【7】==========

題意より, 出力記号を, A(A さんの勝ち), B(B さんの勝ち), C(未決)で表し, これらの集合(出力の集合)を $\boldsymbol{O}$ と書く. 次に, グー, チョキ, パーをそれぞれ R(rock), S(scissors), P(paper)で表し, A さんが出した手を先に書くことにすると, 入力記号は, RR, RS, RP, SR, SS, SP, PR, PS, PP の 9 種類になる. そこで, これらの入力の集合を $\boldsymbol{I}$ と書き, 3 つの部分集合: A さんが勝ちの集合 $\boldsymbol{I}_A$ = { RS, SP, PR }, B さんが勝ちの集合 $\boldsymbol{I}_B$ = { RP, SR, PS }, あいこの集合 $\boldsymbol{I}_C$ = { RR, SS, PP } に分割しておく. そうすると, 勝ち負けが分かりやすい.

このとき, 次の 3 状態を定義すれば, どちらが 2 回勝ち越したかを判定する回路を設計できる.

Q0 ： 勝ち負けの回数が同じ （初期状態）

QA ： A さんが 1 回勝ち越している

QB ： B さんが 1 回勝ち越している

すなわち, 入力の集合 $\boldsymbol{I}$, 出力の集合 $\boldsymbol{O}$, および状態の集合 $\boldsymbol{Q}$ がそれぞれ以下のように与えられたとき,

$\boldsymbol{I}$ = $\boldsymbol{I}_A \cup \boldsymbol{I}_B \cup \boldsymbol{I}_C$ = { RS, SP, PR } ∪ { RP, SR, PS } ∪ { RR, SS, PP },

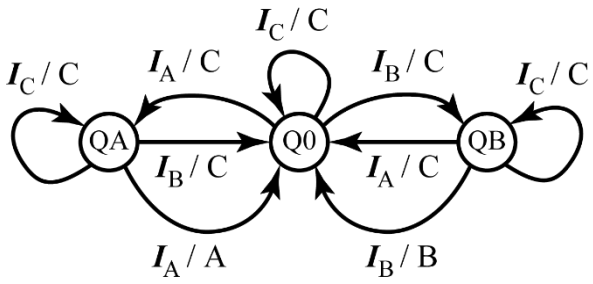
$\boldsymbol{O}$ = { A, B, C }

$\boldsymbol{Q}$ = { Q0, QA, QB }

下記の状態遷移表のように状態遷移をし, 出力表のような出力をする順序回路を作成すれば, 2 回勝ち越した方を判定できる. ここでは, 2 回勝ち越して結果が分かったら, 続けて新たな入力を受け付けることができるよう, 状態は初期状態 Q0 に戻るものとしている.

現状態	次状態				出力		
入力	$\boldsymbol{I}_A$ = {RS, SP, PR}	$\boldsymbol{I}_B$ = {RP, SR, PS}	$\boldsymbol{I}_C$ = {RR, SS, PP}		$\boldsymbol{I}_A$ = {RS, SP, PR}	$\boldsymbol{I}_B$ = {RP, SR, PS}	$\boldsymbol{I}_C$ = {RR, SS, PP}
Q0	QA	QB	Q0		C	C	C
QA	Q0	Q0	QA		A	C	C
QB	Q0	Q0	QB		C	B	C

そこで, これを状態遷移図として表すと, 下図となる. ただし, 図を簡潔にするため, 各枝に付けた入力のラベルには, 入力記号ではなく, 部分集合名を書いている.



© Shuji Tsukiyama, Takashi Kambe, Masahiro Fukui, 2017

順序回路を設計するには, 入力の集合 $\boldsymbol{I}$, 出力の集合 $\boldsymbol{O}$, および状態の集合 $\boldsymbol{Q}$ をそれぞれ符号化しなければならないが, 符号化の自由度は大きく, この時点で回路規模が小さくなるような最適な符号を選択することは困難である. 従って, 以下の符号化は一例で, 順序回路設計手順を確認するためだけに書かれていると, 思って欲しい.

入力の集合 $\boldsymbol{I}$ には 9 個の要素が含まれ, それらが 3 つの部分集合に分割されているので, 4 個の論理変数 x_A, x_B, x_C, x_D を用い, 以下のように符号化されているものとする. すなわち, $\boldsymbol{I}_A$ の入力には $x_A = 1$, $x_B = 0$ なる符号を, $\boldsymbol{I}_B$ の入力には $x_A = 0$, $x_B = 1$ なる符号を, $\boldsymbol{I}_C$ の入力には $x_A = 0$, $x_B = 0$ なる符号を割り当て, $x_A = 1$, $x_B = 1$ なる符号は用いないものとする.

	$\boldsymbol{I}_A$ = {RS, SP, PR}	$\boldsymbol{I}_B$ = {RP, SR, PS}	$\boldsymbol{I}_C$ = {RR, SS, PP}	don't care
(x_A, x_B, x_C, x_D)	(1, 0, -, -)	(0, 1, -, -)	(0, 0, -, -)	(1, 1, -, -)

出力の集合 $\boldsymbol{O}$ には 3 個の要素が含まれているので, 2 つの論理変数 z_A, z_B を用いて以下のように符号化されているものとする. すなわち, $z_A = 1$, $z_B = 1$ なる出力は生成しないものとする.

	A	B	C
(z_A, z_B)	(1, 0)	(0, 1)	(0, 0)

最後に, 状態の集合 $\boldsymbol{Q}$ には 3 個の要素が含まれているので, 2 つの論理変数 q_A, q_B を用いて以下のように符号化する. すなわち, $q_A = 1$, $q_B = 1$ なる状態は生じないもの(ドントケア)とする.

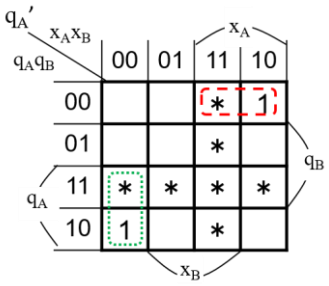
	Q0	QA	QB	don't care
(q_A, q_B)	(0, 0)	(1, 0)	(0, 1)	(1, 1)

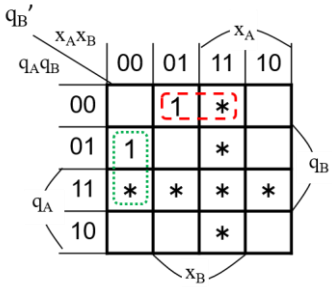
こうすると, 上記の状態遷移表および出力表は下のように論理値で書くことができる.

現状態 q_A, q_B		次状態 q_A', q_B'				出力 z_A, z_B			
	x_A, x_B	0, 0 ($\boldsymbol{I}_C$)	0, 1 ($\boldsymbol{I}_B$)	1, 0 ($\boldsymbol{I}_A$)	1, 1	0, 0 ($\boldsymbol{I}_C$)	0, 1 ($\boldsymbol{I}_B$)	1, 0 ($\boldsymbol{I}_A$)	1, 1
0, 0	(Q0)	0, 0	0, 1	1, 0	＊, ＊	0, 0	0, 0	0, 0	＊, ＊
0, 1	(QB)	0, 1	0, 0	0, 0	＊, ＊	0, 0	0, 1	0, 0	＊, ＊
1, 0	(QA)	1, 0	0, 0	0, 0	＊, ＊	0, 0	0, 0	1, 0	＊, ＊
1, 1		＊, ＊	＊, ＊	＊, ＊	＊, ＊	＊, ＊	＊, ＊	＊, ＊	＊, ＊

これらの状態遷移表と出力表から, 次状態の状態変数 q_A', q_B' および出力変数 z_A, z_B のカルノー図は, 下図となる. 従って, これらより, 最簡な積和形論理式が得られる. 各論理式はカルノー図の下に示す. さらに, これらの論理式を NAND 演算に変えた式も, 積和形論理式の下に示す.

© Shuji Tsukiyama, Takashi Kambe, Masahiro Fukui, 2017



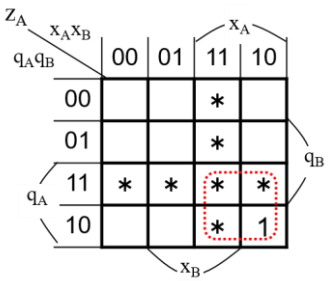


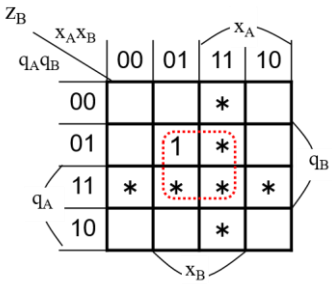
$$q_A' = \overline{x_A} \cdot \overline{x_B} \cdot q_A + x_A \cdot \overline{q_A} \cdot \overline{q_B}$$

$$q_B' = \overline{x_A} \cdot \overline{x_B} \cdot q_B + x_B \cdot \overline{q_A} \cdot \overline{q_B}$$

$$q_A' = \overline{\overline{\overline{x_A} \cdot \overline{x_B} \cdot q_A + x_A \cdot \overline{q_A} \cdot \overline{q_B}}} = \overline{(\overline{x_A} \cdot \overline{x_B} \cdot q_A) \cdot (x_A \cdot \overline{q_A} \cdot \overline{q_B})}$$

$$q_B' = \overline{\overline{\overline{x_A} \cdot \overline{x_B} \cdot q_B + x_B \cdot \overline{q_A} \cdot \overline{q_B}}} = \overline{(\overline{x_A} \cdot \overline{x_B} \cdot q_B) \cdot (x_B \cdot \overline{q_A} \cdot \overline{q_B})}$$





$$z_A = x_A \cdot q_A$$

$$z_A = \overline{(\overline{x_A} \cdot \overline{q_A})}$$

$$z_B = x_B \cdot q_B$$

$$z_B = \overline{(\overline{x_B} \cdot \overline{q_B})}$$

さらに，初期化信号 reset が 1 のときに，2 つの D フリップフロップを初期状態 $q_A = q_B = 0$ にする初期化回路は，各 D フリップフロップの入力 d_A および d_B に，それぞれ次式の値を入れればよい．

$$d_A = \overline{\text{reset}} \cdot D_A = \overline{(\overline{\text{reset}} \cdot D_A)} \ ,$$

$$d_B = \overline{\text{reset}} \cdot D_B = \overline{(\overline{\text{reset}} \cdot D_B)}$$

ここで，D_A および D_B は，それぞれ q_A' および q_B' の右辺

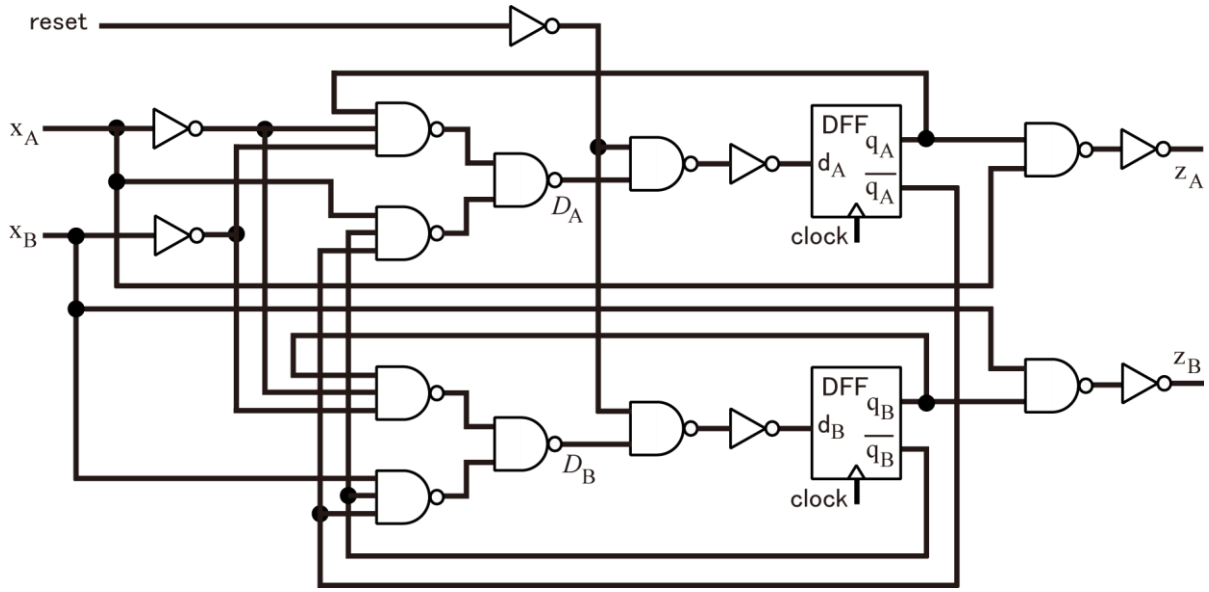
$$D_A = \overline{(\overline{x_A} \cdot \overline{x_B} \cdot q_A) \cdot (x_A \cdot \overline{q_A} \cdot \overline{q_B})} = q_A' \ ,$$

$$D_B = \overline{(\overline{x_A} \cdot \overline{x_B} \cdot q_B) \cdot (x_B \cdot \overline{q_A} \cdot \overline{q_B})} = q_B'$$

である．

これらを用いて，順序回路を構成すると，下図のようになる．

© Shuji Tsukiyama, Takashi Kambe, Masahiro Fukui, 2017



問題文では，各時刻に 9 種類の入力記号の一つが入力されるものとしており，上の解答例でもその前提で設計した．これに対して，本章 5.7 で取り上げた自動販売機の制御回路では，時刻を指定するクロックの周波数が高く，全ての時刻に 50 円玉あるいは 100 円玉が投入されると限らないため，どちらの貨幣も入力されない “入金無し” という入力（入力記号 X_0）を考えた．上で設計した同期式順序回路において，各時刻に入力されるジャンケンの結果が，人が（同時進行で）行っているジャンケンの結果であるならば，クロックの周波数は低くなければならず，ジャンケンの結果が一定周期で入力されなければならない．

このような入力を想定することが困難な場合，ジャンケン判定回路に対して，入力を読み取るタイミングを指定する入力 T を新たに考え，T が 0 から 1 に変化した時点が時刻を指定していると考えると，不定期に入力があってもよく，各時刻に入力を仮定することができる．すなわち，T が 0 から 1 に変化した時に，入力を受け付け，それが 9 種類の内のどれであるかを読み取るようにするのである．

このように，ジャンケン判定回路のような順序回路を設計する場合，入力と出力が明確に定義されている必要がある．

一方，どのような入力装置を用意すれば良いかを考えるのも楽しい．昨今のように，画像処理技術が進歩してくると，ジャンケンをカメラで撮影し，その画像から 9 種類の入力記号のどれになったかを判別し，それをジャンケン判定回路に送信することもできるであろう．また，その入力装置には，「後出し」 のような不正を検出する機能を付与しておくこともできる．

このように，実用に供する装置を設計する場合，ユーザや他の装置とのインターフェイス（interface：境界面）が重要であるが，本書では扱わない．

© Shuji Tsukiyama, Takashi Kambe, Masahiro Fukui, 2017