

アナログ電子回路補足

目次

- 補数とは
- 負数の表現
- 2の補数表現
- 1の補数表現
- サインマグニチュード(絶対値表現)
- オフセットバイナリ(オフセット表現)
- 代数系
- ブール代数
- 双対定理
- その他のブール代数の性質
- 論理演算とブール代数の関係
- デジタル回路におけるブール代数
- カルノー図の気持ち
- Static CMOS回路
- オープンドレイン回路
- 様々な回路
- 論理式の変形のコツ(完全にテスト対策専用知識)

補数とは

n 進法において、桁上りが発生する最小の数 を(n 進数における) n の補数という。普段は(n 進数における)は省略する。

ついでに、桁上りが発生しない最小の数 を(n 進数における) $n-1$ の補数という。普段は(n 進数における)は省略する。

例えば2進数 $10010_2 = 18_{10}$ について 2の補数は 1110_2 となり、1の補数は 1101_2

負数の表現

以下2進数の話とする。負数の表現は2進数 a の表現方法を2の補数表現、1の補数表現、サインマグニチュード(絶対値表現)、オフセットバイナリ(オフセット表現)などによって異なってくる。

2の補数表現

2進数nbitで表された数 $a(0 < a < 2^n - 1)$ に対し、 a はmbitで表せるとする。 a に対し、 $-a$ を a の2の補数(足りない部分mbitまでは0埋めそこから先は1埋め)としたものとして定義する表現方法。例えば $n=8$ 、

$$a = 00001101_2 (= 13_{10})$$

とすると13の2補数は

$$13の2の補数 = 11_2$$

a は4bitで表せる数なので $m=4$ であるので、 $-a$ は4bitまでは0埋めそこから先は1埋めとなり

$$-a := 11110011_2$$

となる。

1の補数表現

2進数nbitで表された正数 $a(0 < a < 2^{n-1})$ に対し、 a はmbitで表せるとする。 a に対し、 $-a$ を a の1の補数(足りない部分mbitまでは0埋めそこから先は1埋め)としたものとして定義する表現方法。例えば $n=8$ 、

$$a = 00001101_2 (= 13_{10})$$

とすると13の1補数は

$$13の2の補数 = 10_2$$

a は4bitで表せる数なので $m=4$ であるので、 $-a$ は4bitまでは0埋めそこから先は1埋めとなり

$$-a := 11110010_2$$

となる。

サインマグニチュード(絶対値表現)

2進数nbitで表された正数 $a(0 < a < 2^{n-1})$ に対し、 $-a$ を先頭bitを1にしたものとして定義する表現方法。例えば $n=8$ とすると 例えば

$$a = 00001111_2 \rightarrow -a := 10001111_2$$

となる。

オフセットバイナリ(オフセット表現)

読み取った値 — オフセット値として数を定義する方法。例えばオフセット値を10000000とし、3を表したい場合

$$10000011 - 10000000 = 00000011$$

なので $3 := 10000011$ と定義される。

代数系

まず、少しだけ代数系の記号の意味について説明していく。

無限集合と有限集合

集合 A の要素が無限個ある場合、 A を無限集合といい、有限個の場合を有限集合という。有限集合の場合、演算子等の定義が楽に出来る。

演算

A を集合とし、 $x, y \in A$ とする。写像 $\circ(x, y)$ を考え、

$$\text{写像 } \circ : A \times A \rightarrow A \mid \circ(x, y) = z$$

を A 上の演算という、 $\circ(x, y)$ を $x \circ y$ と表記する。集合 A と一つ以上の演算 \circ が定義されている時 (A, \circ) を代数系という。

作用

A を集合とし、 $x \in A$ とする。写像 $\star(x)$ を考え、

$$\text{写像 } \star : A \rightarrow A \mid \star(x) = y$$

を作用という。

ブール代数

B は少なくとも記号 $(0, 1)$ を含むとする。 $x, y \in B$ に対し、演算 $+$ 、 \cdot と作用 $-$ が定義され、以下の4つの性質が成り立つとき、 $(B, +, \cdot, -)$ をブール代数、または単に B をブール代数という。

1. 交換則

$$x \cdot y = y \cdot x, \quad x + y = y + x$$

2. 単位元の存在(単位元律)

$$x \cdot 1 = x, \quad x + 0 = x$$

3. 相補則

$$x \cdot \bar{x} = 0, \quad x + \bar{x} = 1$$

4. 分配測

$$a \cdot (b + c) = a \cdot b + a \cdot c, \quad a + b \cdot c = (a + b) \cdot (a + c)$$

この性質を満たせばブール代数を名乗れるため、演算 $+$ 、 \cdot と作用 $\bar{}$ の定義方法は色々あるが例えば

$$0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 1$$

$$0 \cdot 0 = 0, 0 \cdot 1 = 0, 1 \cdot 0 = 0, 1 \cdot 1 = 1$$

$$\bar{0} = 1, \bar{1} = 0$$

が各演算及び作用の定義となる。

双対定理

双対定理はたまに役に立つので紹介する。ブール代数において” $+$ ”と” \cdot ”、” 0 ”と” 1 ”を交換しても恒等式は成り立つという定理が公理から導ける。ただし計算順序はそのままにしなければならない。

例えば

$$a \cdot (b + c) + 0 = (a \cdot b) + (a \cdot c) + 0$$

が成り立っているとき

$$a + (b \cdot c) \cdot 1 = (a + b) \cdot (a + c) \cdot 1$$

も成り立つ。

例えば問題として

$$A + \bar{A} \cdot B = A + B \text{ を示せ。}$$

とあった場合、双対を取って

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

を示せば問題を解いたことになる。

その他のブール代数の性質

具体的な説明は省くが、等式を証明する場合、機械的な方法で証明が出来るシャノン展開という方法もある。興味があったら調べてみよう。

完全系という概念を導入すると(and-or-not)系は(nand)系と同様完全なことが分かったり(つまりすべての論理式はnandのみで表せる)とそーゆうのもある。

最小項、最大項、主加法標準形、主乗法標準、ドモルガンの法則など。。

論理演算とブール代数の関係

命題変数 p は真なら1、偽なら0を取るとする。例えば

$$p(x) = \begin{cases} 1 & (\text{電圧 } x \text{ がしきい値を超える。}) \\ 0 & (\text{電圧 } x \text{ がしきい値を超えない。}) \end{cases}$$

などとなる。ここで集合 B を

$$B = \{ p \mid p : \text{命題変数} \} \cup \{ 0, 1 \}$$

とし(つまり B は論理変数 p と論理定数 $0, 1$ の和集合)、演算 $+$ 、 \cdot と作用 \neg を

$$+ : \text{論理和}, \quad \cdot : \text{論理積}, \quad \neg : \text{否定}$$

と定義すると B はブール代数となる。よって論理演算がブール代数の構造を持つのでブール代数が適応できる。

デジタル回路におけるブール代数

さすがに記号だけじゃわかりにくいかなと思ったので具体例ものせた追加資料参照。

カルノー図の気持ち

例えば以下のような $f(x, y, z)$ を考えてみる。

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0

x	y	z	f(x,y,z)
1	1	0	1
1	1	1	1

これに対し f_1 、 f_2 、 f_3 、 f_4 を以下のように定義する。

x	y	z	f(x,y,z)	f_1	f_2	f_3	f_4
0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0
0	1	0	0	0	0	0	0
0	1	1	1	0	1	0	0
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	1	0	0	1	0
1	1	1	1	0	0	0	1

とすると

$$f(x, y, z) = f_1(x, y, z) + f_2(x, y, z) + f_3(x, y, z) + f_4(x, y, z)$$

となり、後は見れば簡単に

$$f(x, y, z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot y \cdot \bar{z} + x \cdot y \cdot z$$

となる。ちなみにこの表し方を主加法標準形と言ったりする。ここで式を簡略化することを考える。具体的には、吸収則

$$a \cdot b + a \cdot \bar{b} = a$$

の性質を用いる。すると

$$\bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z = \bar{x} \cdot z$$

$$x \cdot y \cdot \bar{z} + x \cdot y \cdot z = x \cdot y$$

より

$$f(x, y, z) = \bar{x} \cdot z + x \cdot y$$

となる。ここで先ほどの真理値表の順番を意図的に変えてもう一度見てみる。

x	y	z	f(x,y,z)
0	0	0	0
0	1	0	0
0	0	1	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

f(x,y,z)が1の部分に注目する。まず

x	y	z	f(x,y,z)
0	0	1	1
0	1	1	1

を見るとx、y、zは1bitのずれ(yだけ異なる。)でf(x,y,z)=1である。よって吸収則を適用し、f(x,y,z)は $\bar{x} \cdot z$ を含む事がすぐわかる。同様に

x	y	z	f(x,y,z)
1	1	0	1
1	1	1	1

もx、y、zは1bitのずれ(zだけ異なる。)でf(x,y,z)=1である。よって吸収則を適用し、f(x,y,z)は $x \cdot y$ を含む事がすぐわかる。

ここでポイントとなるのは **1bit分だけ入力異なる** という点にある。真理値表の問題点はf(x,y,z)の場合すべての(x,y,z)について1bit分だけ入力異なる入力が3つ存在するにもかかわらず、隣り合せに出来ない点にある。カルノー図はそこを克服している。

x/yz	00	01	11	10
0	0	1	0	0
1	0	1	1	1

カルノー図を見るとすべての入力に対し境界がつながっていると仮定すれば、1bit分だけ入力が異なる入力が3つ隣り合っている。本質は1bit差で隣り合うことだけなので表記は色々方法があり例えば

x/yz	11	01	00	10
1	1	1	0	1
0	0	1	0	0

でも

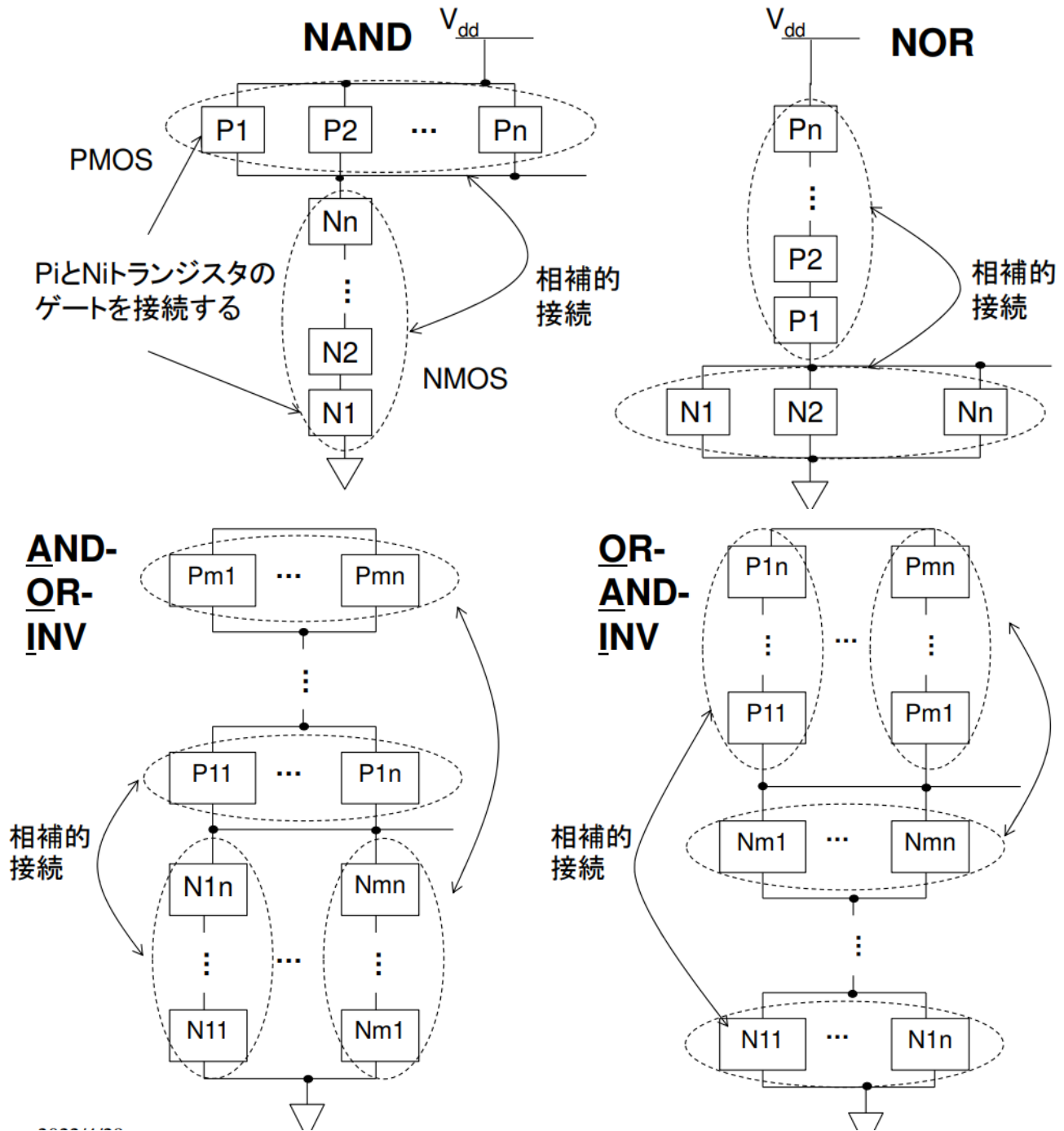
x/yz	00	10	11	01
0	0	0	0	1
1	0	1	1	1

でも構わないことが分かるだろう。

Static CMOS回路

cmos回路はnmos回路とpmos回路を用いて構成されるが、特にnmosとpmosを相補的に接続した場合、それをStatic CMOS回路という。ゲート(nand、nor、not、AOI、OAI、.....etc)はすべて相補的に接続されている。相補的に接続すると必ず電流が流れるパスがなくなり消費電力を抑えられる。

相補的に接続とはPMOSはプルアップ(電源側につける)し、NMOSはプルダウン(接地側につける)を施しさらにPMOSが直列ならNMOSは並列にするといったつなぎ方をする。

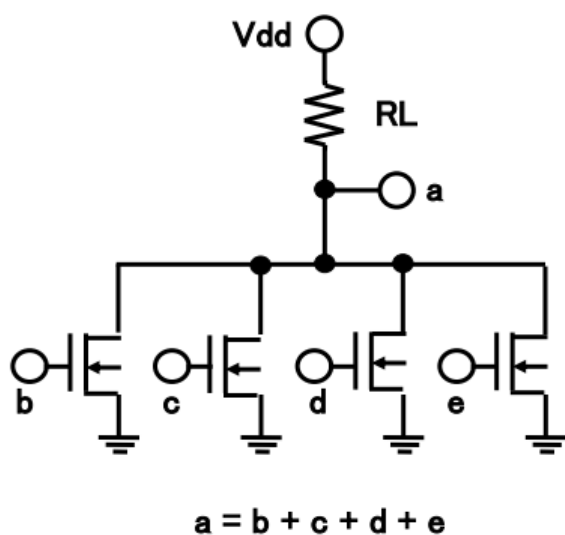


オープンドレイン回路

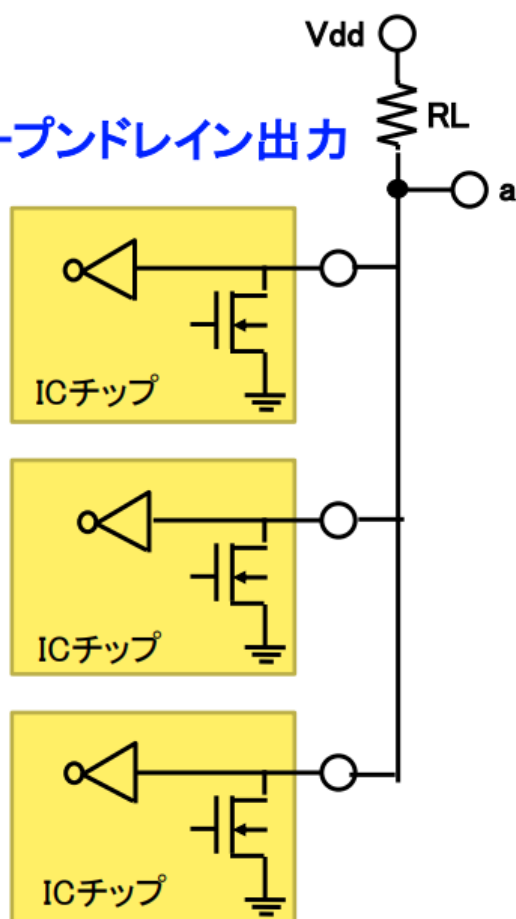
オープンドレイン回路とはそのままの意味でドレインに何もつながっておらず、プルアップ抵抗を用いて外部電源に接続し、high及びlowを制御する回路である。プルアップ抵抗を付けない場合、導通の時にショート

してしまう。

Trを用いた他入力nor回路 (ワイヤードnor回路)



オープンドレイン出力

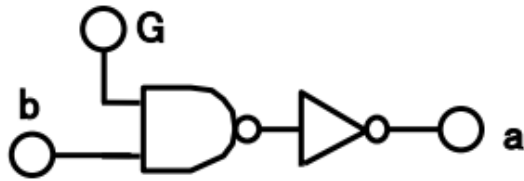


様々な回路

ゲート回路とセレクト回路。

そのままの意味。

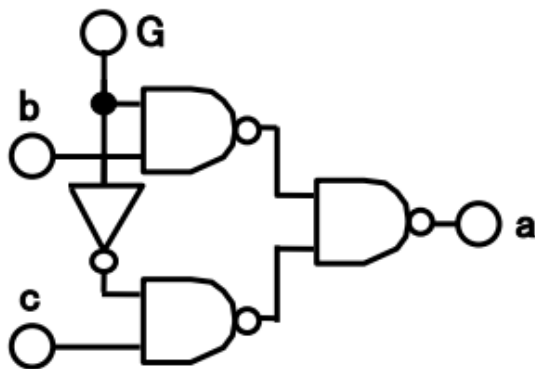
ゲート回路



$$a = \text{not}[(G)\text{nand}(b)] = (G) \cdot (b)$$

G	b	a
1	1	1
1	0	0
0	1	0
0	0	0

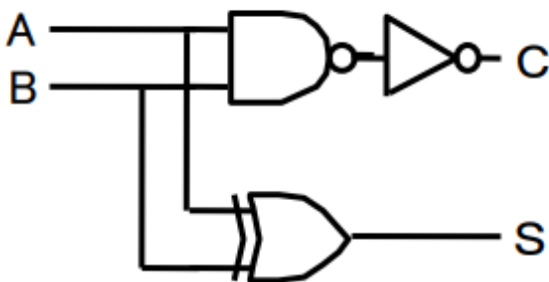
セレクト回路



G	b	c	a
1	1	x	1
1	0	x	0
0	x	1	1
0	x	0	0

半加算回路

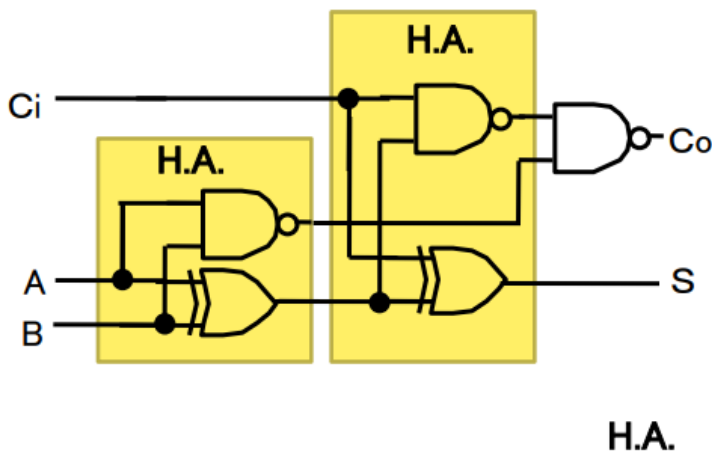
半加算回路は $A+B$ を計算して C は桁上り、 S はsumの略で計算結果となる。



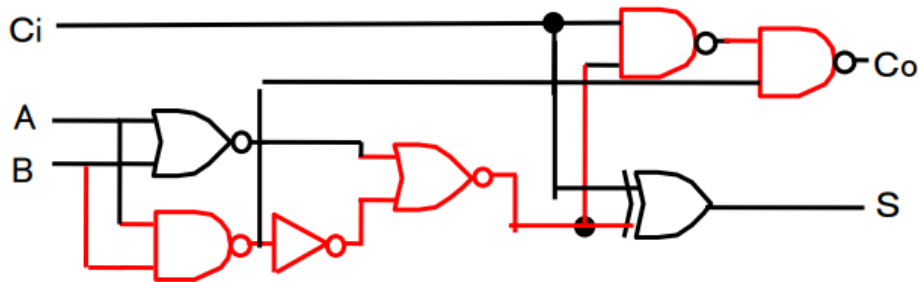
Input		Output	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1
		EOR	AND

全加算器

全加算回路は $A + B + C_{in}$ を計算して C_{out} は桁上り、 S はsumの略で計算結果となる。全加算回路を複数つなげると加算器が作れる。

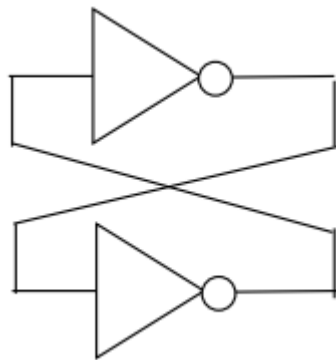


Input			Output	
A	B	Ci	S	Co
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



かぎ型構造

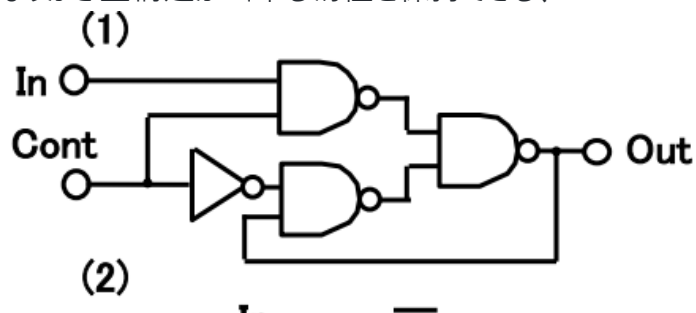
以下の回路は2つの安定状態を持つ。よって2値を保持できる回路となっている。記憶素子は値を保持中



の時はこの構造になっている。

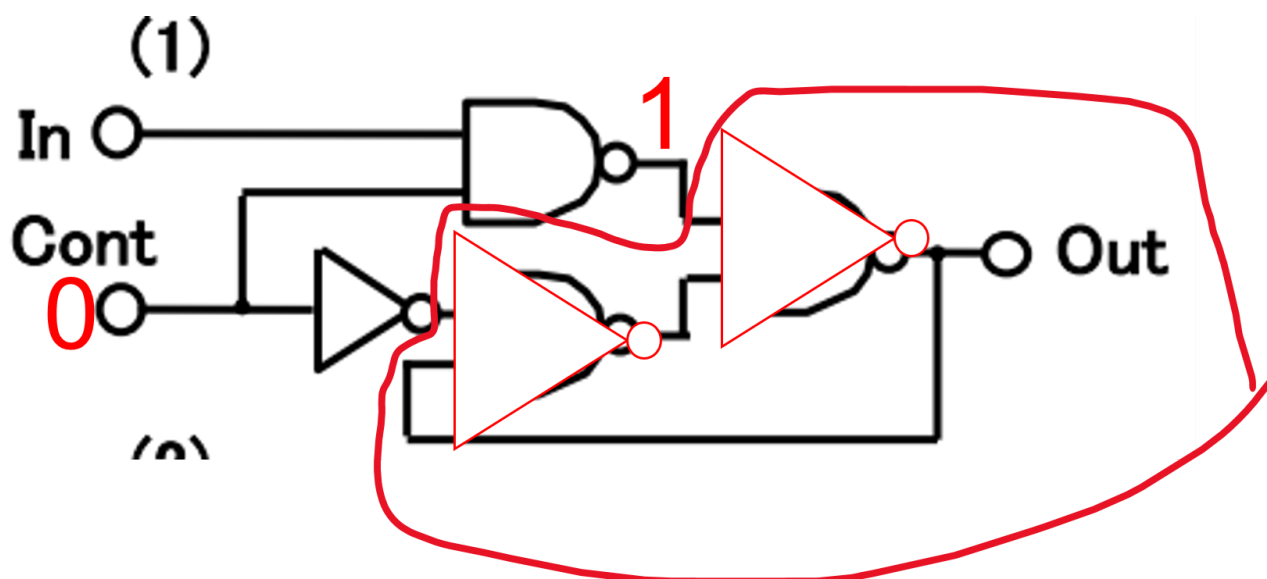
ラッチ回路

Contが1の時は信号Inをそのまま通し、Countが0の時は $\overline{A} \cdot 1 = \overline{A}$ の性質に着目すると下の図のようになりかぎ型構造が出来る為値を保持できる、



Input		Output
Cont	In	Out
1	1	1
1	0	0
0	1	$In \cdot z^{-1}$
0	0	$In \cdot z^{-1}$

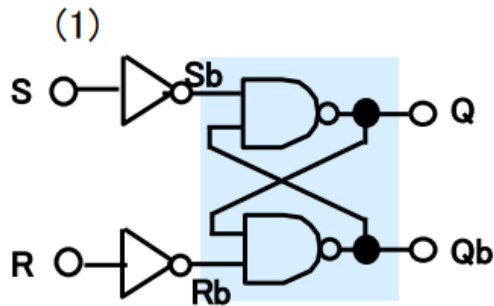
Countが0の時↓



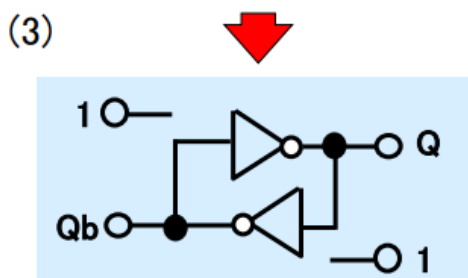
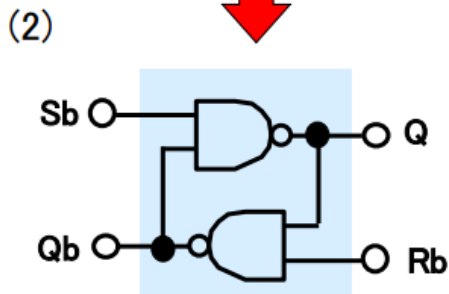
かぎ型構造

RS-FF回路

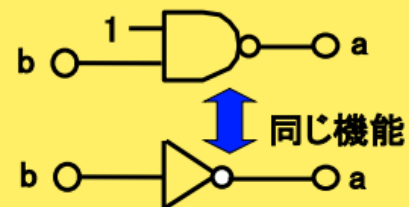
セットリセット付きフリップフロップ。セット(S)が1の時はQに1をセットし、リセット(R)が1の時はQを0にリセットする。 $S = R = 0$ の時はかぎ型構造となるため値を保持。 $S = R = 1$ の時は $Q = Q_b = 1$ となるが、 $S = R = 1 \rightarrow S = R = 0$ と遷移させるとかぎ型構造の構成上 $\bar{Q} = Q_b$ となるため $Q = Q_b = 1$ は保持させることが出来ずどちらかに遷移してしまう。具体的にどちらに遷移するかは $R \rightarrow 0$ が早ければ $Q=1$ 、 $S \rightarrow 0$ が早ければ $Q=0$ が保持されることとなりどちらになるかはわからないため禁止入力としている。



S	R	Q
0	0	$Q \cdot z^{-1}$
0	1	0
1	0	1
1	1	禁止



(4) 簡単化のための考え方



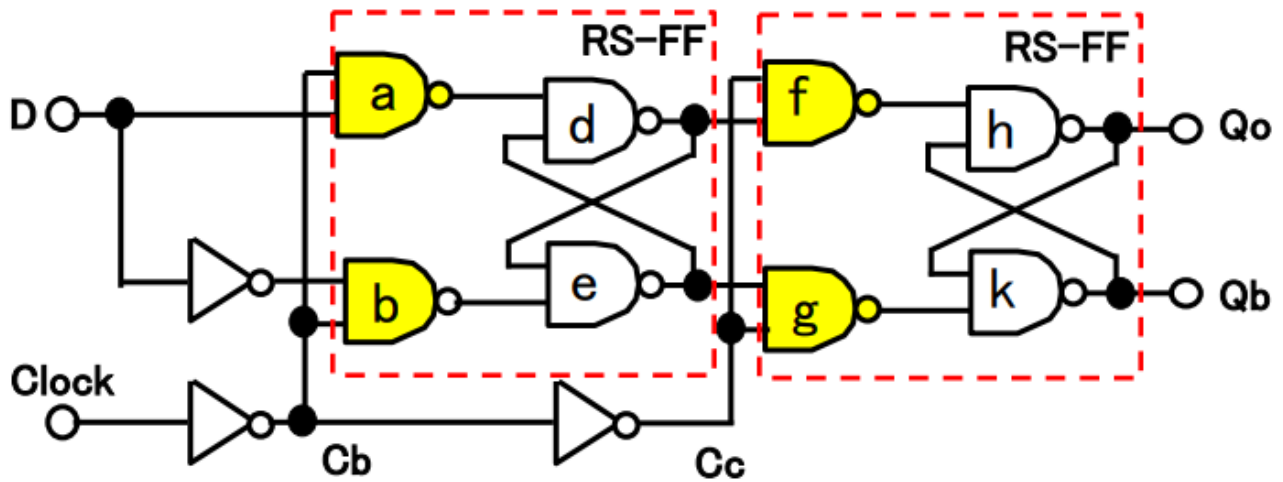
マスタースレーブフリップフロップ(DFF)

前段をマスター(主人)、後段をスレーブ(奴隷)と呼ぶ。

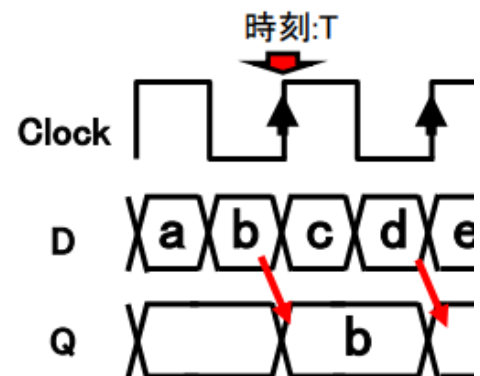
1. Clockが0の時マスター側のaとbはインバータになるため $d = D$ 、 $e = \bar{D}$ となる。それに対しスレーブ側は $f = g = 1$ となるためかぎ型構造が形成され保持状態。つまりDの値にかかわらず Q_o を保持。スレーブ側は保持状態よりDの入力を受け付けない。よってClockが0の時にDを変化させても Q_o は変化しない。
2. Clockが1に立ち上がった時、マスター側は $a = b = 1$ よりかぎ型構造が形成され保持状態。つまりClockが1に立ち上がる瞬間のDの値をdは保持する。それに対しスレーブ側はfとgはインバータになるため $Q_o = d =$ 立ち上がる瞬間のDの値となる。
3. Clockが1の時、マスター側は保持状態よりその後段のスレーブもDの入力を受け付けない。よってClockが1の時にDを変化させても Q_o は変化しない。

実際はフリップフロップには値が滲みだす時間(遅延時間のこと。造語だがイメージしやすいと思う)があるためセットアップ時間(Clockが1に立ち上がる少し前までにDが確定していないと思った通りの値にならない)等の制約があるのだが特に言及されていないため詳しくは話さない。

DFF回路



Clock	D	Qo
↑	1	1
↑	0	0
0	1	$D \cdot z^{-1}$
0	0	$D \cdot z^{-1}$

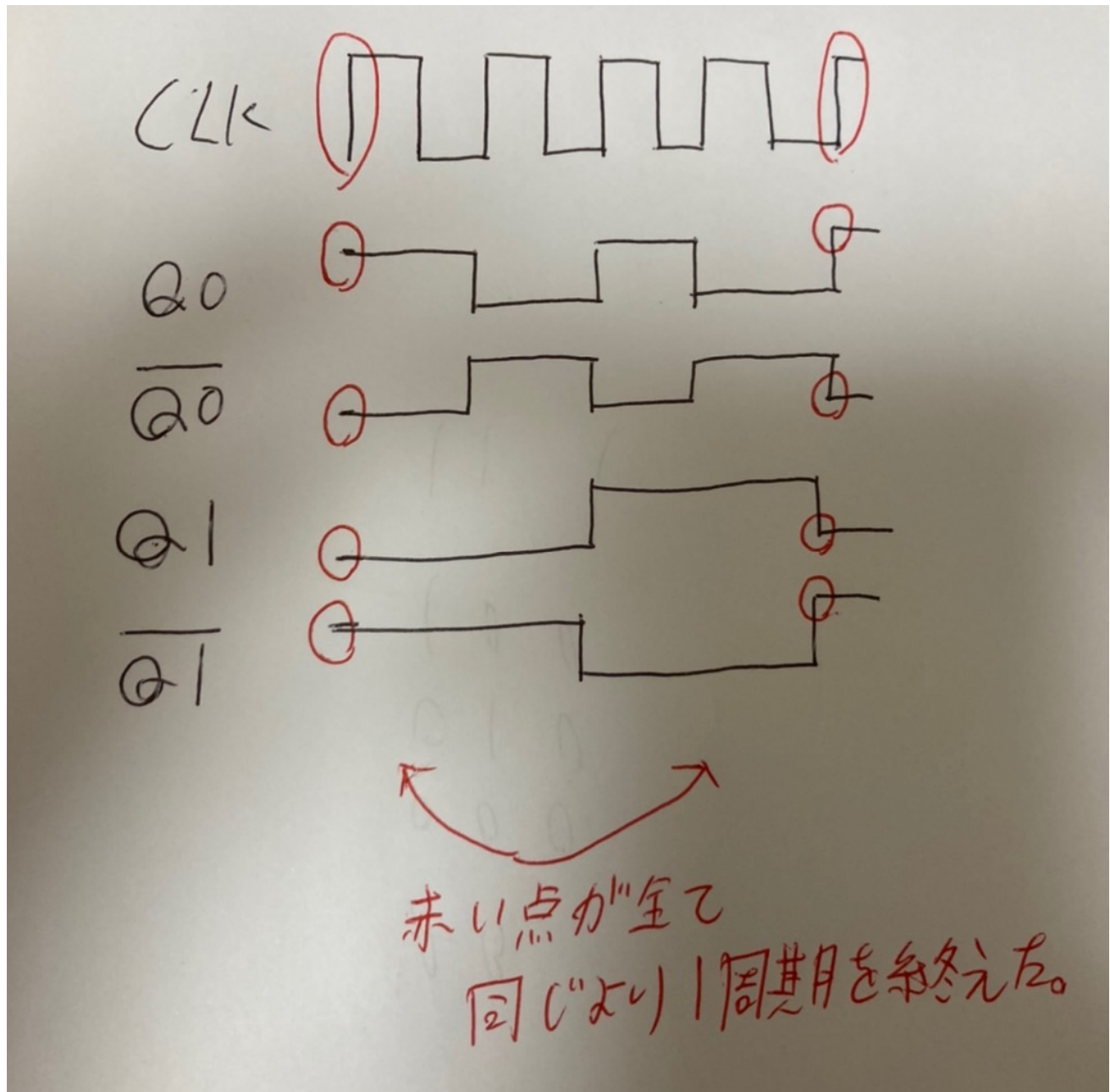
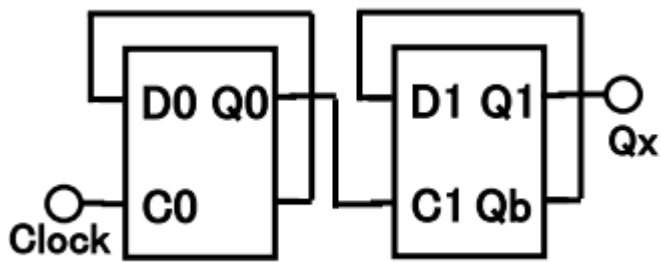


2^n 分周回路

これから様々な分周回路を説明していくが初期状態はデジタル回路の各点において矛盾が無ければ内部状態Qの値は任意であるので好きな値を設定する。

以下の図において初期状態(0回目の立ち上がり、スタート地点)を $Q_0 = 1$ 、 $Q_1 = 0$ とする。(言っている意味がわからない場合下にあるタイミングチャートのスタート地点を参照)

- 一回目の立ち上がりで $Q_0 = Q_{0b} = 0$ に変化する。一方 C_1 目線は立ち下りなので Q_1 に変化はない。
- 二回目の立ち上がりで $Q_0 = Q_{0b} = 1$ に変化する。一方 C_1 目線は立ち上がりなので $Q_1 = Q_b = 1$ に変化する。
- 三回目の立ち上がりで $Q_0 = Q_{0b} = 0$ に変化する。一方 C_1 目線は立ち下りなので Q_1 に変化はない。
- 四回目の立ち上がりで $Q_0 = Q_{0b} = 1$ に変化する。一方 C_1 目線は立ち上がりなので $Q_1 = Q_b = 0$ に変化する。よってこれを図にすると以下のタイミングチャートになる。今回は素子を2つ使ったため $2^2 = 4$ 回のCLK立ち上がりで1周期となる4分周回路となったがn個付けると 2^n 分周回路となる。



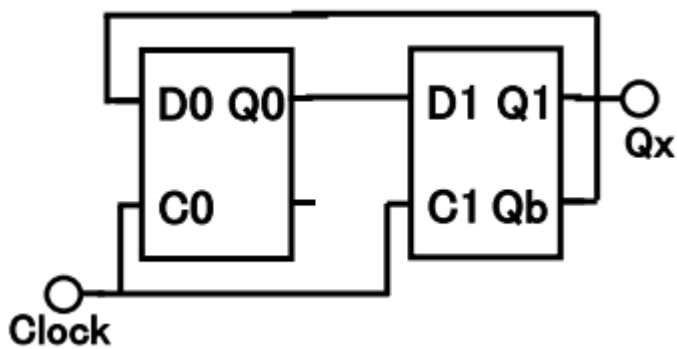
2n分周回路

初期状態(0回目の立ち上がり、スタート地点)を $Q_0 = 0$ 、 $Q_1 = 0$ とする。

1. 一回目の立ち上がりで $Q_0 = Q_b = 1$ 、 $Q_1 = Q_{0前} = 0$
2. 二回目の立ち上がりで $Q_0 = Q_b = 1$ 、 $Q_1 = Q_{0前} = 1$
3. 三回目の立ち上がりで $Q_0 = Q_b = 0$ 、 $Q_1 = Q_{0前} = 1$

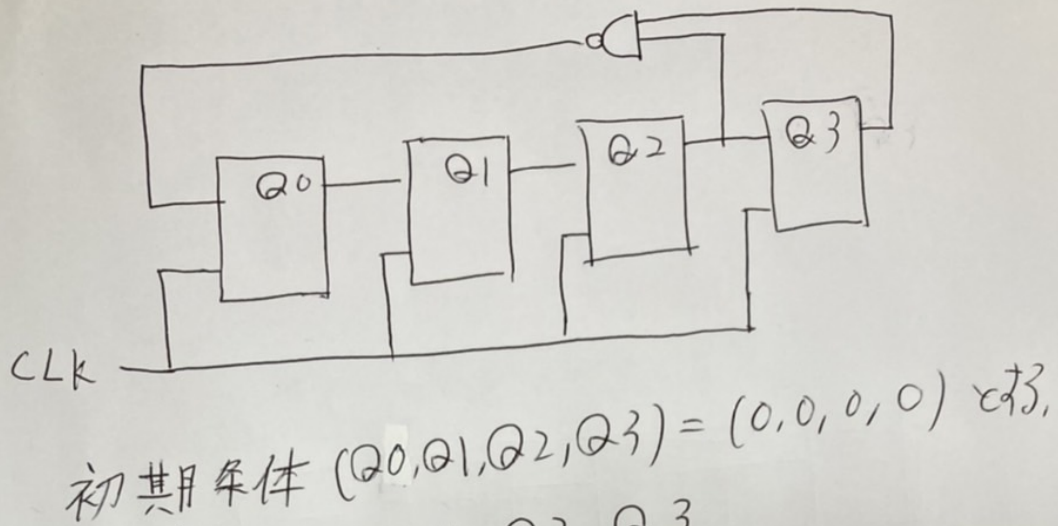
4. 四回目の立ち上がりで $Q_0 = Q_b = 0$ 、 $Q_1 = Q_{0前} = 0$

よって初期状態に戻った。 $2 \times 2 = 4$ 回のCLK立ち上がりで1周期となる4分周回路となったがn個付けると $2 \times n$ 分周回路となる。



2n-1分周回路

手書きの図で説明する。Dffをn個とすると2n-1分周回路となる。




	Q_0	Q_1	Q_2	Q_3
0回目	0	0	0	0
1回目	1	0	0	0
2回目	1	1	0	0
3回目	1	1	1	0
4回目	0	1	1	1
5回目	0	0	1	1
6回目	0	0	0	1
7回目	0	0	0	0

よって7分周

n-bitリプルキャリー回路

さすがに有名なので割愛。やってるのはそのままのひっ算である。ちなみにこの回路で $x-y$ を行いたい場合は最下段の c_i である $C_{i0} = 1$ とし、 y を"1"の補数に変換すれば出来る。また、最上段の C_o をフラグレジスタにつなげることもよくある。が特に説明がないのでここでも説明しない。

 画像がなかった時の代替テキスト

論理式の変形のコツ(完全にテスト対策専用知識)

2入力nand、2入力nor、invの3素子合計n個で表せとかいうくそ問題について。

- 否定(invのこと)が無くなるまでドモルガンを繰り返す
- ドモルガンをしている最中に論理変数が減らせる場合は必ず減らす。具体的には以下の公式は常に狙うこと。

$$A \cdot B + A \cdot B = A \cdot (B + C)$$

$$(A + B) \cdot (A + C) = A + B \cdot C \quad (\text{上式の双対である})$$

$$A + A \cdot B = A \quad (A = 1 \text{の時は} 1, A = 0 \text{の時は} 0)$$

$$A \cdot (A + B) = A \quad (\text{上式の双対である})$$

- 否定を消せない場合、アプローチ変える。
- 具体的には加法形(カルノー図で1に着目したパターン)から始めたなら、乗法形(カルノー図で0に着目したパターン)

これを意識して解いてみよう。特に4つ目は肝心である。加法形から行くと本当に地獄を見るパターンもたまにある。