# 1　目的

減算回路のリバースエンジニアリングを行い、その動作原理を真理値表の作成、回路図の作成を通して理解する。

# 2　原理

今回実験で必要な知識について説明を行う。

## 2.1　補数

補数とは、ある数値とその数値を足すとちょうど特定の基数に達する数字のことを指す。例えば 10 進数では、69 に対する補数は $(100-69)=31$ である。今回では 0 から 7 の数字を扱うため、8 進数を用いて論理式の計算を行い、回路上での論理回路では 2 進数を用いて計算を行う。8 進数での補数表現では、$n$ に対する補数は $(8^m-n)$ である。$m$ は桁数である 2 進数での補数表現は、$n$ に対する補数は $(2^m-n)$ である。$m$ は桁数である。この補数を求める操作は、ビット反転と 1 の加算を行うことで求めることができる。動作原理として、$n$ に対する進数から 1 を引いた進数では値を桁上がりをしない範囲で最大にするものであるので、$2^m$ から $n$ を引いた値を求めることで補数を求めることができる。これを二進数で行うとビットの反転であり、2 の補数にするためには値を 1 加算することで求めることができることがわかる。例えば、$1-5$ を行う場合には、5 の補数に 1 を加算することで求めることができる。5 の補数は $(2^3-5)_{10}=(3)_{10}$ または、$(101)$ を反転して $(010)$ となる。これに $(001)_2$ を足すと $(011)_2=(3)_{10}$ であるため、$(3+1)_{10}=(4)_{10}$ となる。

## 2.2　減算

減算を行うときには補数を使用する。例えば $1-5$ を行う場合には、$1+(2^3-5)=1+3$ となる。補数を使った計算の場合符号が $m$ の次のビットに現れて、1 の場合は正の数であることを示し、0 の場合は負の数であることを示す。例として $5-1$ を行うと $(1)_2$ を補数にして $(101)_2+(111)_2=(1100)_2$ となり、4bit 目に符号が現れる。原理として、$a-b\geq 0$ の場合 $a+(8-b)\geq 8$ となる。そのため、4bit 目が 1 になる。逆に $a-b<0$ の場合、$a+(8-b)<8$ となる。そのため、4bit 目が 0 になる。

## 2.3　計算結果が補数になる原理

減算の原理を示したが、$(1-6)_{10}$ の計算を行うとき、補数と加算を使った計算をすると計算結果が補数となる。このようになる条件としては、減算の結果が負の数になるときである。$(1-6)_{10}$ の例で示すと、$(001)_2+(010)_2=(011)_2$ となる。この結果は $(3)_{10}$ であるが、これは $-5$ の補

数である。これを二進数の絶対値に戻すには $(0)_{10} - (-5)_{10}$ を行うとできる。$(011)_2$ の補数は $(101)_2 = (5)_{10}$ となるので、$(0)_{10} + (5)_{10} = (5)_{10}$ となり、絶対値に直すことが出来る。

### 2.4 加算器

二進数の加算を行う方法について説明をする。各桁の足し算だけを考えたときの真理値表を表 1 に示す。C は繰り上がりの出力を示し、S は各桁の和を示す。この真理値表から論理式を求めると、$C = A \cdot B$、$S = A \oplus B$ となる。これを半加算器と呼ぶ。これをすべての桁を包括した演算に拡張して考えると、繰り上がりの計算を行う必要がある。$A + B$ を行った後に、前の桁の繰り上がりを足すことにより、全体の和を求めることができる。そのために、半加算器を二つ用いる。1 個目の半加算器の $S$ の出力を 2 個目の半加算器の $A$ に入力し、前の繰り上がりの $C$ を 2 個目の半加算器の $B$ に入力することで、$A + B + C$ を行うことが出来る。そして、二つの半加算器の $C$ を $OR$ 演算したものを全加算器と呼ぶ。$OR$ 演算をする理由としては $ABC$ のどれか二つ以上が 1 のときに繰り上がりが発生するため、繰り上がりの出力二つを $OR$ 演算することで、繰り上がりの出力を行うことが出来る。

表 1: 半加算器の真理値表

| $A$ | $B$ | $C$ | $S$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

## 3 実験方法

リバースエンジニアリングを行う手順を説明する。

### 3.1 真理値表の作成

減算回路の真理値表を作成するために、減算基板確認用基板を用いて入力を行い、セレクター基盤を経由してデコード回路に信号を入力させて、真理値表を作成する。また、減算回路の基板には IC がついてないため、IC の配置を確認し、基板に IC を取り付ける。IC の場所を画像 1 に示す。また、IC の取り付ける場所を表 2 に示す。

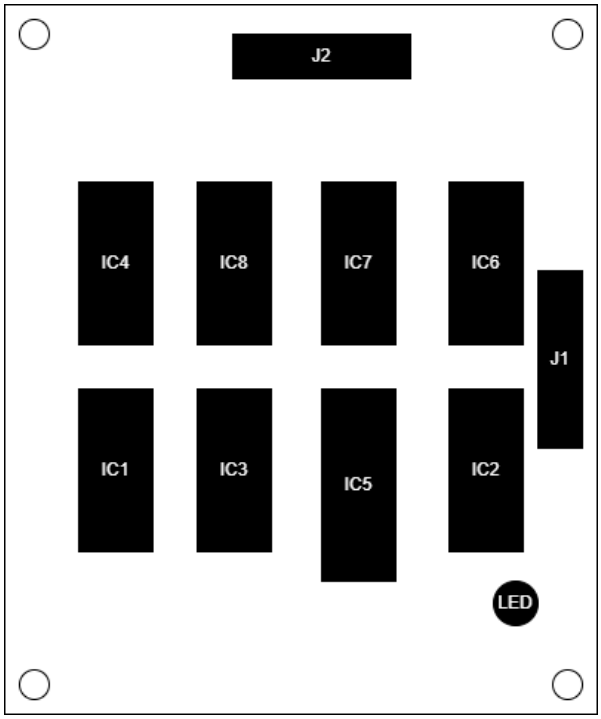


図 1: IC の配置

表 2: IC の配置

| IC 番号 | IC 名 |
|---|---|
| IC1 | 74LS04 |
| IC2 | 74LS08 |
| IC3 | 74LS08 |
| IC4 | 74LS32 |
| IC5 | 74LS83 |
| IC6 | 74LS86 |
| IC7 | 74LS86 |
| IC8 | 74LS86 |

## 3.2 テスターを用いて回路図を作成する

テスターから電圧を印加させて回路の導通を確認し、導通しているピンを基に回路図の作成を行う。

# 4　結果

実験方法で示した通りに行った実験の結果を示す。

## 4.1　真理値表

真理値表を表 3 に示す。

表 3: 真理値表

| $A < B$ | $C$ | $LED$ |
|---|---|---|
| 0 | A - B | 1 |
| 1 | B - A | 0 |
| $A = B$ | $C$ | $LED$ |
| 0 | $\|A - B\|$ | * |
| 1 | 0 | 1 |

## 4.2　回路図

回路図を図 2 に示す。

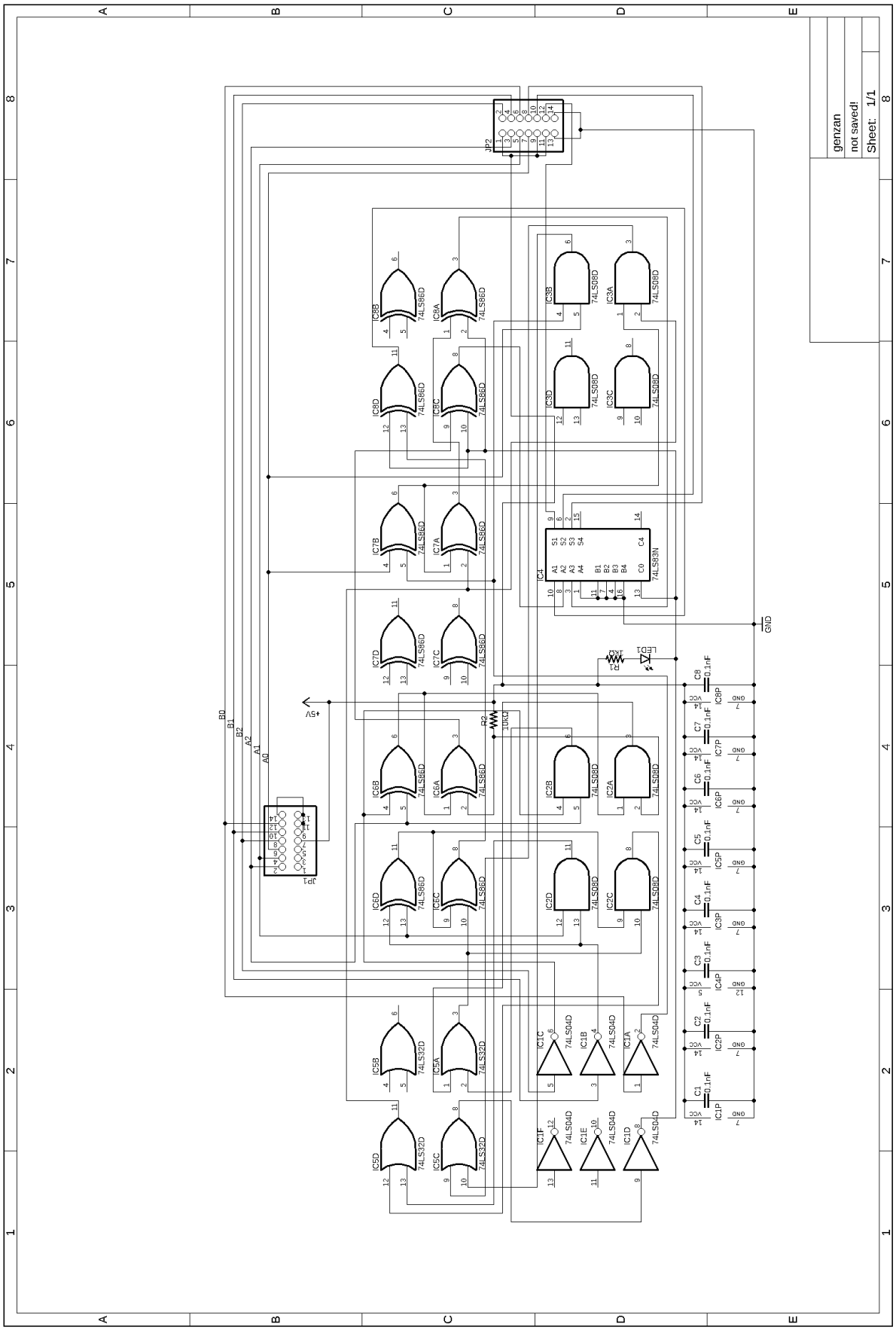
genzan
not saved!
Sheet: 1/1
JP2
JP1
+5V
GND
B0
B1
B2
A2
A1
A0
R2
10kΩ
R1
1kΩ
LED1
IC4
74LS83N
IC1A
IC1B
IC1C
IC1D
IC1E
IC1F
74LS04D
IC2A
IC2B
IC2C
IC2D
IC3A
IC3B
IC3C
IC3D
74LS08D
IC5A
IC5B
IC5C
IC5D
74LS32D
IC6A
IC6B
IC6C
IC6D
IC7A
IC7B
IC7C
IC7D
IC8A
IC8B
IC8C
IC8D
74LS86D
C1
C2
C3
C4
C5
C6
C7
C8
0.1nF

図 2: 回路図

# 5　考察

減算回路のリバースエンジニアリングを行った結果御もとに考察を書いていく。

## 5.1　真理値表

LED の出力結果が演算結果が正の値のときに点灯になった理由として、原理で説明した減算の演算方法によって、4bit 目の値が 1 になった、これを not 回路に通すことで 0 となりそのピンは $0V$ となるため $5V$ からの電流により LED が点灯したと考える。

## 5.2　回路図

回路図を見ると、全加算器が各 bit に 2 つずつ接続されていることがわかる。これは原理で説明した減算結果が補数になる場合に絶対値に戻す手法の実現のためである。XOR 回路に 4bit 目の反転した値と各 bit の一回目の全加算器の演算結果が入力されている。これにより、演算結果が補数になったときにだけ値が反転されるようになる。そして、74LS83 を用いて、反転した値と 1 を全加算器で加算の演算を行っている。この 1 は 4bit 目の出力を反転したものが 74LS83 の C0 入力に入っていることにより実現している。これを行うことにより、補数になった値を絶対値に戻すことができる。