電子制御工学実験報告書

実験題目　：論理回路の設計

報告者　　：3 年 32 番　平田 蓮

提出日　　：2019 年 12 月 17 日

実験日　　：2019 年 11 月 25 日, 12 月 9 日, 12 月 16 日

実験班　　：第 4 班

共同実験者：　8 番 小林歩夢

※指導教員記入欄

評価項目	配点	一次チェック ・　・	二次チェック ・　・
記載量	20		
図・表・グラフ	20		
見出し，ページ番号，その他体裁	10		
その他の減点	–		
合計	50		

コメント：

1　目的

本実験では, ロジック学習装置 KENTAC 2600 (昭和電業社) を用いて基本的な論理回路の設計を行い, コンピュータのハードウェア設計の基礎となる論理回路設計の理解を深めることを目的とする. まず, マルチプレクサや半加算器などの組み合わせ論理回路の設計に取り組み, 真理値表と論理式と論理回路の対応へのイメージを深める. そして, 各種フリップフロップの動作確認とシフトレジスタ及びカウンタの設計を通して, コンピュータの記憶装置を用いた機能の基礎となる順序回路への理解を深める.

2　KENTAC 2600 について

本実験で使用するロジック学習装置 KENTAC 2600 の主な特徴を以下に挙げる.

- 入出力の端子と LED ディスプレイを接続することで, High/Low を目で確認できる.
- 回路素子記号がパネル上に描かれているため, わかりやすい.
- 信号発信器を内蔵しているので, 外部の発信器が必要ない.

3　組み合わせ論理回路の設計

3.1　プライオリティエンコーダの設計

情報を保存したり伝送したりする際には, なるべく使用する記憶素子や伝送路を少なくするのが望ましい. そこで用いられるのが符号化 (エンコード) という手法であり, 符号化を行う回路をエンコーダという. 中でも, 複数の入力があった場合に最も添字が大きい入力を優先させる回路をプライオリティエンコーダという.

この節では, 4 入力のプライオリティエンコーダを設計する. 入力をそれぞれ A_i $(i \in \{0,1,2,3\})$, 出力を Z_1, Z_2, E とし, $(Z_2\ Z_1)_2$ で入力されている最大の添字を表す. E はどこからも 1 の入力がなければ 1, それ以外の場合は 0 である.

まず, このプライオリティエンコーダの真理値表を表 1 示す. 表内の $\times$ はドントケア入力であることを示す.

表 1　4 入力プライオリティエンコーダの真理値表

A_0	A_1	A_2	A_3	Z_2	Z_1	E
0	0	0	0	0	0	1
1	0	0	0	0	0	0
$\times$	1	0	0	0	1	0
$\times$	$\times$	1	0	1	0	0
$\times$	$\times$	$\times$	1	1	1	0

次に, この表からカルノー図を作成して各出力の式を導出する. カルノー図を以下に示す.

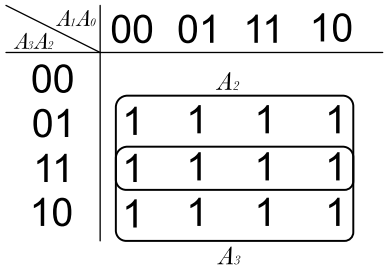

図 1　Z_2 についてのカルノー図

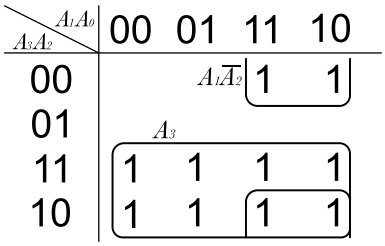

図 2　Z_1 についてのカルノー図

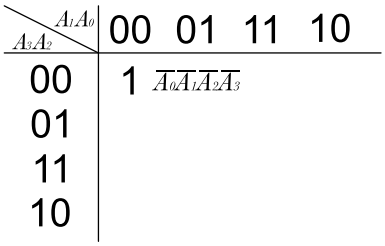

図 3　E についてのカルノー図

また, これらの式は以下のように真理値表から直接求めることもできる.

$$Z_2 = A_2\overline{A_3} + A_3 = A_2\overline{A_3} + A_2A_3 + \overline{A_2}A_3 = A_2 + A_3$$
$$Z_1 = A_1\overline{A_2}\ \overline{A_3} + A_3 = A_1\overline{A_2}\ \overline{A_3} + A_1\overline{A_2}A_3 + \overline{A_1\overline{A_2}}A_3 = A_1\overline{A_2} + A_3$$
$$E = \overline{A_0}\ \overline{A_1}\ \overline{A_2}\ \overline{A_3}$$

最後に, これらの式から回路を構成する. 構成した回路を図 4 に示す.

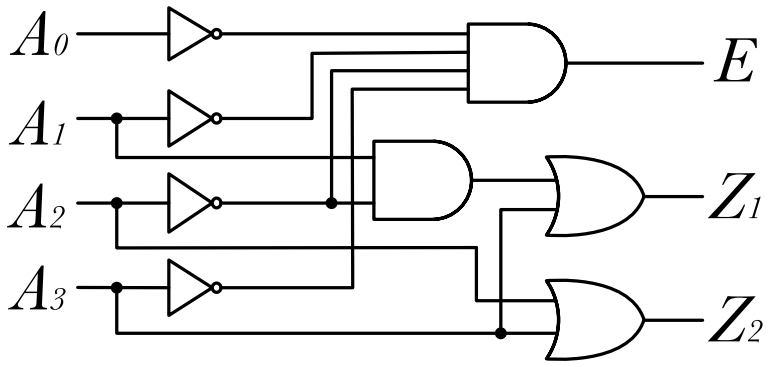

図 4　4 入力プライオリティエンコーダの論理回路

3.2　加算器

論理回路によって桁上がり入力を考慮しない 1 ビットの加算を行う回路を半加算器 (HA: Half Adder) という. HA は, 入力 A, B に対してその和 $(C^+\ S)_2$ を出力する (C^+ は桁上がり).

HA は下位ビットからの桁上がりを考慮しておらず複数桁の演算をできないため, 桁上がり入力 C を追加することを考える. この回路を全加算器 (FA: Full Adder) という.

FA は, $A + B + C$ を演算するので, HA を 2 つ使用して以下のように構成できる. ただし, 2 つの HA の出力をそれぞれ $(C_1^+\ S_1)_2$, $(C_2^+\ S)_2$ とする.

1. HA を使用して $A + B$ を演算して S_1 と C_1^+ を得る.

2. HA を使用して $S_1 + C$ を演算して S と C_2^+ を得る.
3. 上のどちらかの演算で桁上がりが発生していたら全体の演算でも桁上がりが発生するので, C_1^+ と C_2^+ の論理和を取って C^+ を得る.

このように構成した回路と FA のタイミングチャートをを図 5, 6 に示す.

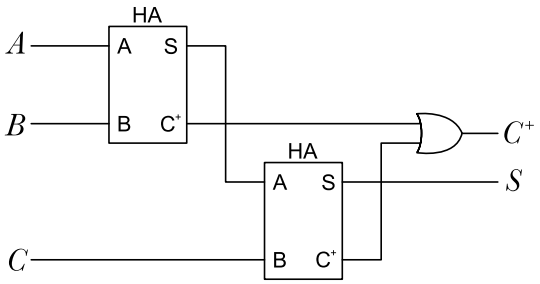

図 5　FA の論理回路

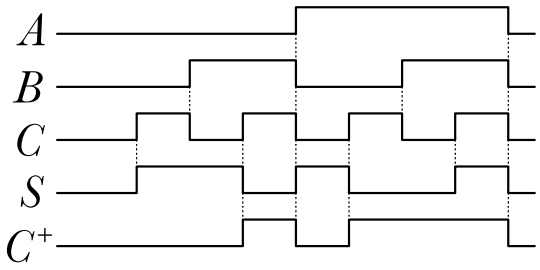

図 6　FA のタイミングチャート

3.3　2 ビット加算器

3.2 節で作成した FA を使用して, $(A_2\ A_1)_2 + (B_2\ B_1)_2 = (S_3\ S_2\ S_1)_2$ の演算を行う 2 ビットの加算器を作成する.

筆算と同じ要領で, 先に下位ビットを HA で演算した後, その桁上がりと上位ビットを FA で演算することで構成できる. 図 7 に作成した回路示す.

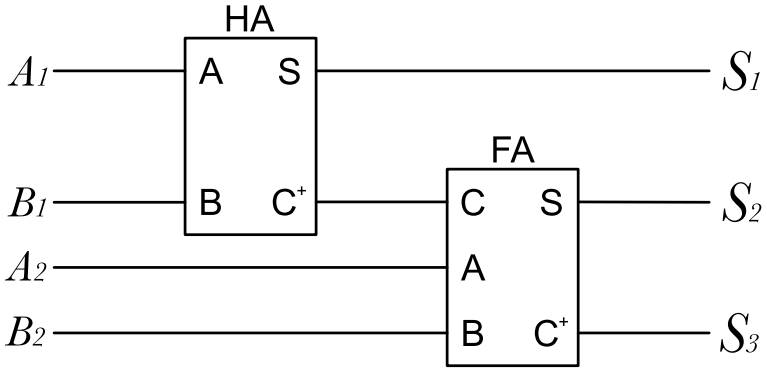

図 7　2 ビット加算器の論理回路

4　順序論理回路の設計

組み合わせ論理回路はその時の入力によって出力がただ 1 つに定まる回路であった. しかし, コンピュータの動作の中にはその時の入力だけでなく, その時の内部状態も出力に関わるものが多い. このような回路を順序回路という. 順序回路はその時の入力と内部状態によって出力 (次の内部状態) が決まるので, 組み合わせ論理回路と違い, 入力に対して出力が一意に定まらない.

順序回路を構成する際に, 1 ビットの記憶素子としてフリップフロップ (FF: Flip Flop) を使用する. FF はその特徴によって様々な種類が存在するが, 本節では JK-FF と D-FF に触れる.

4.1 JK-FF と D-FF

JK-FF, D-FF の入力をそれぞれ $(J, K), D$, 両 FF のクロック入力, 内部状態, 次の内部状態をそれぞれ CLK, Q, Q^+ とする. 両 FF のタイミングチャート, 状態遷移表を以下に示す. 状態遷移表の $\times$ はドントケア入力を表す.

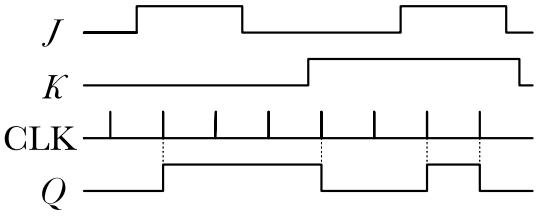

図 8 JK-FF のタイミングチャート

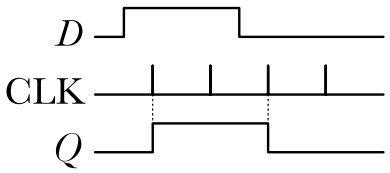

図 9 D-FF のタイミングチャート

表 2 JK-FF の状態遷移表

J	K	CLK	Q^+
$\times$	$\times$	0	Q
0	0	1	Q
0	1	1	0
1	0	1	1
1	1	1	$\overline{Q}$

表 3 D-FF の状態遷移表

CLK	Q^+
0	Q
1	D

JK-FF は, $J = K = 1$ の時以外は $R = J$, $S = K$ と見たときの RS-FF と同じ動作をする. $J = K = 1$ の時は内部状態 Q を反転させる.

D-FF は, は CLK が 0 の時は現在の内部状態を保持し, それ以外は内部状態 Q を D にする.

4.2 シフトレジスタの作成

簡単な順序回路の例として 3 ビットシフトレジスタを設計する. シフトレジスタは, クロック入力ごとに保持しているビット列を 1 桁ずつシフトさせる回路である. この操作の度に最上位ビットは捨てられ, 最下位ビットにはクロック入力時の入力が入る. 構成方法としては, フリップフロップを直列に並べ, 下位ビットの内部状態を上位ビットに書き込むようにすればいい.

D-FF を使用すれば簡単に構成することができるが, KENTAC 2600 には D-FF が 2 個しか内蔵されていないので, JK-FF を用いて D-FF と同じ動作をする回路を作成する.

作成した擬似 D-FF を 3 つ直列に接続してシフトレジスタを作成する. このシフトレジスタの内部状態を Q_3, Q_2, Q_1 とする. Q_3 が最上位ビットである. また, 直近の入力 A が $1 \to 1 \to 0$ だった時に出力 X が 1 になるようにする.

これは, (Q_3, Q_2, Q_1) が (1, 1, 0) のときに X に 1 を出力すればいい.

作成した回路を図 10, 11 に示す. ただし, 擬似 D-FF の入力, クロック入力をそれぞれ D, CLK, 出力を $(Q, \overline{Q})$ とする.

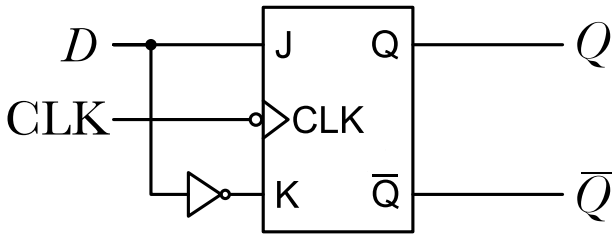

図 10　JK-FF による D-FF

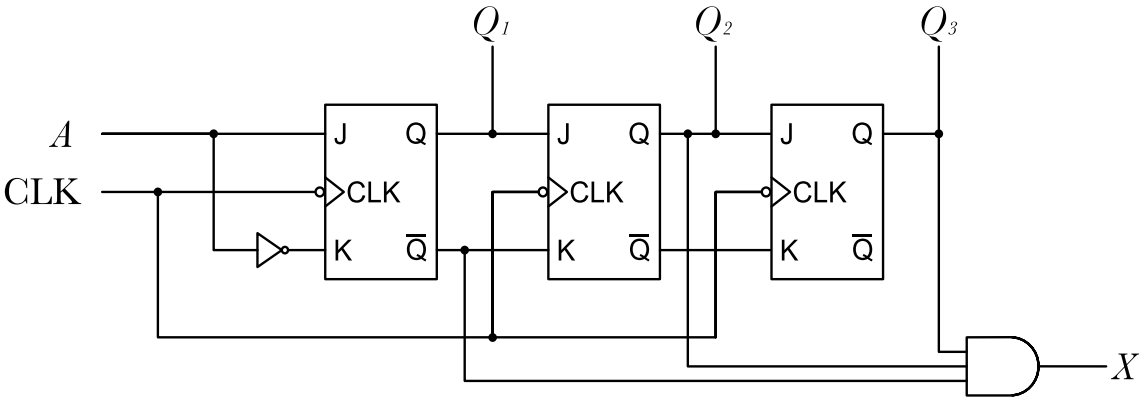

図 11　3 ビットシフトレジスタの論理回路

4.3　カウンタの作成

カウンタはクロック入力の回数を数える順序回路である. 今回は 3 ビットの 8 進カウンタを設計する. 始めに, 現在の内部状態を $(Q_3\ Q_2\ Q_1)_2$, 次の内部状態を $(Q_3^+\ Q_2^+\ Q_1^+)_2$ として状態遷移表を表 4 に示す.

次に, この表から Q_3, Q_2, Q_1 それぞれを出力する JK-FF の入力の論理式を導く. 各フリップフロップは, 自身より下位のビットが全て 1 になった時に反転すれば良いので, 下位ビット全ての論理積を取り, JK-FF の両方の入力端子に入力[*1]するように設計する. Q_3, Q_2, Q_1 それぞれの JK-FF に対する入力を (J_3, K_3), (J_2, K_2), (J_1, K_1) として求めた論理式をまとめると,

$$J_1 = K_1 = 1$$
$$J_2 = K_2 = Q_1$$
$$J_3 = K_3 = Q_1 Q_2$$

となる.

この式をもとに作成した回路とタイミングチャートを図 12, 13 に示す.

[*1] 4.1 より, J, K 両方に 1 を入力をすると反転する

表 4　8 進カウンタの状態遷移表

Q_3	Q_2	Q_1	Q_3^+	Q_2^+	Q_1^+
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

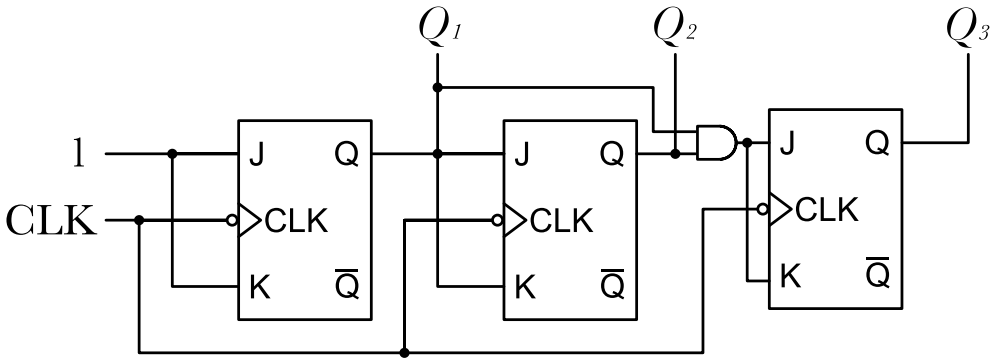

図 12　8 進カウンタの論理回路

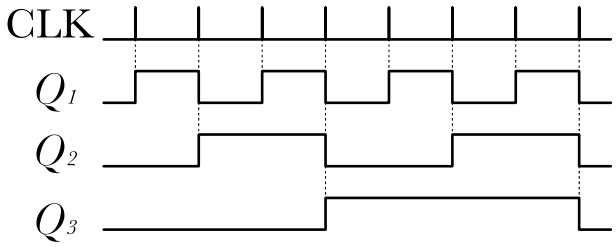

図 13　8 進カウンタのタイミングチャート

5　調査課題「SRAM について」

SRAM(Static Random Access Memory) は, コンピュータ内で CPU と主記憶の間にあるキャッシュメモリに使われている. RAM とは, 読み書き可能だがコンピュータの電源を切るとデータが消えてしまうメモリであるが, 反対に書き込み不可能だが電源を切っていてもデータを保持できる ROM(Read Only Memory) もある.

また, SRAM と似た装置として DRAM(Dynamic Random Access Memory) があり, これも SRAM と同様にキャッシュメモリに使用されている.

特に SRAM は自然放電によるデータ損失がないので, コンピュータの高速処理を実現している.

この 2 つのそれぞれの特徴を表 5 にまとめる.

表 5　SRAM と DRAM

	SRAM	DRAM
動作速度	速い	遅い
消費電力	小さい	大きい
容量	大きくしづらい	大きくしやすい
コスト	高い	安い

この表からわかるように, 性能は SRAM の方が高い. さらに, 上の表に加えて, DRAM は SRAM に比べて定期的に電流を流さないとデータが消えてしまうので, 処理に遅延が発生する. 性能だけを見ると SRAM が高いが, コストも高いのでキャッシュメモリの中でも特に重要な部分を SRAM, そうでない部分を DRAM と使い分けている.

6　感想 · 意見

今回の実験では, 動作をもとに回路を設計することを中心に行った. 今までの授業などでは, 真理値表とカルノー図などから論理的に式を起こして回路を実装していたが, 今回の回路の設計を通して直感的に回路を組むことを覚えた.

今回学んだ内容を次回以降の実験や, 今後の授業に生かしていきたい.

参考文献

1. 令和元年度電子制御工学実験 · 3 年後期テキスト
2. わわわ IT 用語辞典 「SRAM」, https://wa3.i-3-i.info/word17052.html, 2019/12/12
3. IT ポスト.com 「メモリとは、RAM と ROM の違い」,
   http://xn–l8jcc0l6q2czg5j.com/2017/10/31/post-1263/, 2019/12/12