

まえがき

本書では、コンピュータの基本である論理回路の豊富で高度な内容を、初めての方でも十分納得できるように、やさしく丁寧に体系的に述べている。できる限り直感的に納得できるように考慮しながらも、同時に論理的であるように心がけて、基礎から高度な内容までスムーズに理解できるように構成している。

本文中で説明すると難解になりそうなところでは、例題や演習問題を解くことによって理解が深まるようにすると共に、その理解度をチェックできるようにしている。論理回路の本質をわかりやすく、直感的にも理論的にも理解できるような教科書となることが本書の重要な目的である。「わかりやすいけれど浅い、高度な内容だけれどわかりにくい」教科書とならないことに気をつけて構成している。

論理回路の基礎的な事柄として、ブール代数、論理関数の表現と簡単化、順序回路の原理について述べ、コンピュータのハードウェアの基礎として、論理素子を組み合わせた組合せ論理回路や、フリップフロップを記憶素子として用いた順序回路の設計法について述べている。特に、一般の教科書では詳しく取り上げられることが少ない主乗法標準形の意味やその簡単化についても記述している。

また、コンピュータにますます高速処理が要求されてきている状況に鑑み、高速演算方式についても一章を設け、近年発展が著しいプログラマブル論理素子についても説明し、アナログ世界とデジタル世界を関連づけるために、アナログ、デジタル変換の原理についても述べている。最近のデジタル回路(論理回路)はどんどん複雑になり、多くのデジタル回路は“専用の言語”を使って設計されるようになってきている。そのため、「ハードウェア記述言語

HDLによる論理設計」を付録としてつけた。

本書はデジタル回路である論理回路に重点を置いているので、その基本素子である AND 素子や OR 素子の構成などの電子回路に関する記述を最後の章(9章)にしている。電子回路に関する知識が先に必要であると思われる方は、9章をはじめに学習してから1章にお進みください。

2013年8月

曾和 将容
範 公可

目 次

1 数の表現

1.1 2 進 数	1
1.2 2進数の負数表現	3
1.3 アスキーコード	5
演 習 問 題	6

2 ブール代数と論理関数

2.1 ブール代数入門	7
2.2 論 理 関 数	8
2.2.1 論 理 積	8
2.2.2 論 理 和	9
2.2.3 否 定 論 理	10
2.3 ブール代数の公理と基本定理	11
2.3.1 公 理	11
2.3.2 基 本 定 理	15
2.3.3 双 対 性	16
2.4 論理関数の標準形	17
2.4.1 主加法標準形	17
2.4.2 主乗法標準形	19
2.5 展 開 定 理	21
2.6 基本定理の証明	23

2.7 主加法標準形の導出	25
2.8 主乗法標準形の導出	27
演習問題	30

3 論理関数の簡単化

3.1 カルノー図による主加法標準形の簡単化	31
3.2 カルノー図による主乗法標準形の簡単化	37
3.3 クワイン・マクラスキー法による簡単化	40
3.4 コンセンサス法による簡単化	44
演習問題	45

4 いろいろな組合せ論理回路

4.1 正論理と負論理, 論理回路と基本素子	46
4.2 排他的論理和回路	48
4.3 半加算器, 全加算器	49
4.4 比較器	51
4.5 マルチプレクサ (セレクタ) とデマルチプレクサ	51
4.6 エンコーダとデコーダ	53
4.7 加減算器	54
4.8 桁上げ先見加算器	55
4.9 乗算器	57
4.10 除算器	58
4.11 論理回路のハザード	61
演習問題	63

5 フリップフロップ

5.1 RS フリップフロップ	64
5.2 JK フリップフロップ	67
5.3 T フリップフロップ	68
5.4 D フリップフロップ	68
5.5 同期式フリップフロップ	69
5.6 マスタスレーブフリップフロップ	70
5.7 直接セットリセット端子つき同期式フリップフロップ	73
5.8 簡単なフリップフロップ応用回路	74
5.8.1 レジスタ	74
5.8.2 シフトレジスタ	74
5.8.3 カウンタ	76
5.8.4 セルフスタートつきリングカウンタ	77
5.8.5 バレルシフトレジスタ	78
演習問題	79

6 順序回路

6.1 順序回路の基礎	80
6.2 順序回路の表現	82
6.3 4進カウンタの設計	83
6.4 状態の簡単化	86
演習問題	88

7 アナログ-デジタル変換

7.1 デジタル-アナログ変換 (D-A 変換)	89
7.2 アナログ-デジタル変換 (A-D 変換)	91
演 習 問 題	93

8 高速演算方式

8.1 桁上げ保存加算器	94
8.2 SD 表現による並列加減算	95
8.3 配列型乗算器	98
8.4 複数ビット走査型乗算	100
8.5 符号つきディジット 2 進数表示による乗算	101
8.6 Booth の乗算器	102
8.7 SRT 除 算 法	104
8.8 浮動小数点乗算および加減算法	105
演 習 問 題	108

9 基本論理素子の電子回路

9.1 基本半導体素子	109
9.2 NOT 論理素子	110
9.3 コンプリメンタリ回路	112
9.4 ハイインピーダンス状態	113
9.5 ダイオードによる論理回路	114
9.6 FET コンプリメンタリ論理回路	114

9.7	トランジスタによる論理回路	116
9.8	ワイヤード OR 回路とバスゲート	117
9.9	ダイナミック論理回路	118
9.10	雑音余裕度とファンアウト, 伝搬遅延	119
9.11	メモリ (RAM)	122
9.12	フラッシュメモリと ROM	125
9.13	プログラマブルデバイス	127
	演習問題	128

付 録 ハードウェア記述言語 HDL による論理設計

A.1	Verilog HDL による論理設計概要	130
A.2	Verilog HDL による論理記述	132
A.3	Verilog HDL によるシミュレーション	133
A.4	簡単な組合せ論理回路の Verilog HDL 表現	136
A.5	簡単な順序回路の Verilog HDL による表現	138

引用・参考文献 141

演習問題解答 142

索 引 159

COMPUTER SCIENCE TEXTBOOK SERIES □

1 数の表現



論理回路では情報を0と1の2文字によって表す。ここでは、0と1とで数値や文字を表す方法について述べる。

1.1 2 進 数

0と1を用いて数値を表現するには2進数を使う。2進数は0110のように表される。2進数の各桁は**ビット (bit)**と呼ばれており、8桁を1**バイト (byte)**と呼ぶ。2進数の値、例えば101の値は、各桁の値に2のべき乗を掛けることにより

$$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 4 + 0 + 1 = 5$$

と求められる。

3ビットの2進数では**表 1.1**のように0から7まで表すことができる。4ビットでは0から15まで (**表 1.2**)、8ビットでは0から255、16ビットでは0か

表 1.1 2進数

10進数	2進数
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

2 1. 数の表現

表 1.2 4ビット2進数と8進数, 16進数表示

10進数	2進数表示	16進数表示	8進数表示
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

ら 65 535 で表すことができる。

数が大きくなる場合は、10進数のキロである 1 000 に近い 1 024 (2^{10}) が **1 K** (ケイ) または **キロ** として使われる。1 024 K は **1 M** (メガ), 1 024 M は **1 G** (ギガ), 1 024 G は **1 T** (テラ) と呼ばれる。

2進数 n ビットで表すことのできる値は 0 から $(2^n - 1)$ の 2^n 個である。2進数の最上位桁は **MSB** (most significant

bit), 最下位桁は **LSB** (least significant bit) と呼ばれる。

2進数は 0 と 1 の羅列で表すので、人間にとっては扱いにくい数値表現である。そのため、表 1.2 に示すように、3桁または4桁をひとかたまりとして扱う方法が考えられている。3桁では 0 から 7 まで表現できるので **8進数** (octal number) として、また4桁では 0 から 15 まで表現できるので **16進数** (hexadecimal number) として表現する。

16進数では 0 から 15 を表す文字が必要である。0 から 9 までは 10進数の文字そのものが使えるが、10 から 15 を表す文字は 10進数にはないので新たな文字が必要である。この文字として A, B, C, D, E, F を使う。すなわち、10 を 16進数で表すのに A を、11 を表すのに B を、12 を表すのに C を、13 を表すのに D を、14 を表すのに E を、15 を表すのに F を用いる。

例えば、2進数 01010010101 は 16進数表示では

$$0101\ 1010\ 1111 \rightarrow 5AF$$

のように、5AF と表される。8進数表示では

$$010\ 110\ 101\ 111 \rightarrow 2657$$

のように、2657と表される。

4ビットで表すことができる0から15のうち、0から9までを使って10進数を表す方法がある。これを**2進化10進符号**（BCDコード：binary coded decimal code）と呼ぶ。例えば、2進化10進符号で10進数365を表現するのに、2進数の3（0011）、6（0110）、5（0101）をそのまま並べて

0011 0110 0101

と表現する。この値は

$$1 \times 2^9 + 1 \times 2^8 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^2 + 1 \times 2^0 = 512 + 256 + 64 + 32 + 4 + 1 = 869$$

ではなく、10進数365であることに注意が必要である。

1.2 2進数の負数表現

2進数による負の値の表現には、表1.3(a)、(b)に示すように、**1の補数表現**（補数表現）と**2の補数表現**（縮小基数表現）が使われる。

表1.3 負の数の表現方法

(a) 1の補数表現		(b) 2の補数表現	
10進数	2進数	10進数	2進数
-3	100	-4	100
-2	101	-3	101
-1	110	-2	110
-0	111	-1	111
+0	000	+0	000
+1	001	+1	001
+2	010	+2	010
+3	011	+3	011

0の表現
2個 →

← 負の表現が
1個増える

← 0の表現
1個

1の補数表現の各桁は、(基数-1)から正の2進数の各桁を引くことによって求められる。2進数では基数は2であるので、(基数-1)は(2-1=1)から1である。したがって、1の補数表現の各桁は、1から正の2進数の各桁を減じて求めるということができる。

例えば、011の最上位桁は(2-1)-0=1、第2桁は(2-1)-1=0、第1桁

4 1. 数の表現

は $(2-1)-1=0$ となり、1の補数表現として100が求められる。

正の2進数表現	011
1の補数表現	100

この例からわかるように、1の補数は結果的にもとの2進数の1、0を入れ替えたものとなる。

3桁の2進数では8個の数値表現が可能であり、表1.3(a)に示すように、1の補数では-3から+3までを表現することができる。8個のうち2個が+0(2進数では000)と-0(2進数では111)の表現に使われおり、少し表現効率が悪い。

2の補数表現は、1の補数に1を加算したものである。表(b)のように零の表現には000一つが使われているだけで、表現できる数が一つ増える。そのため2の補数表現では、-4から+3まで表現することができる。この利点からコンピュータでの計算には2の補数表現が使われることが多い。

1の補数、2の補数表現では、正数のとき最上位ビットが0、負数では1となるので、このビットは正負の符号を表す**符号ビット**と呼ばれることも多い。

【例題 1.1】

- (1) 4ビットで表現できる数の範囲を、正数表現のみの場合と、1と2の補数表現を用いる場合について書け。

解答	正の整数のみ	0から15
	1の補数表現	-7から+7
	2の補数表現	-8から+7

- (2) 0101 1101, 0111 0010の2の補数表現を求めよ。

解答	0101 1101	→	1010 0011
	0111 0011	→	1000 1101

◇

2の補数を使った計算では、減算を負数の加算として行う。例えば、10進数 $3-1=2$ の計算は、3ビットの2の補数を使うと

$$\begin{array}{r} 3 \quad 011 \\ +) -1 \quad +) 111 \\ \hline 2 \quad 1\boxed{010} \end{array}$$

となって、010 (10進数で2) が得られる。結果の4ビット目は、計算が3ビット2進数の計算であるので無視する。

2の補数を使った計算では、その補数が表現できる範囲(値域)を超えた結果となる計算はできない。例えば、3ビットの場合、10進数 $3+2=5$ の計算は

$$\begin{array}{r} 3 \quad 011 \\ +) 2 \quad +) 010 \\ \hline 5 \quad 101 \rightarrow -3 \end{array}$$

となり、5となるべきところが-3になってしまう。これは5を表現したくても5の表現方法がないからである。このような現象を、計算結果が3ビットの表現値域を超えたということで**オーバーフロー** (overflow) という。

したがって、3ビットの場合、2の補数を使う計算では、計算結果が-4~+3までの間になるようにしなければならない。2の補数を使った計算の結果がオーバーフローしているかどうかは、次のようにして確かめることができる。

「最上位桁からの桁上げと、最上位桁への桁上げが両方あるか両方ないとき、計算はオーバーフローしていない。」

1.3 アスキーコード

人間世界で使っているアルファベットや記号などの文字と、0, 1で表されるデジタル世界との情報交換用に**アスキー** (ASCII: American Standard Code for Information Interchange) **コード**と呼ばれる米国の情報交換用標準コードが使われてきた。アスキーコードは、表1.4に示すように7ビットでアルファベットや記号を表す。例えば、ronriはアスキーコードでは

$$\begin{array}{cccccc} r & o & n & r & i & \\ 111\ 0010 & 110\ 1111 & 110\ 1110 & 111\ 0010 & 110\ 1001 & \end{array}$$

となる。行替え (LF: 0001010) や1字戻り (BS: 0001000), ベル (BEL: 0000111) などもある。数字1の表現は011 0001, 2の表現は011 0010である。

コンピュータでは1バイトが基準となることが多いので、最上位ビットに0

表 1.4 ASCII コード

				上位				下位			
				6	0	0	0	0	0	1	1
3	2	1	0	4	0	1	0	1	0	1	0
0	0	0	0		NUL	DLE	SP	0	@	P	`
0	0	0	1		SOH	DC1	!	1	A	Q	a
0	0	1	0		STX	DC2	"	2	B	R	b
0	0	1	1		ETX	DC3	#	3	C	S	c
0	1	0	0		EOT	DC4	\$	4	D	T	d
0	1	0	1		ENQ	NAK	%	5	E	U	e
0	1	1	0		ACK	SYN	&	6	F	V	f
0	1	1	1		BEL	ETB	'	7	G	W	g
1	0	0	0		BS	CAN	(8	H	X	h
1	0	0	1		HT	EM)	9	I	Y	i
1	0	1	0		LF	SUB	*	:	J	Z	j
1	0	1	1		VT	ESC	+	;	K	[{
1	1	0	0		FF	FS	,	<	L	\	l
1	1	0	1		CR	GS	-	=	M]	m
1	1	1	0		SO	RS	.	>	N	^	n
1	1	1	1		SI	US	/	?	O	_	o
											DEL

を加えて8ビットに拡張することも多い。

演 習 問 題

- 【1】 4ビットの1の補数, 2の補数を書け。
- 【2】 ASCIIコードで kairo を表せ。
- 【3】 3ビットの2の補数表現を用いて次の計算を行い, 結果の正否を確かめよ。
 (1) $3+(-1)=$ (2) $(-2)+(-1)=$ (3) $2+3=$
- 【4】 8ビットからなる2の補数を使って次の計算を行い, 結果の正否を確認せよ。
 (1) $124-67=$ (2) $124+72=$
 (3) $-15-21=$ (4) $127-124=$
- 【5】 2の補数を使った計算で, 「最上位桁への桁上げと最上位桁からの桁上げの両方があるか両方がないとき, 計算はオーバーフローしていない」ことを説明せよ。

索 引

【あ】	クロックパルス	69
アスキーコード	5	
アンダシュート	69	
【い】	クワイン・マクラスキー法	40
インスタンス化	134	
インスタンス名	134	
インバータ回路	111	
インピーダンス	120	
【え】	加え戻し法	60
エッジトリガ FF	72	
エンコーダ	53	
【お】	【け、こ】	
オーバシュート	69	
オーバフロー	5	
オープンコレクタ接続	117	
オープンドレイン接続	117	
【か】	ケ イ	2
カウンタ	76	
荷重抵抗型 D-A 変換回路	91	
カラムアドレス	122	
カルノー図	31	
貫通電流	113	
【き】	桁上げ先見加算器	55
記憶素子	64	
ギ ガ	2	
行セレクト線	122	
キ ロ	2	
【く】	桁上げ保存加算器	95
組合せ論理回路	64	
	ゲート回路	52
	現在の回路の状態	80
	コンセンサス	44
	【さ】	
	最小 SD 表示	96
	最小項	17
	最大項	17
	サンプリング定理	91
	【し】	
	時間の経過	64
	しきい値素子	72
	シフトレジスタ	74
	シミュレータ	130
	主加法標準形	17
	主項表	41
	主乗法標準形	17
	出力関数	81
	出力表	83
	出力ポート	132
	順序回路	64, 80
	状態遷移関数	81
	状態遷移図	83
	状態遷移表	83
	状態遷延回路	81
	真理値表	8
	【せ】	
	正論理回路	46
	正論理論理回路	46
	積 項	17
	セルフスタートつきリング	
	カウンタ	77
	セレクト	51
	全加算器	49
	【そ】	
	双対性	16
	双対性定理	16
	相補性回路	113
	【た】	
	ダイオード	109
	ダイナミックメモリ	123
	ダイナミック論理回路	118
	立上り遅延時間	122
	立下り遅延時間	122
	【ち】	
	値 域	5
	遅延回路	81
	遅延形フリップフロップ	69
	チャタリング	69
	直接セットリセット端子	
	つきフリップフロップ	73
	直並列データ変換回路	75
	【つ】	
	次の状態	81
	【て】	
	デコーダ	53

テストベンチ	130	半加算器	49	マスタスレーブ FF	71
デマルチプレクサ	52			マルチプレクサ	51
デュアルポート RAM	124	【ひ】		【め】	
テラ	2	比較器	51	命題算	7
電界効果トランジスタ	109	引き離し法	60	メガ	2
展開定理	21	必須主項	43	メモリセル	122
伝播遅延	122	ビット	1	【も】	
【と】		ビット線	122	モジュール	132
同期式 RS-FF	70	ビット列リコード型乗算	101	【よ】	
同期式論理回路	69	否定	8, 11	予約語	132
トグル	68	否定論理	11	【り】	
トグルフリップフロップ	68	非同期式論理回路	69	リフレッシュ動作	124
ドミノ論理回路	119	【ふ】		リングカウンタ	77
ド・モルガンの定理	15	ファンアウト数	121	【れ】	
トライステート	113	ファンイン数	121	励振表	84
トランジスタ	109	符号つきディジット	95	レジスタ	74
ドントケア項	34	符号ビット	4	レジスタ転送レベル	131
【に】		フラッシュメモリ	125	接続演算子	138
入力インピーダンス	109	プリセット端子つき		【ろ】	
入力ポート	132	フリップフロップ	73	ロードアドレス	122
【ね】		フリップフロップ	64	論理演算	8
ネガティブエッジ	72	ブール代数	7	論理関数	8
ネットリスト	131	ブール変数	7	論理合成ツール	131
【は】		プログラマブルデバイス	127	論理積	8, 9
ハイインピーダンス状態	113	フローティングゲート	125	論理変数	7
排他的論理和	48	負論理回路	46	論理和	8, 9
バイト	1	負論理論理回路	46	【わ】	
ハザード	61	【へ, ほ】		ワイアード OR 回路	117
バスの衝突	117	並列データ	74	和項	17
ハードウェア記述言語	130	ベン図	9		
パラレルデータ	74	ポジティブエッジ	72		
パレルシフトレジスタ	79	【ま】			
反 SD 数	96	交わり部分	9		
		マスク ROM	127		

【A】	AND ゲート	70	ASCII	5
AND 演算	AND 論理	9		
	ANSI	9		

【B】		【J】		pMOS 型	109
BCD コード	3	JK フリップフロップ	67	pnp 型	110
bit	1			PROM	127
Booth の乗算器	102	【K】		【R】	
byte	1	K	2	RAM	122
【C】		【L】		ROM	127
CLK	69	L レベル	7, 46	RS フリップフロップ	64
【D】		——の雑音余裕度	120	RTL	131
D フリップフロップ	69	LSB	2	【S】	
DTL	116	【M】		SD	95
【E】		M	2	System C	130
EPROM	127	MSB	2	【T】	
【F】		【N】		T	2
FA	49	NAND 回路	46	T フリップフロップ	68
FET	109	nMOS 型	109	TTL	116
FPGA	127, 128	NOR 回路	46	【V】	
【G】		NOT 演算	8	Verilog HDL	130
G	2	NOT 論理	11	VHDL	130
【H】		npn 型	110	【X】	
H レベル	7, 46	【O】		XOR	48
——の雑音余裕度	120	OFF	7	【数字】	
HA	49	ON	7	1 の補数表現	3
HDL	130	OR 演算	8	2 進化 10 進符号	3
		OR 論理	9	2 の補数表現	3
		【P】		8 進数	2
		PLA	127	16 進数	2

—— 著者略歴 ——

曾和 将容 (そわ まさひろ)

1974年 名古屋大学大学院博士課程修了
(電気・電子工学専攻)
工学博士
1974年 群馬大学助手
1976年 群馬大学助教授
1987年 名古屋工業大学教授
1993年 電気通信大学教授
2009年 電気通信大学名誉教授

範 公可 (ファム コンカ)

1988年 上智大学理工学部電気・電子工学
科卒業
1992年 上智大学大学院博士後期課程修了
(電気・電子工学専攻)
博士(工学)
1992年 上智大学助手
1996年 東京情報大学講師
2000年 電気通信大学助教授
2007年 電気通信大学准教授
現在に至る

論 理 回 路

Logical Circuits

© Masahiro Sowa, Cong-Kha Pham 2013

2013年9月20日 初版第1刷発行

検印省略

著 者 曾 和 将 容
範 公 可
発 行 者 株式会社 コロナ社
代 表 者 牛 来 真 也
印 刷 所 新日本印刷株式会社

112-0011 東京都文京区千石4-46-10

発行所 株式会社 コロナ社

CORONA PUBLISHING CO., LTD.

Tokyo Japan

振替 00140-8-14844・電話 (03) 3941-3131 (代)

ホームページ <http://www.coronasha.co.jp>

ISBN 978-4-339-02705-1

(阿部) (製本: 愛千製本所)

Printed in Japan



本書のコピー、スキャン、デジタル化等の無断複製・転載は著作権法上での例外を除き禁じられております。購入者以外の第三者による本書の電子データ化及び電子書籍化は、いかなる場合も認めておりません。

落丁・乱丁本はお取替えいたします