情報科学実験 (2022後期)

ーデジタル回路Ⅱー

佐久間　史典
(理化学研究所)

Ver. 20220915

課題

- デジタル回路を実験で確かめながら理解する
  - 組み合わせ回路のおさらい
  - 続･組み合わせ回路
  - 順序回路 = フリップ･フロップ回路

- 論理回路実習トレーナー → **第1週目実験**

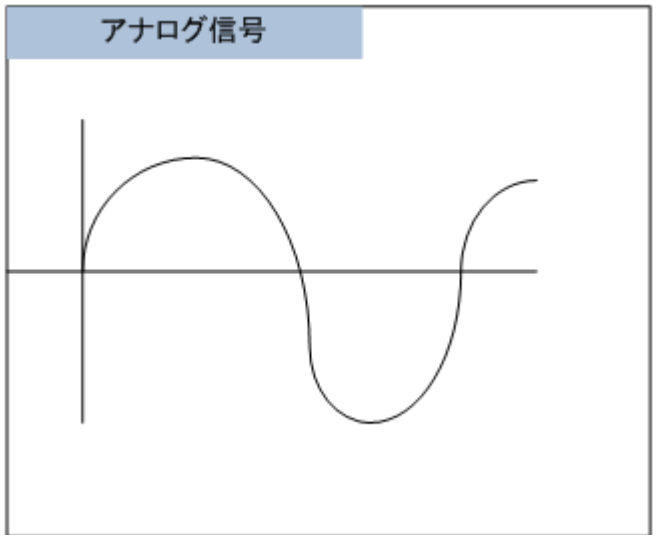
- ブレッドボード → **第2週目実験**

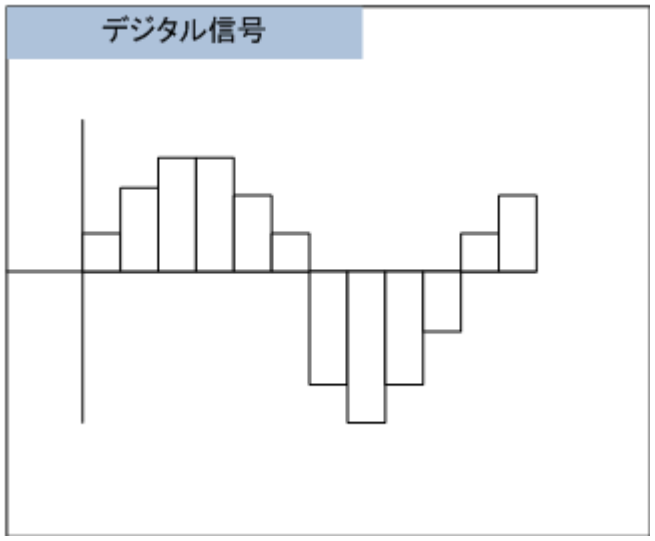
レポート・注意事項

- レポート
  - テキスト最後の[課題]をまとめて、レポートを提出してください
    - **1週目は2週目の予習 (レポートに入れる必要なし)**
      - 実験前の予想真理値表を書き込んでから実験をすること
      - 加えて、問12,13は論理式を出してから実験をすること
      - テキストに書き込んで良い
    - **2週目の実習が主なレポートの内容になる**
      - あらかじめ回路図を書き込んでくること

- 注意事項
  - 乱雑に回路を扱うと壊れますので、注意!

アナログ vs. デジタル





最小単位が無い

0と1で量子化 = 最小単位がある

アナログ vs. デジタル

	○	×
アナログ	• 本来は情報量が多い • 少ない情報で直感的(=早い)	• 劣化しやすい • 再現性が難 • ノイズが入りやすく、長距離伝送が苦手
デジタル	• **劣化しにくい** • **再現性に優れる** • **ノイズが入りにくく、長距離伝送が可能**	• **最小単位があり、決まった情報量のみ** • **情報量を増やすとデータ量が膨大になる(=遅い)**

科学技術の進歩で日々改善

昔のデジタルの例

モールス信号

0と1の組み合わせで通信

のろし

上がると1

上がらないと0

オルゴール

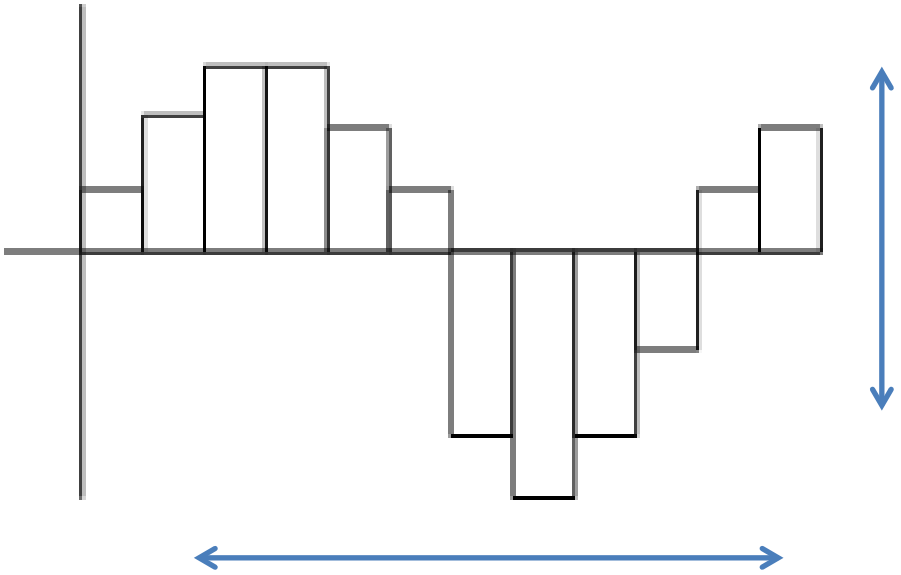
平らな部分=0

出っ張り=1

いつでも復元可能

＝デジタルデータ

今のデジタルの例:
CD [パルス符号変調(PCM)方式]



量子化bit数 16bit

2^{16} = 65,536段階の音圧

0/1が2^{16}個の1列 = 1点のデータ

サンプリング周波数 44.1kHz

= 1秒を44,100で分割

22kHzまで正確に復元(標本化定理)
~人間の可聴域

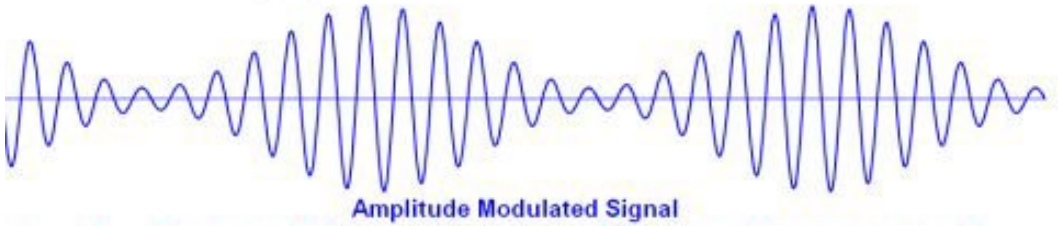
基本的にはパソコンも
これと同じ
(3GHz/64bit等のCPU)

豆知識：アナログ/デジタル変調

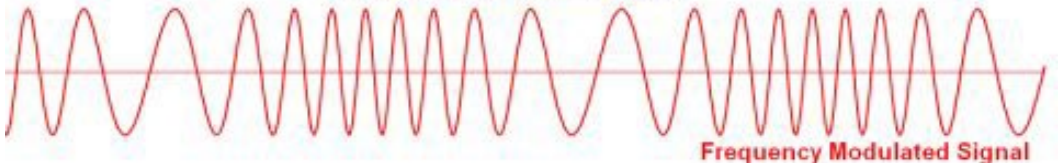
アナログ変調

元のシグナル

AM(振幅変調)

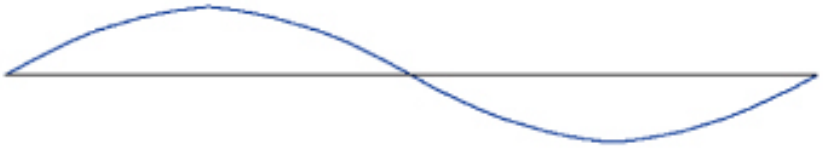


FM(周波数変調)



デジタル変調

元のシグナル



PCM(パルス符号変調)

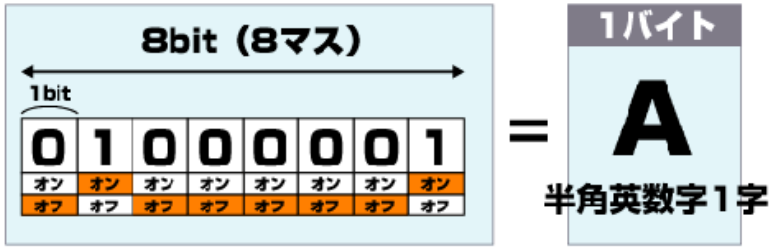
44.1kHz
/16bit

DSD(ダイレクトストリームデジタル)

2.8224MHz
/1bit

bitとByte

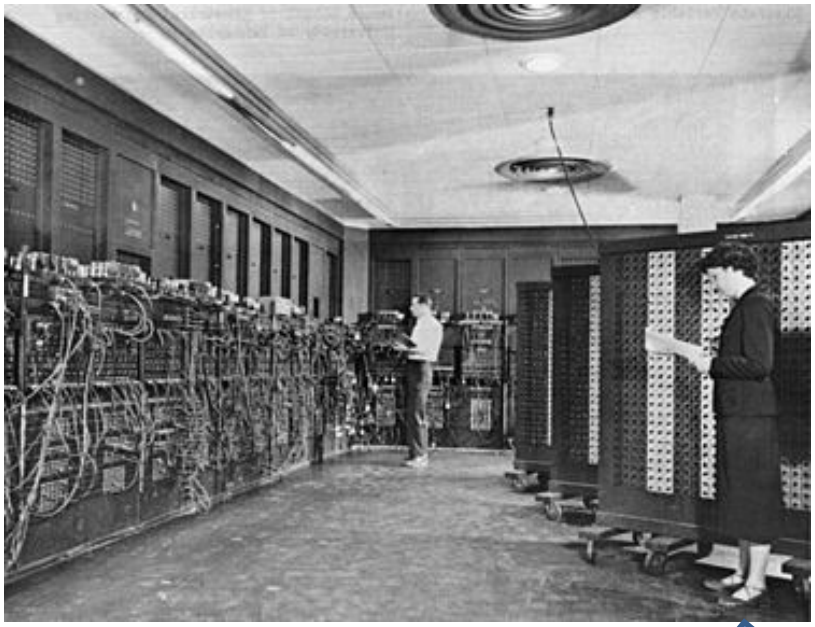
- **bit**：データの最小単位
  - 0 or 1
    - 64bit PC = 64個の0/1が処理系のデータ単位 (1 word)



- **Byte**：情報量の単位
  - 1Byte = 8 bit
    - 1Byte文字 = 半角英数字、半角カタカナ等
    - 2Byte文字 =ひらがな、漢字等

コンピュータの世界では
10^3=1,000ではなく2^{10}=1,024!

コンピューター今昔



ENIAC

1946年アメリカ
床面積：60 畳
重さ：車約 20 台分
1kW のストーブ 150 台相当

70 年後

CPUはトランジスタ
150億個の塊

スマートフォン

手のひらサイズ
数年前のPCより高性能

2つのコンピューター

- **普通のコンピューター = 古典コンピューター**
  - アラン・チューリング(英)によってその動作原理が考案(1936)
    - 仮想のマシン＝チューリングマシン
  - ジョン・フォン・ノイマン(米)によってその基礎が構築
    - 現実のマシン＝フォン・ノイマン型コンピューター
  - **「古典ビット」**で計算・プログラムとデータを同列で扱う

- **量子コンピューター**
  - リチャード・ファインマン(米)による概念がはじまり(1982)
  - 0と1の両方の状態を**同時に**取ることのできる**「量子ビット」**で計算
    - 重ね合わせ状態で表現 = すべては確率 (量子力学)
    - 複数の状態を同時に保持でき、かつそれぞれにつき計算可能
    - 2^{量子ビットの数}の状態が同時に計算可能 = 組み合わせ(暗号)問題等で威力を発揮
      - 5000 量子ビット @ D-Wave Advantage 量子コンピューター

コンピューターの数 = 2進数

- コンピュータは0と1しか原則扱えないため、2進数での考えが基本

10進数	2進数
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

0と1だけなら
“論理的”に扱える
0+0=0
0+1=1
1+1=1

論理演算

- 0(L)か1(H)のみを扱う2値論理 = ブール代数
  - 以下、A等の記号は0か1を表す

否定(NOT)

$$\overline{A} \qquad \overline{1}=0 \quad \overline{0}=1$$

論理和(OR)

$$A+B$$

論理積(AND)

$$A \cdot B$$

注) "・"は省略可能

基本法則

$$A+A=A \qquad 0+0=0,\ 1+1=1$$

$$A \cdot A=A \qquad 0 \cdot 0=0,\ 1 \cdot 1=1$$

$$A+1=1 \qquad 0+1=1,\ 1+1=1$$

$$A+\overline{A}=1 \qquad 0+1=1,\ 1+0=1$$

$$A \cdot \overline{A}=0 \qquad 0 \cdot 1=0,\ 1 \cdot 0=0$$

$$\overline{\overline{A}}=A \qquad \overline{\overline{0}}=\overline{1}=0,\ \overline{\overline{1}}=\overline{0}=1$$

ド・モルガンの定理

$$\overline{A+B}=\overline{A} \cdot \overline{B}$$

$$\overline{A \cdot B}=\overline{A}+\overline{B}$$

注) ABCDと増えていっても同様

通常の四則演算と同様、乗算(・)が加算(+)より優先
交換・結合・分配則も成立

論理式の簡略化

- 論理式は基本法則を用いて簡略化できる

例1 $A(A+B) = \underset{A}{AA} + AB = A + AB = A\underset{1}{(1+B)} = \underset{A}{A \cdot 1} = A$

例2 $A + \overline{A} + B = \underset{1}{(A + \overline{A})} + B = \underset{1}{1 + B} = 1$

例3 $(A+B)(\overline{A}+\overline{B}) = \underset{0}{A\overline{A}} + A\overline{B} + B\overline{A} + \underset{0}{B\overline{B}} = A\overline{B} + \overline{A}B$

例4 $A + \overline{A}B = A \cdot 1 + \overline{A}B = A\underset{1}{(B + \overline{B})} + \overline{A}B = AB + A\overline{B} + \overline{A}B$

$= \underset{AB}{AB + AB} + A\overline{B} + \overline{A}B = (AB + A\overline{B}) + (AB + \overline{A}B)$

$= A\underset{1}{(B + \overline{B})} + B\underset{1}{(A + \overline{A})} = A + B$

カルノー図を用いた方法も重要

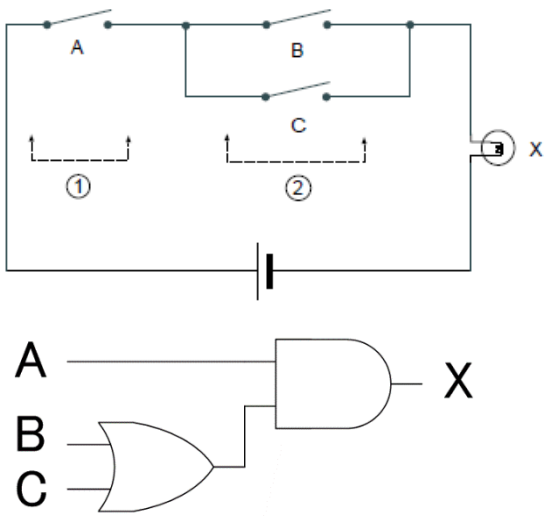
論理回路、論理記号、真理値表

- 論理回路：論理演算に対応した回路
- 論理記号：MIL規格(military standard)が一般
  - アメリカの軍隊で使用
- 真理値表：入力-出力の対応表

論理式

$$X = A(B + C)$$

論理回路



真理値表

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

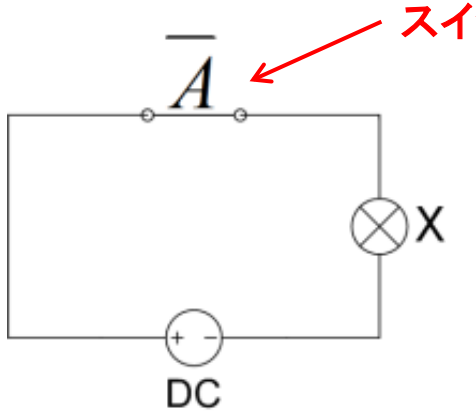
NOT

- 否定 (インバーターとも言う)

論理式

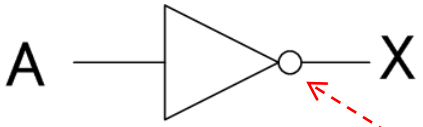
$$X = \overline{A}$$

論理回路



真理値表

A	X
0	1
1	0



何かに○を付けると否定

入力・出力、どちらにも付く

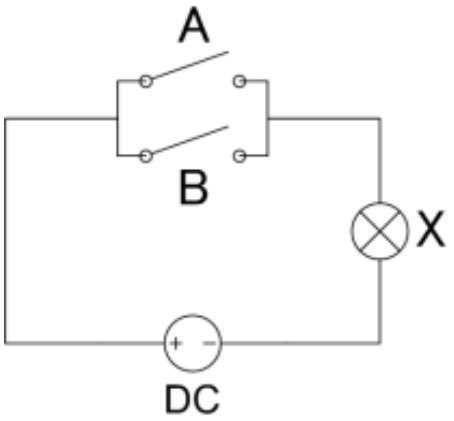
OR

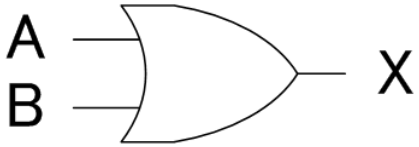
- 論理和

論理式

$$X = A + B$$

論理回路





真理値表

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

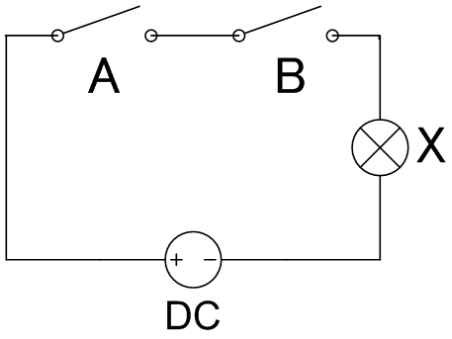
AND

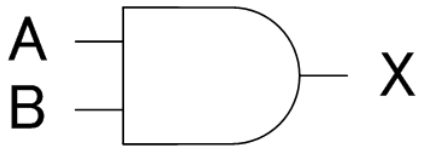
- 論理積

論理式

$$X = A \cdot B = AB$$

論理回路





真理値表

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

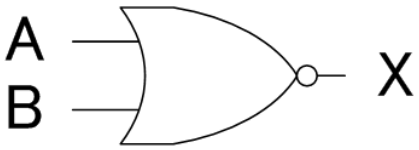
NOR

- 否定論理和

論理式

$$X = \overline{A + B}$$

論理回路



真理値表

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

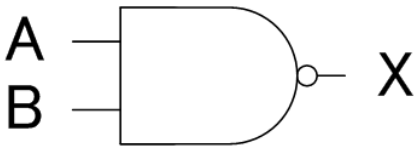
NAND

- 否定論理積

論理式

$$X = \overline{AB}$$

論理回路



真理値表

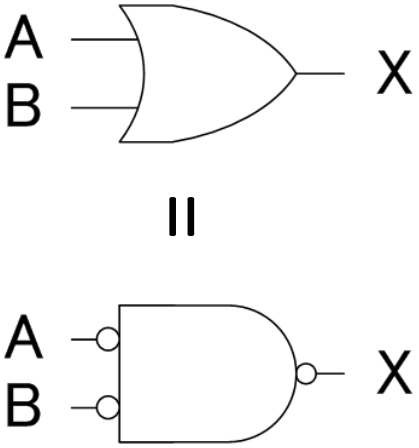
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

トランジスタとしてすごく重要な回路!
(後述)

論理回路の変形

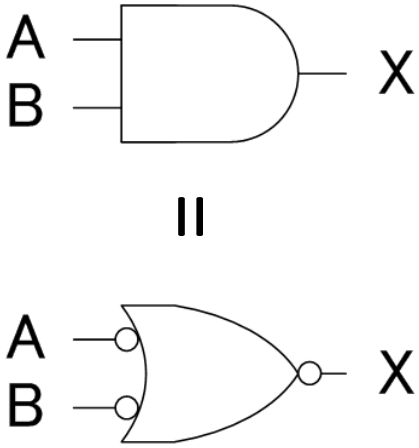
- ド･モルガンの定理より、簡単に変形できる

OR



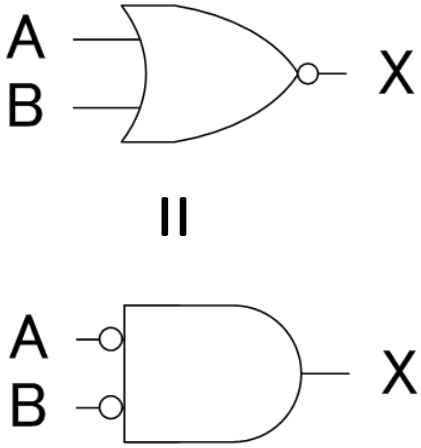
$$A + B = \overline{\overline{A + B}} = \overline{\overline{A}\,\overline{B}}$$

AND



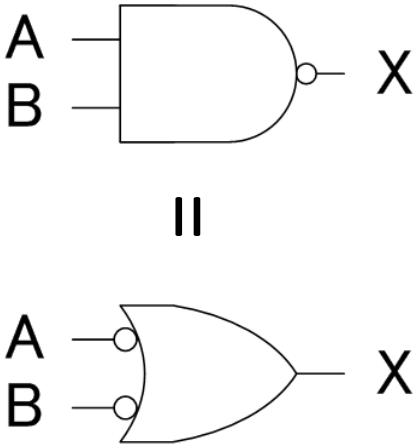
$$AB = \overline{\overline{AB}} = \overline{\overline{A} + \overline{B}}$$

NOR



$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

NAND



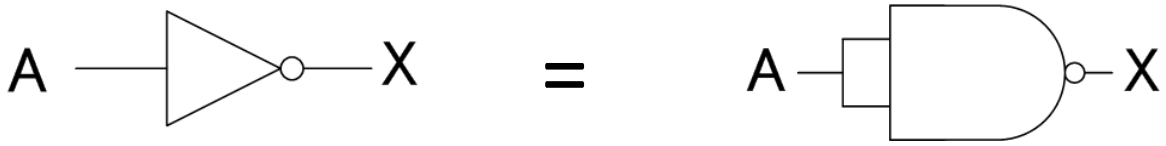
$$\overline{AB} = \overline{A} + \overline{B}$$

NANDで全て出来る

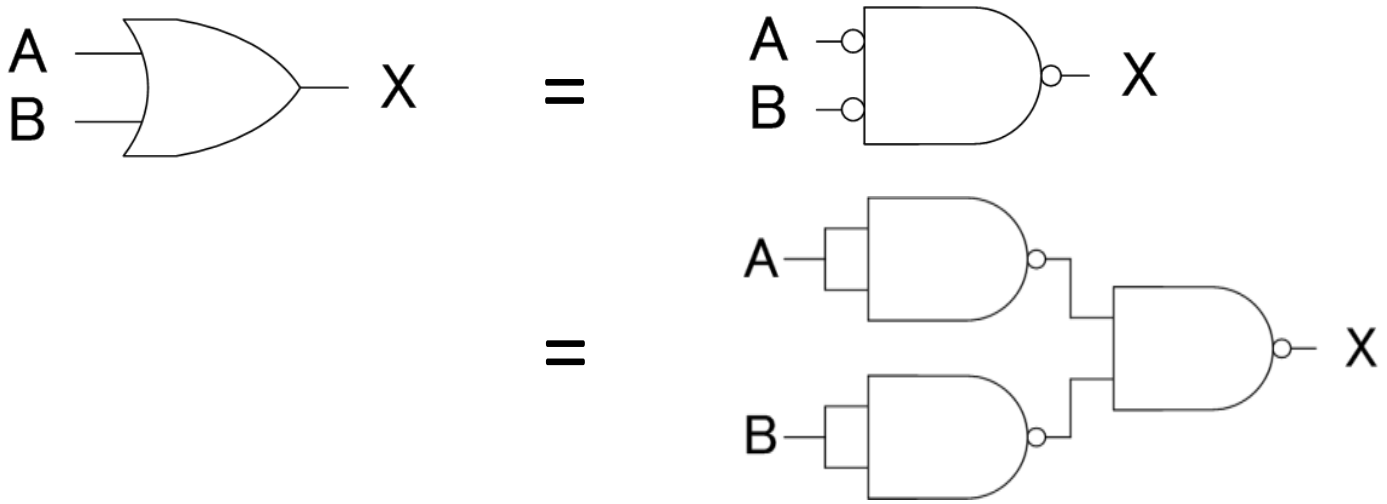
- それ自身の組合せで全て表せる = 完全系

例

NOT

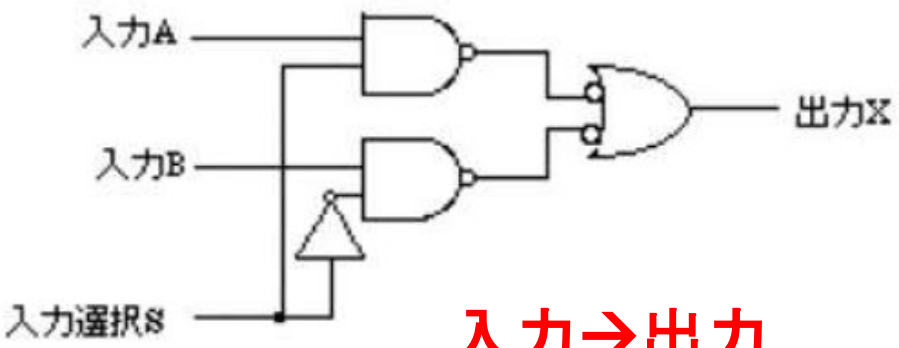


OR



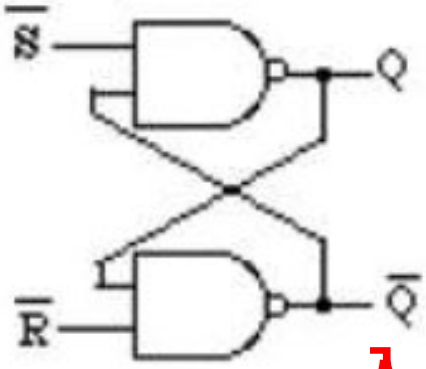
組合せ回路と順序回路

組合せ回路	順序回路
現在のみ考慮する	**過去も考慮**
出力がその時点の入力で決まる	出力に過去の出力状態が影響する
例：切替器、加算器	例：レジスタ、カウンタ



入力→出力

図 2: 1 ビット 2 チャンネル入力切換回路



入力→出力

図 10: SR−FF

順序回路

- 記憶素子(メモリー)に用いる
  - **フリップ・フロップ(FF)**で構成
    - 一時記憶回路(レジスタ)や数値計測(カウンタ)として使用
    - SR-FF, T-FF, JK-FF, D-FFの4種

「Flip Flop」は
- サンダルの「**パタパタ**」
- シーソーの「**ギッコンバッタン**」

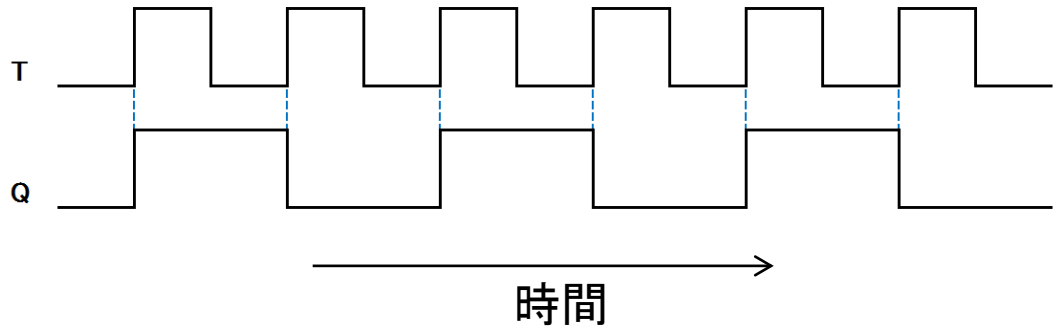
などの擬音語

T-FFの遷移表

入力T	現在の状態Q	次の状態Q^+	
0	0	0	
1	0 →	1	反転
0	1	1	
1	1 →	0	反転

$$Q^+ = T\overline{Q} + \overline{T}Q$$

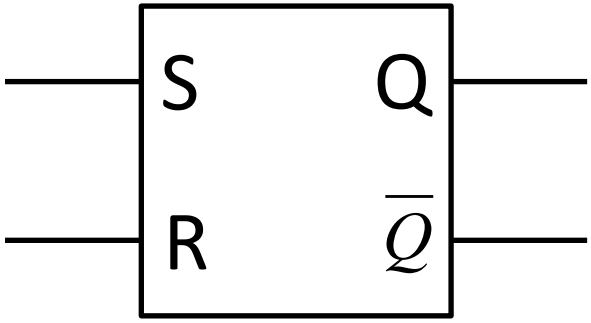
T-FFのタイミングチャート



SRフリップ・フロップ(SR-FF)

- SR-FFの動作
  - **外部からSまたはRに入力が無いと、Qから前の状態(0か1の出力)が出続ける**

→ 記憶装置



- **ホールド**：S=R=0入力(入力無)で、Qから前の状態を出力
- **セット**：S=1入力でQから1を出力
- **リセット**：R=1入力でQから0を出力
- **禁止**：S=R=1の同時入力は禁止(SR=0)
- $\overline{Q}$からは、Qの反転を出力

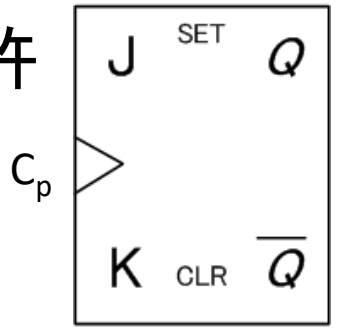
SR-FFの遷移表

入力S	入力R	現在の状態Q	次の状態Q^+	
0	0	0	0	ホールド
0	0	1	1	
0	1	0	0	リセット
0	1	1	0	
1	0	0	1	セット
1	0	1	1	
1	1	0	X	禁止
1	1	1	X	

S*R=0でなければならない

JKフリップ・フロップ(JK-FF)

- SR-FFにクロックC_pを追加して、禁止入力SR=1を許したもの
  - JK=1のときは反転出力
  - T-FFと同様、JK=1(反転)の時にはC_pが十分短くないと発振

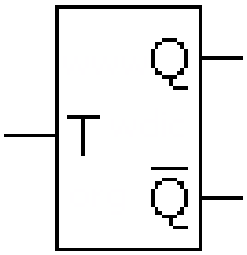


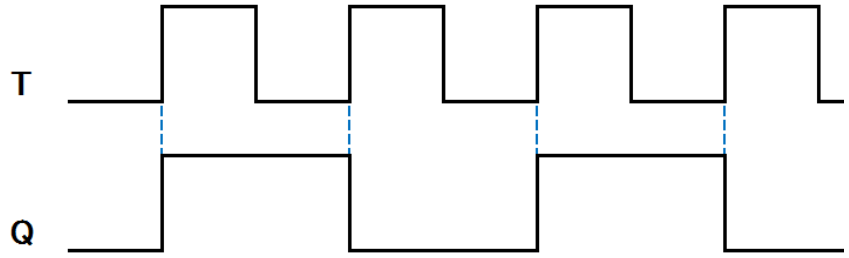
JK-FFの遷移表

入力J	入力K	現在の状態Q	次の状態Q^+	
			C_p=1	C_p=0
0	0	0	0 ホールド	0 ホールド
0	0	1	1	1
0	1	0	0 リセット	0 ホールド
0	1	1	0	1
1	0	0	1 セット	0 ホールド
1	0	1	1	1
1	1	0	1 反転	0 ホールド
1	1	1	0	1

Tフリップ・フロップ(T-FF)

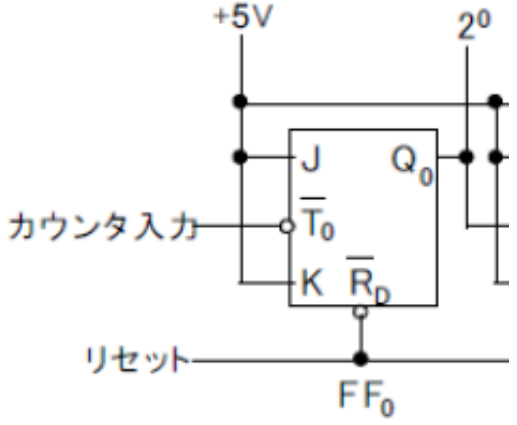
- 入力Tがある度に出力Qが反転する(トグル動作)
  - 1bit-2進カウンタに相当
  - Tのパルス幅が十分短くないと発振
    - T=1(反転)が長いと1→0→1→0→…と発振する





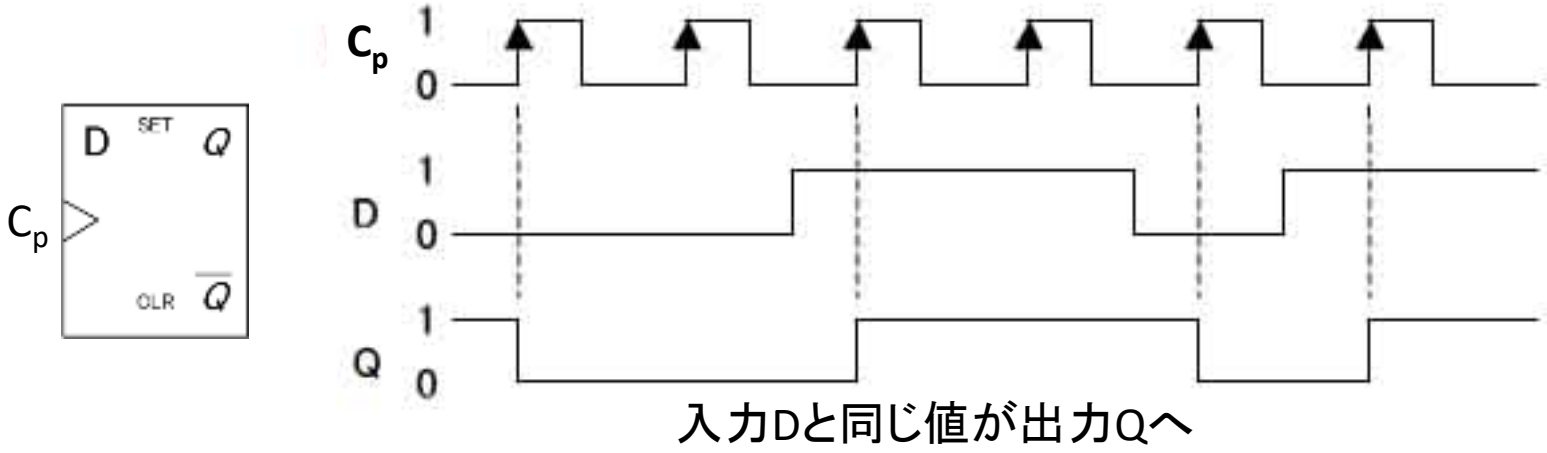
クロック毎に出力が反転

2週目実験7で行うように、JK-FFのJ-Kどちらも+5VにするとT-FFと同じ動作 (JK=11で反転動作)



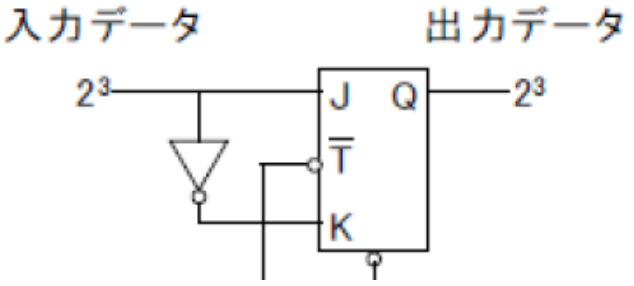
Dフリップ・フロップ(D-FF)

- クロックC_pが入力する直前の入力Dと同一の出力がQに出力する



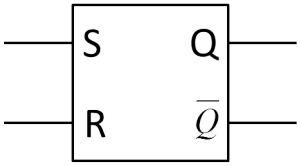
入力Dと同じ値が出力Qへ

2週目実験5,6で行うように、JK-FFのJ-Kをインバーターを通じてつなげるとD-FFと同じ動作(JK=10/01で出力も1/0)



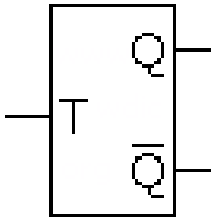
フリップ・フロップまとめ

SR-FF



S	R	Q
0	0	保持
0	1	0
1	0	1
1	1	禁止

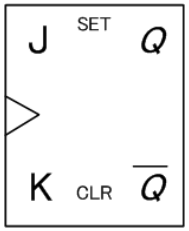
T-FF



＊Qは次の状態のQを示す

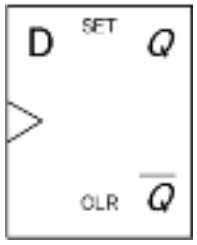
T	Q
0	保持
1	反転

JK-FF



J	K	Q
0	0	保持
0	1	0
1	0	1
1	1	反転

D-FF

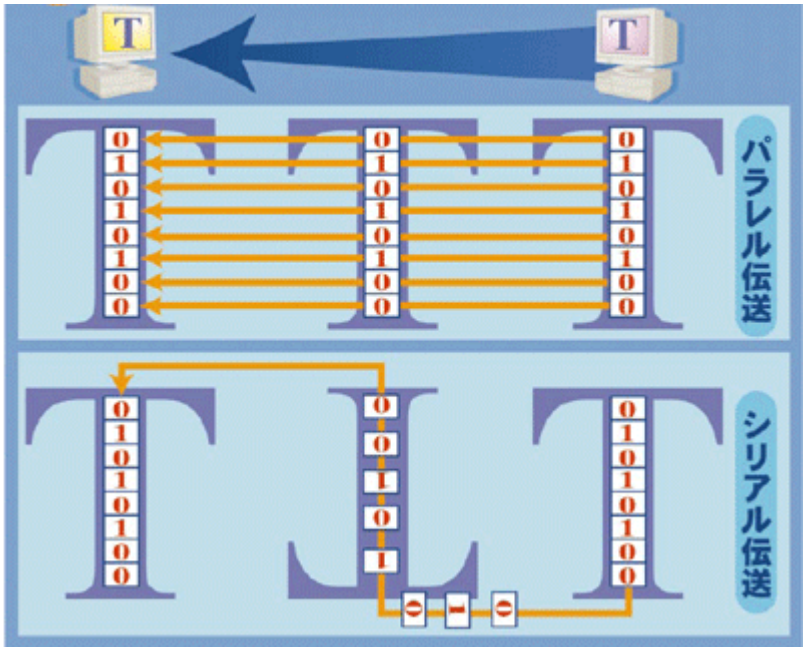


D	CLK	Q
0	立上	0
1	立上	1
0/1	立下	保持

フリップ・フロップの使用例

--- パラレル(並列)とシリアル(直列) ---

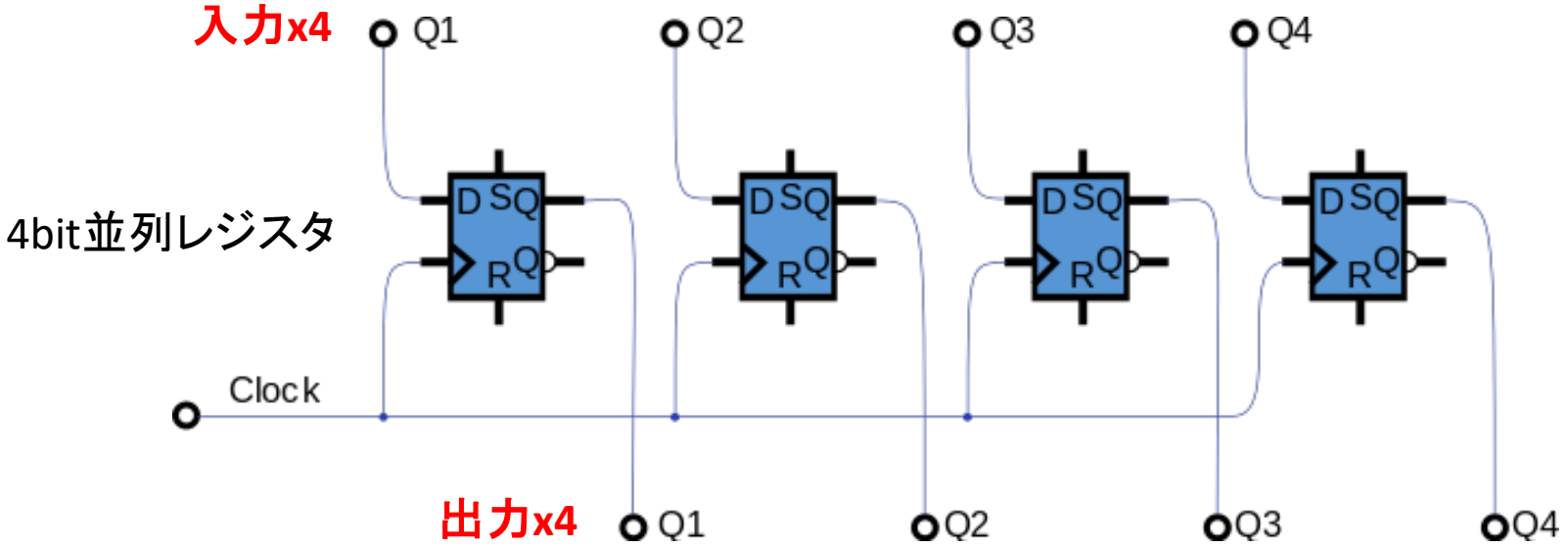
- デジタル信号の伝送には
  パラレルとシリアルがある
  - パラレルは一度に多bit伝送
    - データ遅延が問題
  - シリアルは1bitずつ伝送
    - 現在の主流



http://itpro.nikkeibp.co.jp

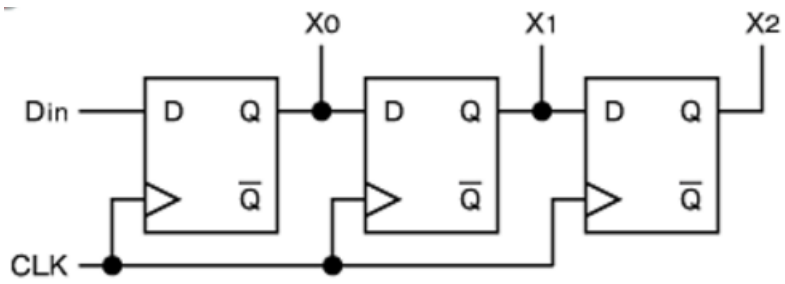
レジスタ

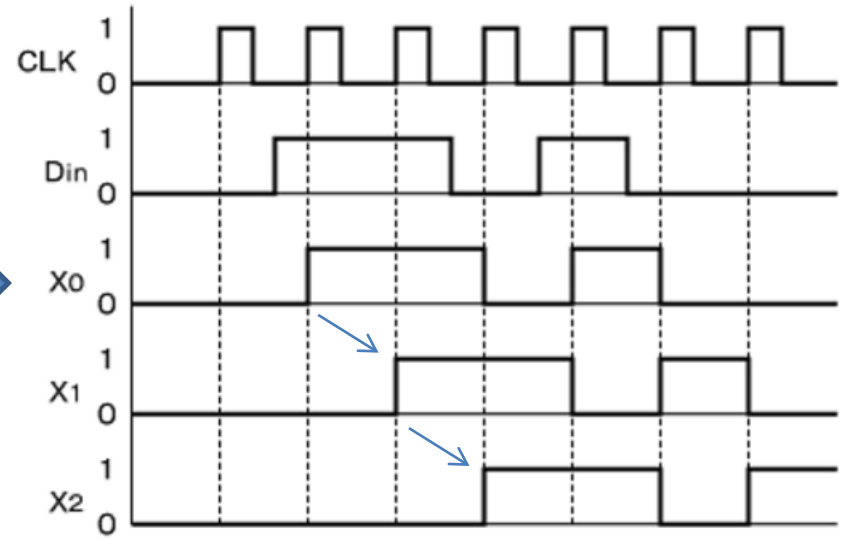
- レジスタ = データを一時的に記憶する一種のメモリで、複数のFFで構成
- 並列レジスタ
  - FFを並列に接続、D-FFで簡単に構成可能
  - パラレルデータを入力として読み込み、記憶する
  - (リセットするか)新たなデータが入力されるまで保持



レジスタ

- 直列レジスタ(シフト・レジスタ)
  - FFを直列に接続、JK-FFで簡単に構成可能
  - シリアルデータを入力として読み込み、記憶する
  - (パラレル・シリアルデータどちらも入出力可能)



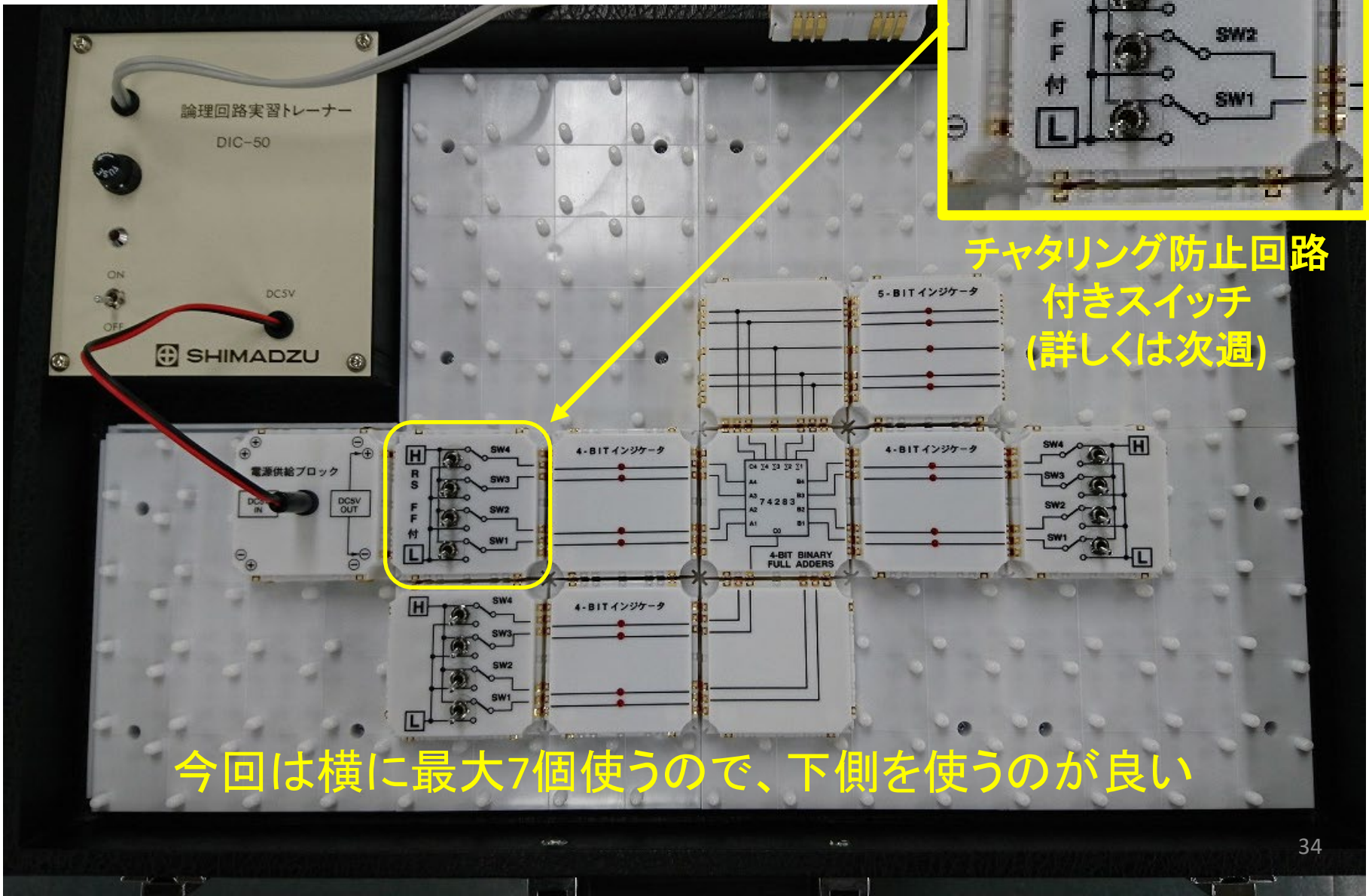


1クロックごとにデータがシフトしていく

論理回路実習トレーナー

- テキスト“DIC-50論理回路実験”に従って実験を行う
  - テキスト”DIC-50論理回路実験ロック配置図”を参考にブロックを配置し、実際に回路を動かす
  - 課題の答えを書き込む

実際の配置



チャタリング防止回路
付きスイッチ
(詳しくは次週)

今回は横に最大7個使うので、下側を使うのが良い

課題6(4ビット全加算回路)について

6　全加算回路（4 ビット バイナリー フルアダー）

加算回路には下の桁からの桁上がりを考えない 2 入力加算の半加算器（デジタル I で学んだ）と、下の桁からの桁上がりを考えた 3 入力加算の全加算器とがある。本実験では 4 ビットの 2 つの入力データ(A_1~A_4 と B_1~B_4)および下の桁からの桁上がり(C_0)の加算結果を、4 ビットの出力データ（Σ_1~Σ_4）と上の桁への桁上がり(C_4) を出力する全加算回路の実験である。

真理値表

入力			出力	
A_n	B_n	下から桁上り C_{n-1}	和 Σ_n	桁上り C_n
0	0	0	0	0
1	0	0		
0	1	0		
1	1	0	0	1
0	0	1	1	0
1	0	1		
0	1	1		
1	1	1		

実験予定

入力									出力				
A				B				桁上り	和 Σ				桁上り
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	0	0	1	1	0	0	0					
								0					
								1					

桁上がり

- ビット演算の基礎

- A+B = Σ & 桁上がりC
  - 1+0 = 1 & 桁上がり無し
  - 1+1 = 0 & 桁上がり1

- A,Bを2bitデータと考えると
  - 01+00 = 01
  - 01+01 = 10

真理値表

入力			出力	
A_n	B_n	下から桁上り C_{n-1}	和 Σ_n	桁上り C_n
0	0	0	0	0
1	0	0		
0	1	0		
1	1	0	0	1
0	0	1	1	0
1	0	1		
0	1	1		
1	1	1		

同じことをいっている

＊「下からの桁上がり」は単純にA+B+Cと考えてOK
(C=01)

桁上がりの加算例

入力									出力				
A				B				桁	和Σ				桁
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	1	1	1	0	0	1	1					

ビット演算と同じ → A4-A3-A2-A1の順で下に行く
AとBを逆順に並べて、桁上がり（C0）をAとBの下位ビット（A1/B1）に足していく

	5	4	3	2	1
A		1	1	0	1
B		1	0	0	1
桁上り					1
Σ					

桁上がりの加算例

入力									出力				
A				B				桁	和Σ				桁
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	1	1	1	0	0	1	1	1				

ビット演算と同じ → A4-A3-A2-A1の順で下に行く
AとBを逆順に並べて、桁上がり（C0）をAとBの下位ビット（A1/B1）に足していく

	5	4	3	2	1
A		1	1	0	1
B		1	0	0	1
桁上り				1	1
Σ					1

桁上がりの加算例

入力									出力				
A				B				桁	和Σ				桁
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	1	1	1	0	0	1	1	1	1			

ビット演算と同じ → A4-A3-A2-A1の順で下に行く
AとBを逆順に並べて、桁上がり（C0）をAとBの下位ビット（A1/B1）に足していく

	5	4	3	2	1
A		1	1	0	1
B		1	0	0	1
桁上り			**0**	**1**	1
Σ				**1**	**1**

桁上がりの加算例

入力									出力				
A				B				桁	和Σ				桁
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	1	1	1	0	0	1	1	1	1	1		

ビット演算と同じ → A4-A3-A2-A1の順で下に行く
AとBを逆順に並べて、桁上がり（C0）をAとBの下位ビット（A1/B1）に足していく

	5	4	3	2	1
A		1	1	0	1
B		1	0	0	1
桁上り		**0**	**0**	**1**	1
Σ			**1**	**1**	**1**

桁上がりの加算例

入力									出力				
A				B				桁	和Σ				桁
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	1	1	1	0	0	1	1	1	1	1	0	

ビット演算と同じ → A4-A3-A2-A1の順で下に行く
AとBを逆順に並べて、桁上がり（C0）をAとBの下位ビット（A1/B1）に足していく

	5	4	3	2	1
A		1	1	0	1
B		1	0	0	1
桁上り	**1**	**0**	**0**	**1**	1
Σ		**0**	**1**	**1**	**1**

桁上がりの加算例

入力									出力				
A				B				桁	和Σ				桁
A1	A2	A3	A4	B1	B2	B3	B4	C0	Σ1	Σ2	Σ3	Σ4	C4
1	0	1	1	1	0	0	1	1	1	1	1	0	1

ビット演算と同じ → A4-A3-A2-A1の順で下に行く
AとBを逆順に並べて、桁上がり（C0）をAとBの下位ビット（A1/B1）に足していく

	5	4	3	2	1
A		1	1	0	1
B		1	0	0	1
桁上り	1	0	0	1	1
Σ	1	0	1	1	1

参考資料

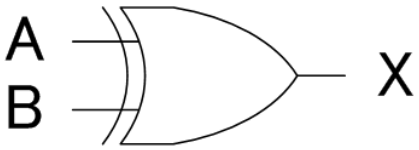
Exclusive OR (XOR)

- 排他的論理和

論理式

$$X = A \oplus B$$
$$= \overline{A}B + A\overline{B}$$
$$= (A + B)(\overline{A} + \overline{B})$$

論理回路



真理値表

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

同じものが入ると”0”
= 不一致回路

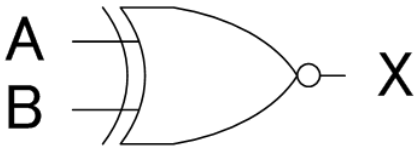
Exclusive NOR (XNOR)

- 否定排他的論理和

論理式

$$X = A \odot B$$
$$= \overline{A \oplus B}$$
$$= AB + \overline{A}\,\overline{B}$$
$$= (A + \overline{B})(\overline{A} + B)$$

論理回路



真理値表

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

同じものが入ると”1”
= 一致回路

入力切替回路(マルチプレクサ)1

- 入力A/Bを出力Xに切り替えて出力したい
  - 制御信号Sを用いる

S	A	B	X
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

X=1の時を抜き出して論理式を作り、簡略化

$$X = \overline{S}\,\overline{A}B + \overline{S}A\overline{B} + SA\overline{B} + SAB$$

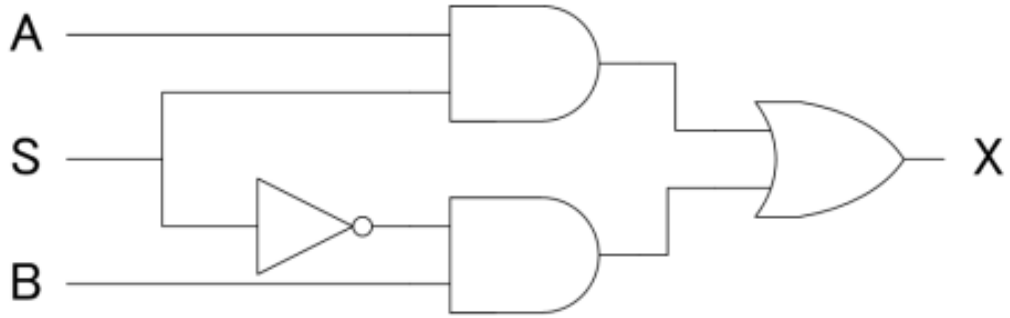
$$= AS + B\overline{S}$$

S=1のとき：X = A･1 + B･0 = A
S=0のとき：X = A･0 + B･1 = B

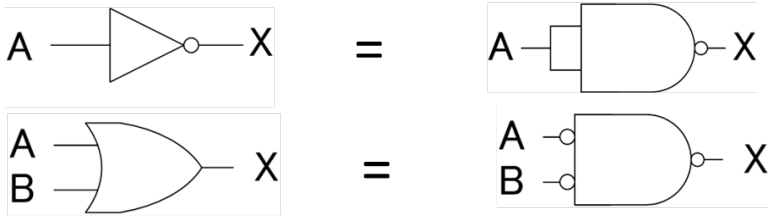
入力切替回路(マルチプレクサ)2

- 論理式より回路が組める

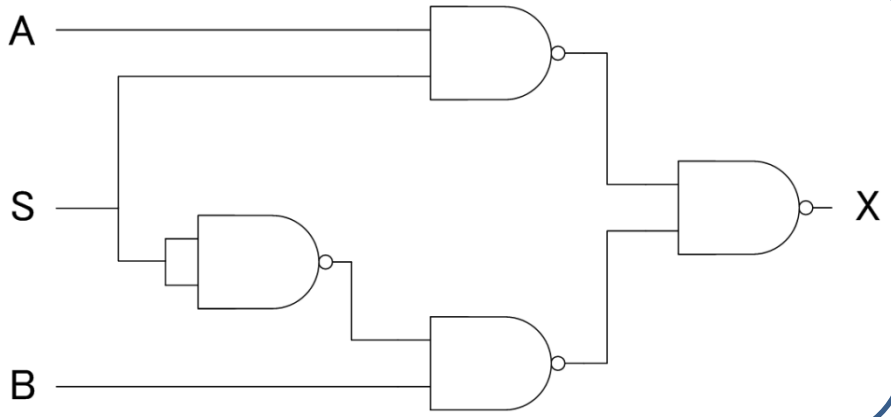
$$X = AS + B\overline{S}$$



実際の回路図



を用いて



出力切替回路(デマルチプレクサ)1

- 入力Aを出力X1/X2に切り替えて出力したい
  - 制御信号Sを用いる

S	A	X1	X2
0	0	0	0
0	1	0	1
1	0	0	0
1	1	1	0

論理式は

$$X1 = AS$$

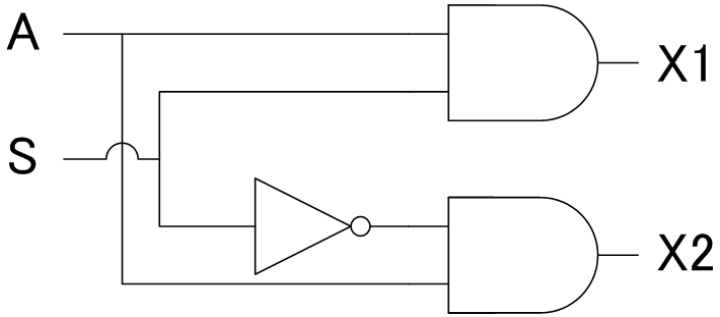
$$X2 = A\overline{S}$$

S=1のとき：X1 = A, X2 = 0
S=0のとき：X1 = 0, X2 = A

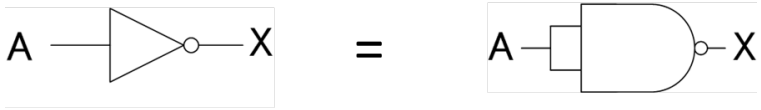
出力切替回路(デマルチプレクサ)2

- 論理式より回路が組める

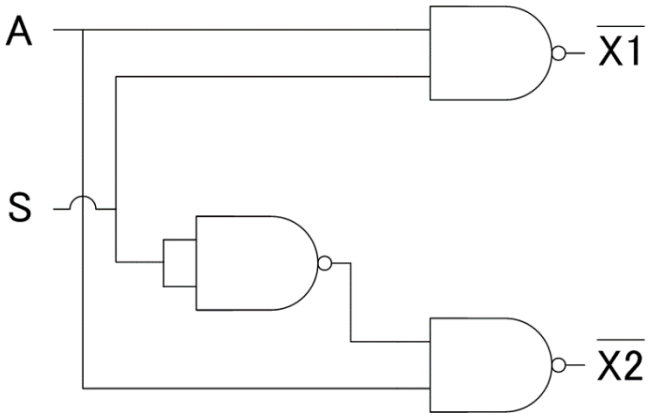
$$X1 = AS,\ X2 = A\overline{S}$$



実際の回路図



を用いて



半加算回路1

- 前の段からの桁上がりを考えない加算
  - 全加算の元

A+Bの計算

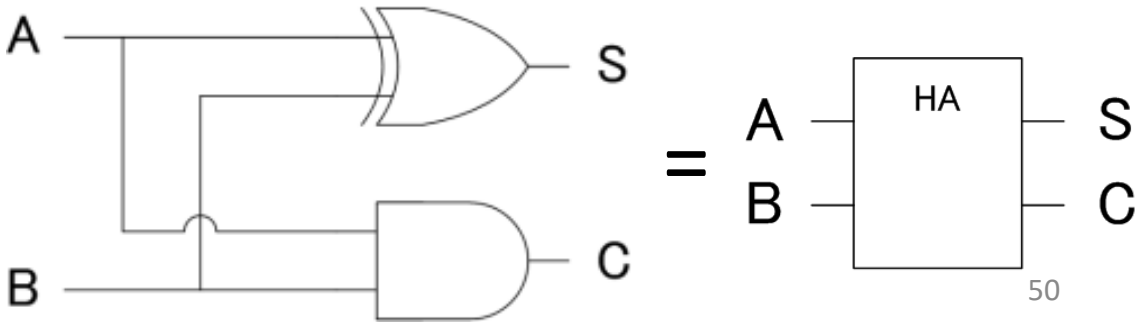
A	B	和S sum	桁C carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

論理式は

$$S = \overline{A}B + A\overline{B} = A \oplus B$$

XOR (Exclusive OR)

$$C = AB$$

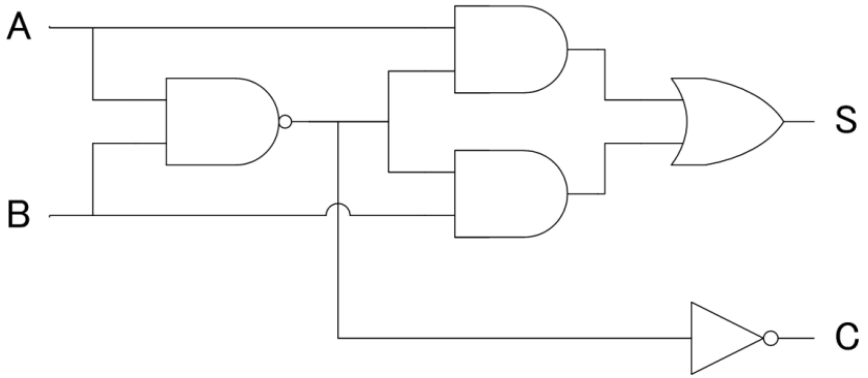


半加算回路2

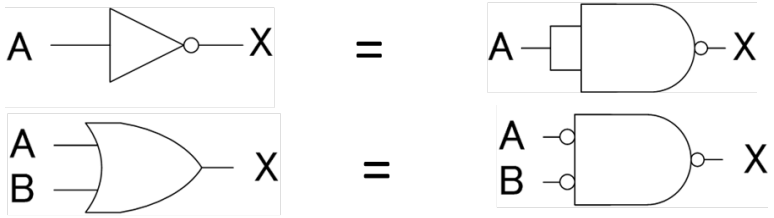
- 論理式より回路構成を考える

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

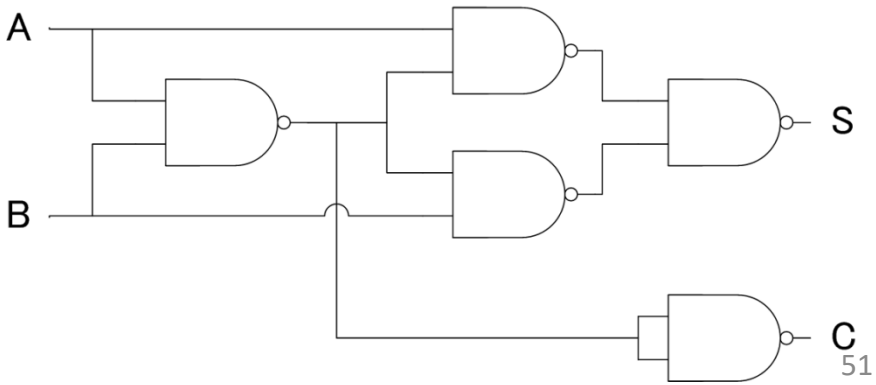
$$C = AB$$



実際の回路図



を用いて



全加算回路1

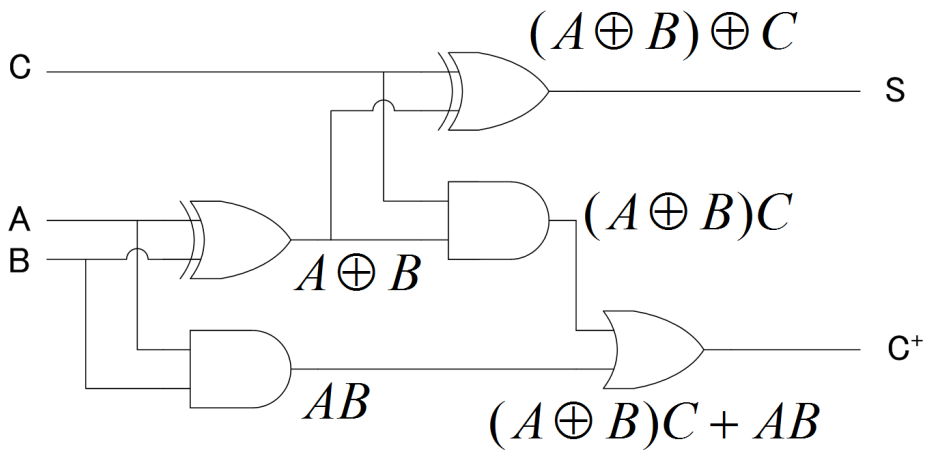
- 前の段からの桁上がりを考える加算
  - コンピューター演算の基本

A+Bの計算(Cは前の段からの桁上がり)

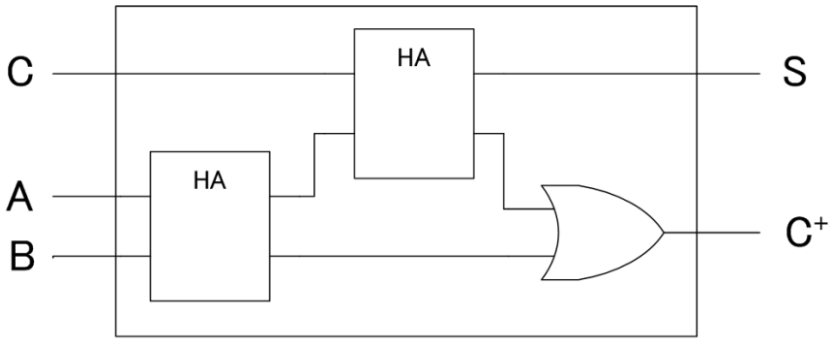
A	B	C	和S sum	桁C+ carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$
\begin{aligned}
S &= \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC \\
&= C(\overline{A}\,\overline{B} + AB) + \overline{C}(\overline{A}B + A\overline{B}) \\
&= C(\overline{A \oplus B}) + \overline{C}(A \oplus B) \\
&= C \oplus (A \oplus B) \\
C^{+} &= \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC \\
&= C(\overline{A}B + A\overline{B}) + AB(\overline{C} + C) \\
&= C(A \oplus B) + AB
\end{aligned}
$$

全加算回路2

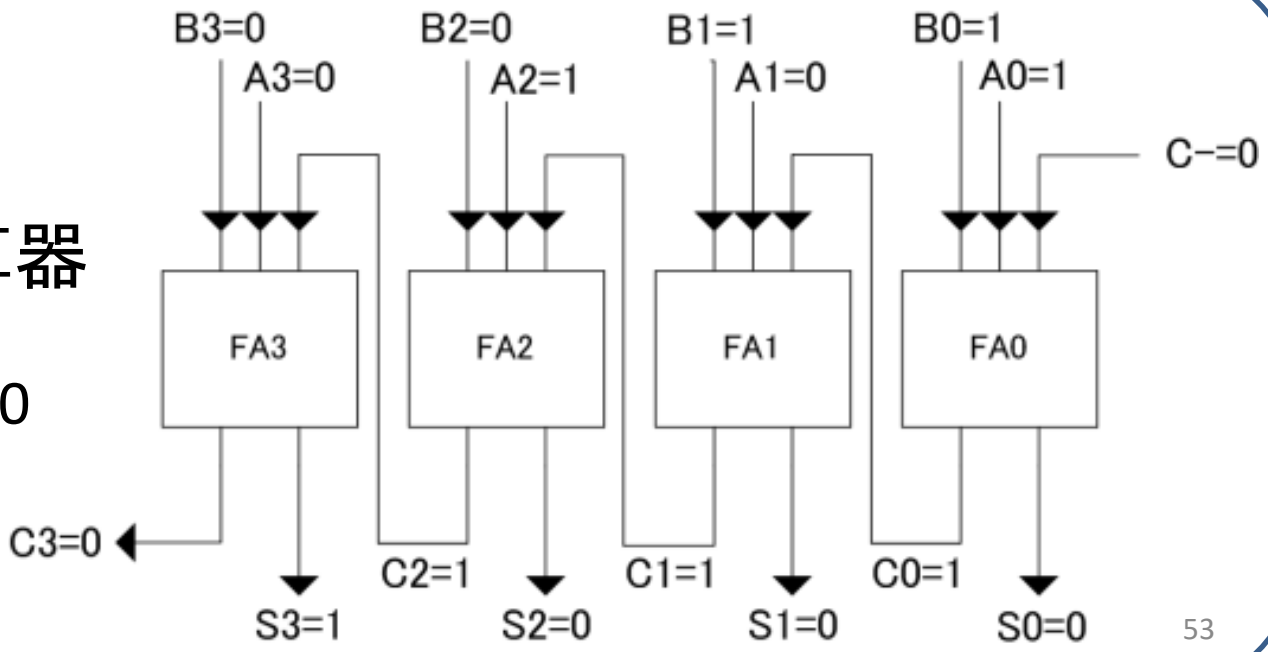


=



例) 4bit並列加算器

0101 + 0011 = 1000



SRフリップ・フロップ(SR-FF)

- 遷移表より論理関数を求める

SR-FFの遷移表

入力S	入力R	現在の状態Q	次の状態Q^+	
0	0	0	0	ホールド
0	0	1	1	
0	1	0	0	リセット
0	1	1	0	
1	0	0	1	セット
1	0	1	1	
1	1	0	X	禁止
1	1	1	X	

SR=0でなければならない

$$Q^+ = \overline{S}\overline{R}Q + S\overline{R}\overline{Q} + S\overline{R}Q$$
$$= S\overline{R}\overline{Q} + \overline{R}Q(S + \overline{S})$$
$$= S\overline{R}\overline{Q} + \overline{R}Q$$

S=1の時は強制的にQ^+=1だから
(S=R=1は禁止である)

$$S\overline{R}\overline{Q} = S$$

つまり

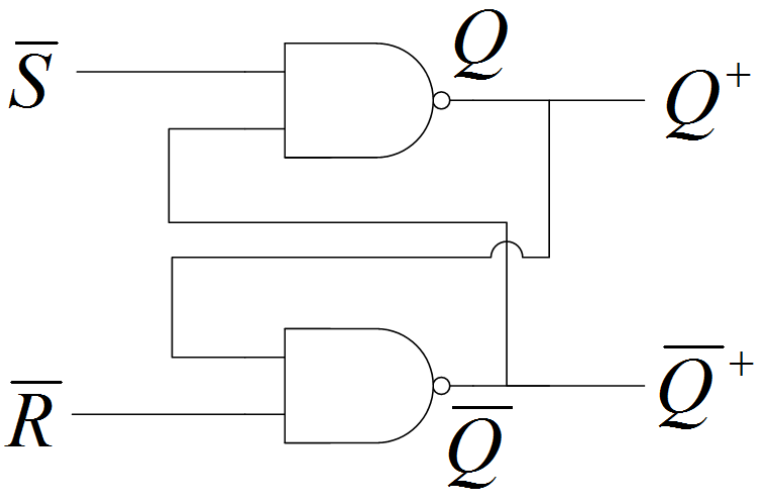
$$\boxed{Q^+ = S + \overline{R}Q}$$

＊カルノー図を使えばもう少し簡単に求まる

SRフリップ・フロップ(SR-FF)2

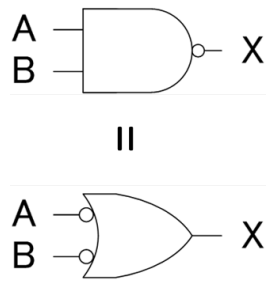
- 論理式より回路を組む

$$Q^+ = \overline{\overline{Q^+}} = \overline{\overline{S + \overline{R}Q}} = \overline{\overline{S}\,\overline{\overline{R}Q}}$$



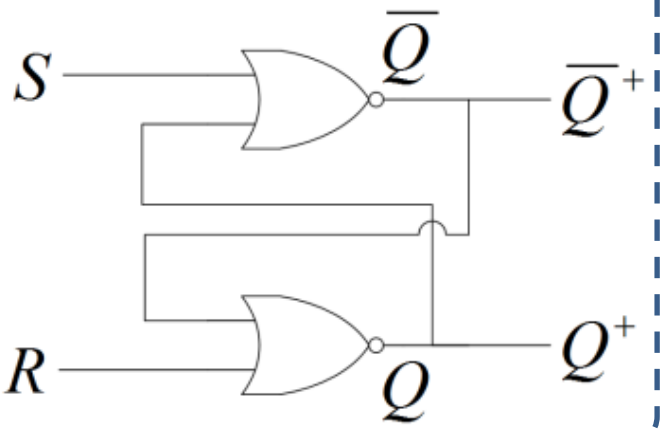
入力がS^{bar},R^{bar}に注意

S^{bar}	R^{bar}	Q	Q^+
1	1	0	0
1	1	1	1
1	0	0	0
1	0	1	0
0	1	0	1
0	1	1	1
0	0	0	X
0	0	1	X

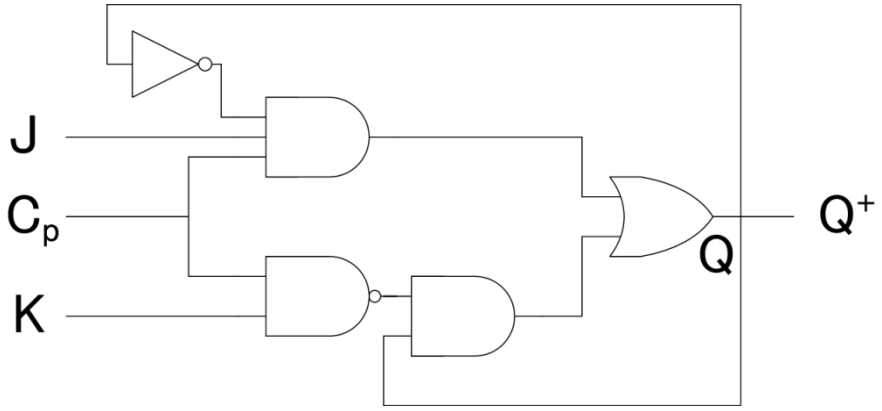


II

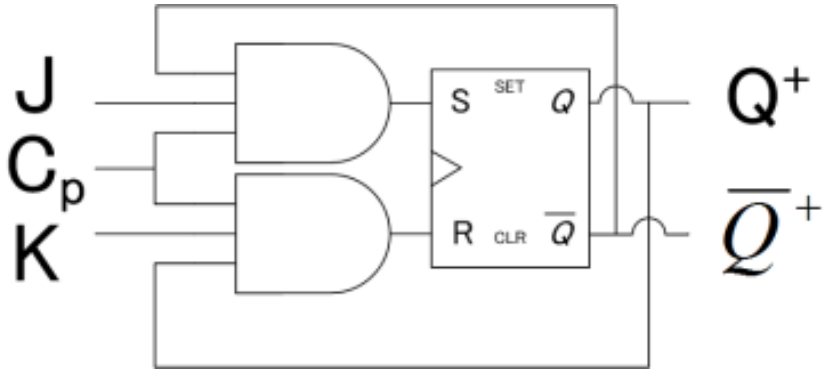
を用いると、下と同等



JKフリップ・フロップ(JK-FF)2

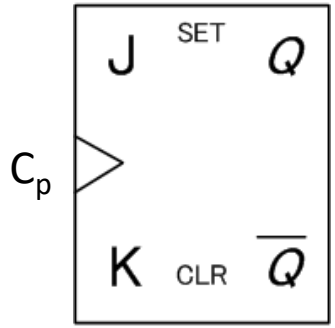


,



Cp=1のとき $Q^+ = \overline{K}Q + J\overline{Q}$

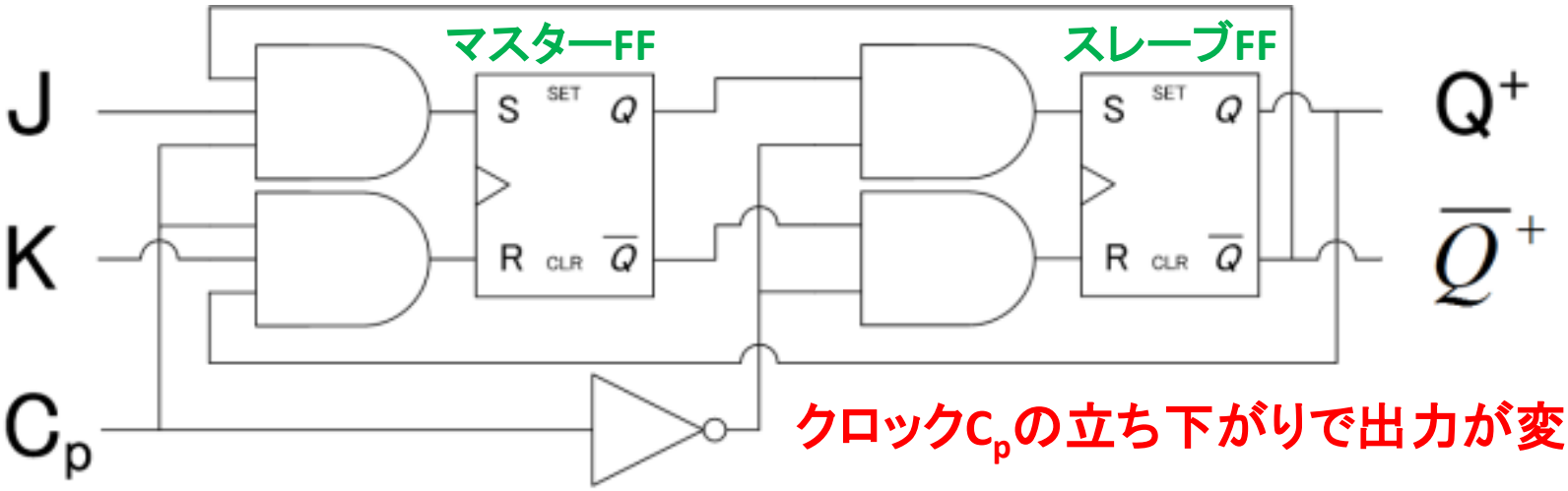
=



JK-FFのJ-Kをつなげて入力するとT-FFと同じ動作

JKフリップ・フロップ(JK-FF)3

- 発振を抑えるための実用回路として以下が知られる
  - エッジトリガタイプ
    - ICの内部でC_pを短いパルスに変換
  - マスタースレーブタイプ
    - マスターFFとスレーブFFを切り替えることにより発振を抑える



Tフリップ・フロップ(T-FF)

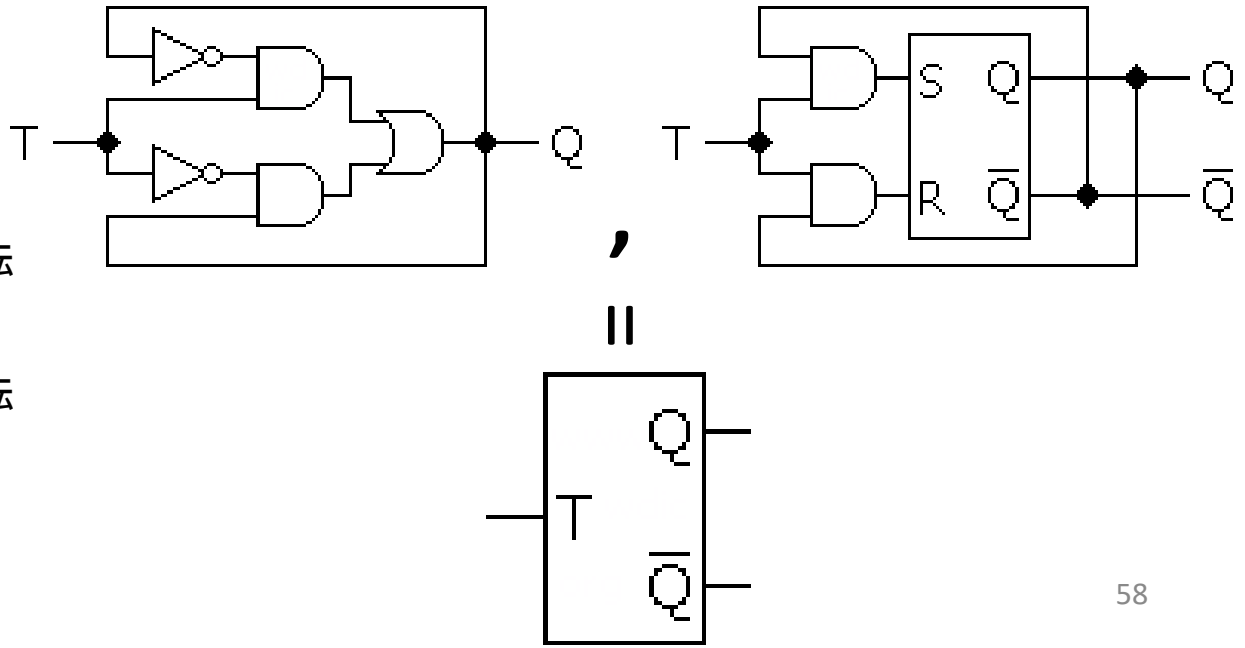
- 入力Tがある度に出力Qが反転する(トグル動作)
  - 1bit-2進カウンタに相当
  - Tのパルス幅が十分短くないと発振
    - T=1(反転)が長いと1→0→1→0→…と発振する

T-FFの遷移表

入力T	現在の状態Q	次の状態Q⁺
0	0	0
1	0 →	1
0	1	1
1	1 →	0

反転

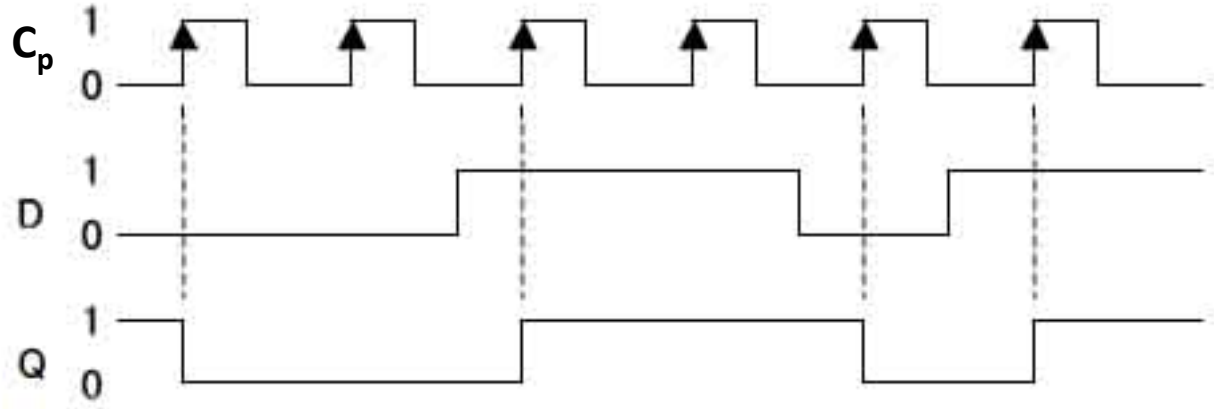
反転

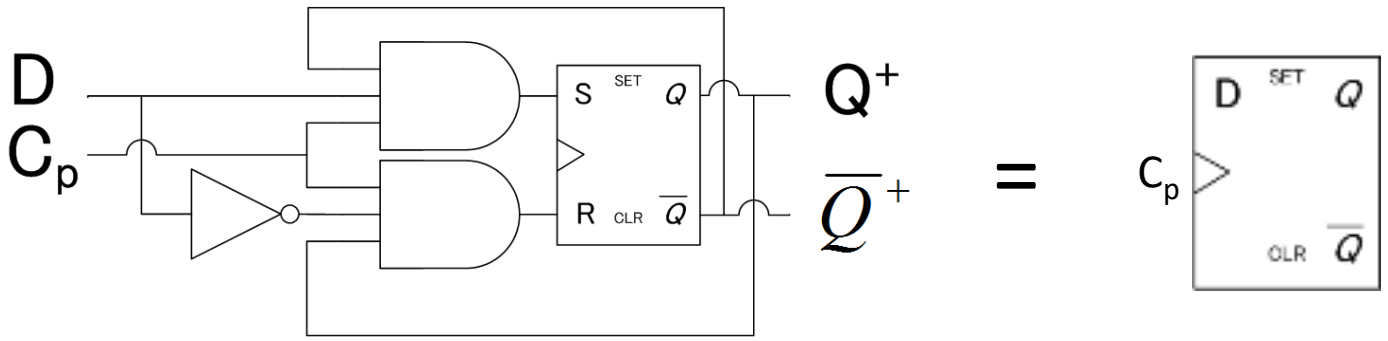


$$Q^{+} = T\overline{Q} + \overline{T}Q$$

Dフリップ・フロップ(D-FF)

- クロックC_pが入力する直前の入力Dと同一の出力がQに出力する



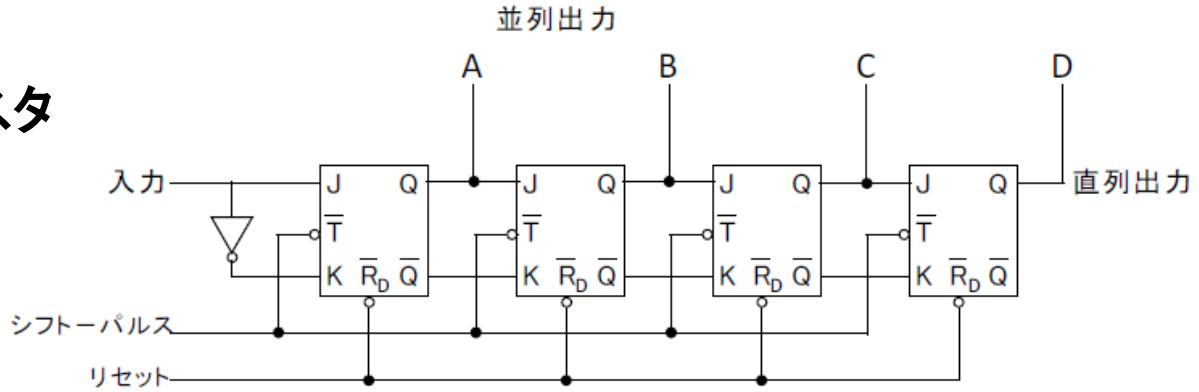


JK-FFのJ-Kをインバーターを通じてつなげるとD-FFと同じ動作

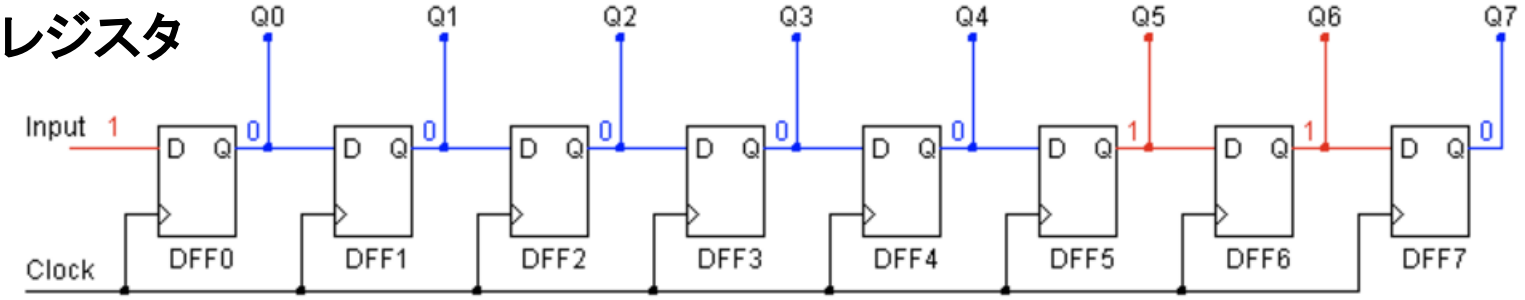
直列レジスタ(シフト・レジスタ)2

- シリアルとパラレルのインタフェース変換
  - ネットワーク通信などに用いる

4bitシフト・レジスタ



8bitシフト・レジスタ



カウンタ

- 非同期式2進N段カウンタ
  - T-FFを使用 (JK-FFのJ-Kをつないでも同じ)
  - N段で0~2^N-1までカウント
    - 4個では0~2^4-1 = 0~15までの16進カウンタ
  - FFの数だけの遅延が問題 → 同期式では無問題

4bit16進カウンタ