

東北大学 工学部 機械知能・航空工学科
2019年度 クラス C D

情報科学基礎 I

12. 順序回路の設計と応用 (教科書5章)

大学院情報科学研究科

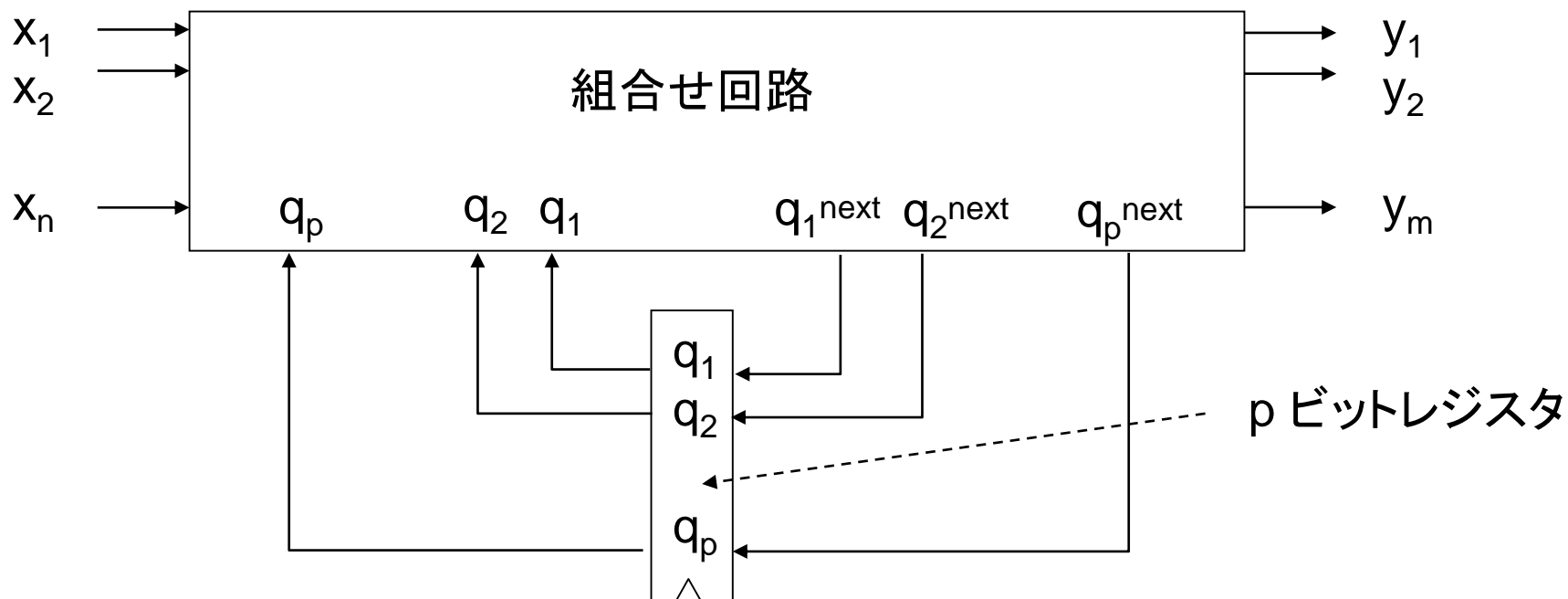
鏡 慎吾

<http://www.ic.is.tohoku.ac.jp/~swk/lecture/>

順序回路の設計

- 組合せ論理回路の設計法の分類
 - 構造や規則性に着目した手設計(先人の知恵を使う)
 - 入力・出力の関係に基づく自動合成(カルノー図など)
- 順序回路の設計法の分類
 - 構造や規則性に着目した手設計(前回の各例)
 - 入力・出力・状態の関係に基づく自動合成(今回)

#fis1 同期式順序回路の入力・出力・状態の関係



$$q_i(t+1) = f_i(\mathbf{q}(t), \mathbf{x}(t)), \quad i = 1, 2, \dots, p$$

$$y_j(t) = g_j(\mathbf{q}(t), \mathbf{x}(t)), \quad j = 1, 2, \dots, m$$

ただし $\mathbf{q}(t) = [q_1(t), q_2(t), \dots, q_p(t)]$

$$\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_n(t)]$$

f_i : 状態遷移関数

g_j : 出力関数

t : 離散時刻

有限状態機械

- 取り得る状態の数が有限であるようなシステム
- すべての状態を列挙して, どんな入力のあるときにどの状態からどの状態に遷移するのか, そのとき何が出力されるのかを考え尽くすことができる
- 設計手順
 - 入力・出力・状態の関係を状態遷移図で表す
 - 入力・出力・状態に2進符号を割り当てる
 - 状態遷移表・出力表を作成する
 - 論理回路に置き換える(必要ならば簡単化する)

例1: 自動販売機

- 100円硬貨だけを受け付ける
- 投入金額が300円に達すると品物が出てくる
- 投入済金額を返却するボタンがある(硬貨投入と同時に押せない)

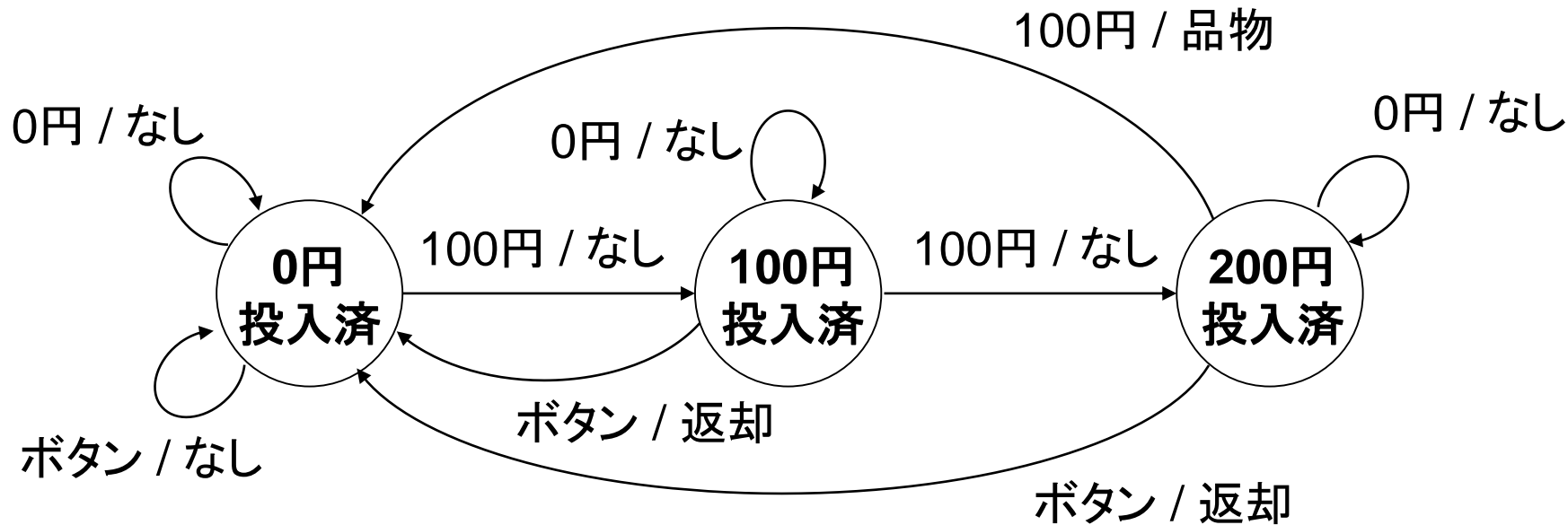
入力: { なし, 100円硬貨, 返却ボタン }

出力: { なし, 品物, 投入金額返却 }

状態: { 0円投入済, 100円投入済, 200円投入済 }

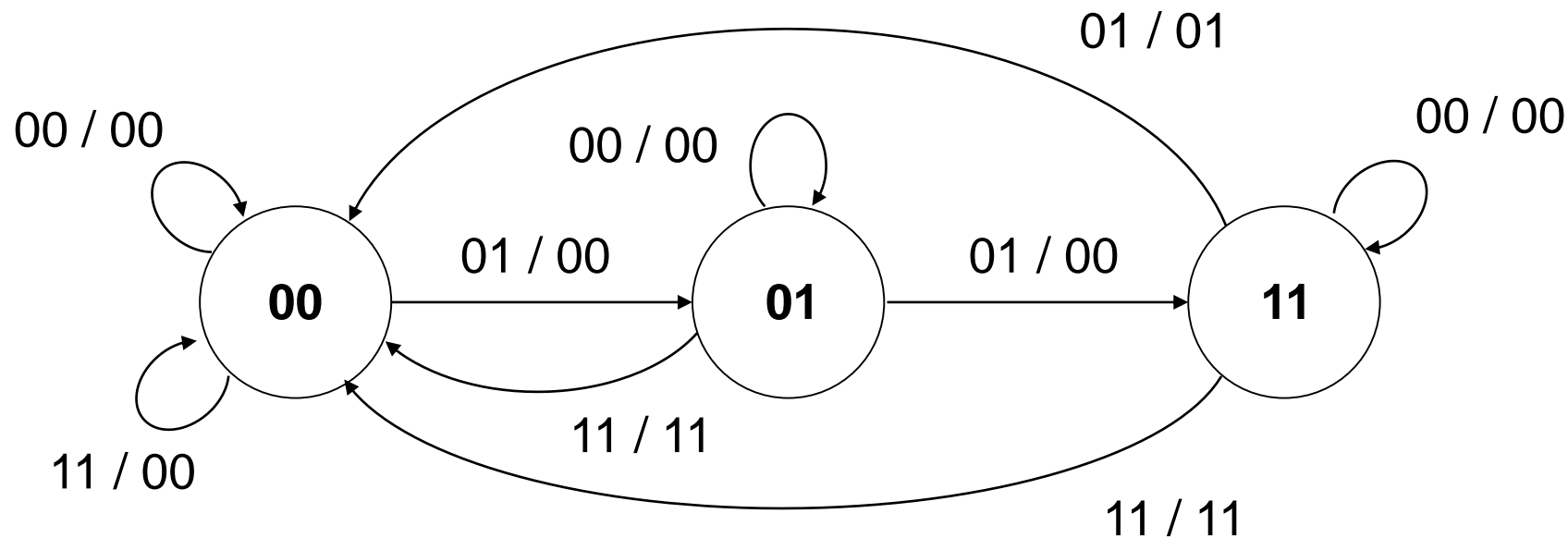
例1: 状態遷移図

- 状態を円(ノード)で表す
状態名を記入する
- 状態遷移を矢印(アーク)で表す
入力 / 出力 を付記する



例: 状態・入力・出力の符号化

入力: {	なし,	100円硬貨,	返却ボタン}	
	00	01	11	= $x_1 x_2$
出力: {	なし,	品物,	投入金額返却}	
	00	01	11	= $z_1 z_2$
状態: {	0円投入済,	100円投入済,	200円投入済}	
	00	01	11	= $y_1 y_2$



例1: 状態遷移表・出力表

次時刻の状態

入力×状態
→ 状態遷移先
の表を状態遷移表,

入力×状態
→ 出力
の表を出力表と呼ぶ

両方合わせて状態
遷移表と呼ぶことも
多い

y_1	y_2	x_1	x_2	Y_1	Y_2	z_1	z_2
0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0
0	0	1	0	*	*	*	*
0	0	1	1	0	0	0	0
0	1	0	0	0	1	0	0
0	1	0	1	1	1	0	0
0	1	1	0	*	*	*	*
0	1	1	1	0	0	1	1
1	0	0	0	*	*	*	*
1	0	0	1	*	*	*	*
1	0	1	0	*	*	*	*
1	0	1	1	*	*	*	*
1	1	0	0	1	1	0	0
1	1	0	1	0	0	0	1
1	1	1	0	*	*	*	*
1	1	1	1	0	0	1	1

例1: カルノー図による簡単化

Y ₁		X ₁ X ₂			
		00	01	11	10
y ₁ y ₂	00	0	0	0	*
	01	0	1	0	*
	11	1	0	0	*
	10	*	*	*	*

Y ₂		X ₁ X ₂			
		00	01	11	10
y ₁ y ₂	00	0	1	0	*
	01	1	1	0	*
	11	1	0	0	*
	10	*	*	*	*

Z ₁		X ₁ X ₂			
		00	01	11	10
y ₁ y ₂	00	0	0	0	*
	01	0	0	1	*
	11	0	0	1	*
	10	*	*	*	*

Z ₂		X ₁ X ₂			
		00	01	11	10
y ₁ y ₂	00	0	0	0	*
	01	0	0	1	*
	11	0	1	1	*
	10	*	*	*	*

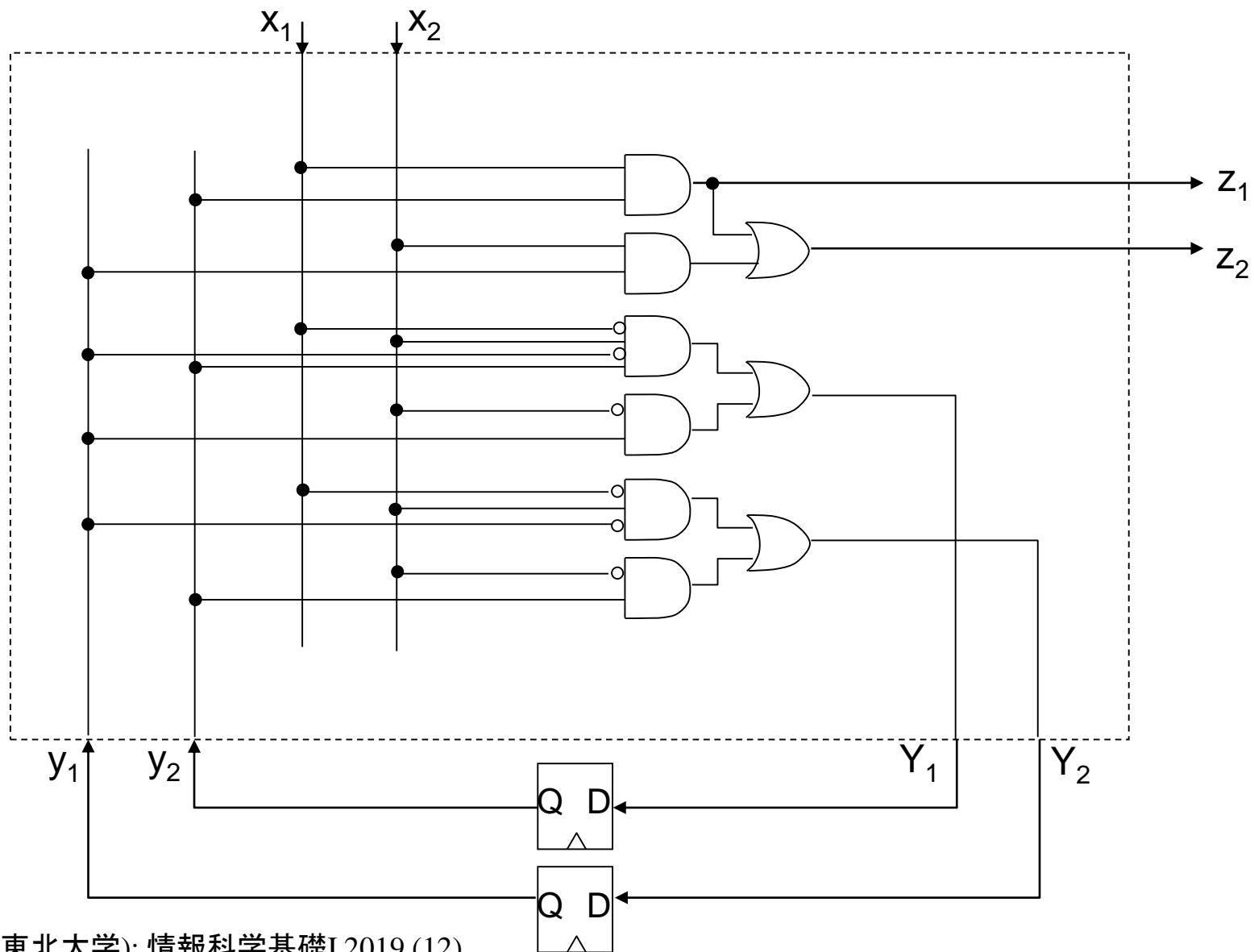
$$Y_1 = \overline{x_1}x_2\overline{y_1}y_2 + \overline{x_2}y_1$$

$$Y_2 = \overline{x_1}x_2\overline{y_1} + \overline{x_2}y_2$$

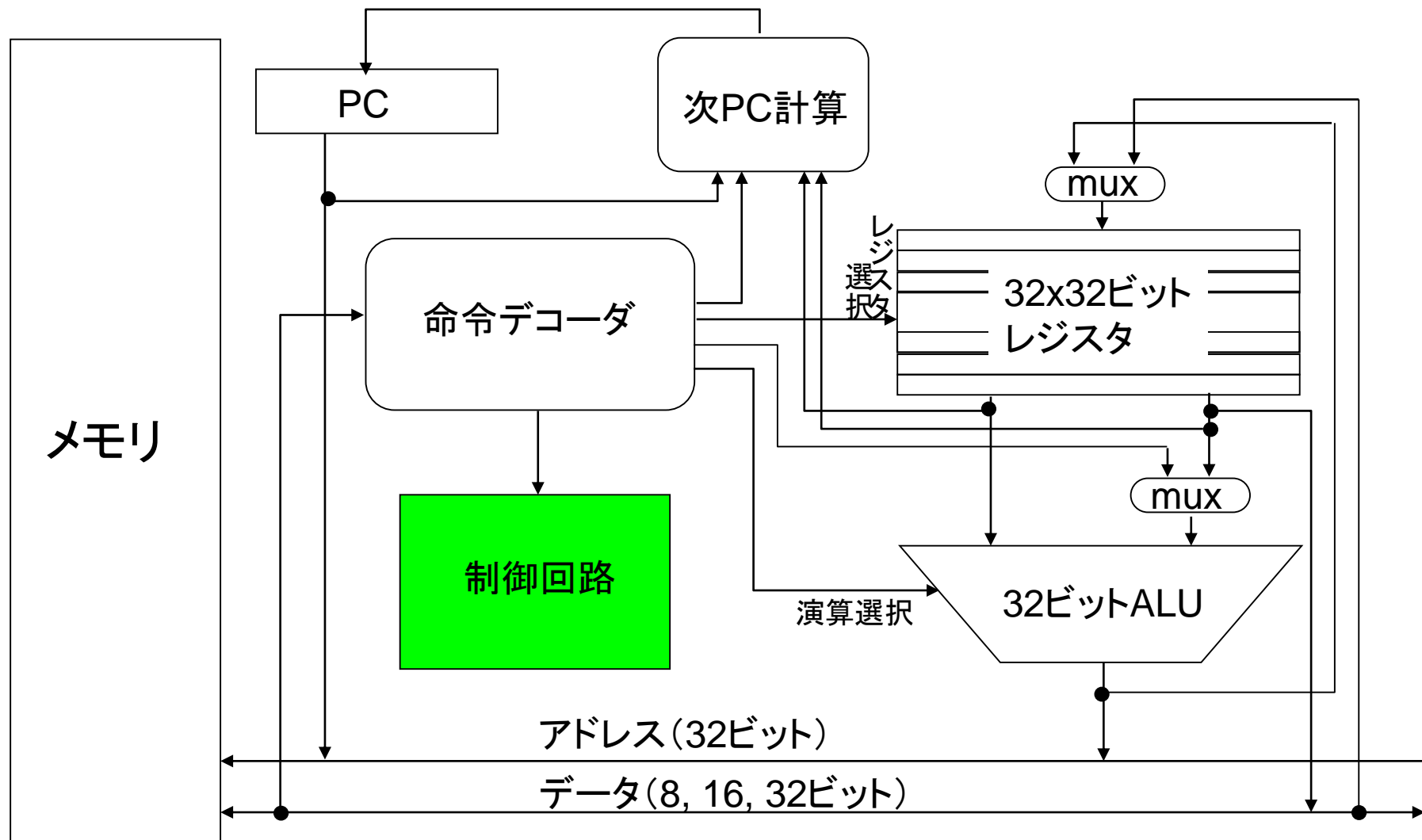
$$z_1 = x_1y_2$$

$$z_2 = x_1y_2 + x_2y_1$$

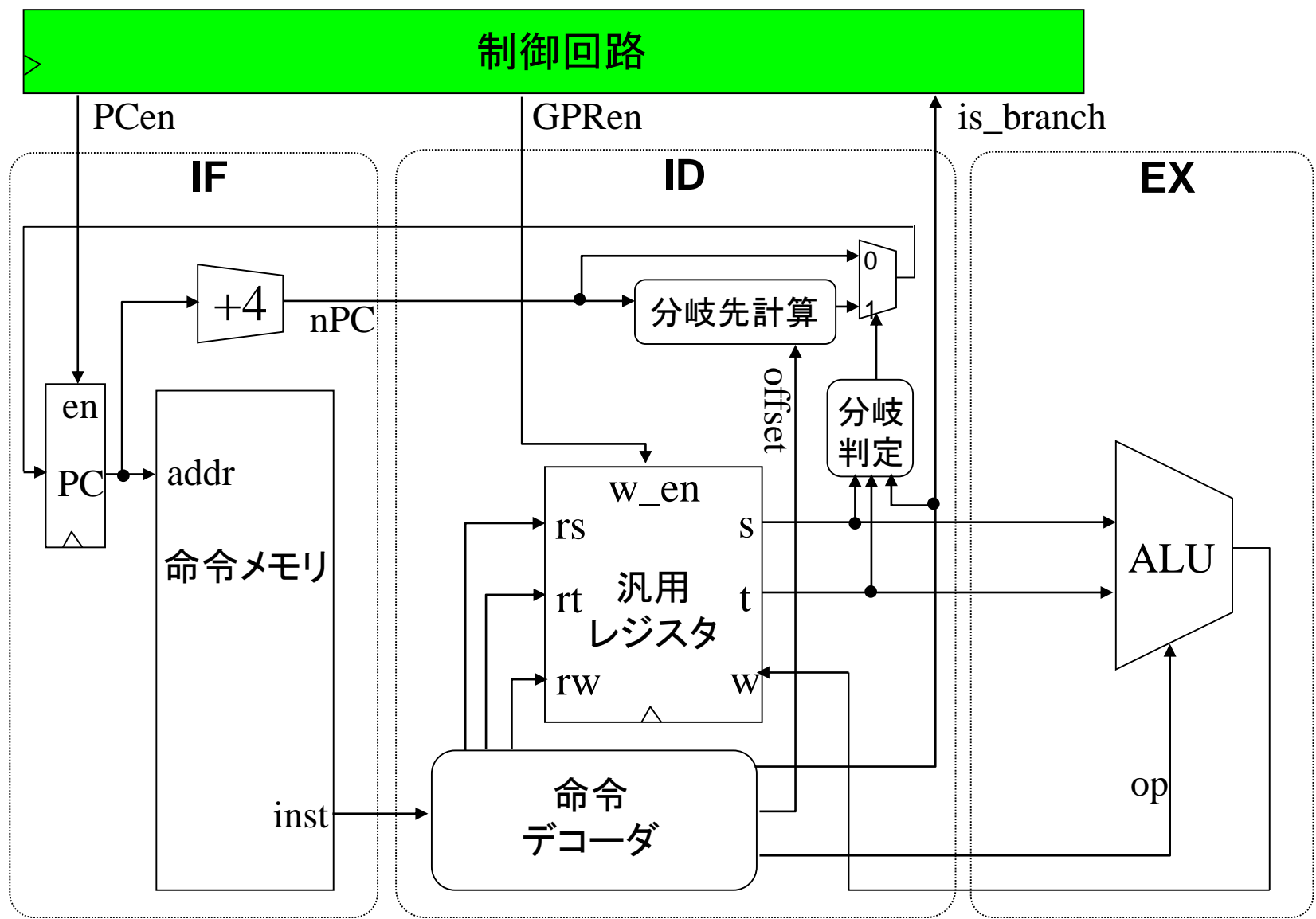
例1: 回路図



復習: MIPSの構造



例2: 簡易版MIPSの制御回路(付録E章)



例2: 定義

レジスタ間演算命令と分岐命令だけの簡略版MIPSの制御回路を設計する

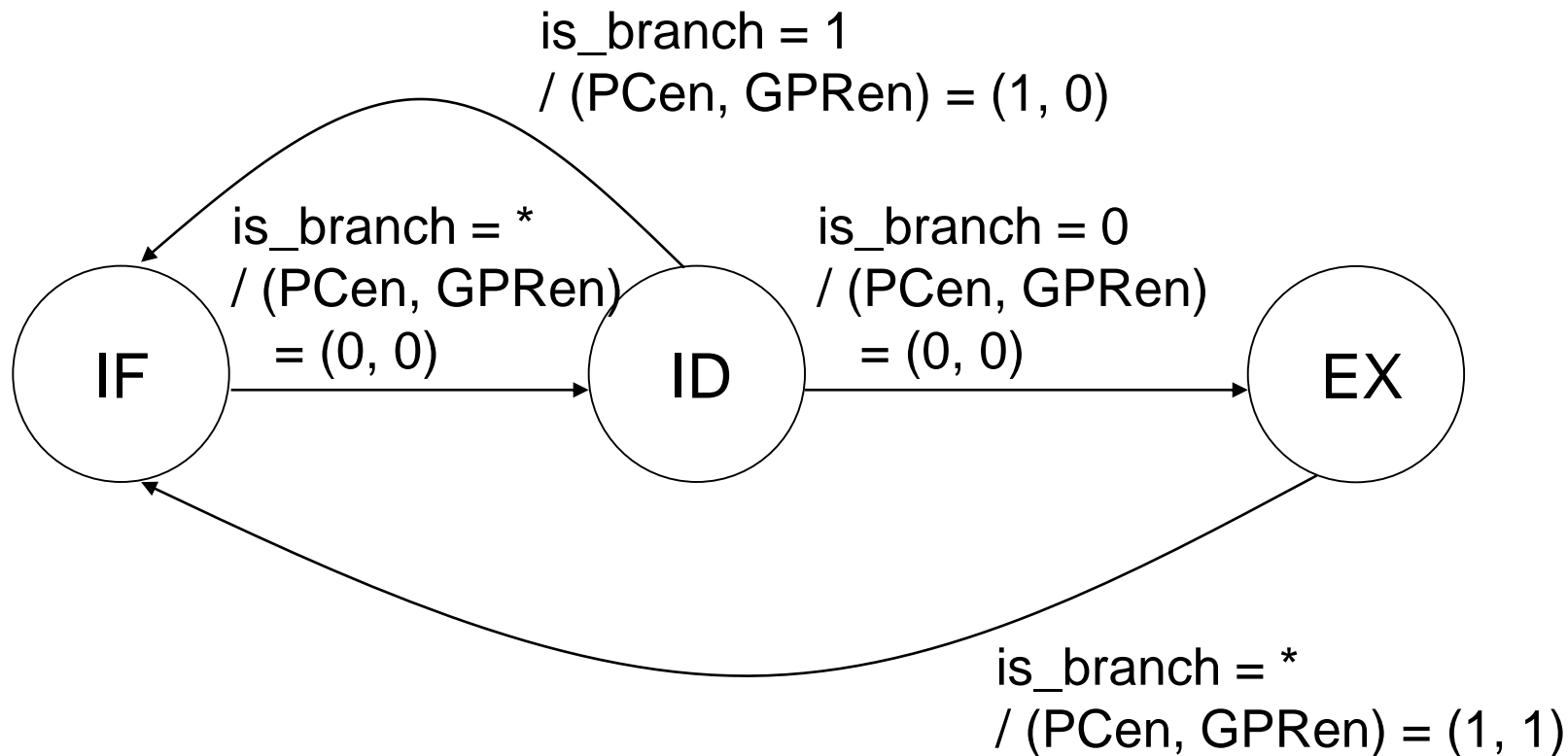
入力: 命令デコードの結果生成される is_branch 信号
(分岐命令なら1, 演算命令なら0)

出力: イネーブル信号 PCen, GPRen

状態:

- IF (Instruction Fetch): 命令読出し, 後続命令アドレス計算
- ID (Instruction Decode): 命令デコード, レジスタ読出, 分岐判定
- EX (EXecution): 演算実行, レジスタ書き込み

例2: 状態遷移図



「入力 = *」は、入力が1の場合も0の場合も出力・遷移先状態が同じことを表すとする

例2: 状態遷移表・出力表

符号割り当て:

IF: $Q_1Q_0 = 00$ ID: $Q_1Q_0 = 01$ EX: $Q_1Q_0 = 10$

Q_1	Q_0	is_branch	Q_1'	Q_0'	PCen	GPRen
0	0	0	0	1	0	0
0	0	1	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	1	0
1	0	0	0	0	1	1
1	0	1	0	0	1	1
1	1	0	*	*	*	*
1	1	1	*	*	*	*

例2: カルノー図による簡単化

Q_0'		$Q_1 Q_0$			
		00	01	11	10
is_branch	0	1	0	*	0
	1	1	0	*	0

PCen		$Q_1 Q_0$			
		00	01	11	10
is_branch	0	0	0	*	1
	1	0	1	*	1

Q_1'		$Q_1 Q_0$			
		00	01	11	10
is_branch	0	0	1	*	0
	1	0	0	*	0

GPRen		$Q_1 Q_0$			
		00	01	11	10
is_branch	0	0	0	*	1
	1	0	0	*	1

$$Q_0' = \overline{Q_1} \cdot \overline{Q_0}$$

$$Q_1' = Q_0 \cdot \overline{\text{is_branch}}$$

$$\text{PCen} = Q_1 + Q_0 \cdot \text{is_branch}$$

$$\text{GPRen} = Q_1$$

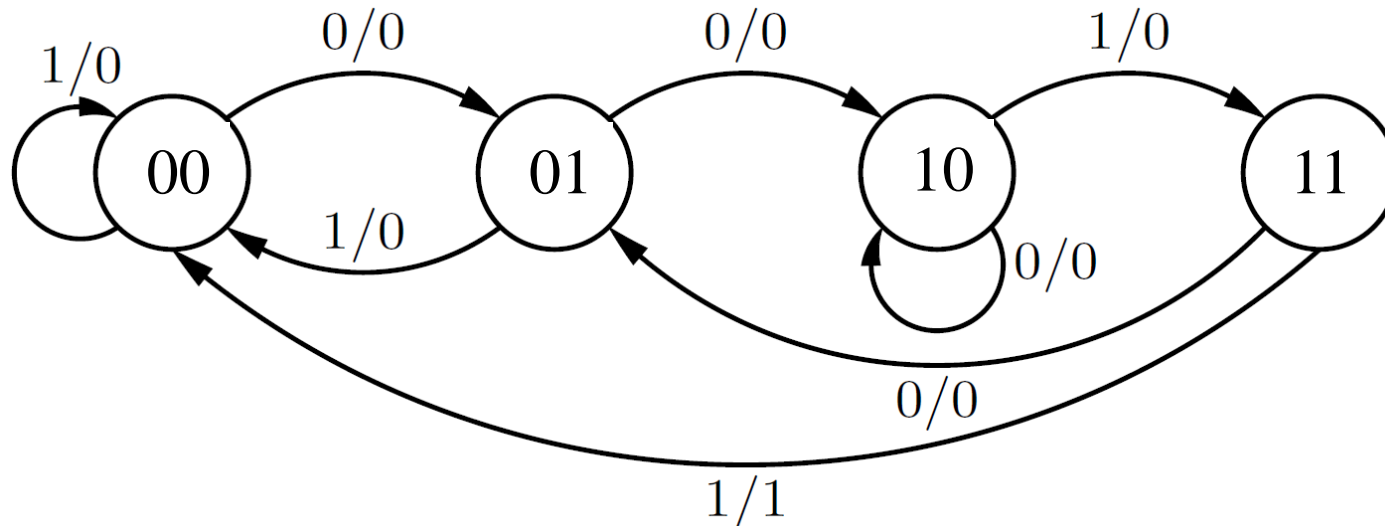
参考

- 今回学んだ Mealy 型の有限状態機械のほかに、出力が状態にだけ依存する(入りに直接依存しない) Moore 型の有限状態機械がある.
 - Moore 型も Mealy 型も表現能力は同じである. ただし, Moore 型は入りが出力に反映されるのに1時刻以上かかる.
 - 入りが出力に直接影響するのが好ましくない場合は Moore 型で設計する必要がある.
- 今回は, 状態遷移図から安直に回路を生成した, 実際は
 - より状態数の少ない有限状態機械で表せるかも知れない
 - 符号の割り当て方によって, 回路がもっと簡単になるかも知れないなどといったことも考える必要がある.

練習問題

論理値の系列(例えば 0100110101...) を入力として受け取って、「0011」という系列が現れたとき初めて1を出力し、それ以外では0を出力する順序回路を設計せよ。

入力を x , 状態を y_1y_2 (次時刻の状態を Y_1Y_2), 出力を z とする。



解答例

x	y_1	y_2	Y_1	Y_2	z
0	0	0	0	1	0
0	0	1	1	0	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	1	1	0
1	1	1	0	0	1

$$Y_1 = \bar{x}\bar{y}_1y_2 + y_1\bar{y}_2$$

$$Y_2 = \bar{x}\bar{y}_1\bar{y}_2 + \bar{x}y_1y_2 + xy_1\bar{y}_2$$

$$z = xy_1y_2$$

Y_1

		y_1y_2			
		00	01	11	10
x	0		1		1
	1				1

Y_2

		y_1y_2			
		00	01	11	10
x	0	1		1	
	1				1

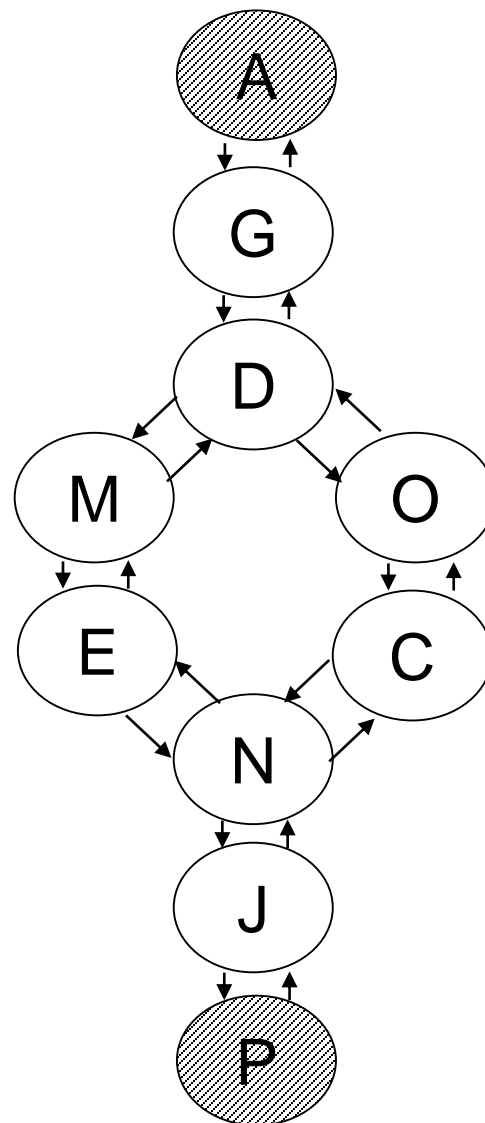
例題

ある人が狼, 羊, 牧草とともに旅をしていたところ, 川にさしかかった. 小さな舟を漕いで渡るしかない. 舟には, 漕ぎ手である人のほか, 狼, 羊, 牧草のいずれか高々1つしか乗せるスペースがない. ただし, 人がいないと狼は羊を食べてしまい, また羊は牧草を食べてしまう. 人, 狼, 羊, 牧草すべて無事に川を渡るにはどうすればよいか.

- (1) こちらの岸と向こう岸に存在し得る人, 狼, 羊, 牧草の組合せを列挙せよ. これが有限状態機械の状態となる. (渡っている途中の状況は, 状態としてなくてよい. 理由を考えよ)
- (2) 状態遷移図を作成せよ. ただし入力, 出力を考える必要はない. 状態に2進符号を割り当てる必要もない. (ある状態からはどの状態に遷移し得るかのみを考慮する)
- (3) 状態遷移図をもとに, 最短で向こう岸に渡るための手順を示せ. 答えは一つとは限らない. (ヒント: 初期状態から順に, あるいは最終状態から逆にたどるとよい. あるいは状態遷移図を整理しながら考えてもよい)

解答例

状態	こちら岸	向こう岸
A	人狼羊草	- ϕ
(B	狼羊草	- 人)
C	人羊草	- 狼
D	人狼草	- 羊
E	人狼羊	- 草
(F	羊草	- 人狼)
G	狼草	- 人羊
(H	狼羊	- 人草)
(I	人草	- 狼羊)
J	人羊	- 狼草
(K	人狼	- 羊草)
(L	人	- 狼羊草)
M	狼	- 人羊草
N	羊	- 人狼草
O	草	- 人狼羊
P	ϕ	- 人狼羊草



最短経路は二つ