計算機システムⅠ・第8回

2022年6月8日

今回の内容

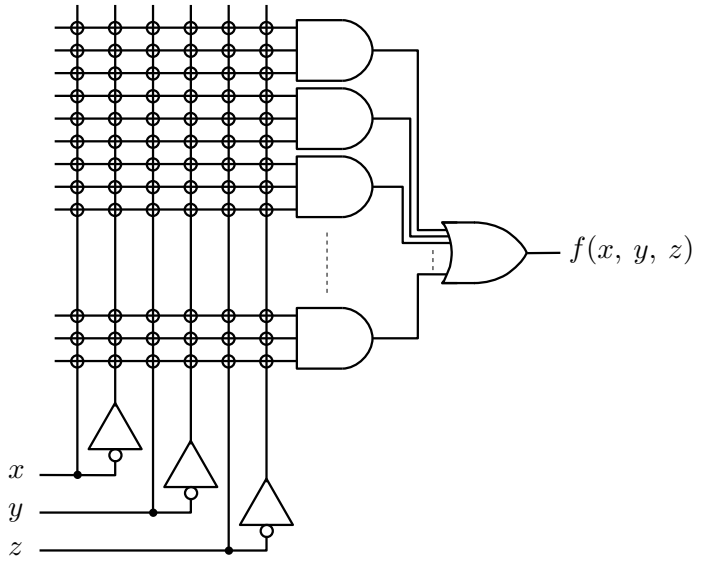
8.1 組み合わせ論理回路の一般的な構成方法

入力となるいくつかの真理値で決まる真理値を常に計算して出力し続ける論理回路を**組み合わせ論理回路**と呼びます。組み合わせ論理回路によって、任意の論理関数の値を計算させることができます。前回までに、さまざまな論理関数を論理回路として実現してきましたが、論理関数を論理回路化する一般的な方法を考えることができます。

メモ

AND-OR 構成による組み合わせ回路の実現　一般に、真理値表の形で論理関数の定義が与えられた場合、次のような手順に従えば必ず論理回路として実現することができます。

1. 真理値表から (主) 論理和標準形の論理式を作る。このとき、カルノー図などを利用して、より簡単な論理和標準形の論理式にしても構いません。
2. 論理和標準形の論理式の各積項を構成しているリテラルに対応するように、次のような論理回路 (3変数の場合の例) の ○ の部分を接続する (もちろん、各 AND ゲートの各入力に対して接続するのは1箇所のみです)。



AND ゲートは、論理和標準形を構成している積項の数だけ必要となりますが、積項の1つが論理定数1である場合は、そもそも $f(x, y, z) = 1$ となりますので、論理回路の出力としては (入力の

値にかかわらず) 1 とすればよいことになります。また、積項がない場合、つまり論理和標準形の論理式全体が論理定数 0 である場合は、そもそも $f(x,\, y,\, z) = 0$ ですので、論理回路の出力は (入力の値にかかわらず) 0 とすれば十分です。

論理関数の定義が論理式で与えられた場合も、第 5 回で解説したように、その論理式を (主) 論理和標準形に変形することができますので、同様に AND-OR 構成で論理回路を作ることができます。

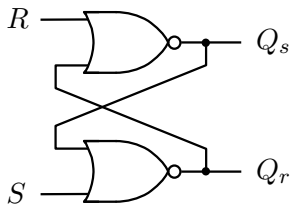
メモ

8.2 順序論理回路

組み合わせ論理回路の出力の値は、その論理回路の入力だけで決まりますが、論理回路自身が過去の入力に依存した内部状態 (何らかの記憶) を持っていて、現在の入力値とその内部状態に依存して出力値が決まるものもあります。このような論理回路は**順序論理回路**と呼ばれます。

メモ

SR-ラッチ　内部状態 (記憶) を持つ論理回路の基礎となるのは右のような 2 入力 2 出力の論理回路です。この論理回路は **SR-ラッチ (Set Reset Latch)** と呼ばれます[1]。

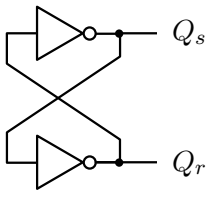


SR-ラッチ

通常、2 つの入力 S と R はともに 0 としておきます。このときの SR-ラッチは右のような論理回路と等価になります。出力 Q_s と Q_r は、お互いがお互いを否定したものとなっていますので、これらの値は

$$(Q_s,\, Q_r) = (1,\ 0) \quad \text{あるいは} \quad (Q_s,\, Q_r) = (0,\ 1)$$

のいずれかとなります。ただし、このどちらになるかは全く偶然によって決まります。



$S = R = 0$ のときの等価回路

この状態から、例えば、S を 1 とすると、NOR ゲートによって Q_r の値が必ず 0 になります。また、これに伴って Q_s の値は 1 になります。一旦、この $(Q_s,\, Q_r) = (1,\ 0)$ の状態になると、S を 0 に戻しても、同じ出力値が維持されます。逆に、$S = R = 0$ の状態から、R の方を 1 に変更すると、$(Q_s,\, Q_r) = (0,\ 1)$ となり、R を 0 に戻しても、この状態が維持されます。これが、この論理回路が記憶を持つという意味です。

[1]「RS-ラッチ」あるいは「(非同期) SR-フリップフロップ」、「(非同期) RS-フリップフロップ」と呼ばれることもあります。

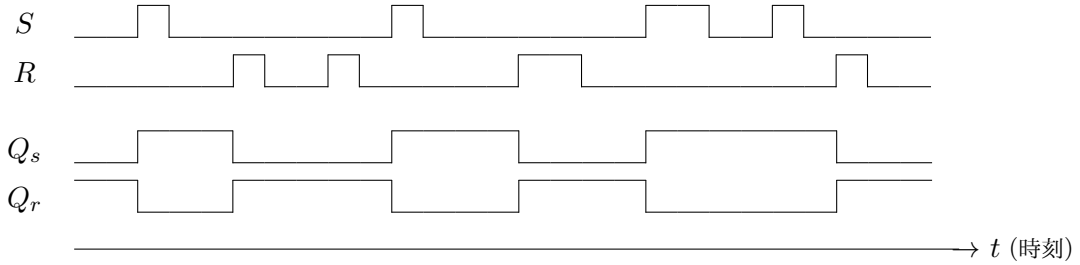
SR-ラッチの真理値表は次のようなものとなります。

S	R	Q_s	Q_r
0	0	変化せず	変化せず
0	1	0	1
1	0	1	0
1	1	0	0

$S = R = 1$ のときを除けば、2 つの出力 Q_s と Q_r は、一方の値がもう一方の値の否定をとったものとなります。SR-ラッチでは、入力 S と R を同時に 1 とするような使い方は想定されていません。このため、Q_s を単に Q と呼び、Q_r を $\overline{Q}$ を呼ぶこともあります。$S = R = 0$ のときに、出力が「変化せず」となっているのは、それまでの値を保ったままになるという意味ですが、$S = R = 1$ の状態から、$S = 1,\ R = 0$ や $S = 0,\ R = 1$ を経ずに、$S = R = 0$ となった場合の出力は $(Q_s,\ Q_r) = (1,\ 0)$ あるいは、$(Q_s,\ Q_r) = (0,\ 1)$ のどちらかになります。どちらになるかは予想できません。

通常は、$S = R = 0$ を入力した状態にしておき、Q_s および Q_r の値を保持させます。Q_s や Q_r の値を変えたい場合は、S または R のどちらか一方を一瞬だけ 1 にして、すぐに $S = R = 0$ に戻します。S を 1 にしたのであれば $(Q_s,\ Q_r) = (1,\ 0)$、R を 1 にしたのであれば $(Q_s,\ Q_r) = (0,\ 1)$ となり、その後、$S = R = 0$ である限りはその状態が維持されます。

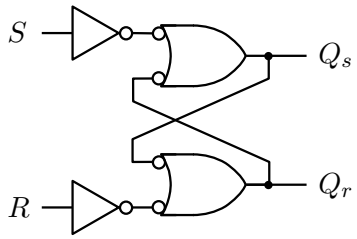
次は、SR-ラッチの入力 S と R の時間的変化に対する出力 Q_s、Q_r の時間的変化の例です。



順序論理回路の入力と出力の時間的変化の関係をこのようなグラフで表したものを、一般に、**タイミングチャート**と呼びます。

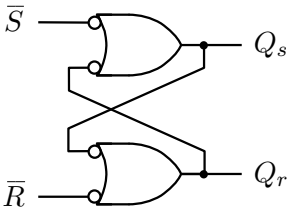
メモ

SR-ラッチの他の実現方法　順序回路の基礎となる SR-ラッチは、同様の機能を持つものを次のような論理回路として実現することもあります。



S	R	Q_s	Q_r
0	0	変化せず	変化せず
0	1	0	1
1	0	1	0
1	1	1	1

NAND ゲートによる SR-ラッチとその真理値表



$\overline{S}$	$\overline{R}$	Q_s	Q_r
0	0	1	1
0	1	1	0
1	0	0	1
1	1	変化せず	変化せず

$\overline{\mathrm{S}}\overline{\mathrm{R}}$-ラッチとその真理値表

2 ページで紹介した (NOR ゲートを使った) SR-ラッチと比べると、入力 S と R の位置関係が変わっていることに注意してください。NOR ゲートを使った SR-ラッチでは、$S = R = 1$ のときは $(Q_s,\, Q_r) = (0,\, 0)$ となりましたが、上左図の NAND ゲートによる SR-ラッチでは、$S = R = 1$ のとき、$(Q_s,\, Q_r) = (1,\, 1)$ になるという違いがあります。また、上右図の $\overline{\mathrm{S}}\overline{\mathrm{R}}$-ラッチでは、入力 $\overline{S}$ と $\overline{R}$ をともに 1 にしておくことで出力の状態を保持し、どちらかを 0 にすることで出力を変化させるという使い方をします。

計算機システム I ・第 8 回・終わり