

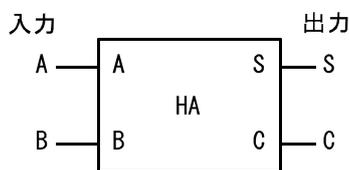
・半加算器

1桁の2進数2つを入力し、それらを算術加算した2桁の2進数を出力する回路を半加算器(HA: half adder)と呼ぶ。半加算器の真理値表を図1(a)に、回路記号を図1(b)に示す。

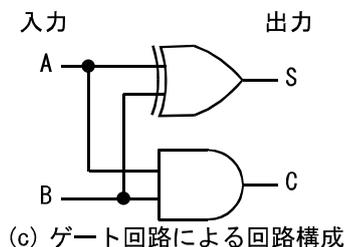
A	B	C	S	備考
0	0	0	0	0B+0B=00B に対応
0	1	0	1	0B+1B=01B に対応
1	0	0	1	1B+0B=01B に対応
1	1	1	0	1B+1B=10B に対応

(a) 真理値表

$$\begin{array}{r} A \\ + B \\ \hline C S \end{array}$$



(b) 回路記号



(c) ゲート回路による回路構成

図1、半加算器

真理値表において、AとBとCの関係を見ると、CはAとBの論理積(AND)になっている事が分かる。また、AとBとSの関係を見ると、SはAとBの排他的論理和(XOR)になっている事が分かる。よって、半加算器をゲート回路で構成すると、図1(c)のようになる。

半加算器を使えば1桁の2進数の足し算を行うことができるが、半加算器を2つ組み合わせて2桁の2進数の足し算をする回路を作ろうと思っても、下の桁から上の桁への繰り上がり信号を扱う事ができないので、無理である。2桁以上の足し算を行うには、足したい数AとB以外に、下の桁からの繰り上がり信号も足せる加算器が必要となる。

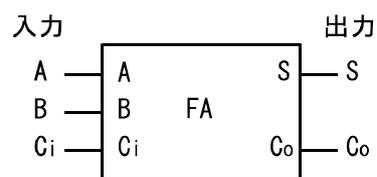
・全加算器

1桁の2進数3つを入力し、それらを算術加算した2桁の2進数を出力する回路を全加算器(FA: full adder)と呼ぶ。全加算器の真理値表を図2(a)に、回路記号を図2(b)に示す。

A	B	C _i	C _o	S	備考
0	0	0	0	0	0B+0B+0B=00B に対応
0	0	1	0	1	0B+0B+1B=01B に対応
0	1	0	0	1	0B+1B+0B=01B に対応
0	1	1	1	0	0B+1B+1B=10B に対応
1	0	0	0	1	1B+0B+0B=01B に対応
1	0	1	1	0	1B+0B+1B=10B に対応
1	1	0	1	0	1B+1B+0B=10B に対応
1	1	1	1	1	1B+1B+1B=11B に対応

(a) 真理値表

$$\begin{array}{r} A \\ B \\ + C_i \\ \hline C_o S \end{array}$$



(b) 回路記号

図2、全加算器

AとBは足したい数、C_iは下の桁からの繰り上りを意味する。またSはその桁の合計、C_oは上の桁への繰り上りを意味する。

AとBとC_iとSの関係に注目すると、これら4つの信号の中で1の数が必ず偶数になるようにSが決定されている事が分かる。このような回路を偶数パリティ回路と呼ぶ。

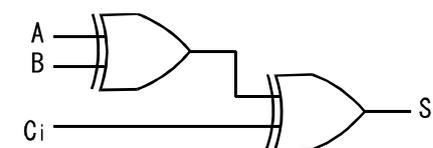


図3、3入力偶数パリティ回路

偶数パリティ回路を作るには、XOR回路を入力の数(この場合は3)より1少ない数だけ用意すればいいことが知られている。3入力偶数パリティ回路の回路図は、図3のようになる。この回路が図2(a)の真理値表を満たす事は、各自で確認せよ。

また、図2(a)の真理値表において、AとBとC_iとC_oの関係に注目すると、前回の講義で説明した3入力多数決回路になっている事が分かる。3入力多数決回路の回路図は、前回の講義で説明したとおり、図4の様になる。ただし、図4では次の項目で説明する3入力のOR回路を使っている事に注意せよ。

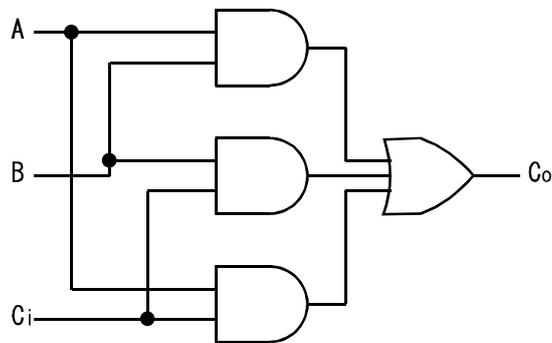


図4、3入力多数決回路

・多入力のAND回路とOR回路

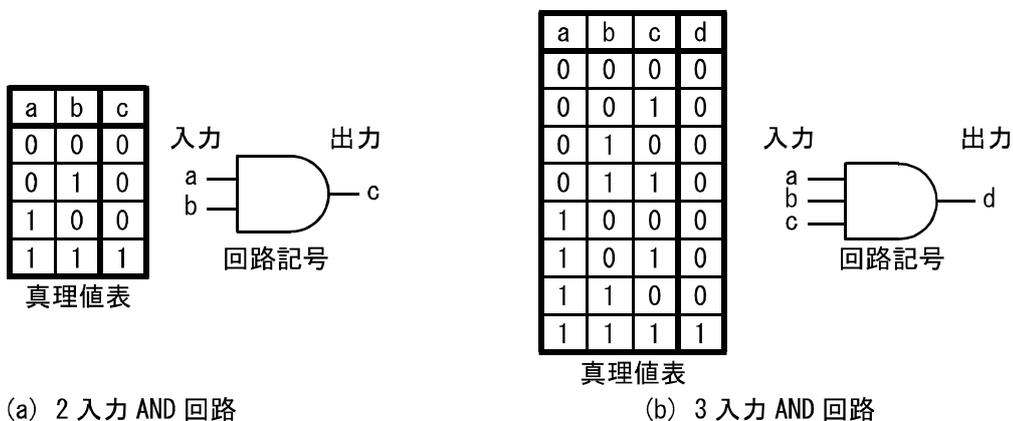


図5、2入力AND回路と多入力AND回路

図5(a)に示す2入力のAND回路の真理値表を見ると、出力が1になるのは、全ての入力(aとb)が1の場合のみである。3入力以上の回路の場合でも、全ての入力が1になる場合のみ出力が1になる回路の事をAND回路と呼ぶ事にすれば、多入力(3つ以上の入力)のAND回路を定義できる。図5(b)に3入力のAND回路の例を示す。

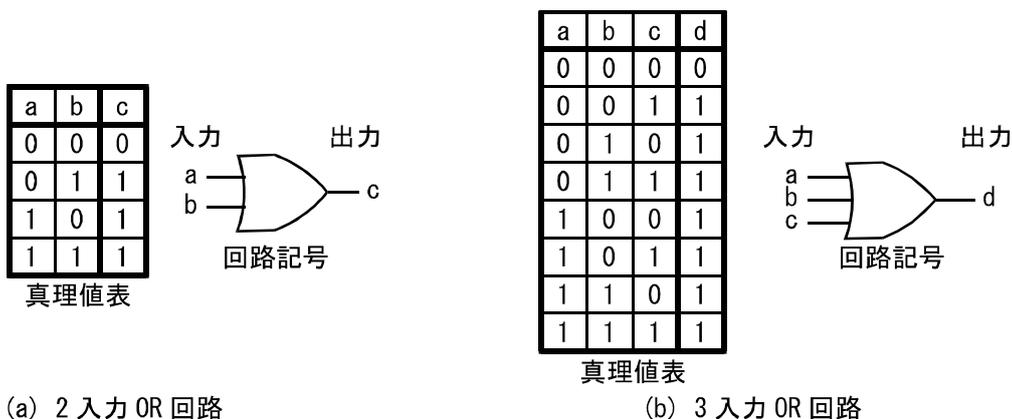


図6、2入力OR回路と多入力OR回路

図6(b)に示す2入力のOR回路の真理値表を見ると、出力が1になるのは、入力(aとb)の中で1つでも1が入力された場合である。3入力以上の回路の場合でも、入力の中で1つでも1が入力されたものがあれば出力が1となり、全ての入力が0の場合は0を出力する様な回路をOR回路と呼ぶ事にすれば、多入力

の OR 回路を定義できる。図 6(b)に 3 入力の AND 回路の例を示す。

・多入力の AND 回路や OR 回路の伝播遅延時間

組み合わせ回路(出力が入力の状態のみで決まるデジタル回路)では、理論的には出力の値が真理値表に従うとされているが、現実には真理値表に合わない出力状態の瞬間が存在する。

例えば図 7(a)の様な NOT 回路を考える。図 7(b)の様に入力信号が変化した場合、入力 a が変化しても、その変化が出力 b に反映されるには、わずかではあるが時間がかかる。この時間を伝播遅延時間と呼ぶ。

SN74HC04 という CMOS の NOT 回路の場合、電源電圧 4.5V、温度 25°Cにおいて、伝播遅延時間の典型値が 9ns、最大値 19ns(ns は 10⁻⁹s)と規定されており、伝播遅延時間はおおよそ 1 億分の 1 秒のオーダーとなっている事がわかる。

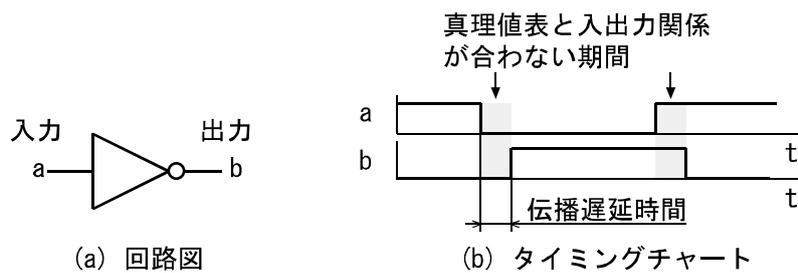


図 7、NOT 回路の伝播遅延時間

図 8(a)の様に、2 つの 2 入力 AND 回路を組み合わせると、3 入力 AND 回路ができる。この回路はトランジスタで直接 3 入力の AND 回路を組んだ場合と比較すると、真理値表は同じになるが、a や b に対する d の伝播遅延時間が、c に対する d の伝播遅延時間の 2 倍になるという特徴がある。

一方で、図 9 に示すように、トランジスタで直接 3 入力の AND 回路を構成すると、a、b、c に対する d の伝播遅延時間は、2 入力 AND 回路の伝播遅延時間と同じになる。すなわち、2 入力の AND 回路を 2 つ

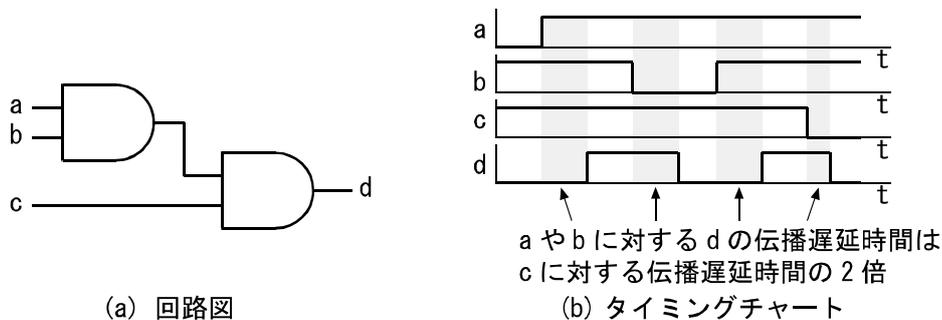


図 8、2 つの 2 入力 AND 回路で 3 入力 AND 回路を構成した場合の伝播遅延時間

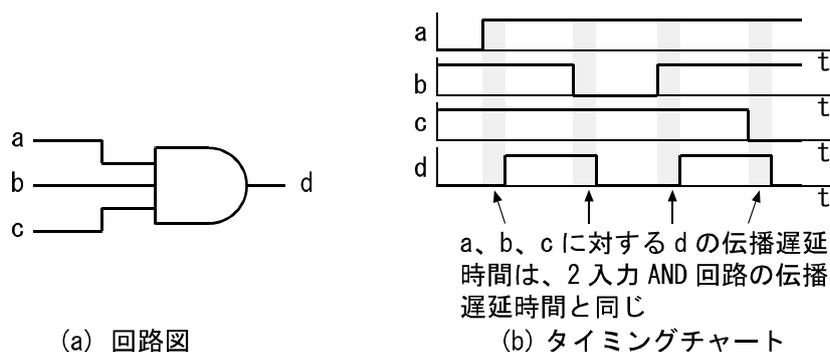


図 9、直接 3 入力 AND 回路を構成した場合の伝播遅延時間

組み合わせるよりも、3入力のAND回路を直接構成する方が、高速で対称性の良い回路が作れる。この話は、4つ以上の入力になっても、あるいはOR回路になっても同様である。

・多数桁の加算回路の構成(並列型加算回路)

図10に示すように全加算器3つと半加算器1つを組み合わせると、4桁の2進数2つを算術加算する加算回路が構成できる。この様に桁数と等しい数の全加算器および半加算器を並列に動作させるタイプの加算回路を並列加算回路と呼ぶ。

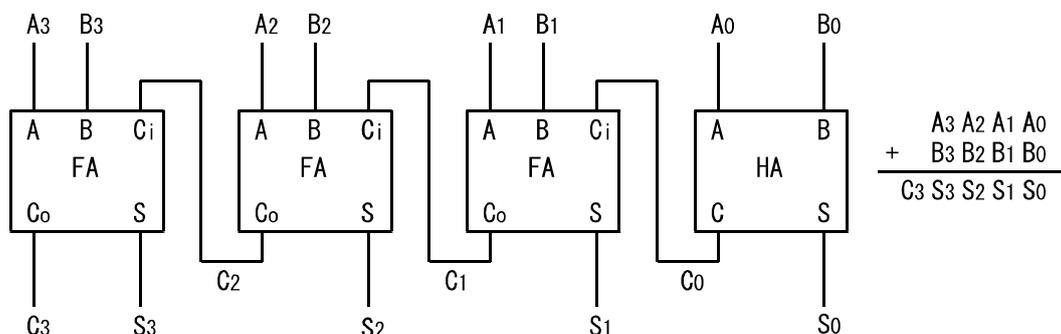


図10、4桁の並列加算回路(リップルキャリー型)

図10の加算回路で、例えば1111B+0001B=10000Bの計算を行う場合、A₀(1)とB₀(1)をHAで足してS₀=0とC₀=1が確定し、次に右側のFAでA₁(1)とB₁(0)とC₀(1)を足してS₁=0とC₁=1が確定し、さらに真ん中のFAでA₂(1)とB₂(0)とC₁(1)を足してS₂=0とC₂=1が確定し…という具合に、繰り上がり信号が下の桁から上の桁に順次伝播してゆき、計算結果は下の桁から確定していく。繰り上がり(carry)がさざ波(ripple)の様に伝わってゆく様から、このような繰り上がり信号の処理の仕組みをもつ加算回路を、リップルキャリー型加算回路と呼ぶ。

・キャリールックアヘッド型の加算回路

リップルキャリー型加算回路の場合、下位の桁の計算結果が確定するまで上位の桁の計算が行えないことから、桁数が多い場合に計算に時間がかかる。この問題を解決するために、繰り上がり信号を下位の桁の加算器からもらうのではなく、足したい数(A₀~A₃とB₀~B₃)から直接計算するタイプの加算回路が考案された。このタイプの加算回路をキャリールックアヘッド型の加算回路と呼ぶ。

キャリールックアヘッド型の加算回路は、桁数にかかわらず一定時間で計算が終わり、高速であるという特徴を持つが、桁数の多い加算回路を作ろうとすると、回路が複雑化する欠点がある。

ここで、実際に4桁のキャリールックアヘッド型の加算回路を設計してみる。

まずC₀について考える。C₀が1になる条件はA₀=1かつB₀=1なので、次の論理式が成立する。

$$C_0 = A_0 \cdot B_0 \cdots (1)$$

次にC₁について考える。C₁は、A₁、B₁、C₀を入力とする多数決回路の出力であるから、次の式が成立する。

$$C_1 = A_1 \cdot B_1 + A_1 \cdot C_0 + B_1 \cdot C_0 = A_1 \cdot B_1 + (A_1 + B_1) \cdot C_0 \cdots (2)$$

この式に式(1)を代入すると、次式を得る。

$$C_1 = A_1 \cdot B_1 + (A_1 + B_1) \cdot A_0 \cdot B_0 \cdots (3)$$

C_2 については式(2)と同様に、次の式が成立する。

$$C_2 = A_2 \cdot B_2 + A_2 \cdot C_1 + B_2 \cdot C_1 = A_2 \cdot B_2 + (A_2 + B_2) \cdot C_1 \cdots (4)$$

この式に式(3)を代入すると次の式を得る。

$$C_2 = A_2 \cdot B_2 + (A_2 + B_2) \cdot A_1 \cdot B_1 + (A_2 + B_2) \cdot (A_1 + B_1) \cdot A_0 \cdot B_0 \cdots (5)$$

C_3 については式(2)と同様に、次の式が成立する。

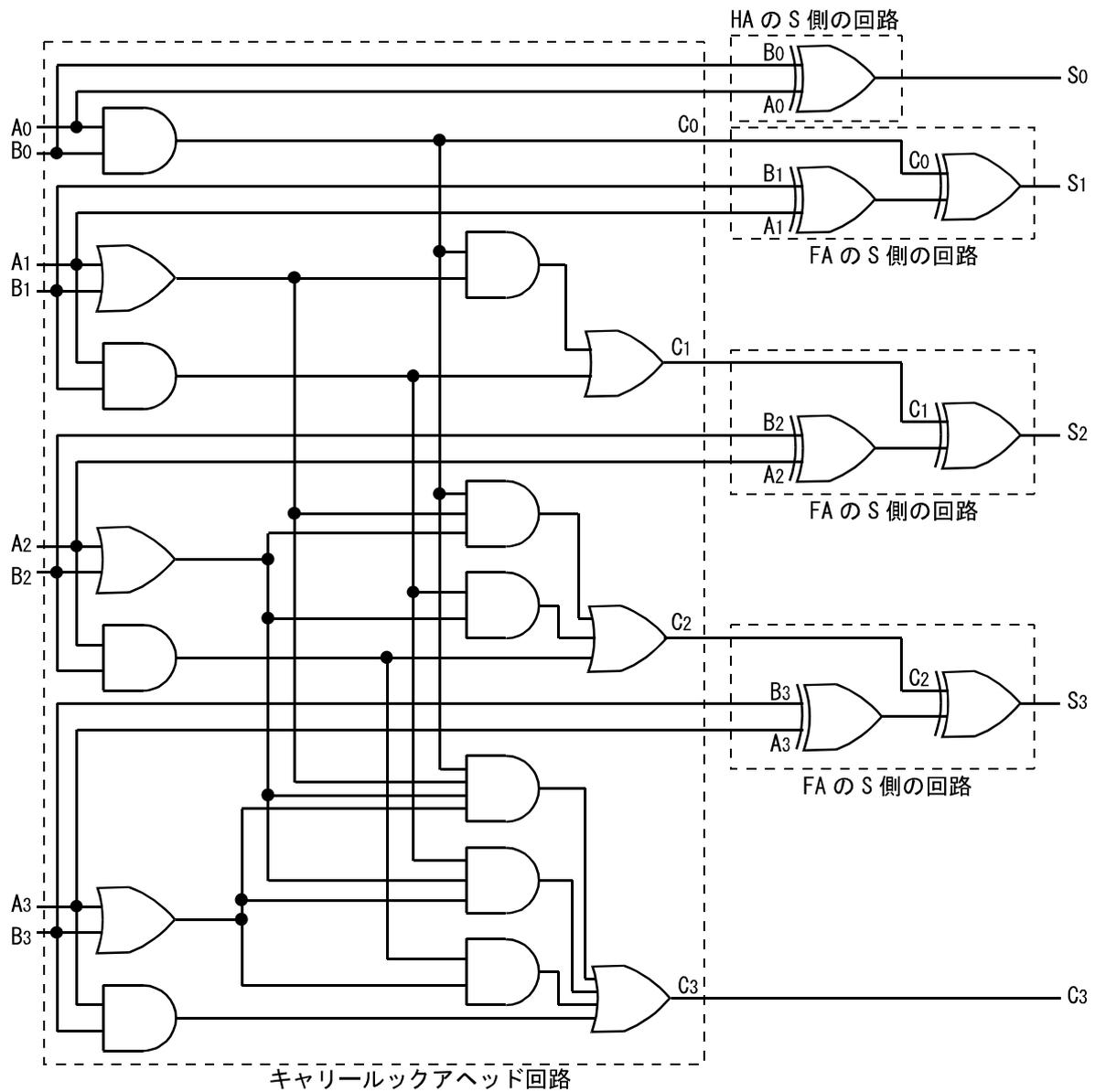


図 11、4桁の並列加算回路（キャリールックアヘッド型）

$$C_3 = A_3 \cdot B_3 + A_3 \cdot C_2 + B_3 \cdot C_2 = A_3 \cdot B_3 + (A_3 + B_3) \cdot C_2 \cdots (6)$$

この式に式(5)代入すると次の式を得る。

$$C_3 = A_3 \cdot B_3 + (A_3 + B_3) \cdot A_2 \cdot B_2 + (A_3 + B_3) \cdot (A_2 + B_2) \cdot A_1 \cdot B_1 + (A_3 + B_3) \cdot (A_2 + B_2) \cdot (A_1 + B_1) \cdot A_0 \cdot B_0 \cdots (7)$$

式(1)、(3)、(5)、(7)より、キャリールックアヘッド型の4桁並列加算器を設計すると、図11の様になる。

図11において、点線で囲まれ、キャリールックアヘッド回路と書かれた部分が、 $A_0 \sim A_3$ および $B_0 \sim B_3$ より繰り上がり信号 $C_0 \sim C_3$ を計算する回路である。最大でも3段のANDもしくはOR回路しか通っていないため、高速に繰り上がり信号が計算できる。上位の桁の回路ほど、多入力のAND回路やOR回路を使っており、またゲート数も多い事に注目すると、キャリールックアヘッド型の加算回路は、桁数が多くなると、回路がどんどん複雑化していくことが分かる。

$A_0 \sim A_3$ 、 $B_0 \sim B_3$ 、 $C_0 \sim C_2$ の各信号は、それぞれの桁に応じた全加算器もしくは半加算器に入力されるが、これらの加算器は、繰り上がり信号を計算する必要がないため、簡略化されている。