論理回路基礎 (第12回)

鹿間　信介

摂南大学 理工学部　電気電子工学科

- 組合せ論理回路

7.4　加算器

7.5　補数による減算器

演習

2進数法の加算法則

重要

- **最下位桁の加算（半加算）**

0	0	1	1
+) 0	+) 1	+) 0	+) 1
0	1	1	1 0

桁上げC_0

その桁の和S

- **下の桁からの桁上げを考慮（全加算）**

被加数　X	0	0	0	1	1	1	0	1
加数　Y	0	0	1	0	1	0	1	1
下からの C	+) 0	+) 1	+) 0	+) 0	+) 0	+) 1	+) 1	+) 1
桁上げ	0 0	0 1	0 1	0 1	1 0	1 0	1 0	1 1

桁上げ

2進数法の加算例

- $(13)_{10}+(27)_{10}$　を2進法で行うと
  - $(13)_{10}=(1101)_2$
  - $(27)_{10}=(11011)_2$

2進法	10進法
1101	13
+) 11011	+) 27
101000	40

1 1 1 1 1 ← 桁上げ → 1

出典: 猪飼「ディジタル・システムの設計」p.270

加算器の真理値表

重要

- **半加算器：　下からの桁上げ考慮せず**
- **全加算器：　下からの桁上げも加算**

半加算器(HA)の真理値表

A	B	C_0	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

全加算器(FA)の真理値表

A	B	C_0	C	S
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

半加算器(HA)の論理回路

重要

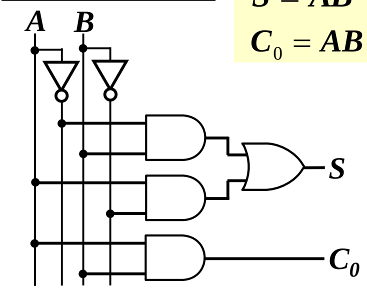
HAの真理値表

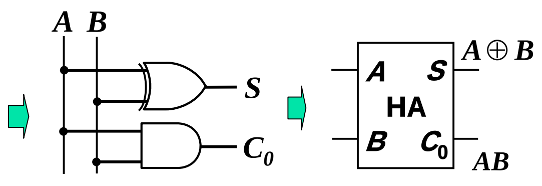
A	*B*	C_0	*S*
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

- 真理値表 ⇒ 論理式
- 論理式 ⇒ 論理回路
- XOR使って回路簡単化
- HAの論理記号で機能を表現

<主加法標準形>

$$S = \bar{A}B + A\bar{B}\ (= A \oplus B)$$
$$C_0 = AB$$





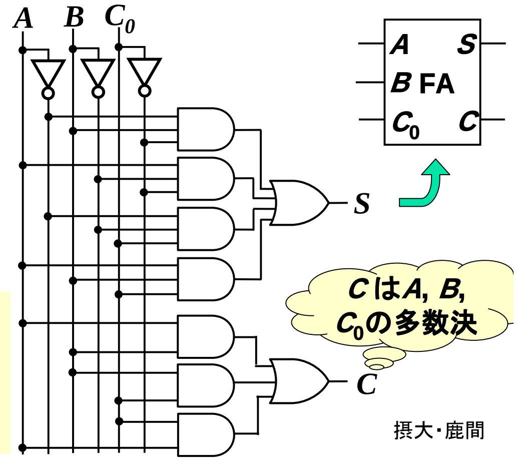
全加算器(FA)の論理回路

重要

FAの真理値表

A	*B*	C_0	*C*	*S*
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

- 真理値表 ⇒ 論理式
- 論理式 ⇒ 論理回路
- FAの論理記号で機能を表現



$$S = \bar{A}\cdot B\cdot \bar{C_0} + A\cdot \bar{B}\cdot \bar{C_0} + \bar{A}\cdot \bar{B}\cdot C_0 + A\cdot B\cdot C_0$$
$$C = AB\bar{C_0} + \bar{A}BC_0 + A\bar{B}C_0 + ABC_0$$
$$= AB + BC_0 + AC_0$$

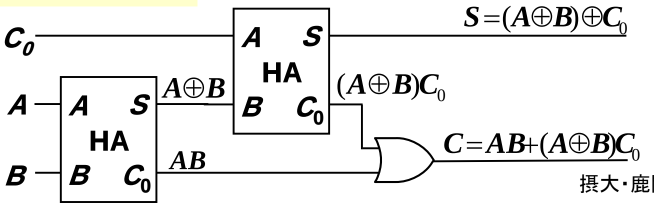
例題7-5 FAがHA2段で構成できることを示し回路を構成せよ

$$S = \bar{A}\cdot B\cdot \bar{C_0} + A\cdot \bar{B}\cdot \bar{C_0} + \bar{A}\cdot \bar{B}\cdot C_0 + A\cdot B\cdot C_0$$
$$= (\bar{A}\cdot B + A\cdot \bar{B})\cdot \bar{C_0} + (\bar{A}\cdot \bar{B} + A\cdot B)\cdot C_0$$
$$= (A\oplus B)\cdot \bar{C_0} + \overline{(A\oplus B)}\cdot C_0$$
$$= (A\oplus B)\oplus C_0$$

$$C = AB\bar{C_0} + \bar{A}BC_0 + A\bar{B}C_0 + ABC_0$$
$$= AB(\bar{C_0} + C_0) + (\bar{A}B + A\bar{B})C_0$$
$$= AB + (A\oplus B)C_0$$

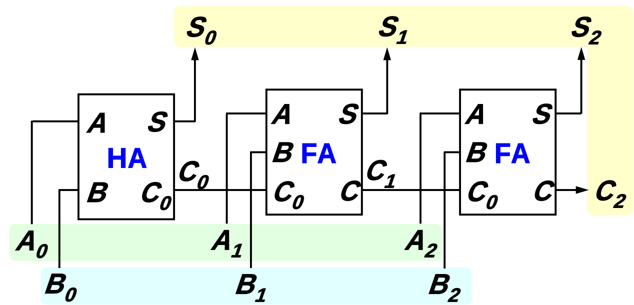
- FAの論理式
  ⇒ XOR演算使う形に変形
- 論理式よりHA2段で構成

$$\overline{(A\oplus B)}$$
$$= \overline{\bar{A}B + A\bar{B}}$$
$$= (\overline{\bar{A}B})(\overline{A\bar{B}})$$
$$= (A + \bar{B})(\bar{A} + B)$$
$$= AB + \bar{A}\cdot \bar{B}$$



リプルキャリー方式加算器

- 3ビット加算器： 入力 $(A_2\ A_1\ A_0)$，$(B_2\ B_1\ B_0)$
  出力 $(C_2\ S_2\ S_1\ S_0)$
- 上位桁に向かって順次桁上げし、加算を実行
  - ripple carry方式：水面に石を放り込んだときにさざ波(ripple)が伝わるのに似ていることが語源
  - 加算器の段数が増えると加算結果の時間遅れ増加

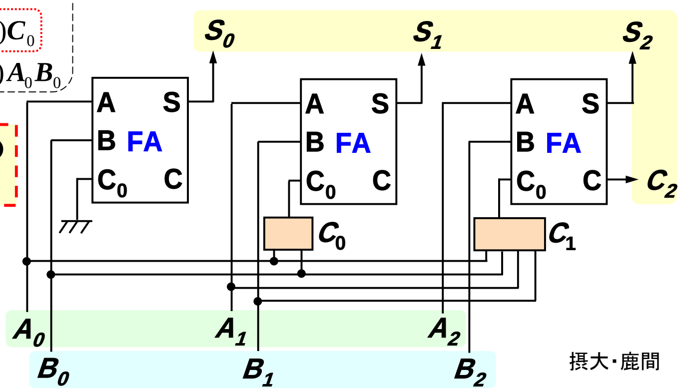


加算時間遅れ解消：ルックアヘッドキャリー方式

- リプルキャリー方式の時間遅れ解消
- 下位の全ての桁上がりを前もって演算し、その桁に加算
  - Look ahead carry（先見桁上げ）方式
  - 桁ごとに、専用の桁上がり演算回路（C_0, C_1, ･･･）を有す

$$C_0 = A_0 B_0$$
$$C_1 = \boxed{A_1 B_1 + (A_1 + B_1) C_0}$$
$$= A_1 B_1 + (A_1 + B_1) A_0 B_0$$

C_1はFA入力の多数決論理！



参考：猪飼, 定本 p.280, T. Bartee, CA p.232

2進数法の減算法則

重要

- 最下位桁の減算（半減算）

0	0	1	1
-) 0	-) 1	-) 0	-) 1
0	1 1	1	0

上の桁からの借りB_0　その桁の差D_0

- 下の桁への貸しを考慮（全減算）

被減数	X	0	0	0	1	1	1	0	1
減数	Y	0	0	1	0	1	0	1	1
下の桁への貸し	b	-) 0	-) 1	-) 0	-) 0	-) 0	-) 1	-) 1	-) 1
		0 0	1 1	1 1	0 1	0 0	0 0	1 0	1 1

上の桁からの借り

2進数法の減算例

- $(72)_{10}$ − $(33)_{10}$ を2進法で行うと
  - $(72)_{10}=(1001000)_2$
  - $(33)_{10}=\ (100001)_2$

2進法	10進法
1001000	72
-) 100001	-) 33
100111	39
1　1 1 1	1

← 上の桁から借り →

出典：猪飼「ディジタル･システムの設計」p.270

負数の表示（10進数の場合）

- （大きい数）-（小さい数）：上の桁からの借りを利用して計算
- （小さい数A）-（大きい数B）をどうする？
  - #1：AとBをひっくり返して減算し、引いた結果に"－"をつける
    - 正数X, 負数-|X| ← 数の絶対値に符号を付ける方法！
    - AとBの大小比較判定回路が必要：回路複雑化
  - #2：ひっくりかえさず強引に引く ⇒ $(13)_{10}$－$(28)_{10}$の例

```
      13
-)    28
  999985
   11111
    借り
```

- 負数の表示
  （･･99）985で-15を表す
- 正数の表示
  +15 ⇒"015"と3桁で表す

正数と負数の加算
(+18)+(-15)

```
    018
+)  985
   1003
```

借りを返して003

★正数と負数の加算が、正数同士の加算と同じように実行可能：正数-正数

出典：猪飼「ディジタル･システムの設計 定本？？」p.271

補数表示について（10進数）

- 補数表示
  - 一定値からある数の絶対値を引いて表示
    - 例：-15を985と表す：1000から-15の絶対値を引いた値
  - 一定値（1000）は扱う負の数の桁が多くなると、10000，100000と大きくなる
  - 一定値を999としてもよい： 9の補数表示
- 補数表示を用いると、減算も加算で処理できる。

出典: 猪飼「ディジタル・システムの設計」p.270

2進数の補数

重要

- 補数： 互いに補い合う数字
  - $Y+C$ = 定数
  - Yが与えられると補数 C が一意に決まる
- 2進数の2の補数
  - 4桁の2進数Yの補数 C_2 は？
  - $Y+C_2=(10000)_2$ -- C_2 を足すと桁上げ（「2の補数」の由来）
  - $C_2=(10000)_2-Y$
- 2進数の1の補数 C_1
  - $Y+C_1=(1111)_2$ --- C_1 の加算で桁上げなし（2の補数より1小さい）

出典: 角山「ディジタル回路の基礎」p.21

2進数の補数表示（その1）

- マイナス符号
  - 10進数では”9”
  - 2進数では”1”
- プラス符号
  - “0”（10進数と同様）
- 表示例 （2の補数表示）

+1101 → 01101

-1101 → 10011
（100000から引いた値）

```
   100000
-) 01101
   10011
```

定義どおりの作り方

```
   01101
   ↓↓↓↓↓     各桁の”0”
   10010     と”1”を反転
+)     1     ”1”を加える
   10011
```

簡単な補数の作り方

出典: 猪飼「定本ディジタル・システムの設計」p.272

2進数の補数表示（その2）

重要

- n桁の2進数Y
  - **1の補数： $C_1=(2^n-1)-Y$**
  - **2の補数： $C_2=2^n-Y$**

$\boxed{C_2=C_1+1}$

- 例（0101の補数）
  - **C_1= (2^4-1)-0101 = (10000-1)-0101 = 1111-0101 = 1010**
  - **C_2= 2^4-0101 = 10000-0101 = 1011**

- **C_1：元の数の各桁の1と0を反転**
- **C_2：C_1 に1を加算**

```
          0101
          ↓↓↓↓    各桁の”0”
C1 --     1010    と”1”を反転
+)           1    ”1”を加える
C2 --     1011
```

補数の作り方

再度 C_2 にすると，正数に戻る
```
    1011
C1-- 0100
C2-- 0101
```

2の補数による数値表示範囲

10進	2の補数(4ビット)
-8	1000
-7	1001
-6	1010
-5	1011
-4	1100
-3	1101
-2	1110
-1	1111
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111

- 最上位桁(MSB)が**0**で”正”
- MSBが**1**で“負”
- 4ビットで$(-8)_{10}$〜$(7)_{10}$を表現

例題7-6　Y=(011010)の補数C_1, C_2を求めよ

符号桁(正数: Y)

Y -- 011010

↓↓↓↓↓↓

C_1 -- 100101　**各桁の”0”と”1”を反転**

+) 1　**”1”を加える**

C_2 -- 100110

符号桁(負数: -Y)

- C_1: Yの各桁の1と0を反転
- C_2: C_1に1を加える
- C_1: インバータを通して作る
  - 0の補数が(11111)となり本来の0(00000)と二通りになり、まぎらわしい
  - 現在はC_1に1を加えた”2の補数”表示が主流

2進数X, Yの減算

重要

- X-Yは、X+(-Y)として、補数の加算で演算する。

【計算手順】

1. **減数Yの1の補数C_{1Y}を求める**
2. **C_{1Y}+1 ⇒　2の補数C_{2Y}**
3. **X＋C_{2Y}の加算を実行**
4. **キャリーC(最上位桁からの桁上がり)あり:**
   ⇒ Cを無視し、残りの数を正の解とする
5. **加算結果キャリーCなし:**
   ⇒ 加算結果の2の補数C_2を求め、負の解とする

例題7-7(a)　2進数の減算X-Yを2の補数の加算で求めよ

(a) X>Y: X=$(0110)_2$ = $(6)_{10}$
Y=$(0010)_2$ = $(2)_{10}$

C_{1Y}=1101

C_{2Y}=C_{1Y}+1=1110

X　0110
+) C_{2Y}　1110
10100
無視する

解: $(0100)_2$=$(4)_{10}$

- **Yの2の補数C_{2Y}を求め、Xに加算する**
- **加算結果に最上位桁からのキャリーCがあるから、Cを無視して残りの数を正の解とする**

例題7-7(b)　2進数の減算X-Yを2の補数の加算で求めよ

(b) X<Y: $X=(0010)_2=(2)_{10}$
$Y=(0110)_2=(6)_{10}$

$C_{1Y}=1001$

$C_{2Y}=C_{1Y}+1=1010$

X	0010
+) C_{2Y}	1010
	1100

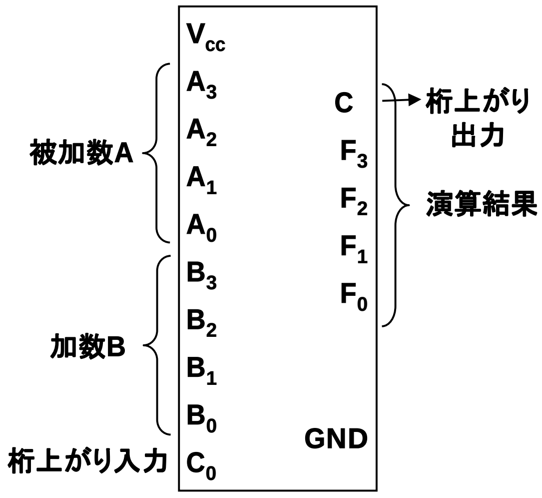
C_1　0011

C_2　0100

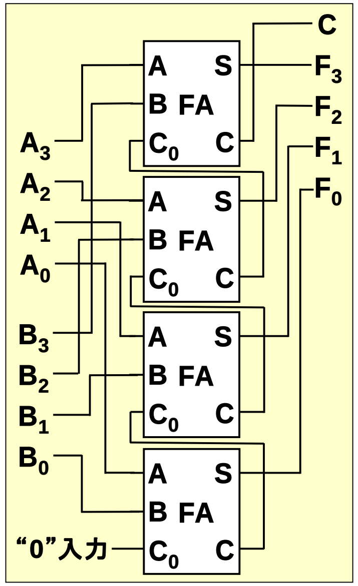
解: $-(0100)_2=-(4)_{10}$

- Yの2の補数C_{2Y}を求め、Xに加算する
- 加算結果に最上位桁からのキャリーCがないから、加算結果の**2の補数C_2を求め、負の解とする**

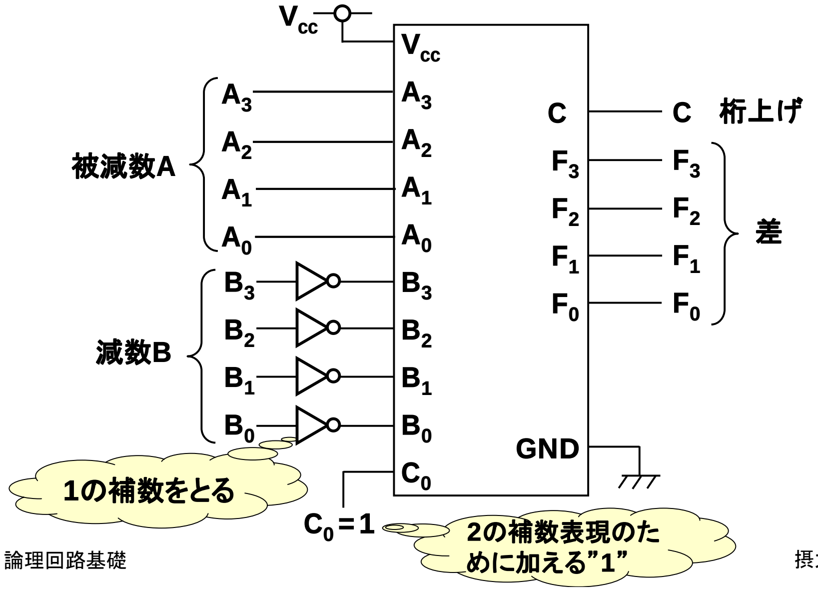
4桁全加算器



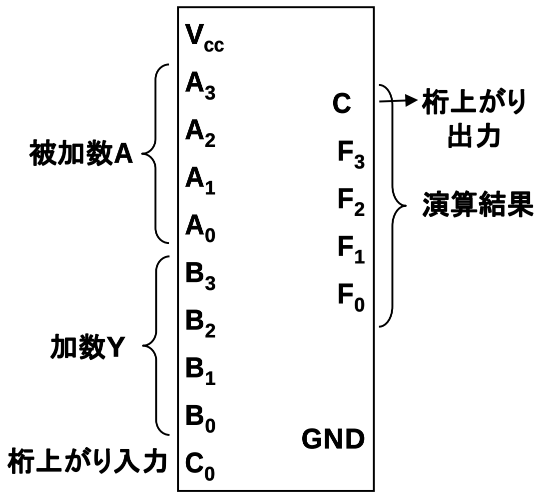
4桁全加算器の内部構成



4桁全加算器による減算回路



例題7-8　4桁全加算器による加算・減算器を設計せよ(その1)

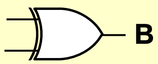


【設計条件】

- 左図の全加算器を使用
- 減算は2の補数で演算
- 加算・減算切り替えを入力信号 SUBで制御
  - SUB=0:　加算 A+Y
  - SUB=1:　減算 A-Y
  - A, Yは正の数

例題7-8　4桁全加算器による加算・減算器を設計せよ（その2）

XORゲートによる Y入力制御



$$B = Y \cdot \overline{SUB} + \overline{Y} \cdot SUB$$

SUB	Y	B
0	0	0
0	1	1
1	0	1
1	1	0