# 知能機械工学実験

題名　8. 論理回路

学籍番号　1610097

氏名　宇治 大智

実験日　平成 30 年 6 月 22 日

平成 30 年 6 月 29 日

# 8. 論理回路

1610097　宇治 大智

平成 30 年 6 月 22 日
平成 30 年 6 月 29 日

## 1　目的・方法

### 1.1　目的

我々は様々な現象を条件分岐で直し，定量化している．その値を計算機の処理できる値とし，計算機が人間を超える速度での演算を行っている．その処理についての基礎とそれを担う回路の使用法，設計法，動作について理解することが本実験の目的である..

### 1.2　方法

基本的な方法として，ブレッドボードに回路を実装し，通電後にどのような結果が出力されるかを調べた．
課題ごとの回路は必要な集積回路を用いて組み立てた．それぞれの素子は，次の集積回路を用いた．

- NAND 素子：74HC00
- AND 素子：74HC08
- ExOR 素子：74HC86
- J-K フリップフロップ：74HC112
- D フリップフロップ：74FC74

## 2　実験項目

次の実験を実施し，課題に関しては各自で取り組んだ．

### 2.1　ゲート回路（6 種類）のまとめ

巻末に付したゲート回路のまとめに関する資料とその中の表 4.1 に主なことを記した．

#### 2.1.1　理論

ゲート回路の機能は，基本的に集合によって体系化されている論理を物理的に，物質的に記述するための素子である．それぞれの素子と集合の対比をここでする．

インバータ素子は，not 素子とも呼ばれるように，入力に対してその否定を返す．集合では，ある集合 A に対し，その否定を補集合と呼び，$\overline{\mathrm{A}}$ と記述する．$\overline{\mathrm{A}}$ は A が含まれる全体集合 U を考えたときに U − A に対応している．ベン図で記すと図 1 で表せる．黒い部分が今回示したい集合である．

AND 素子は，2 つの入力が等しいときのみ 1 を出力する素子である．集合では，ある 2 つの集合 A，B が重なり合うところを A かつ B と言ったり，A∩B と表し，AND や論理積と呼ばれている．これは，A にも B にも存在しているという意味で，素子と同じ働きをしている．ベン図で記すと図 2 で表せる．黒い部分が今回示したい集合である．

OR 素子は，2 つの入力のどちらか一方に 1 があるときに 1 を出力する素子である．集合では，ある 2 つの集合 A，B がある時にそれらの和集合，つまり，少なくともどちらか一方に属している部分を指す．これを A または B と言ったり，A∪B と表し，OR や論理和と呼ばれている．これは，A か B の少なくともどちらか一方に存在しているという意味で，素子と同じ働きをしている．ベン図で示すと図 3 で表せる．黒い部分が今回示したい集合である．

NAND 素子は，2 つの入力のどちらか一方に 0 があるときに 1 を出力する素子である．集合では，ある 2 つの集合 A，B があるときにそれらの補集合が少なくとも 1 つ含まれている部分の集合であるので，全体集合 U から A，B の積集合を除いた部分，つまり，積集合の補集合である．否定論理積と呼ばれる．それは，$\overline{\mathrm{A}\cap\mathrm{B}} = U - (A \cap B)$ と表される．ベン図で示すと図 4 となる．黒い部分が今回示したい集合である．

NOR 素子は，2 つの入力が共に 0 であるときに 1 を出力する素子である．集合では，ある 2 つの集合 A，B があるときにそれらの補集合の積集合を指す．つまり，全体集合 U があるときに集合 A，B の和集合を除いた部分である．否定論理和と呼ばれる．それは，$\overline{\mathrm{A}}\cap\overline{\mathrm{B}} = U - (A \cup B)$ と表される．ベン図で示すと図 5 となる．黒い部分が今回示したい集合である．

ExOR 素子は，2 つの入力が異なるときに 1 を出力する素子である．集合では，ある 2 つの集合 A，B があるときに，A，B の和集合から積集合を除いた部分である．排他的論理和と呼ばれる．それは $(A \cup B) - (A \cap B)$ となる．ベン図で示すと図 6 となる．黒い部分が今回示したい集合である．

## 2.2　2 入力 ExOR ゲート

図 7 で示される回路を用いて NAND 素子によって ExOR 素子を再現した．回路と真理値表等から動作について考察した．

### 2.2.1　動作表

ExOR 回路は，2.1.1 節で述べたような回路である．それの動作は，次の動作表で示された．（表 1 参照）

### 2.2.2　論理式

図 7 の回路図に記載されている端子の状態から論理式を導いた．（ピン番号：状態）という順序で記した．

$(1:\mathrm{B})$ NAND $(2:\mathrm{A}) \rightarrow (3:\overline{\mathrm{A}\cap\mathrm{B}})$

3 番ピンと 4 番ピンが短絡のため電圧等しい．

$(4:\overline{\mathrm{A}\cap\mathrm{B}})$ NAND $(5:\mathrm{A}) \rightarrow (6:\overline{\overline{\mathrm{A}\cap\mathrm{B}}\cap\mathrm{A}})$

表 1　2 入力 ExOR ゲート動作表

| | 入力 | | 出力 |
|---|---|---|---|
| 接続端子 | S0 | S1 | L0 |
| 端子名 | A | B | Y |
| 電圧 | L | L | L |
| | L | H | H |
| | H | L | H |
| | H | H | L |

3 番ピンと 12 番ピンが短絡のため電圧等しい．

$(12 : \overline{\mathrm{A} \cap \mathrm{B}})$ NAND $(13 : \mathrm{B}) \rightarrow (11 : \overline{\overline{\mathrm{A} \cap \mathrm{B}} \cap \mathrm{B}})$

6 番ピンと 9 番ピンが短絡のため電圧等しい．

11 番ピンと 10 番ピンが短絡のため電圧等しい．

$(9 : \overline{\overline{\mathrm{A} \cap \mathrm{B}} \cap \mathrm{A}})$ NAND $(10 : \overline{\overline{\mathrm{A} \cap \mathrm{B}} \cap \mathrm{B}}) \rightarrow (8 : \overline{\overline{\overline{\mathrm{A} \cap \mathrm{B}} \cap \mathrm{A}} \cap \overline{\overline{\mathrm{A} \cap \mathrm{B}} \cap \mathrm{B}}})$

つまり，出力 Y は，8 番ピンの出力と等しいため，

$$
\begin{aligned}
Y &= 8 \text{ 番ピンの出力} \\
Y &= \overline{\overline{\mathrm{A} \cdot \mathrm{B}} \cdot \mathrm{A}} \cdot \overline{\overline{\mathrm{A} \cdot \mathrm{B}} \cdot \mathrm{B}} \\
&= \overline{\overline{A \cdot B} \cdot A} + \overline{\overline{A \cdot B} \cdot B} \\
&= (\overline{A} + \overline{B}) \cdot A + (\overline{A} + \overline{B}) \cdot B \\
&= A \cdot \overline{B} + \overline{A} \cdot B \quad \left( A \cdot \overline{A} = B \cdot \overline{B} = 0 \right) \\
&= A \oplus B
\end{aligned}
$$

となった．

### 2.2.3　ExOR 機能の真理値表

ExOR 機能の真理値表について述べる．次の表 2 で真理値表を示した．

表 2　2 入力 ExOR ゲート真理値表

| | 入力 | | 出力 |
|---|---|---|---|
| 端子名 | A | B | Y |
| 真理値 | 0 | 0 | 0 |
| | 0 | 1 | 1 |
| | 1 | 0 | 1 |
| | 1 | 1 | 0 |

#### 2.2.4 考察

今回の実験によって，図 7 の回路図が動作するときに，求めている ExOR の挙動を示すことがわかった．これは，2.2.2 節で示した様な論理式で示されるからである．論理式にあるように，$Y = A \cdot \overline{B} + \overline{A} \cdot B = \mathrm{A} \oplus \mathrm{B}$ とあり，ある一方の入力ともう一方の入力の否定の論理積を取ることにより，自分自身ともう一方の入力が異なる論理を持っているときにのみ 1 が出力されるようになっている．このために，今回の実験と理論的な動作が一致していてその説明もつくとわかる．

上記にもあるように，この回路の機能は，2 つの入力が異なるときにのみ 1 を返す．そのため，2 つの観察対象があり，その観察対象が互いに異なっていることを取得したいときにこの回路を用いることができる．これは，メモリのアドレス記憶とその取得に使われており，一般にアドレス一致検出回路として広く用いられている．

### 2.3 課題 1

課題内容は，ExOR 回路を実験とは別の回路（図 8 参照）で書き換え，それが ExOR 回路として機能するかを考察することである．今回用いた素子は，AND 素子と負論理入力の OR 素子を用いた．

次のように書き換えた．負論理入力の OR 素子の入力に 2 つの初期入力 A，B を入力した．負論理入力の OR 素子の出力を一つの入力とし，もう一方を初期入力とする NAND 素子を 2 つ組み込んだ．NAND 素子が 2 つであるのは，初期入力が A，B と 2 つあるためだ．それらの 2 つの NAND 素子の出力を負論理入力の OR 素子に入力した．以上のように新たに回路を書き換えた．

負論理入力の素子の場合，入力自体が正論理から見ると反転しているので，$\overline{\mathrm{A}}$ のように”バー”を付して表した．NAND 回路は，出力を負論理と考え，$\overline{\mathrm{D}}$ のように”バー”を付して表した．

$$\mathrm{C} = \overline{\mathrm{A}} + \overline{\mathrm{B}} \tag{1}$$

$$\begin{aligned}
\overline{\mathrm{D}} &= \mathrm{A} \cdot \mathrm{C} && (2)\\
&= \mathrm{A} \cdot (\overline{\mathrm{A}} + \overline{\mathrm{B}}) && (3)\\
&= \mathrm{A} \cdot \overline{\mathrm{A}} + \mathrm{A} \cdot \overline{\mathrm{B}} && (4)\\
&= \mathrm{A} \cdot \overline{\mathrm{B}} \quad \left( \mathrm{A} \cdot \overline{\mathrm{A}} = 0 \right) && (5)
\end{aligned}$$

$$\begin{aligned}
\overline{\mathrm{E}} &= \mathrm{B} \cdot \mathrm{C} && (6)\\
&= \mathrm{B} \cdot (\overline{\mathrm{A}} + \overline{\mathrm{B}}) && (7)\\
&= \mathrm{B} \cdot \overline{\mathrm{A}} + \mathrm{B} \cdot \overline{\mathrm{B}} && (8)\\
&= \mathrm{B} \cdot \overline{\mathrm{A}} && (9)\\
&= \overline{\mathrm{A}} \cdot \mathrm{B} \quad \left( \mathrm{B} \cdot \overline{\mathrm{B}} = 0 \right) && (10)
\end{aligned}$$

$$\begin{aligned}
Y &= \overline{\mathrm{D}} + \overline{\mathrm{E}} && (11)\\
&= A \cdot \overline{B} + \overline{A} \cdot B && (12)\\
&= A \oplus B && (13)
\end{aligned}$$

以上の論理式は，次のように変形された．

(1) 式で，C の出力を論理式で示した．
(2) 式で，2 つの入力 A，C の論理積を取っていることを論理式で示した．
(3) 式で，(1) 式を C に代入し，入力 A，B のみによって記述した．
(4) 式で，分配法則を用いて，論理積を論理和に書き換えた．
(5) 式で，論理として 0 となる部分を消去した．
(6) 式で，2 つの入力 B，C の論理積を取っていることを論理式で示した．
(7) 式で，(1) 式を C に代入し，入力 A，B のみによって記述した．
(8) 式で，分配法則を用いて，論理積を論理和に書き換えた．
(9) 式で，論理として 0 となる部分を消去した．
(10) 式で，次の過程で見やすくなるように，積の順序を入れ替えた．
(11) 式で，2 つの入力 $\overline{\mathrm{D}}$，$\overline{\mathrm{E}}$ の論理和を取っていることを論理式で示した．
(12) 式で，(5) 式と (10) 式を代入し，入力 A，B のみで表されるようにした．
(13) 式で，排他的論理和として書ける部分を排他的論理和で記述した．

以上のように変形することで，論理式の上で，図の回路図で示された回路は，ExOR 回路と同等の機能を有するとわかる．

## 2.4 デコーダとエンコーダ

我々は普段思考するときに 10 進数を用いて考える．これは，指が 10 本あることに由来すると考えられている．しかし，計算機も同様に考えることが適切であるかと言われるとそうではない．計算機の場合，スイッチの ON，OFF や，論理の 0，1 に対応する 2 進数を用いている．この 2 つの変換がうまくしなければ，人間の考えを計算機に伝えることはできないし，計算機の計算結果を人間が読み取る事もできない．その変換を担うのがデコーダとエンコーダである．つまり，2 進数を 10 進数に変換する，10 進数を 2 進数に変換することの機能を担っている．

計算機が用いる言語であるコードから人間の言語にすることとして，コードを脱するということから，「排除」や「脱すること」を示す接頭辞 de と，「それを実行するもの」を示す接尾辞 er を用いて，de/code/er（デコーダ）と言われている．ここでは，2 進数の入力（S1,S0）を 10 進数を示す 4 つのパターン（L0,L1,L2,L3）に変換することを表している．

反対に，人間の言語から計算機が用いる言語であるコードに変換することとして，コードにするということから，「それをなす」や動詞化を示す接頭辞 en と，「それを実行するもの」を示す接尾辞 er を用いて，en/code/er（エンコーダ）と言われている．ここでは，10 進数を示す 4 つのパターン（S0,S1,S2,S3）の入力を 2 進数の数字に表すことを指している．

実験では，このようなデコーダの機能について確認し，理論との関係を考察した．課題ではエンコーダの構造と動作について考察した．今回次の図 9 で示されている回路を用いて実験した．

### 2.4.1 動作表

2 入力 4 出力デコーダ回路の動作は，次の動作表で示された通りになった．（表 3 参照）
結果として，

- S1 と S0 の両方が OFF のとき，L0 が点灯

表 3　デコーダの動作表

| | 入力 | | 出力 | | | |
|---|---|---|---|---|---|---|
| 接続端子 | S1 | S0 | L0 | L1 | L2 | L3 |
| 電圧 | L | L | H | L | L | L |
| | L | H | L | H | L | L |
| | H | L | L | L | H | L |
| | H | H | L | L | L | H |

- S1 が OFF で S0 が ON のとき，L1 が点灯
- S1 が ON で S1 が OFF のとき，L2 が点灯
- S1 と S0 の両方が ON のとき，L3 が点灯

のようになった．S1 が 2 進数の $2^1$ の桁に対応し，S0 が 2 進数の $2^0$ の桁に対応した．また，L0 から L4 は，それぞれの 4 つのパターンを示した．

### 2.4.2　真理値表

2 入力 4 出力デコーダ回路の真理値は，次の真理値表で示された通りになった．（表 4 参照）

表 4　デコーダの真理値表

| | 入力 | | 出力 | | | |
|---|---|---|---|---|---|---|
| 接続端子 | S1 | S0 | L0 | L1 | L2 | L3 |
| 真理値 | 0 | 0 | 1 | 0 | 0 | 0 |
| | 0 | 1 | 0 | 1 | 0 | 0 |
| | 1 | 0 | 0 | 0 | 1 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 1 |

### 2.4.3　論理式

図 4 の真理値表を参考にし，次の論理式を導いた．

$$\mathrm{L0} = \overline{\mathrm{S1}} \cdot \overline{\mathrm{S0}}$$
$$\mathrm{L1} = \mathrm{S1} \cdot \overline{\mathrm{S0}}$$
$$\mathrm{L2} = \overline{\mathrm{S1}} \cdot \mathrm{S0}$$
$$\mathrm{L3} = \mathrm{S1} \cdot \mathrm{S0}$$

L0: それぞれの否定の論理積
L1: S1 の否定と S0 の論理積
L2: S1 と S0 の否定の論理積
L3: それぞれの論理積

で表された．

### 2.4.4　考察

2 桁の 2 進数では，10 進数を示す 4 つのパターン（数値）だけを表すことでできる. 具体的な数値としては 0 から 3 までの 4 つの数値であるが，今回は，数値を LED の点灯によって表示している．LED の点灯に 10 進数の最初の 4 つの数値を意味づけしている．この 10 進数を示す 4 つの LED のパターンは，図 4 で示されている入力によって作られる 4 つのパターンで構成されており，それぞれの 0，1 のパターン（機械的にスイッチの ON，OFF で操作される）が 2 種類ずつであるため，$2 \times 2 = 4$ となることから 2 桁の 2 進数では，10 進数を表す場合，10 進数を示す 4 つのパターン（数値）だけを表すことでできるとわかる．

論理積は，2 つの入力が互いに 1 でなければ 1 が出力されない．そのために 1 つずつパターンを抜き出すのには適している．表 4 より，これらのパターンは，表の行に対応しており，入力の組み合わせでどのように抜き出すべきかが見て取れる．例えば，1 行目を抜き出したければ，2 つの入力のそれぞれの否定の論理積を取ることで，出力が 1 となる．（この機能は，負論理入力の AND 素子であり，NOR 素子も同様の機能を有する．） 表 4 で述べた組み合わせで論理積を用いることで，全パターン抜き出す事ができる．つまり，計算機の言語である 2 進数を人間が読みやすい 10 進数に変換することができているので，解読できていると考えられる．

デコーダは，計算機から人間に対するインターフェイスで用いられており，ディスプレイ表示やプリンタ（RGB 値変換）なども広義でデコーダの役割に当たると考えられる．それぞれは，文書にしたり，視覚的に手助けしたり，と役目は異なるものの役割としての根本の部分はどれも等しい．

## 2.5　課題 2

### 2.5.1　真理値表

10 進数を示す 4 つのパターンから 2 進数に変換するのが課題 2 である．まず，10 進数を示す 4 つのパターンそれぞれを入力することによってどのような 2 進数が出てくるべきかを真理値表にまとめた．（表 5 参照）

表 5　エンコーダ真理値表

| | 入力 | | | | 出力 | |
|---|---|---|---|---|---|---|
| 接続端子 | S0 | S1 | S2 | S3 | L1 | L0 |
| 真理値 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 1 | 0 | 0 | 0 | 1 |
| | 0 | 0 | 1 | 0 | 1 | 0 |
| | 0 | 0 | 0 | 1 | 1 | 1 |

これは，それぞれの 10 進数を示す 4 つのパターンの添え字の小さい順に（S0,S1,S2,S3 の順に）2 進数の小さい数字に合わせていったものである．

### 2.5.2　論理式

次に真理値表から論理式を導く．次の条件を満たすような論理式を立式すれば良い．

- L0 も L1 も，少なくとも，S0 が 0 のときに 1 を持つ．

- L0 は，少なくとも，S2 が 0 のときに 1 を持つ．
- L0 は，S1，もしくは，S3 が 1 のときに 1 を持つ．
- L1 は，少なくとも，S1 が 0 のときに 1 を持つ．
- L1 は，S2，もしくは，S3 が 1 のときに 1 を持つ．

このときに，条件を出力毎に分解すると，

L0 ：（ S0 が 0 ）かつ（ S2 が 0 ）かつ（ S1，もしくは，S3 が 1 ）
　＝（ S0 が 0 ）AND（ S1 が 0 ）AND（ S1，OR，S3 が 1 ）
L1 ：（ S0 が 0 ）かつ（ S1 が 0 ）かつ（ S2，もしくは，S3 が 1 ）
　＝（ S0 が 0 ）AND（ S2 が 0 ）AND（ S2，OR，S3 が 1 ）

となるため，これを論理式の形で記述すると，

$$\mathrm{L0} = \overline{\mathrm{S0}} \cdot \overline{\mathrm{S2}} \cdot (\mathrm{S1} + \mathrm{S3})$$
$$\mathrm{L1} = \overline{\mathrm{S0}} \cdot \overline{\mathrm{S1}} \cdot (\mathrm{S2} + \mathrm{S3})$$

となる．ここで，10 進数を示す 4 つのパターンのうち一つだけしか選択できないとする．この時，入力値が一意に決められるので，入力値が 0 であることを参照する必要がなくなる．したがって，論理式は次のように書ける．

$$\mathrm{L0} = \mathrm{S1} + \mathrm{S3}$$
$$\mathrm{L1} = \mathrm{S2} + \mathrm{S3}$$

#### 2.5.3　回路図

以上の議論で求めた論理式を論理回路で表す．論理積を AND 素子で，論理和を OR 素子で，否定を NOT 素子で置き換えると，10 進数を示す 4 パターンの内，多数個選択できる場合，図 10 の回路で示すことができる．10 進数を示す 4 パターンの内，1 つしか選択できない場合は，図 11 の回路で示すことができる．

以上で，4 入力 2 出力のエンコーダについて，真理値表から回路図を考えることができたと考えられる．

## 2.6　加算回路

加算回路は論理回路の中で，2 進数として扱われているデータの演算を行う回路の一つであり，加算を行っている．加算回路には主に 2 種類あり，半加算器と全加算器がある．半加算器は，1 桁同士の 2 進数の加算であり，桁上げの数を加算することはない．加算の結果，桁上げとして値を出力することはある．1 桁同士であるので，そんなに複雑にはならない．全加算器のプロセスは，2 桁以上 2 進数が含まれる加算の中に含まれており，桁上げの数も加算の入力として扱う．つまり，入力が 3 つ（足す数，足される数，下の位からの桁上げの数）となったときの加算器のことである．

## 2.7　半加算器

今回の実験で実際に回路を組み，動作させた半加算器について述べる．

### 2.7.1　半加算器の機能説明

2.6 節で述べた半加算器について機能の詳細について述べる．入力は 2 つで，A と B である．それに対し，出力も 2 つであり，S と C である．S は，A，B の和の一桁目である．C は，桁上げの数である．要するに筆算式としては，左のようになる．また，加算結果を箇条書きでまとめると右のようになる．

$$
\begin{array}{cccc}
 & & \mathrm{C} & \\
 & & \downarrow & \mathrm{A} \\
+ & & \downarrow & \mathrm{B} \\
\hline
 & & \mathrm{C} & \mathrm{S}
\end{array}
$$

| | 和 | 桁上げ |
|---|---|---|
| $0+0=0$ | | なし |
| $0+1=0$ | | なし |
| $1+0=1$ | | なし |
| $1+1=0$ | | あり |

### 2.7.2　半加算器の真理値表

2.7.1 節でまとめた半加算器の和と桁上げの数の関係を入力と出力の真理値表にまとめると次のようになった．（表 6 参照 ）

表 6　半加算器真理値表

| | 入力 | | 出力 | |
|---|---|---|---|---|
| | | | 和 | 桁上げ |
| 接続端子 | A | B | S | C |
| 真理値 | 0 | 0 | 0 | 0 |
| | 0 | 1 | 1 | 0 |
| | 1 | 0 | 1 | 0 |
| | 1 | 1 | 0 | 1 |

桁上げのある，なしは，1，0 によって表した．

### 2.7.3　半加算器の論理式

2.7.2 節でまとめた真理値表より，半加算器の論理式をまとめた．論理式は，次の条件を満たすような式でなければならないとわかった．

- 和は，入力の値が互いに異なるときに 1 となること．
- 桁上げは，入力の両方が同時に 1 となるときに 1 となること．

これをもう一度書き換えると，

- 和は，（（A = 1）かつ（B = 0））もしくは（（A = 0）かつ（B = 1））のときに 1
- 桁上げは，（A = 1）かつ（B = 1）のときに 1

となった．したがって，これを論理式の形で記述すると，

$$
\begin{aligned}
S &= A \cdot \overline{B} + \overline{A} \cdot B \\
&= A \oplus B \\
C &= A \cdot B
\end{aligned}
$$

となった．

### 2.7.4　半加算器の回路図

2.7.3 節でまとめた論理式を回路図にすると次のようになった．（図 12 参照 ）S を出力するための直和を ExOR 回路を用いて，C を出力するための論理積を AND 素子を用いて設計した．

### 2.7.5　半加算器の動作表

図 12 で示された回路を実装し，動作させたときに示した動作を表にまとめると次のようになった．（表 7 参照）

表 7　半加算器動作表

| | 入力 | | 出力 | |
|---|---|---|---|---|
| | | | 和 | 桁上げ |
| 接続端子 | $S_0$ | $S_1$ | $L_0$ | $L_1$ |
| 端子名 | A | B | S | C |
| 電圧 | L | L | L | L |
| | L | H | H | L |
| | H | L | H | L |
| | H | H | L | H |

### 2.7.6　考察

実験では，事前に準備した真理値表と実験で得られた動作表を比較したときに，真理値表で示したのと同様の動作が実装した回路で確認された．つまり，今回の真理値表から論理式と回路図の設計は間違いなく書き換えが行われていると考えられる．半加算器としての性能をきちんと有している．これを応用することにより，全加算器を設計することが可能であり，全加算器が設計できれば，計算機上で 2 進数の計算をこなすことができるとわかる．

## 2.8　全加算器と 2 進数の加算（課題 3）

今回の課題で真理値表から回路の設計までを行った全加算器について述べる．

### 2.8.1　機能説明

2.6 節で述べた全加算器について機能の詳細について述べる．入力は 3 つで，A と B と $C_1$ である．それに対し，出力は 2 つであり，S と $C_2$ である．$C_1$ は，一つ前の桁から繰り上げられた数である．S は，A，B の

和の一桁目である．$C_2$ は，繰り上げた数である．これを半加算器と組み合わせて，2 桁の 2 進数の計算を考えた時，筆算式は左のようになる．また，加算結果を箇条書きでまとめると右のようになる．

$$\begin{array}{cccc} & C_2 & C_1 & \\ & \downarrow & A_1 & A_0 \\ + & \downarrow & B_1 & B_0 \\ \hline & C_2 & S_1 & S_0 \end{array}$$

1 桁目の加算（半加算器）
　入力：$A_0$ と $B_0$，出力：和 $S_0$ と桁上げ $C_1$
2 桁目の加算（全加算器）
　入力：$A_1$ と $B_1$ と $C_1$，出力：和 $S_1$ と桁上げ $C_2$

### 2.8.2　全加算器の真理値表

2.7.1 節でまとめた全加算器の和と繰り上げ数の関係を入力と出力の真理値表にまとめると次のようになった．（表 8 参照 ）

表 8　全加算器真理値表

| | 入力 | | | 出力 | |
|---|---|---|---|---|---|
| | | | | 和 | 桁上げ |
| 端子名 | A | B | $C_{in}$ | S | $C_{out}$ |
| 真理値 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 1 | 0 | 1 | 0 |
| | 1 | 0 | 0 | 1 | 0 |
| | 1 | 1 | 0 | 0 | 1 |
| | 0 | 0 | 1 | 1 | 0 |
| | 0 | 1 | 1 | 0 | 1 |
| | 1 | 0 | 1 | 0 | 1 |
| | 1 | 1 | 1 | 1 | 1 |

### 2.8.3　全加算器の論理式

2.8.2 節でまとめた真理値表より，全加算器の論理式をまとめた．論理式は，次の条件を満たすような式でなければならないとわかった．

- $C_{in}$ が 0 のとき，和は，入力の値が互いに異なるときに 1 となること．
- $C_{in}$ が 1 のとき，和は，入力の値が等しくなるときに 1 となること．
- $C_{in}$ が 0 のとき，桁上げは，入力の値が両方 1 になるときに 1 となること．
- $C_{in}$ が 1 のとき，桁上げは，2 つの入力の内，少なくとも一方の状態が 1 であるときに 1 になること．

これをもう一度書き換えると，

- $C_{in} = 0$ ならば，和は，$((A = 1)$ かつ $(B = 0))$ または $((A = 0)$ かつ $(B = 1))$ のときに 1
- $C_{in} = 1$ ならば，和は，$((A = 0)$ かつ $(B = 0))$ または $((A = 1)$ かつ $(B = 1))$ のときに 1
- $C_{in} = 0$ ならば，桁上げは，$(A = 1)$ かつ $(B = 1)$ のときに 1

- $\mathrm{C_{in}} = 1$ ならば，桁上げは，（$A = 1$）または（$B = 1$）のときに 1

となった．和に関しては，A と B の ExOR と $\mathrm{C_{in}}$ の ExOR を取ると一つにまとめられることが，上のまとめた条件から読み取れる．桁上げに関しては，$\mathrm{C_{in}}$ の状態によってまとめるため，それぞれの条件を求めて論理和で足し上げることを考えた．したがって，これを論理式の形で記述すると，

$$
\begin{aligned}
\mathrm{S} &= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{\mathrm{A}} \cdot \overline{\mathrm{B}} + \mathrm{A} \cdot \mathrm{B}) && (14)\\
&= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{\overline{\overline{\mathrm{A}} \cdot \overline{\mathrm{B}}} \cdot \overline{\mathrm{A} \cdot \mathrm{B}}}) && (15)\\
&= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{(\overline{\overline{\mathrm{A}} + \overline{\mathrm{B}}}) \cdot (\overline{\mathrm{A}} + \overline{\mathrm{B}})}) && (16)\\
&= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{(\mathrm{A} + \mathrm{B}) \cdot (\overline{\mathrm{A}} + \overline{\mathrm{B}})}) && (17)\\
&= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{\mathrm{A} \cdot \overline{\mathrm{A}} + \mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B} + \mathrm{B} \cdot \overline{\mathrm{B}}}) && (18)\\
&= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}}) \quad \left( \mathrm{A} \cdot \overline{\mathrm{A}} = \mathrm{B} \cdot \overline{\mathrm{B}} = 0 \right) && (19)\\
&= \overline{\mathrm{C_{in}}} \cdot (\mathrm{A} \oplus \mathrm{B}) + \mathrm{C_{in}} \cdot (\overline{\mathrm{A} \oplus \mathrm{B}}) && (20)\\
&= (\mathrm{A} \oplus \mathrm{B}) \oplus \mathrm{C_{in}} && (21)
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{C_{out}} &= \mathrm{A} \cdot \mathrm{B} \cdot \overline{\mathrm{C_{in}}} + \overline{\mathrm{A}} \cdot \mathrm{B} \cdot \mathrm{C_{in}} + \mathrm{A} \cdot \overline{\mathrm{B}} \cdot \mathrm{C_{in}} + \mathrm{A} \cdot \mathrm{B} \cdot \mathrm{C_{in}} && (22)\\
&= (\mathrm{A} \cdot \overline{\mathrm{B}} + \overline{\mathrm{A}} \cdot \mathrm{B}) \cdot \mathrm{C_{in}} + \mathrm{A} \cdot \mathrm{B} \cdot (\mathrm{C_{in}} + \overline{\mathrm{C_{in}}}) \qquad (\mathrm{C_{in}} + \overline{\mathrm{C_{in}}} = 1) && (23)\\
&= (\mathrm{A} \oplus \mathrm{B}) \cdot \mathrm{C_{in}} + (\mathrm{A} \cdot \mathrm{B}) && (24)
\end{aligned}
$$

となった．次のように和を表す論理式の変形を行った．

(14) 式で，$\mathrm{C_{in}}$ の状態による条件を論理式の形で書き下した．

(15) 式で，$\mathrm{C_{in}} = 1$ のときの入力 A，B による部分をド・モルガン則によって論理和から否定の論理積の否定に書き換えた．こうすることで，論理和で表されていたものを意味を変えずに，形のみ変える事ができる．

(16) 式で，$\mathrm{C_{in}} = 1$ のときの入力 A，B による部分をド・モルガン則によって，否定の論理積から論理和の否定に，もしくは，論理積の否定から否定の論理和に書き換えた．

(17) 式で，否定の否定を解消した．

(18) 式で，論理積を分配法則により論理和に書き換えた．

(19) 式で，論理として 0 となる部分を消去した．

(20) 式で，入力 A，B において，排他的論理和として括れる部分を書き換えた．

(21) 式で，入力 $\mathrm{C_{in}}$ を含めた論理式全体で排他的論理和として括れる部分を書き換えた．

このように変形することで，全加算器の和は，(21) 式で表せるとわかる．次に桁上げを示す論理式の変形について見る．

(22) 式で，表 8 より，出力の桁上げが 1 となる条件を足し上げた．そうすることで，桁上げが 1 となる状態を全て書き上げることができた．

(23) 式で，入力 A，B が関わる演算として，論理和の演算と排他的論理和の演算に分け，括りあげた．

(24) 式で，排他的論理和として括れる部分を書き換え，論理として 1 になる部分を省略した．

このように変形することで，全加算器の和は，(24) 式で表せるとわかる．

### 2.8.4　全加算器の回路図

2.8.3 節でまとめた論理式を回路図にすると次のようになった．（図 12 参照 ）直和を ExOR 回路を用いて，論理積を AND 素子を用いて設計した．

### 2.8.5　2 桁の 2 進数の加算回路の真理値表

2.8.1 節でまとめた 2 桁の 2 進数の加算器の和と繰り上げ数の関係を入力と出力の真理値表にまとめると次のようになった．（表 9 参照 ）

表 9　全加算器真理値表

| | 入力 | | | | 出力 | | |
|---|---|---|---|---|---|---|---|
| | | | | | 3 桁目 | 2 桁目 | 1 桁目 |
| 端子名 | $A_1$ | $A_0$ | $B_1$ | $B_0$ | $C_2$ | $S_1$ | $S_0$ |
| 真理値 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

### 2.8.6　2 桁の 2 進数の加算回路の論理式

2.8.5 節でまとめた真理値表と半加算器，全加算器の設計で用いた考え方より，2 桁の 2 進数の加算器の論理式をまとめた．論理式は，次の条件を満たすような式でなければならないとわかった．

- 1 桁目の和は，1 桁目の入力値の排他的論理和であること．
- 1 桁目の桁上げは，1 桁目の入力値の論理積であること．
- 2 桁目の和は，2 桁目の入力値の排他的論理和と 1 桁目の桁上げ値の排他的論理和であること．
- $C_1$ が 0 のとき，2 桁目の桁上げは，2 桁目の入力値が両方 1 になるときに 1 となること．
- $C_1$ が 1 のとき，2 桁目の桁上げは，2 桁目の入力値が互いに異なるときに 1 になること．

したがって，これを論理式の形で記述すると，

$$
\begin{aligned}
S_0 &= A_0 \oplus B_0 \\
S_0 &= (A_1 \oplus B_1) \oplus (A_0 \cdot B_0) \\
C_2 &= (A_1 \oplus B_1) \cdot (A_0 \cdot B_0) + (A_1 \cdot B_1)
\end{aligned}
$$

となった．

#### 2.8.7　2 桁の 2 進数の加算回路の回路図

2.8.6 節でまとめた論理式を回路図にすると次のようになった．（図 12 参照 ）S を出力するための直和を ExOR 回路を用いて，C を出力するための論理積を AND 素子を用いて設計した．

## 2.9　ラッチ回路

Latch という言葉は，門や戸の掛けがね，錠という意味で，掴んだり，状態を保持しているという意味がある．それが，コンピュータの用語として，データや信号を保持するという意味を持つようになった．今回は，ラッチ回路の中でも D ラッチ回路の実験を行った．

#### 2.9.1　D ラッチ回路の機能説明

D ラッチ回路は，RS ラッチの前段にデータ記憶用のゲートを持ち，入力されたデータを保持することができる．この機能は，ストローブ入力と呼ばれる信号によって制御される．つまり，RS ラッチにいつデータを保持するかをストローブ入力により制御しているのが D ラッチ回路である．これを実装しているのが図 15 で示された回路である．この回路にデータとストローブ入力を入れて，どのような応答があるかを実験を通し，動作表とタイムチャートを用いて調べた．

#### 2.9.2　D ラッチ回路の動作表

図 15 で示された回路を実装し，動作させたときに示した動作を表にまとめると次のようになった．（表 10 参照）

表 10　D ラッチ回路の動作表

| | 入力 | | 出力 | |
|---|---|---|---|---|
| 接続端子 | $S_1$ | $S_0$ | $L_1$ | $L_0$ |
| 端子名 | Data | $\overline{\text{Stb}}$ | Q | $\overline{Q}$ |
| 電圧 | L | L | $Q_0$ | $\overline{Q_0}$ |
| | H | L | $Q_0$ | $\overline{Q_0}$ |
| | L | H | L | H |
| | H | H | H | L |

### 2.9.3　D ラッチ回路のタイムチャート

図 15 で示された回路を実装し，動作させたときに示した動作をタイムチャートにまとめると次のようになった．（図 16 参照）タイムチャートより，ストローブ入力が正のときには，以前のデータが保持されて，データを入れ替えても出力 Q が変化しないことがわかった．

### 2.9.4　D ラッチ回路の考察 1

ストローブ信号が H のときは，$\overline{\mathrm{Stb}}$ が H のときである．ストローブ信号が H のときに，D ラッチ回路の記憶保持機能が OFF になるため，データの入力が変化した場合，今までの出力とは関係なく，出力が変化し，入力データの値を反映する．したがって，$\overline{\mathrm{Stb}} = \mathrm{H}$ のとき，出力はデータ入力値となる．

以上のことは，動作表からも読み取れる．表 10 より，$\overline{\mathrm{Stb}} = \mathrm{H}$ のとき，出力がデータ入力値と等しいのが読み取れる．

同様に，実験で得たタイムチャートを見ると，$\overline{\mathrm{Stb}} = \mathrm{H}$ のとき，出力 Q の動作は，入力データの値 L と H によって，変化しているのがわかる．（図 16 参照 ）データの入力値を追いかけるように出力が変化している．

したがって，$\overline{\mathrm{Stb}} = \mathrm{H}$ のとき，出力はデータの入力値と等しくなると考えられる．

### 2.9.5　D ラッチ回路の考察 2

ストローブ信号が L のときは，$\overline{\mathrm{Stb}}$ が L のときである．ストローブ信号が L のときに，D ラッチ回路の記憶保持機能が ON になるため，データの入力が変化しても，今までの出力を保持するように回路が機能し，出力が変化することはない．したがって，$\overline{\mathrm{Stb}} = \mathrm{L}$ のとき，出力が必ず $\mathrm{Q}_0$ となる．

以上のことは，動作表からも読み取れる．表 10 より，$\overline{\mathrm{Stb}} = \mathrm{L}$ のとき，出力が必ず $\mathrm{Q}_0$ となっているのが読み取れる．

同様に，実験で得たタイムチャートを見ると，$\overline{\mathrm{Stb}} = \mathrm{L}$ のとき，出力 Q の動作は，入力データの値が L と H にかかわらず，一定になっているのがわかる．（図 16 参照 ）

したがって，$\overline{\mathrm{Stb}} = \mathrm{L}$ のとき，出力が必ず $\mathrm{Q}_0$ となると考えられる．

### 2.9.6　D ラッチ回路の考察 3

データの入力値を変更するのと同時にストローブ信号を変化させると，次のようなことが起こると考えられる．

ストローブ信号を L にした時，出力 Q は，何も変化しないと考えられる．ストローブ信号を L にすることは，記憶保持機能が OFF の状態から ON の状態になることである．記憶保持機能が OFF の状態では，以前の出力は記憶されていないために，記憶保持機能が ON になっても，ただちに，記憶された出力が出力されるわけでない．したがって，ストローブ信号を L にした時，ただちに，出力 Q は，何も変化しないと考えられる．

次に，ストローブ信号を H にした時，出力 Q は，その時のデータの入力値になると考えられる．記憶保持機能を ON から OFF にすることであり，OFF になった状態では，データの入力値が出力に反映されるために，ストローブ信号を H にした時，出力 Q は，その時のデータの入力値になると考えられる．

以上の考察より，D ラッチ回路におけるストローブ信号の機能は，以前の出力を記憶しておき，データの入力値の如何にかかわらず，その値を保持し続けようという記憶保持機能を制御する役割である．ラッチの機能

は，2.9 節で述べた言葉の通り，一つ前の出力の内容をドアの錠のように開けられるまでその状態を変えないように保持し続けようという機能であった考えられる．

## 2.10　J-K フリップフロップ回路

### 2.10.1　J-K フリップフロップ回路の機能説明

J-K フリップフロップ回路の特徴として同期しているときには，一つ前の状態を保持して，それを次の動作に反映させることである．最もよく知られているのが，トグルという機能で，一つ前の出力を保持し，それの否定を次に出力する機能である．この機能のトリガーとなるのが，$\overline{\mathrm{CLK}}$ 入力であり，この入力がダウンするときに，反転した出力が生じる．つまり，$\overline{\mathrm{CLK}}$ のダウンする任意のタイミングで H と L の状態を繰り返す事ができる．

### 2.10.2　J-K フリップフロップ回路の動作表

図 17 で示された回路を実装し，動作させたときに示した動作を表にまとめると次のようになった．（表 11 参照）

表 11　J-K フリップフロップ回路動作表

| | 入力 | | | | | 出力 | |
|---|---|---|---|---|---|---|---|
| 端子名 | $S_0$ | $S_1$ | $S_2$ | $S_3$ | DA | $L_1$ | $L_0$ |
| 端子名 | $\overline{\mathrm{CLR}}$ | $\overline{\mathrm{PR}}$ | J | K | $\overline{\mathrm{CLK}}$ | Q | $\overline{\mathrm{Q}}$ |
| 電圧 | L | H | X | X | X | L | H |
| | H | L | X | X | X | H | L |
| | L | L | X | X | X | H* | H* |
| | H | H | L | L | ↓ | $Q_0$ | $\overline{Q_0}$ |
| | H | H | L | H | ↓ | L | H |
| | H | H | H | L | ↓ | H | L |
| | H | H | H | H | ↓ | $\overline{Q_0}$ | $Q_0$ |
| | H | H | X | X | H | $Q_0$ | $\overline{Q_0}$ |

### 2.10.3　J-K フリップフロップ回路のタイムチャート

図 17 で示された回路を実装し，動作させたときに示した動作をタイムチャートにまとめると次のようになった．（図 18 参照）

### 2.10.4　JK フリップフロップ回路考察前半

前半部では，$\overline{\mathrm{CLR}}$ 入力と $\overline{\mathrm{PR}}$ 入力による応答について見ていく．

表 11 によると，$\overline{\mathrm{CLR}}$ もしくは，$\overline{\mathrm{PR}}$ に L が入力されている時，他の入力 J，K，$\overline{\mathrm{CLK}}$ の入力値にかかわらず，ある定まった値を出力するようになっている．それをタイムチャートと見比べると次のようなことがわかる．（図 18 参照）タイムチャートの前半部が，$\overline{\mathrm{CLR}}$ と $\overline{\mathrm{PR}}$ に関する変化である．タイムチャートで $\overline{\mathrm{CLR}}$ と $\overline{\mathrm{PR}}$ が順番に L に変化するところでは，そのたびに出力値が変化し，ある固定の値になっているのがわかる．

$\overline{\mathrm{CLR}} = H$ で $\overline{\mathrm{PR}} = H$ のときに出力 Q は L とする．このときに $\overline{\mathrm{PR}} = L$ となると，表 11 より Q は H になる．この後に $\overline{\mathrm{PR}} = H$ に戻し，$\overline{\mathrm{CLR}} = L$ とすると，11 より Q は L になる．これは，$\overline{\mathrm{PR}} = L$ のときと $\overline{\mathrm{CLR}} = L$ のときとで，出力が取りうる値が異なるために起こる．しかし，同じ入力が連続して L に変化するときは，最初にある定状態に変化しているために，そのたびに変化は起こらないことがわかる．

したがって，$\overline{\mathrm{CLR}}$ もしくは，$\overline{\mathrm{PR}}$ が L のときにはある定状態（出力値がある値に決まる状態）が続き，一度その定状態になった場合は，同じ入力が何度 L に入ろうとも，定状態にいる限り変化することはない．ただ，一度でももう一方の入力に L が入力されれば，定状態は崩れて，再度変化することになる．また，その時の定状態は，$\overline{\mathrm{CLR}}$ が L のときと $\overline{\mathrm{PR}}$ が L のときで互いに反対の値を取るため，これらが連続して L に入ると，必ず出力は反転するとわかる．

### 2.10.5　JK フリップフロップ回路考察後半

後半部では，$\overline{\mathrm{CLR}}$ と $\overline{\mathrm{PR}}$ の入力値がともに H であるときの，J と K の入力値とその時の $\overline{\mathrm{CLK}}$ 変化に対する出力応答について見ていく．

動作表（表 11）より，J と K の入力値が一致しないときには，出力はある定状態になることがわかる．ただ，その定状態になるトリガーは，J や K の入力値の変化ではなく，$\overline{\mathrm{CLK}}$ の値が立ち下がるときに起こる．つまり，（J，K）が（L，H）であるといったからと言って，ただちに，出力（Q，$\overline{\mathrm{Q}}$）も（L，H）になるというわけではなく，$\overline{\mathrm{CLK}}$ の変化による．ただ，$\overline{\mathrm{CLK}}$ が一度トリガーを引いた後は，（J，K）の組が変化しない限り出力値（Q，$\overline{\mathrm{Q}}$）の組は変化しない．そのため，トリガーを何度引いても，値が変わらないということが起こる．その様子はタイムチャートからも読み取れる．（図 18 参照）

動作表より，J と K の値が一致しているときには，一つ前の値を参照しているのがわかる．このときに，両方 L のときは，一つ前の値をそのまま出力し，両方 H のときに一つ前の否定の値を出力している．後者をトグルという．どちらも $\overline{\mathrm{CLK}}$ の値が下がるときにトリガーが引かれるのであるが，両方 L で一つ前の状態をそのまま保持する場合は，トリガー以前で状態が変わらないので，あまり重要でない．重要なのは後者の方で，トリガーが引かれる毎に一つ前の値の否定が返され値が変わるからである．つまり，その値の変化を $\overline{\mathrm{CLK}}$ の入力値を変化することで制御できるということである．そのため，トグルが重要であると考えられる．この様子は，タイムチャートでも見ることができる．（図 18 参照）

以上が，J-K フリップフロップ回路の入力値と出力値の関係である．

## 2.11　D フリップフロップ回路

### 2.11.1　D フリップフロップ回路の機能説明

D フリップフロップ回路も J-K フリップフロップ回路と同様に一つ前の状態を保持し，その否定を次の出力として採用するような構造を取っている．そのときに $\overline{\mathrm{CLK}}$ 入力が立ち上がることがトリガーとなる．つまり，$\overline{\mathrm{CLK}}$ 入力を任意のタイミングで立ち上げることで，反転を制御することができる．

### 2.11.2　D フリップフロップ回路の動作表

図 19 で示された回路を実装し，動作させたときに示した動作を表にまとめると次のようになった．（表 12 参照）

表 12　D フリップフロップ回路動作表

| | 入力 | | 出力 |
|---|---|---|---|
| 接続端子 | S0 | DA | L0 |
| 端子名 | $\overline{\mathrm{CLR}}$ | $\overline{\mathrm{CLK}}$ | Q |
| クリア | L | × | L |
| 分周機能 | H | 0L | L |
| | H | 1↑ | H |
| | H | 2↑ | L |
| | H | 3↑ | H |
| | H | 4↑ | L |
| | H | 5↑ | H |
| | H | 6↑ | L |

### 2.11.3　D フリップフロップ回路のタイムチャート

図 19 で示された回路を実装し，動作させたときに示した動作をタイムチャートにまとめると次のようになった．（図 20 参照）

### 2.11.4　D フリップフロップ回路の考察 1

D フリップフロップ回路を用いた時，動作表（表 12）とタイムチャート（図 20）によると，クロック信号の立ち上がりにより，出力 Q の値が反転することがわかる．ただし，それは，$\overline{\mathrm{CLR}}$ が H のときであり，$\overline{\mathrm{CLR}}$ が L のときには，出力 Q は反転しない．これは，$\overline{\mathrm{CLR}}$ が L のときは，必ず出力 Q が L となり，$\overline{\mathrm{PR}}$ が L のときは，出力 Q が H となるようになっているために反転が起こらない．反転が起こるのは，$\overline{\mathrm{CLR}} = H$ かつ $\overline{\mathrm{PR}} = H$ のときに，出力がデータ値を参照することによる．D フリップフロップ回路は，データ値に一つ前の出力の否定（$\overline{\mathrm{Q}}$）を入力しているため，そのデータ値を参照することにより，一つ前の出力の反転した値が出力される．そのときに，CLK 入力が立ち上がるときにデータ値を参照するため，CLK 入力が，出力を変化させるトリガーとなっている．つまり，CLK 入力が D フリップフロップの反転機構を制御していると考えられる．しかし，そのトリガーには安全装置のようなものがあり，$\overline{\mathrm{CLR}}$ が H でないと，トリガーが作動しないようになっていると考えられる．

### 2.11.5　D フリップフロップ回路の考察 2

クロック信号の立ち上がりから立ち上がりまでを一周期とすると，D フリップフロップ回路の出力の変化の周期は，クロック信号の周期の 2 倍となったことがタイムチャート（図 20）より読み取れる．

### 2.11.6　D フリップフロップ回路の考察 3

クロック信号の周期に対し，出力の変化の周期が 2 倍であることから，周波数が周期の逆数であることを用いると，クロック信号の周波数に対し，出力の変化の周波数は，1/2 倍になったと考えられる．これは，タイムチャート（図 20）からも読み取れる．

以上より，D フリップフロップ回路は，クロック信号の立ち上がりに対して出力の変化が引き起こされるため，立ち上がりが二回起こらないと，最初の状態に戻らず，周期は 2 倍，周波数は 1/2 倍になる．このことから，ある交流信号をクロック信号として入力すると，その周波数の 1/2 倍の信号が出力される．このことから分周器と呼ばれる．つまり，分周機能（分割周波数機能）とは，ある入力信号の周波数を 1/2 倍にして出力することであると考えられる．

## 2.12 カウンタ回路

### 2.12.1 非同期 16 進カウンタ回路の機能説明

J-K フリップフロップ回路の出力を次の J-K フリップフロップ回路の $\overline{\mathrm{CLK}}$ 入力に接続することで，一つ前の回路が一周期終わると同時に，次の回路は変化を起こす．こうすることで，一つずつが 2 進数の桁を表すことができる．ある一つの回路が一周期分動くことは，すなわち，2 進数の桁が次の桁に繰り上がるということである．これによって，計算機が計数することができるようになる．この仕組が 16 進カウンタ回路に用いられている．

### 2.12.2 非同期 16 進カウンタ回路の動作表

図 21 で示された回路を実装し，動作させたときに示した動作を表にまとめると次のようになった．（表 13 参照）

### 2.12.3 非同期 16 進カウンタ回路のタイムチャート

図 21 で示された回路を実装し，動作させたときに示した動作をタイムチャートにまとめると次のようになった．（図 22 参照）

### 2.12.4 16 進カウンタ回路の考察 1

16 進カウンタ回路の一つずつの回路は，J-K フリップフロップ回路であるため，S0（$\overline{\mathrm{CLR}}$）が J-K フリップフロップ回路でどのような働きをしているか見れば良い．そこで，J-K フリップフロップ回路の動作表（表 11）をみると，$\overline{\mathrm{CLR}}$ が L のとき，他の入力の如何にかかわらず，反転出力の $\overline{\mathrm{Q}}$ の値は H になることがわかる．したがって，16 進カウンタの 4 つの出力 A，B，C，D の出力値は，H となるとわかる．その結果，NLED1 に入力される値は，A，B，C，D の 4 桁で表される 2 進数の最大値であるため，F であることがわかる．

### 2.12.5 16 進カウンタ回路の考察 2

S0（$\overline{\mathrm{CLR}}$）に H を入力した後は，J，K の入力に依存しており，これらが全て H であるので，反転した以前の出力を出力することになる．一回目の $\overline{\mathrm{CLK}}$ の立ち下がりでは，もともとが H の状態にあった，それぞれの出力を L に戻す動作が実行された．これは，全てが以前の出力の反転を出力するためである．よって，4 桁の 2 進数で最大であった NLED1 も，全てが反転し L のみとなったため，4 桁の 2 進数最小の 0 となる．

### 2.12.6 16 進カウンタ回路の考察 3

S0（$\overline{\mathrm{CLR}}$）に H を入力した後は，$\mathrm{Q}_0$ の値が $\overline{\mathrm{Q}}$ から出力されることがわかる．すると，J-K フリップフロップ回路が一つずつ老いていくに従って，分周が進み，桁数の大きい部分を担うようになる．クロック信号の周期が 1 の時，1 つ目の出力の周期は 2 で，2 つ目の出力の周期は 4 で，3 つ目の出力の周期は 8 で，4 つ目の

表 13　非同期 16 進カウンタ回路動作表

| $\overline{\text{CLR}}$ | $\overline{\text{CLK}}$ | D | C | B | A | NLED1 |
|---|---|---|---|---|---|---|
| S0 | DA | L3 | L2 | L1 | L0 | 7Seg 表示 |
| L | X | H | H | H | H | F |
| H | L | H | H | H | H | F |
| H | 1↓ | L | L | L | L | 0 |
| H | 2↓ | L | L | L | H | 1 |
| H | 3↓ | L | L | H | L | 2 |
| H | 4↓ | L | L | H | H | 3 |
| H | 5↓ | L | H | L | L | 4 |
| H | 6↓ | L | H | L | H | 5 |
| H | 7↓ | L | H | H | L | 6 |
| H | 8↓ | L | H | H | H | 7 |
| H | 9↓ | H | L | L | L | 8 |
| H | 10↓ | H | L | L | H | 9 |
| H | 11↓ | H | L | H | L | A |
| H | 12↓ | H | L | H | H | B |
| H | 13↓ | H | H | L | L | C |
| H | 14↓ | H | H | L | H | D |
| H | 15↓ | H | H | H | L | E |
| H | 16↓ | H | H | H | H | F |
| H | 17↓ | L | L | L | L | 0 |

出力の周期は 16 となる．このようにして，一段ずつ遅れながらも，クロック信号を伝えて，4 桁の 2 進数をカウントしている．順調に，クロック信号の立下りがある度に，0 から 1，2，3 と，カウントされる．これの 2 進数が NLED1 に入力されて，0 から F までの 16 文字に変換される．

### 2.12.7　16 進カウンタ回路の考察 4

2 進数の並びを逆にしているので，これは，補数，つまり，負の数を示すことになる．その時，最左桁は，補数のマイナスなどの符号を扱う部分となるため，D は，符号を，C は，2 進数の 3 桁目（$2^2$），B は，2 進数の 2 桁目（$2^1$），A は，2 進数の 1 桁目（$2^0$）を表す．

また，逆順に並べたものをただの 2 進数と読むこともでき，補数を考えないときには，D は，2 進数の 4 桁目（$2^3$），C は，2 進数の 3 桁目（$2^2$），B は，2 進数の 2 桁目（$2^1$），A は，2 進数の 1 桁目（$2^0$）を表す．

### 2.12.8　16 進カウンタ回路の考察 5

それぞれ，一つずつ J-K フリップフロップ回路がずれる毎に，周期は 2 倍になる．つまり，DA（$\overline{\text{CLK}}$）に対し，A では 2 倍の周期を持ち，A に対し，B では 2 倍の周期を持ち，B に対し，C では 2 倍の周期を持ち，C に対し，D では 2 倍の周期を持つ．つまり，DA（$\overline{\text{CLK}}$）に対しては，A は 2 倍，B は 4 倍，C は 8 倍，D は 16 倍の周期を持つ．

#### 2.12.9　16 進カウンタ回路の考察 6

2.12.8 節より，周期の逆数が周波数であることより，DA（$\overline{\mathrm{CLK}}$）に対し，A では 1/2 倍の周波数を持ち，A に対し，B では 1/2 倍の周波数を持ち，B に対し，C では 1/2 倍の周波数を持ち，C に対し，D では 1/2 倍の周波数を持つ．つまり，DA（$\overline{\mathrm{CLK}}$）に対しては，A は 1/2 倍，B は 1/4 倍，C は 1/8 倍，D は 1/16 倍の周波数を持つ．

## 3　感想

今回の実験で，論理回路の基本的な動作について理解を深められたと感じている．また，実際に回路を設計し，実装し，考察することで，論理式の組み立てとそれによって作れる動作というのを多少なりとも理解できたのではないかと感じている．

## 参考文献

[1] 電気通信大学 II 類・III 類「メカトロニクス基礎実験 知能機械工学実験」，p.52-80，2018

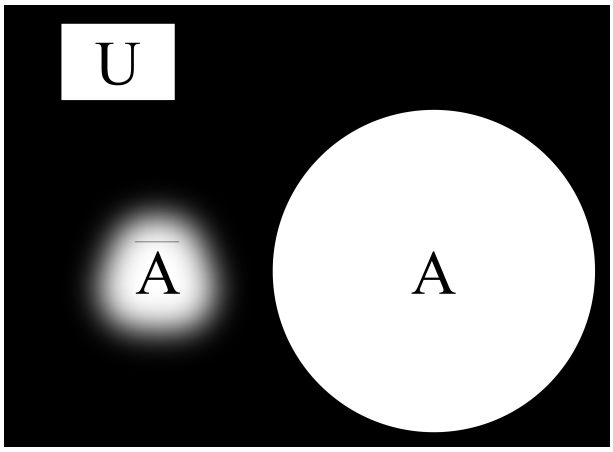


図 1　補集合のベン図

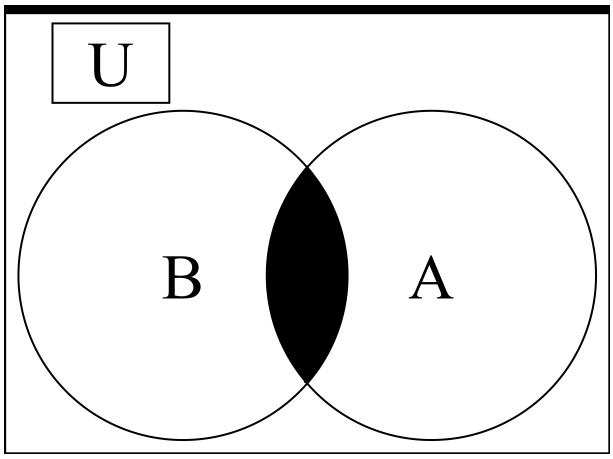


図 2　積集合のベン図

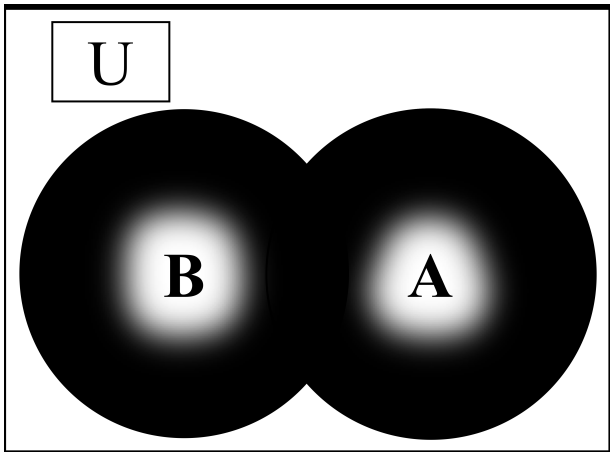


図 3　和集合のベン図

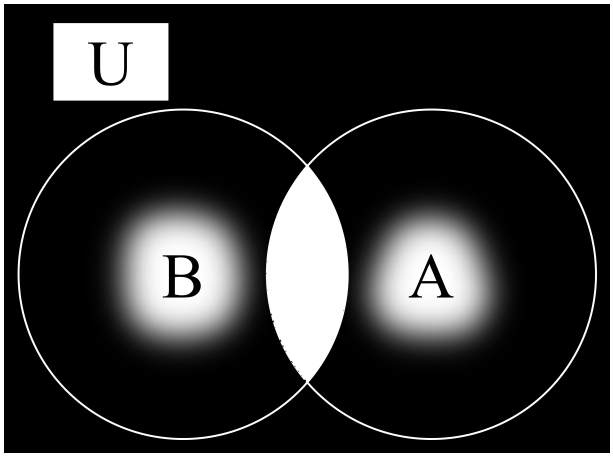


図 4　積集合の補集合のベン図

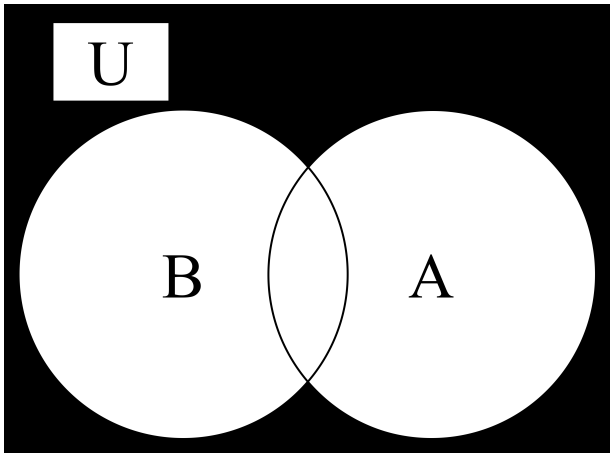


図 5　和集合の補集合ベン図

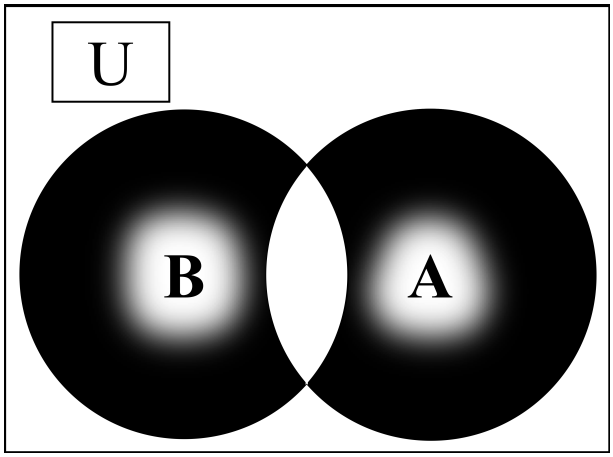


図 6　排他的論理和のベン図

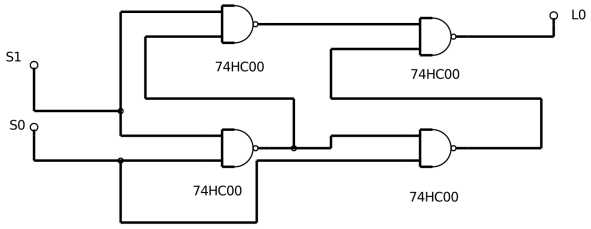


図 7　2 入力 ExOR 回路の回路図

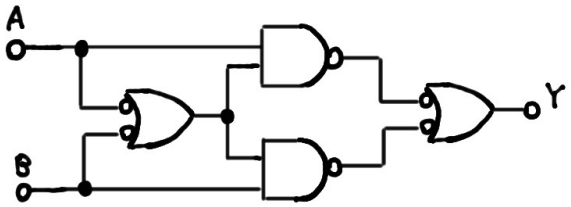


図 8　課題 1 における ExOR 回路の回路図

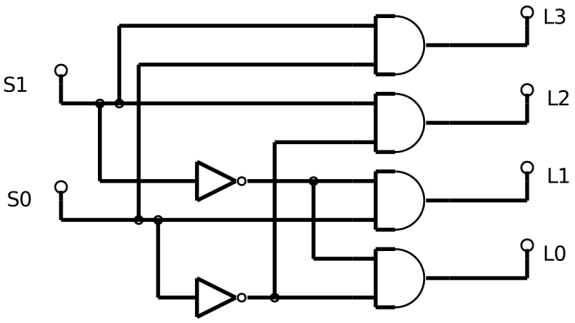


図 9　2 入力 4 出力デコーダの回路図

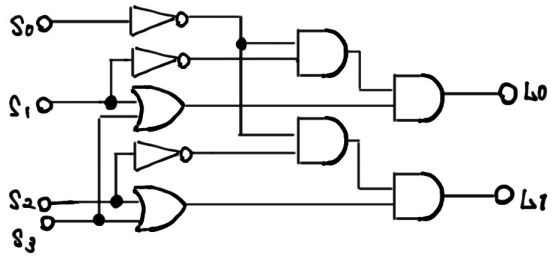


図 10　4 入力 2 出力エンコーダの回路図（複数選択可能）

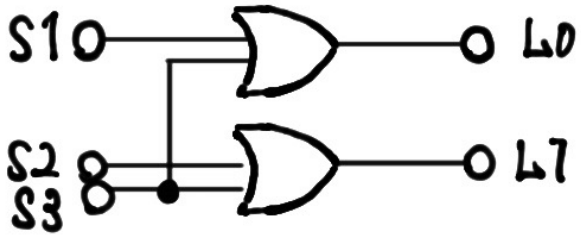


図 11　4 入力 2 出力エンコーダの回路図（選択が 1 つしかできない場合）

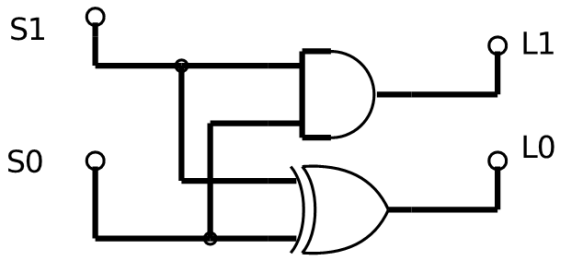


図 12　半加算器回路の回路図

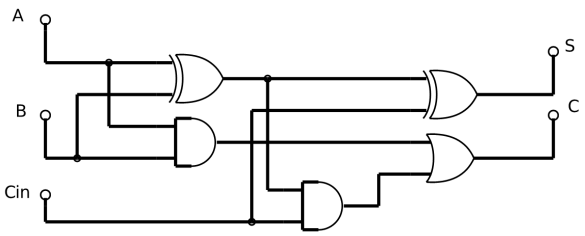


図 13　全加算器回路の回路図

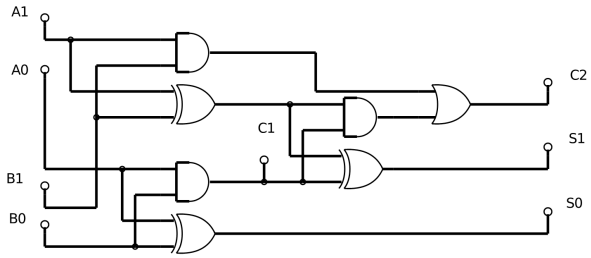


図 14　2 桁の 2 進数の加算器回路の回路図

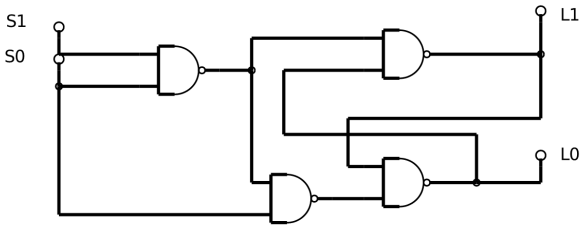


図 15　D ラッチ回路の回路図

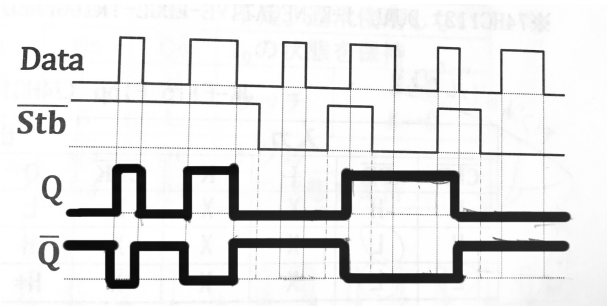


図 16　D ラッチ回路のタイムチャート

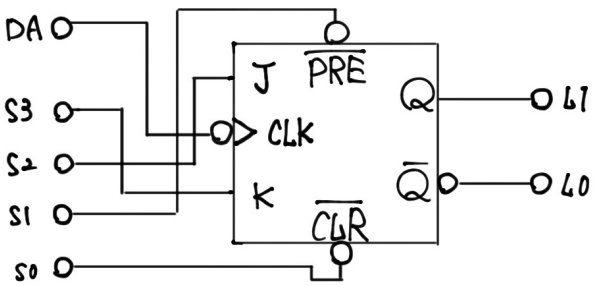


図 17　J-K フリップフロップ回路の回路図

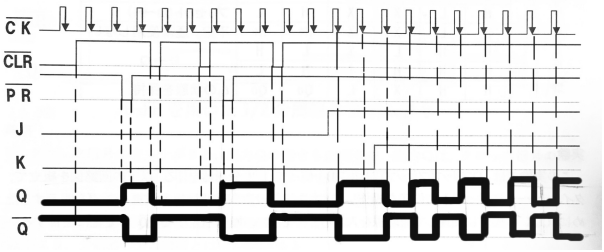


図 18　J-K フリップフロップ回路のタイムチャート

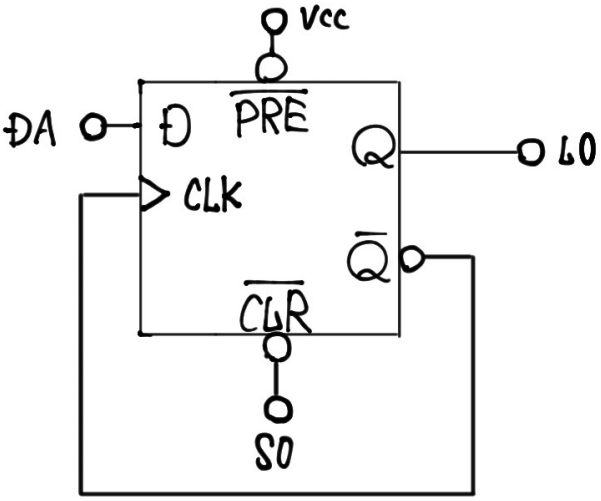


図 19　D フリップフロップ回路の回路図

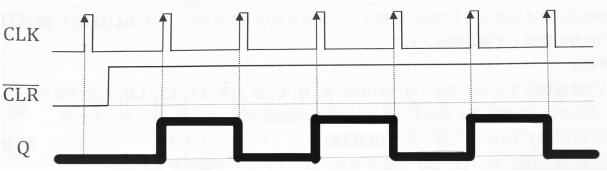


図 20　D フリップフロップ回路のタイムチャート

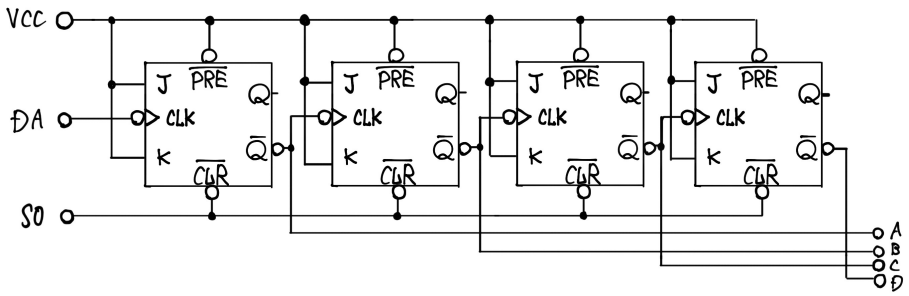


図 21　16 進カウンタ回路の回路図

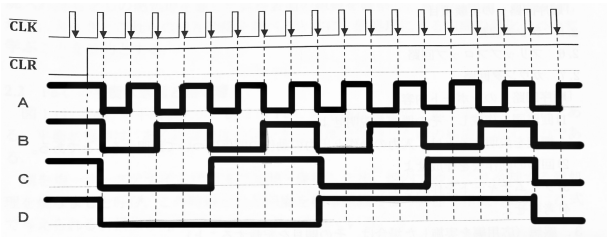


図 22　16 進カウンタ回路のタイムチャート