

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。

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{A} と記述する。 \overline{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。ベン図で記すと図で表せる。

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、 $A \cap B$ と表し、AND 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。ベン図で記すと図で表せる。

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、 $A \cup B$ と表し、OR 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。ベン図で示すと図で表せる。

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。それは、 $\overline{A \cap B} = U - (A \cap B)$ と表される。ベン図で示すと図となる。

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。それは、 $\overline{A \cup B} = U - (A \cup B)$ と表される。ベン図で示すと図となる。

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。それは $(A \cup B) - (A \cap B)$ となる。ベン図で示すと図となる。

2.2 2 入力 ExOR ゲート

図 6 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。その動作は、次の動作表で示された。(表 1 参照)

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

2.2.2 論理式

図 6 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．

$$\begin{aligned}
 &(1 : B) \text{ NAND } (2 : A) \rightarrow (3 : \overline{A \cap B}) \\
 &3 \rightarrow 4 \\
 &(4 : \overline{A \cap B}) \text{ NAND } (5 : A) \rightarrow (6 : \overline{\overline{A \cap B} \cap A}) \\
 &3 \rightarrow 12 \\
 &(12 : \overline{A \cap B}) \text{ NAND } (13 : B) \rightarrow (11 : \overline{\overline{A \cap B} \cap B}) \\
 &6 \rightarrow 9, 11 \rightarrow 10 \\
 &(9 : \overline{\overline{A \cap B} \cap A}) \text{ NAND } (10 : \overline{\overline{A \cap B} \cap B}) \rightarrow (8 : \overline{\overline{\overline{A \cap B} \cap A} \cap \overline{\overline{A \cap B} \cap B}})
 \end{aligned}$$

つまり，

$$\begin{aligned}
 Y &= \overline{\overline{\overline{A \cap B} \cap A} \cap \overline{\overline{A \cap B} \cap B}} \\
 &= \overline{\overline{A \cap B} \cap A} + \overline{\overline{A \cap B} \cap B} \\
 &= (\overline{A \cap B}) \cdot A + (\overline{A \cap B}) \cdot B \\
 &= A \cdot \overline{B} + \overline{A} \cdot B \quad (A \cdot \overline{A} = B \cdot \overline{B} = 0) \\
 &= A \oplus B
 \end{aligned}$$

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 6 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， $Y = A \cdot \overline{B} + \overline{A} \cdot B = A \oplus B$ 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で，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 7 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 回路と負論理入力の OR 回路を用いた。

負論理入力の OR 回路の入力に 2 つの入力を入れる。この出力を一つの入力とする NAND 素子

$$\begin{aligned}C &= \overline{A} + \overline{B} \\ \overline{D} &= A \cdot C = A \cdot (\overline{A} + \overline{B}) \\ \overline{E} &= B \cdot C = B \cdot (\overline{A} + \overline{B}) \\ Y &= \overline{D} + \overline{E} \\ &= A \cdot (\overline{A} + \overline{B}) + B \cdot (\overline{A} + \overline{B}) \\ &= A \cdot \overline{B} + \overline{A} \cdot B \quad (A \cdot \overline{A} = B \cdot \overline{B} = 0) \\ &= A \oplus B\end{aligned}$$

よって、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 つの変換がうまく行かなければ、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。

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デコーダと言われている。ここでは、2 進数のデータを 4 つのパターンに分類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することとして、コードにすることからエンコーダと言われている。ここでは、4 つのパターンを 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 8 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
- S1 が OFF で S0 が ON のとき、L1 が点灯
- S1 が ON で S0 が OFF のとき、L2 が点灯
- S1 と S0 の両方が ON のとき、L3 が点灯

表 3 2 入力 ExOR ゲート動作表

	入力		出力			
接続端子	S0	S1	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

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 L3 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 2 入力 ExOR ゲート真理値表

	入力		出力			
接続端子	S0	S1	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。

$$L0 = \overline{S0} \cdot \overline{S1}$$

$$L1 = S0 \cdot \overline{S1}$$

$$L2 = \overline{S0} \cdot S1$$

$$L3 = S0 \cdot S1$$

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、4 つのパターンを示すことができる。それは、図 4 で示されている入力のパターンであり、それぞれの 0, 1 のパターンが 2 種類ずつであるため、 $2 \times 2 = 4$ となることからわかる。これを 1 つずつ

のパターンに抽出しているのが今回の実験のデコーダである。論理積は、2つの入力がいずれも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。2.4.3節で述べたような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。それは、特定のパターンのときにのみ両方が1となる様に組み立てられているためであり、それによって、デコーダが構成されている。つまり、計算機の言語である2進数を人間が読みやすい4つのパターン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、メモリの読み出しなどもこれ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、と役割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

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、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	L0	L1
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、それぞれのパターンの小さいものから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

- L0 も L1 も、少なくとも、S0 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、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 (S1、もしくは、S3 が 1) = (S0 が 0) AND (S1、OR、S3 が 1)

L1 : (S0 が 0) かつ (S2、もしくは、S3 が 1) = (S0 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、

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

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

となる。

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，図 9 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，繰り上げた数を加算することはない。加算の結果繰り上げとして値を出力することもある。1 桁同士であるので，そんなに複雑にはならない。全加算器は，2 桁以上の数値が含まれる加算の中に含まれており，繰り上げた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，一桁前の繰り上げられた数）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，繰り上げた数である。要するに筆算式としては，左のようになる。また，加算結果を簡条書きでまとめると右のようになる。

	C		和	桁上げ
	↓	A	$0 + 0 = 0$	なし
	↓	B	$0 + 1 = 0$	なし
+	↓	B	$1 + 0 = 1$	なし
	C	S	$1 + 1 = 0$	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 6 参照）

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。

表 6 半加算器真理値表

	入力		出力	
	A	B	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

- 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， $((A = 1) \text{ かつ } (B = 0))$ もしくは $((A = 0) \text{ かつ } (B = 1))$ のときに 1
- 桁上げは， $(A = 1) \text{ かつ } (B = 1)$ 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 10 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 10 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）

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．

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

2.8 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、加算結果を簡条書きでまとめると右のようになる。

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

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、入力の値に少なくとも 1 つ 1 があるときに 1 になること。

表 8 全加算器真理値表

端子名	入力			出力	
	A	B	C _{in}	和 S	桁上げ 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

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)) もしくは ((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)) もしくは ((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
- C_{in} = 1 ならば、桁上げは、(A = 1) もしくは (B = 1) のときに 1

となった。和に関しては、A と B の ExOR と 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、C_{in} の状態によってまとめきれないと判断したため、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、

$$S = (A \oplus B) \oplus C_{in}$$

$$C_{out} = (A \oplus B) \cdot C_{in} + (A \cdot B)$$

となった。

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 10 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)

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ぬとわかった。

表 9 全加算器真理値表

端子名	入力				出力		
	A ₁	A ₀	B ₁	B ₀	3 桁目 C ₂	2 桁目 S ₁	1 桁目 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

- 1 桁目の和は，1 桁目の入力値の直和であること．
- 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 桁目の入力値の直和と 1 桁目の桁上げ値の直和であること．
- C₁ 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．
- C₁ 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 10 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 13 で示された回路である．この回路にデータとストロブ入力を入れて，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 13 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	
	S_1	S_0	L_1	L_0
接続端子 端子名	Data	\overline{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 13 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 14 参照）タイムチャートより，ストロ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

ストロブ信号が H のときは， \overline{Stb} が L のときである．ストロブ信号が H 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， $\overline{Stb} = L$ のとき，出力が必ず 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， $\overline{Stb} = L$ のとき，出力が必ず 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， $\overline{Stb} = L$ 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 14 参照)

したがって、 $\overline{Stb} = L$ のとき、出力が必ず Q_0 となると考えられる。

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{Stb} が H 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 OFF になるため、データの入力に変化した場合、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、 $\overline{Stb} = H$ 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、 $\overline{Stb} = H$ 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、 $\overline{Stb} = H$ 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 14 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、 $\overline{Stb} = H$ 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。

2.9.6 D ラッチ回路の考察 3

データの入力値を変更する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 H 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 ON 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 H にした時、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 L 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。ストローブ信号を 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 L 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と考えられる。

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{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{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 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 11 参照)

表 11 J-K フリップフロップ回路動作表

端子名	入力					出力	
端子名	S_0	S_1	S_2	S_3	DA	L_1	L_0
端子名	$\overline{\text{CLR}}$	$\overline{\text{PR}}$	J	K	$\overline{\text{CLK}}$	Q	\overline{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 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)

2.10.4 JK フリップフロップ回路考察前半

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

表 11 によると、 $\overline{\text{CLR}}$ もしくは、 $\overline{\text{PR}}$ に L が入力されている時、他の入力 J, K, $\overline{\text{CLK}}$ 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 16 参照) タイムチャートの前半部が、 $\overline{\text{CLR}}$ と $\overline{\text{PR}}$ に関する変化である。タイムチャートで $\overline{\text{CLR}}$ と $\overline{\text{PR}}$ 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。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、そのたび毎の変化は起こらないことがわかる。

したがって、 $\overline{\text{CLR}}$ もしくは、 $\overline{\text{PR}}$ が L のときにはある定状態が続き、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、 $\overline{\text{CLR}}$ が L のときと $\overline{\text{PR}}$ 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、 $\overline{\text{CLR}}$ と $\overline{\text{PR}}$ 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 $\overline{\text{CLK}}$ 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、 $\overline{\text{CLK}}$ の値が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{Q}) も (L, H) になるというわけではなく、 $\overline{\text{CLK}}$ の変化による。ただ、 $\overline{\text{CLK}}$ 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{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 16 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 $\overline{\text{CLK}}$ 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 $\overline{\text{CLK}}$ 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 16 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 $\overline{\text{CLK}}$ 入力 that 立ち上がることがトリガーとなる。つまり、 $\overline{\text{CLK}}$ 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 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 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	$\overline{\text{CLR}}$	$\overline{\text{CLK}}$	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 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)

2.11.4 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 18)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、 $\overline{\text{CLR}}$ が H のときであり、 $\overline{\text{CLR}}$ が L のときには、出力 Q は反転しない。つまり、CLK 入力、出力を変化させるトリガーとなっている。しかし、そのトリガーには安全装置のようなものがあり、 $\overline{\text{CLR}}$ 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 18)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 18)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、分周機能とは、ある入力信号の周波数を細かく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 $\overline{\text{CLK}}$ 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。この仕組みが 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 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 13 参照)

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

2.12.3 非同期 16 進カウンタ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)

2.12.4 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 ($\overline{\text{CLR}}$) 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、 $\overline{\text{CLR}}$ が L のとき、他の入力の如何にかかわらず、反転出力の \overline{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 ($\overline{\text{CLR}}$) に H を入力した後は、J, K の入力が依存しており、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 $\overline{\text{CLK}}$ 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 ($\overline{\text{CLR}}$) に H を入力した後は、 Q_0 の値が \overline{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、B は、2 進数の 2 桁目、A は、2 進数の 1 桁目を表す。

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 ($\overline{\text{CLK}}$) 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 ($\overline{\text{CLK}}$) 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 補集合ベン図

図 2 積集合ベン図

図 3 ベン図

図 4 ベン図

図 5 ベン図

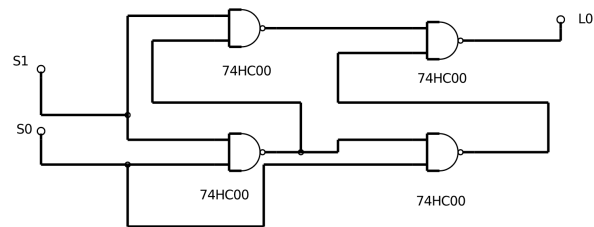


図 6 2 入力 ExOR 回路の回路図

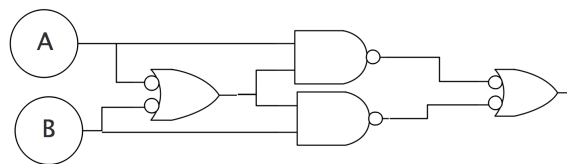


図 7 課題 1 における ExOR 回路の回路図

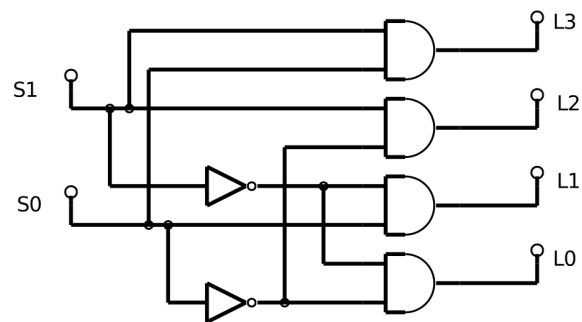


図 8 2 入力 4 出力デコーダの回路図

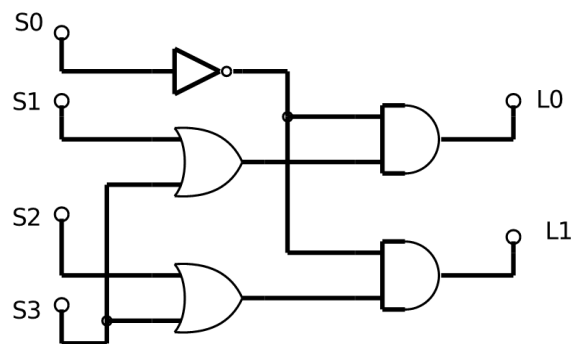


図 9 4 入力 2 出力エンコーダの回路図

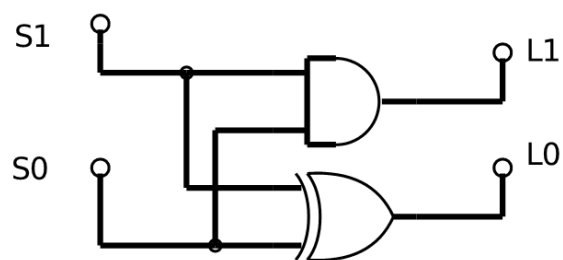


図 10 半加算器回路の回路図

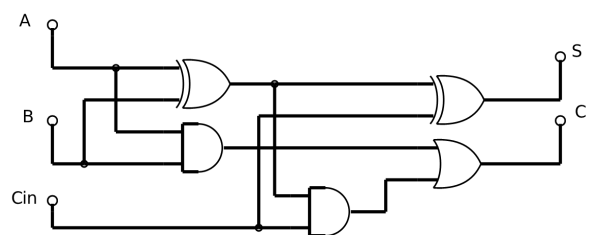


図 11 全加算器回路の回路図

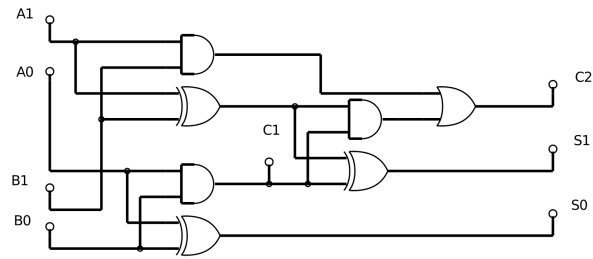


図 12 2桁の2進数の加算器回路の回路図

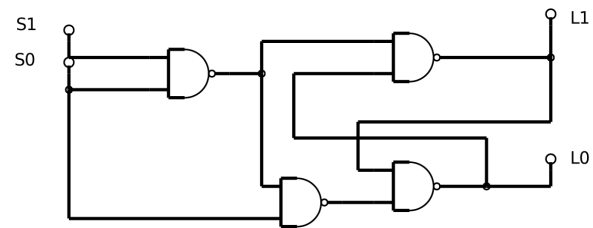


図 13 D ラッチ回路の回路図

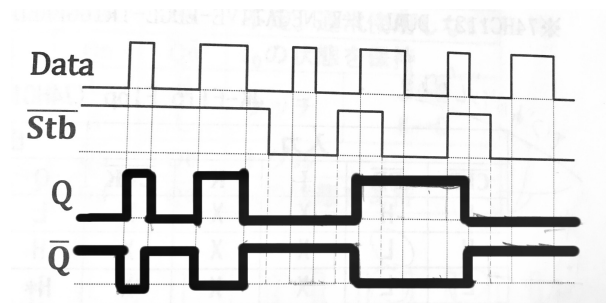


図 14 D ラッチ回路のタイムチャート

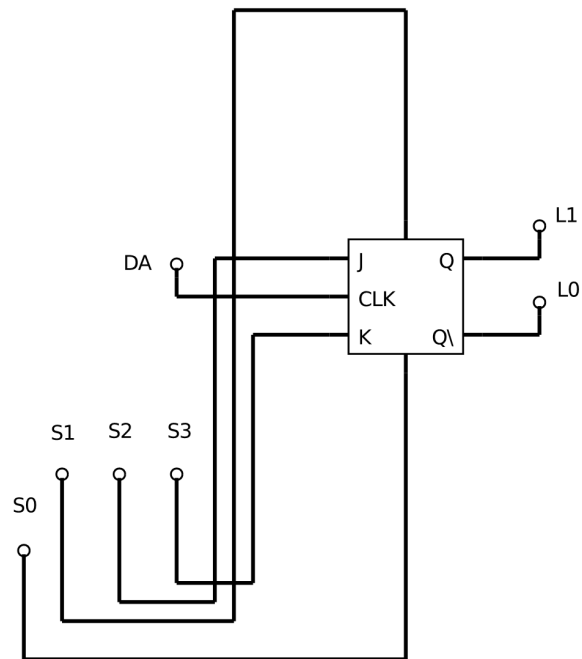


図 15 J-K フリップフロップ回路の回路図

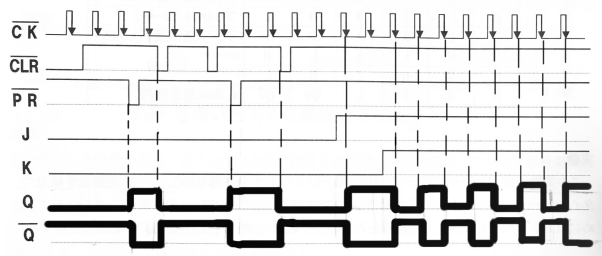


図 16 J-K フリップフロップ回路のタイムチャート

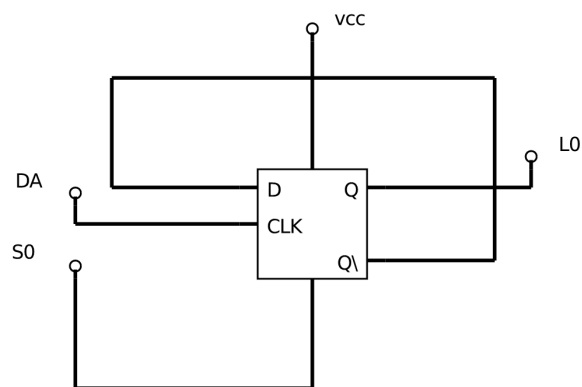


図 17 D フリップフロップ回路の回路図

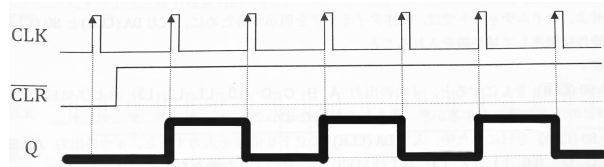


図 18 D フリップフロップ回路のタイムチャート

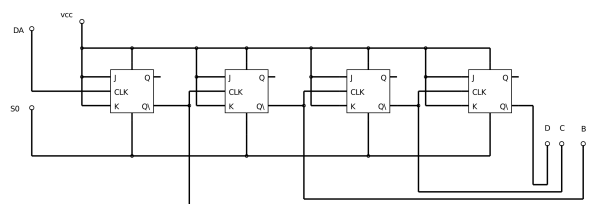


図 19 16 進カウンタ回路の回路図

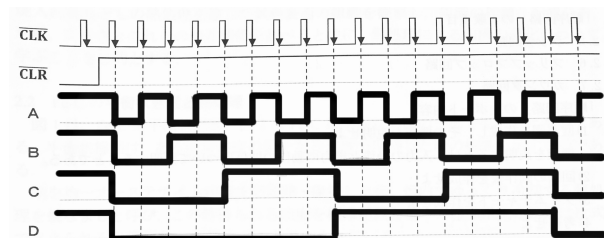


図 20 16 進カウンタ回路のタイムチャート