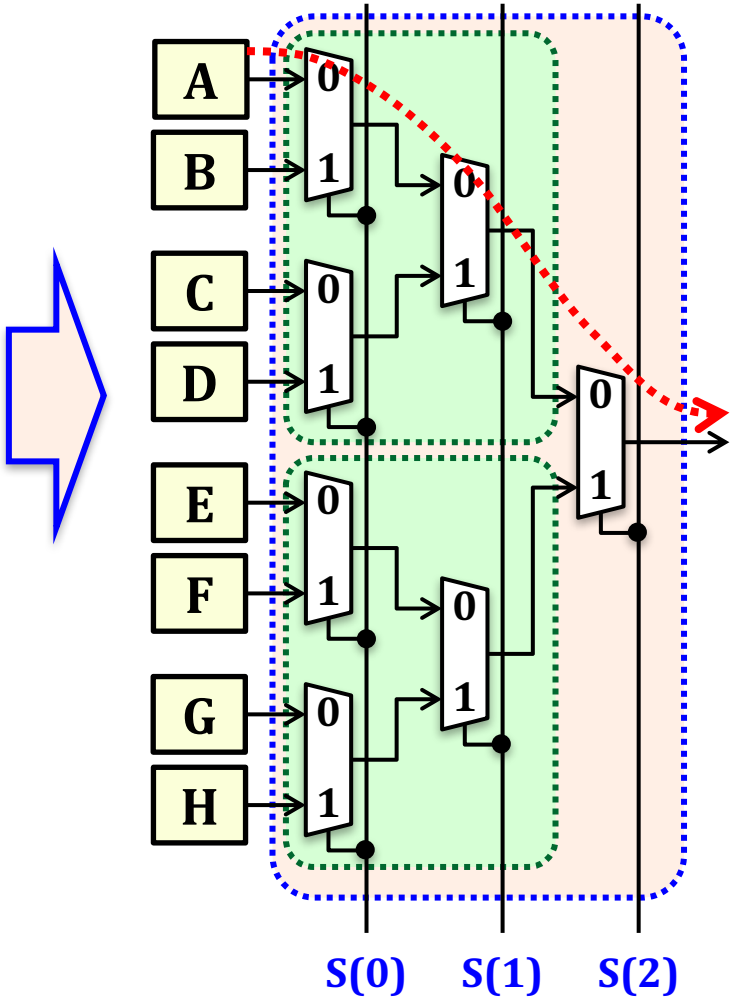
**共通バス方式 : 論理遅延大**
- 共通バスの入力接続数 : 大
- 共通バスの出力接続数 : 大

- 入力接続数（Fan-in） → 入力選択回路（MUX）の深さ（論理遅延）に影響
- 出力接続数（Fan-out） → 出力負荷容量（信号遷移遅延）に影響

# 入力接続数（Fan-in）と論理遅延



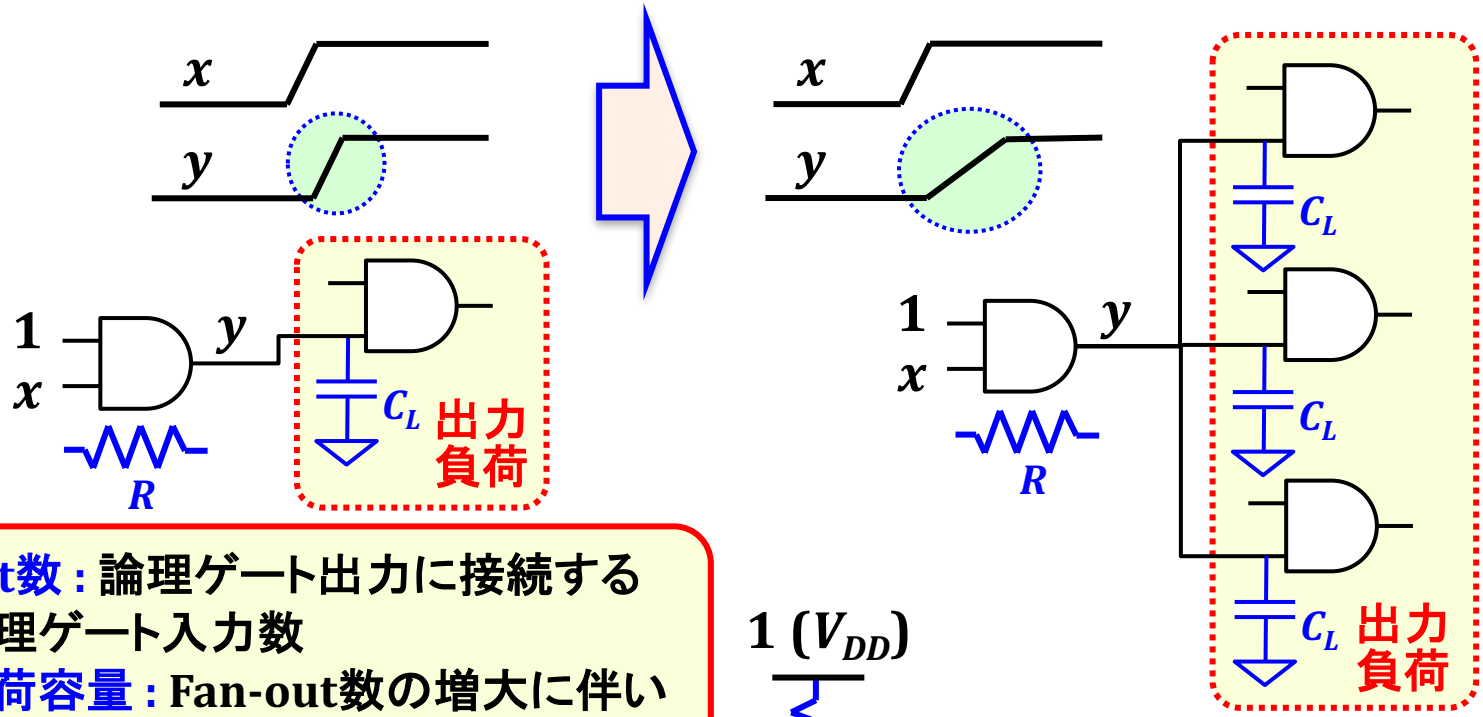




- MUX-Tree型：$2^M$入力MUX-Treeの深さ$M$と共に増大
- AND-Mask型：OR入力数の増大と共に論理遅延も増大

共通バス方式は、すべての出力信号を選択するため、P2P方式に比べMUX-Tree回路の論理遅延が増大する傾向にある

# 出力接続数（Fan-out）と信号遷移遅延

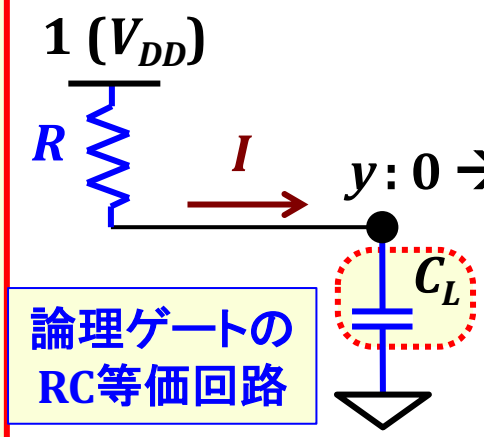



**Fan-out数：** 論理ゲート出力に接続する他の論理ゲート入力数

**出力負荷容量：** Fan-out数の増大に伴い出力負荷容量も増大する

**信号遷移遅延：** 出力負荷容量の増大に伴い、出力線の充電時間（信号遷移時間）も増大する

→ **共通バス方式**は、すべての入力（レジスタ・メモリ）に接続するため**P2P方式**に比べ**信号遷移遅延も増大する傾向**にある




論理ゲートのRC等価回路

出力負荷容量が増加すると充電する電荷量が増える

# データ転送方式のトレードオフ

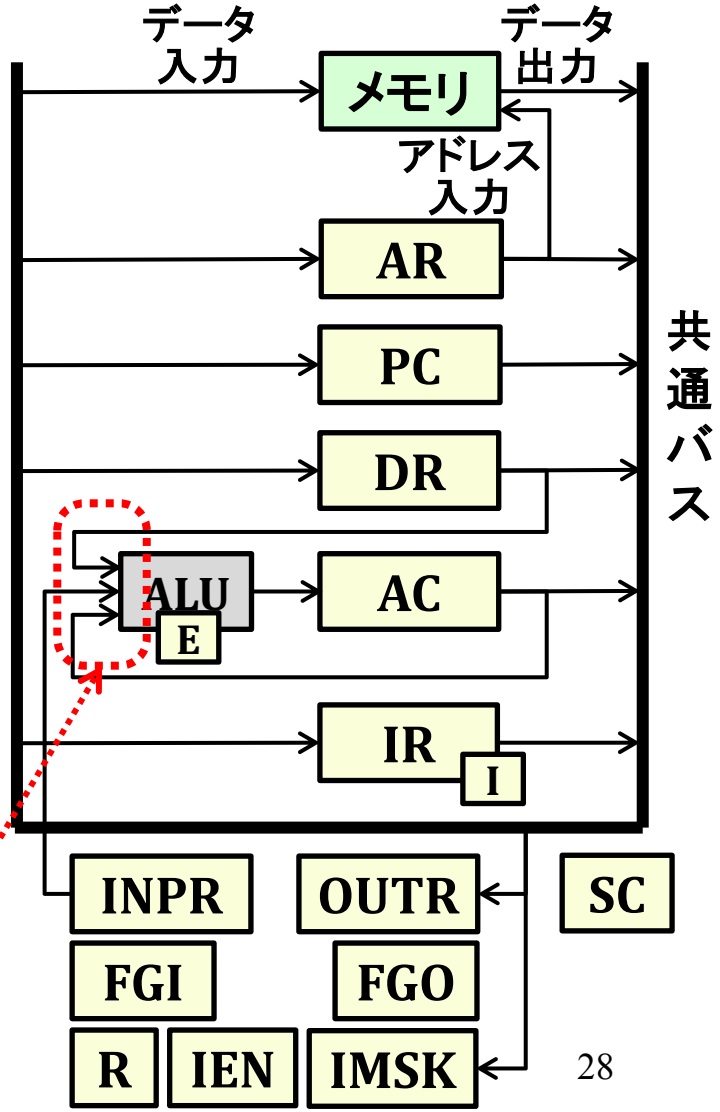
## ハードウェア設計における考慮点

- 回路規模、消費電力
- 性能：計算速度、論理遅延
- 設計の複雑度：回路設計手法に依存

→ これらは複雑なトレードオフ関係にある

## データ転送方式

- Point-to-Point（P2P）方式：設計が複雑、複数の同時転送可能、論理遅延小
- 共通バス方式：設計が単純、複数の同時転送不可、論理遅延大

→ 性能向上のためにはP2P方式が有利

→ 右図では、ALU入力がP2P方式である