電気工学通論Ⅱ（１）

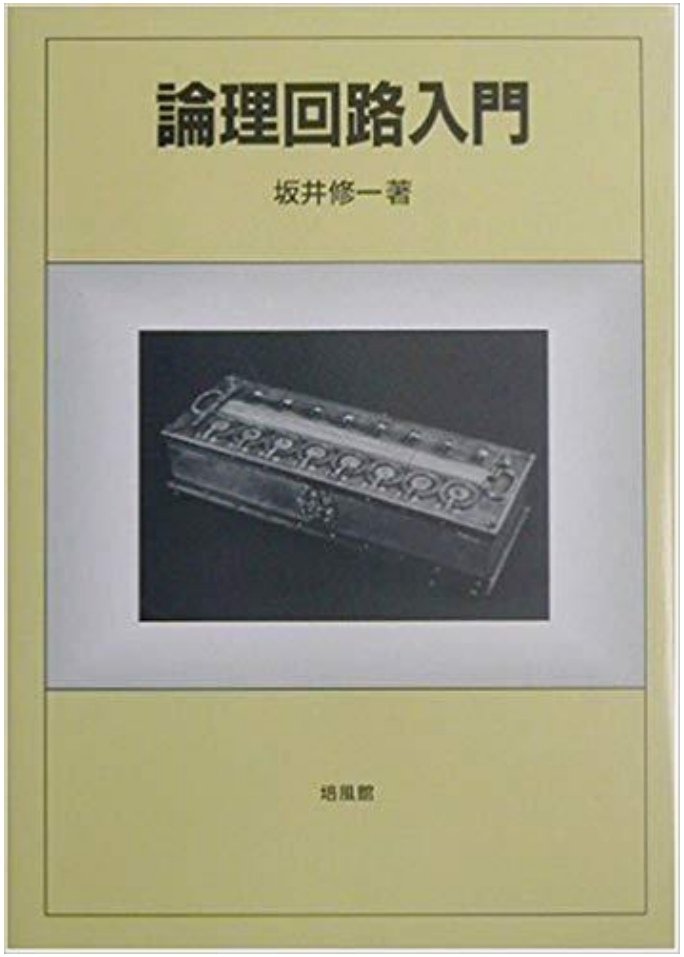
坂井　修一
東京大学大学院 情報理工学系研究科 電子情報学専攻
東京大学　工学部　電子情報工学科

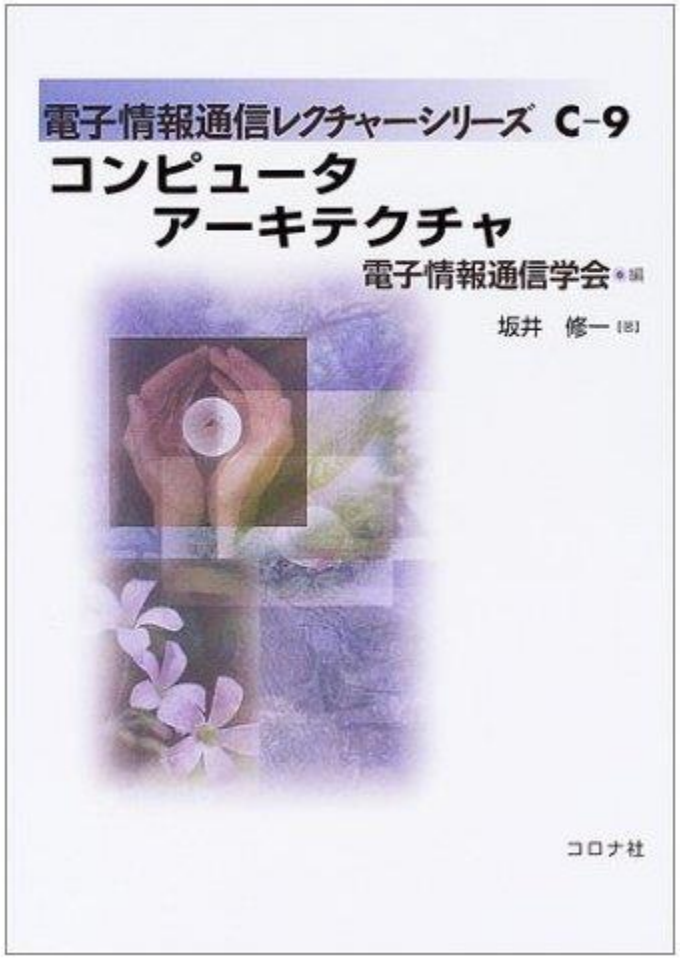
- はじめに
- スケジュール
- ２進数の計算

はじめに

- 本講義の目的
  - ディジタル回路とコンピュータアーキテクチャの基礎を学習する
- 時間・場所
  - 火曜日 8:30 – 10:15、工学部2号館213
- ホームページ（ダウンロード可能）
  - url:　http://www.mtl.t.u-tokyo.ac.jp/~sakai/tsuron2/
- 教科書
  - 坂井修一『論理回路入門』（培風館）
  - 坂井修一　『コンピュータアーキテクチャ』（コロナ社）

教科書


論理回路入門
坂井修一著
培風館


電子情報通信レクチャーシリーズ C-9
コンピュータ
アーキテクチャ
電子情報通信学会 編
坂井 修一 著
コロナ社

講義の概要と予定（１／２）

１．ディジタル回路入門

ディジタルとアナログ、２進数、補数表現、四則演算

２．　論理演算

論理演算とは、３つの基本論理演算、NORとNAND、完備性、ディジタル回路の表現、ブール代数、標準形

３．組み合わせ回路の構成法

真理値表と組合せ回路、組合せ回路の簡単化

４．組合せ回路の実例

加算器、補数、減算器、ＡＬＵ、デコーダ、セレクタなど

５．フリップフロップ

ＦＦ、ＳＲラッチ、Ｄラッチ、非同期と同期、SR-FF、D-FF、マスタスレーブ形とエッジトリガ形, JK-FF, レジスタ

６．基本的な順序回路

７．一般的な順序回路の作り方

講義の概要と予定（２／２）

８．コンピュータアーキテクチャ入門

計算のサイクル、主記憶装置、メモリの構成と分類、レジスタファイル、命令、命令実行の仕組み、実行サイクル、算術論理演算命令、シーケンサ、条件分岐命令

９．命令セットアーキテクチャ

操作とオペランド、命令の表現形式、アセンブリ言語、命令セット、算術論理演算命令、データ移動命令、分岐命令、アドレシング、サブルーチン、RISCとCISC

10．パイプライン

11．キャッシュと仮想記憶

12．入出力と周辺装置

周辺装置、ディスプレイ、二次記憶装置、ハードウェアインタフェース、割り込みとポーリング、アービタ、DMA、例外処理

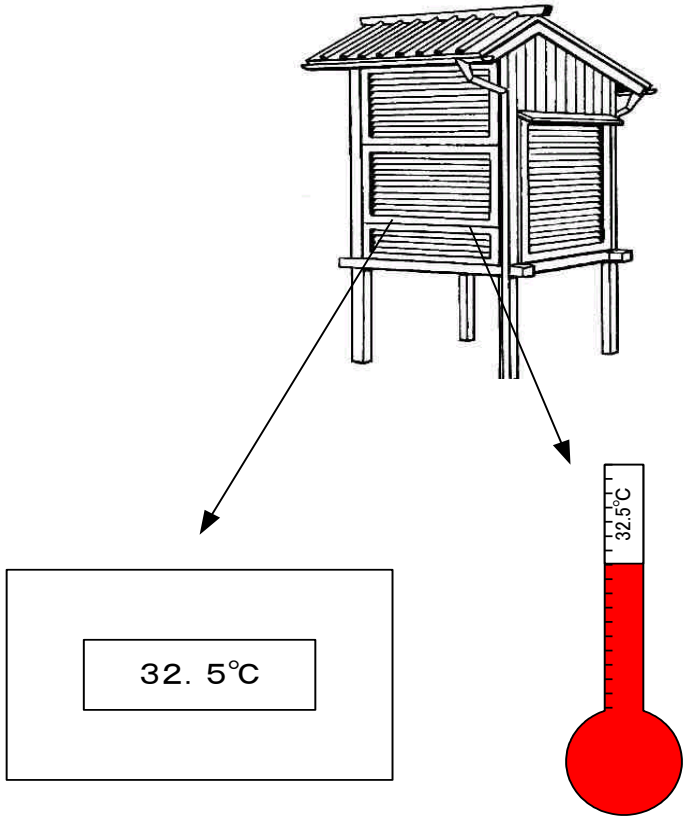
休講：　X月Y日　　試験：Z月W日

成績：出席（小テスト）＋試験　　４年生：追試・レポート無し

１．ディジタル回路入門

- 内容
  - ディジタルとアナログ
  - ２進数
    - 数の表現
    - 数の変換
    - ２進数の演算
  - 組合せ回路と順序回路
  - 素子

ディジタルとアナログ



（a） ディジタルな表現　　（b） アナログな表現

ディジタルとアナログの比較

ディジタルとアナログ

	ディジタル	アナログ
表現法	数字	別の「量」
ハードウェア量	多いが LSI により高集積化可能	少ない
処理速度	速い	問題による
記憶	容易	むずかしい
持ち運び、通信	容易	むずかしい
コスト	安い。特に汎用マイクロプロセッサ	問題・精度に大きく依存
精度	必要な桁数分のハードウェアを投入して達成する	構成要素（素子）に依存。一般にあまり高くない
雑音	強い	弱い
汎用性	高い	低いー
インタフェース	AD/DA 変換が必要	特別な変換は不要

ディジタル論理回路

- ディジタル論理回路
  - 電子計算機（コンピュータ）の基礎となる回路。通常、0，1の2種類の入出力をもつ（不連続な値）
    - 0：　電圧の低い状態（L）
    - 1：　電圧の高い状態（H）

- ビット（bit, binary digit, **b**）
  - 情報の単位。0か1の値を取る

ディジタルな表現

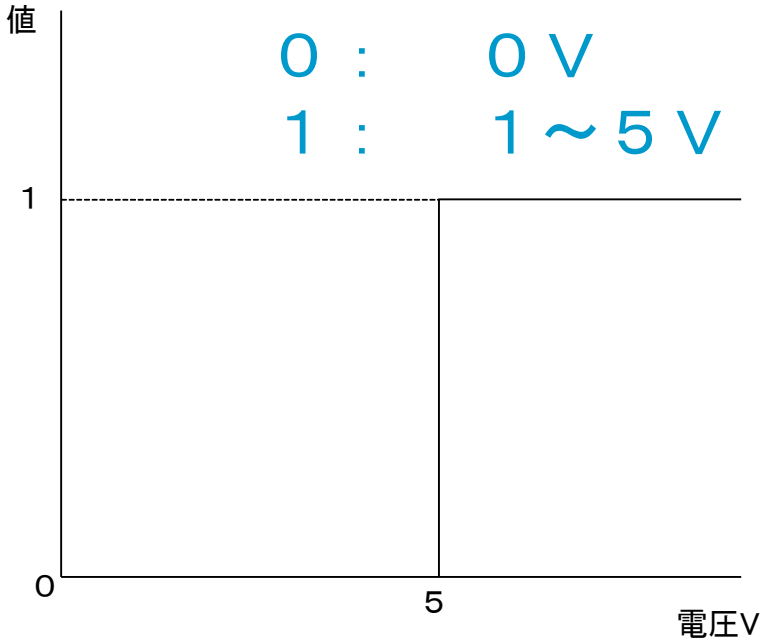
- 2進数 $a_m a_{m-1} \cdots a_1 a_0$

  $X = a_m 2^m + a_{m-1} 2^{m-1} + \cdots + a_1 2 + a_0$ $(a_i = 0, 1)$

- 0,1の表し方
  - 0： 電圧の低い状態（L）
  - 1： 電圧の高い状態（H）

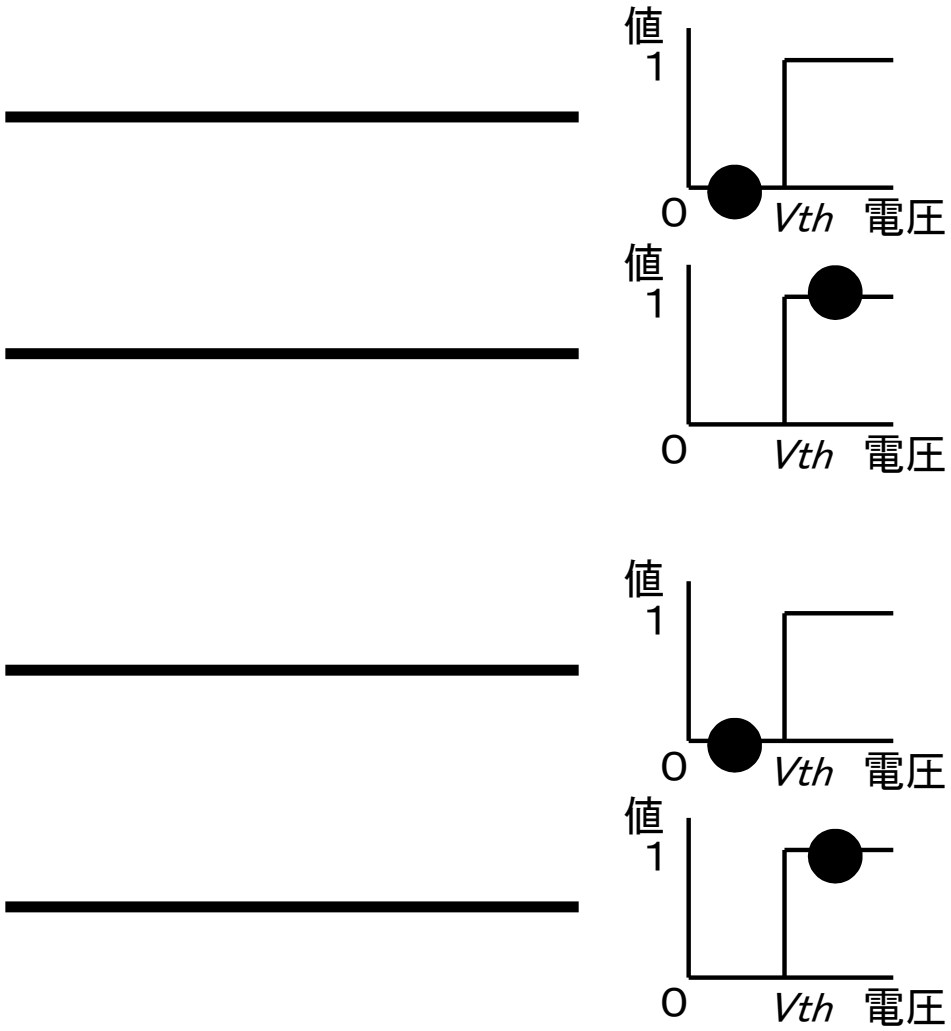
- ビット（bit, binary digit, **b**）
  - 情報の単位。0か1の値を取る

- 電線で自然数を表現する
  - 電線を束ねて、それぞれの線で「桁」を表現していると考える



- バイト（B）
  - 8ビットのこと。2進数で8桁、16進数で2桁

n本の線によるnビット自然数の表現



値　＝　0101　（10進数の5）

数の変換

- ２進数⇒１０進数

$$X_{n-1}2^{n-1}+X_{n-2}2^{n-2}+\cdots+X_1 2+X_0+X_{-1}2^{-1}+X_{-2}2^{-2}+\cdots+X_{-m+1}2^{-m+1}$$

の計算を１０進数の加算・乗算のルールで行えばよい。

- １０進数⇒２進数
  - １０進数を2で割り、商を2で割り、さらに商を2で割り、と、商が1になるまで2で割ることを繰り返し、最後の商、最後の剰余、2番目の剰余、・・・・　最初の剰余と順番に並べればよい。
  - 小数点以下は「2倍にすること」を繰り返して小数点から上にでてきた数（0または1）を並べていく

数の数え方

- 数の単位
  - m(ミリ、10^{-3})、 μ (マイクロ、10^{-6})、n(ナノ、10^{-9})、p(ピコ、10^{-12})、f(フェムト、10^{-15})
  - K(キロ、10^{3})、M(メガ、10^{6})、G(ギガ、10^{9})、T(テラ、10^{12})、P(ペタ、10^{15})、E(エクサ、10^{18})
- 使用例
  - 現在のLSIに含まれるトランジスタの数
    - 1G (giga)個 = 1.0 × 10^{9}個 ⇒ 10億個のトランジスタ
  - 光が真空中を30cm進むのに要する時間
    - 1 ns (nanosecond)= 10^{-9} s
  - 世界最高速級のコンピュータ: Summit （USA）
    - 200 PFLOPS (teraflops) = 2 × 10^{17} FLOPS

      ⇒ 1秒間に20京回浮動小数点数(実数)の計算をする
  - メモリチップの容量
    - 4 Gb (gigabits) = 4 × 10^{9} b

      ⇒ 40億ビット

負の数の表現

- 負の数： 2の補数表現を使う

  $-X \rightarrow 2^n - X$

  （Xの各桁の1と0を反転し、結果に1を加えたもの）

符号	Xが正のときX Xが負のとき$2^n - X$

2の補数表示

符号は、正のとき0，負のとき1

-2^{n-1}から$2^{n-1}-1$までの数を表すことができる

符号あり整数（メモ）

負の整数は２の補数で表現する

ビット反転して１を足す

4は0100だから，−4は0100→1011→1100

4ビットで −8 ~ 7 の整数を表現できる

mビットで

$-2^{m-1} \sim 2^{m-1} - 1$ の整数を表現できる

減算は加算で計算できる

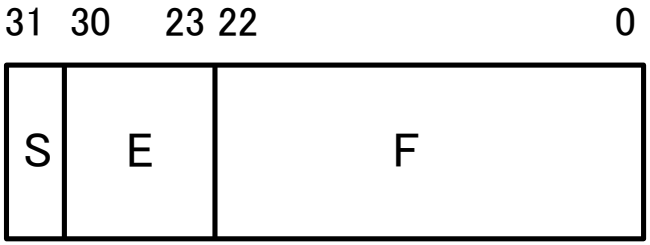
$$7 - 4_{(10)} = 7 + (-4)_{(10)}$$
$$= 0111 + 1100 = 10011 = 3_{(10)}$$

※桁あふれは無視する

数値	ビット列
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
−8	1000
−7	1001
−6	1010
−5	1011
−4	1100
−3	1101
−2	1110
−1	1111

実数の表現

- 固定小数点による表現
  - 整数の表現と同じ。何桁目かに小数点があると「約束」
  - 特別な回路を用意する必要なし
  - 演算のたびに小数点の位置合わせのためのシフト(shift, 桁移動)が必要。プログラムで。
- 浮動小数点による表現
  - 演算のために特別な回路を用意する。プログラムによる補正は不要。



$$
\begin{cases}
E = 0 & \begin{cases} F = 0 & 0 \\ F \neq 0 & (-1)^S \times (0.F) \times 2^{-126} \end{cases} \\
0 < E < 255 & (-1)^S \times (1.F) \times 2^{E-127} \\
E = 255 & \begin{cases} F = 0 & (-1)^S \times \infty \\ F \neq 0 & NaN \text{ (No Number)} \end{cases}
\end{cases}
$$

- 実数は近似値で表現している。誤差解析が必要

2進数の演算（1ビット）

1ビットの四則演算

X	Y	加算 X＋Y	桁上げ	減算 X－Y	借り	乗算 X×Y	除算 X÷Y
0	0	0	0	0	0	0	－
0	1	1	0	1	1	0	0
1	0	1	0	1	0	0	－
1	1	0	1	0	0	1	1

桁上がりのある加算

X	Y	桁上げ入力	X＋Y	桁上げ出力
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Nビット加算　X　＋　Y

- X ≧ 0,　Y ≧ 0
  - 各桁のビットについて加算を行う
  - 最上位のビットが 1 になったら、桁あふれ（オーバフロー、overflow）とみなす

  （例）　0011 + 0010 = 0101（3+2=5）
  　　0011 + 0101 = 1000（3+5→桁あふれ）
- X ≧ 0,　Y ＜ 0　　（ X ＜ 0,　Y ≧ 0）
  - 各桁のビットについて加算を行う
  - 桁あふれは起こらない

  （例）　0011 + 1110 = 0001（3+(-2)=1）
  　　0011 + 1011 = 1110（3+(-5)=-2）
- X ＜ 0,　Y ＜ 0
  - 各桁のビットについて加算を行う
  - 最上位のビットが 0 となったら、桁あふれとみなす

  （例）　1101 + 1110 = 1011　(-3+(-2) = -5)
  　　1101 + 1010 = 0111　(-3+(-6) →桁あふれ)

Nビット減算　X　－　Y

- 減算は、演算数（引く数）の補数をとって被演算数に加える操作を行えばよい
- X ≧ 0,　Y ≧ 0

  （例）0011 - 0010 = 0011 + 1110 = 0001（3-2= 3+(-2) = 1）

  0011 - 0101 = 0011 + 1011 = 1110（3-5= 3+(-5) = -2）
- X ≧ 0,　Y ＜ 0

  （例）0011 - 1110 = 0011 + 0010 = 0101（3-(-2) = 3+(+2) = 5）

  0011 - 1011 = 0011 + 0101 = 1000（3-(-5) = 3+5 →桁あふれ）
- X ＜ 0,　Y ≧ 0

  （例）1110 - 0011 = 1110 + 1101 = 1011（-2 - 3 = -2+(-3) = -5）

  1101 - 0110 = 1101 + 1010 = 1000（−3-6 = -3+(-6) →桁あふれ）
- X ＜ 0,　Y ＜ 0

  （例）1101 - 1110 = 1101 + 0010 = 1111 (-3-(-2) = -3+(+2) = -1)

  1101 - 1010 = 1101 + 0110 = 0011 (-3-(-6) = -3+(+6) = 3)

Nビット乗算 0101＊0011

(1) 被乗数を２Ｎビットに符号拡張する

２N～N+1ビット目までに、下からNビット目の値を入れる（最下位を1ビット目と数えている）

0101 → 00000101　　1011 → 11111011

(2)1ビット目からN－1ビット目まで普通に被乗数×乗数の計算をする。

```
  00000101
 00000101
+      00
----------
  00001111
```

(3) (2) の値からNビット目の乗算結果を引く （乗数が正の場合は(2)で終わり）

乗算の例

```
  00000101
  ×    0011
----------
  00000101
 00000101
 +      00
----------
  00001111
```

(a)　5×3＝15

```
  00000101
  ×    1101
----------
  00000101
+ 000001010
----------
    11001
-   0101
----------
  11110001
```

(a)　5×（－3）＝　－15

```
  11111011
  ×    0011
----------
  11111011
 11111011
 +      00
----------
  11110001
```

(c)　（－5）×3＝－15

```
  11111011
  ×    1101
----------
  11111011
+ 111110110
----------
  11100111
-11111011
----------
  00001111
```

(d)　（－5）×（－3）＝　15

Nビット除算：正数どうし

(1) 被除数と除数の最上位の１がそろうまで除数を左シフト（２倍）する。このときのシフトの回数をＳとする。

(2) 被除数から除数を引く。

(3) 結果が正なら商に１をたてる。負なら０をたてて、除数を足しもどす。

(4) 除数を１ビット右シフトして、(1)(2)を繰り返す。除数のビットをＳ回右シフトしたときの操作で修了。

Nビット除算：一般形

(1) 被除数を２ n－１ビットに符号拡張する。すなわち、２ n－１～ n＋１ビット目までに、下から n ビット目の値を入れる。これをD_0とする。
(2) 除数を n－１ビット左シフトする。すなわち、除数に 2^{n-1} をかける。これをD_1とする。
(3) Q＝０とする（商の初期値の設定）。
(4) D_0とD_1の符号が同じか、またはD_0＝０のとき、D_0＝D_0－D_1。そうでないとき、D_0＝D_0＋D_1とする。
(5) 新しいD_0とD_1の符号が同じか、または新しいD_0＝０のとき、Qの一番右のビットを１にする。そうでないとき、Qの一番右のビットを０とする。
(6) D_1を１ビット右シフトする。Qを１ビット左シフトする。
(7) (4)-(6)を n－１回繰り返す。最後に(4)(5)をもう一度行う。
(8) Qの一番右のビットが０のとき、D_0＝D_0＋D_1とする。
(9) 商はQ、余りはD_0となる。なお、このとき余りの符号は除数の符号に合わせてある。

本方式の正当性を確認せよ！

除算の例(1)

				Q
	0 0 0 0 1 0 1	D_0		0
−	0 0 1 1 0 0 0	D_1		
	1 1 0 1 1 0 1	負	→	0
		↓	シフト	
＋	0 0 0 1 1 0 0			0 0
	1 1 1 1 0 0 1	負	→	0 0
		↓	シフト	
＋	0 0 0 0 1 1 0			0 0 0
	1 1 1 1 1 1 1	負	→	0 0 0
		↓	シフト	
＋	0 0 0 0 0 1 1			0 0 0 0
	0 0 0 0 0 1 0	正	→	0 0 0 1
	あまり			

(a)　5 ÷ 3 = 1 … 2

除算の例(2)

			Q
	0 0 0 **0 1 0 1**		0
＋	**1 1 0 1** 0 0 0		
	1 1 0 1 1 0 1	負 →	1
		↓ シフト	
－	1 1 1 0 1 0 0		1 0
	1 1 1 1 0 0 1	負 →	1 1
		↓ シフト	
－	1 1 1 1 0 1 0		1 1 0
	1 1 1 1 1 1 1	負 →	1 1 1
		↓ シフト	
－	1 1 1 1 1 0 1		1 1 1 0
	0 0 0 0 0 1 0	正 →	**1 1 1 0**
＋	1 1 1 1 1 0 1		
	1 1 1 1 1 1 1 1		

あまり

(b) 5 ÷ (－3) ＝－2…－1

(0101 ÷ 1101=1110… 1111)

除算の例(3)

			Q
	1 1 1 [1 0 1 1]		0
＋	[0 0 1 1] 0 0 0		
	0 0 1 0 0 1 1	正 →	1
		↓ シフト	
－	0 0 0 1 1 0 0		1 0
	0 0 0 0 1 1 1	正 →	1 1
		↓ シフト	
－	0 0 0 0 1 1 0		1 1 0
	0 0 0 0 0 0 1	正 →	1 1 1
		↓ シフト	
－	0 0 0 0 0 1 1		1 1 1 0
	1 1 1 1 1 1 0	負 →	[1 1 1 0]
＋	0 0 0 0 0 1 1		
	[0 0 0 0 0 0 1]		
	あまり		

(c) （－5）÷3＝－2…1

(1011 ÷ 0011=1110… 0000)

除算の例(4)

			Q
	1 1 1 [1 0 1 1]		0
−	[1 1 0 1] 0 0 0		
	0 0 1 0 0 1 1	正 →	0
		↓ シフト	
＋	1 1 1 0 1 0 0		0 0
	0 0 0 0 1 1 1	正 →	0 0
		↓ シフト	
＋	1 1 1 1 0 1 0		0 0 0
	0 0 0 0 0 0 1	正 →	0 0 0
		↓ シフト	
＋	1 1 1 1 1 0 1		0 0 0 0
	[1 1 1 1 1 1 0]	負 →	[0 0 0 1]
	あまり		

(d) （－5）÷（－3）＝1…－2

(1011 ÷ 1101=0001 … 1110)

2進数の演算：例題　＝　宿題

- アナログ方式の温度計とディジタル方式の温度計の特徴を比較せよ。
- 次の2進数を10進数に変換せよ

  101011,　0.11,　110.101
- 次の10進数を2進数に変換せよ。また、それぞれを16進数に変換せよ。

  25,　102.5,　40.675, 0.4
- 引き放し法による除算の仕組みを解説せよ。