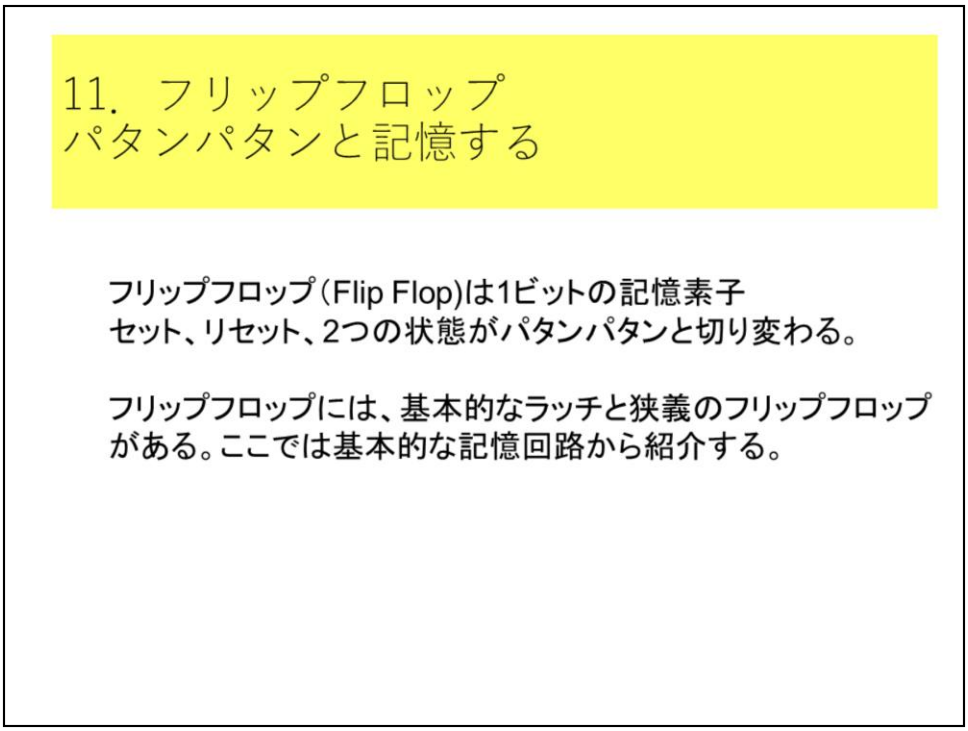
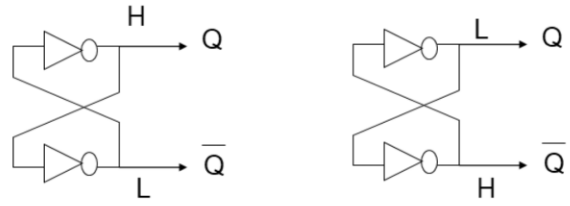


フリップフロップは、1ビットの記憶素子です。セット、リセットの2つの状態を持っていて、どちらの状態になっているかで情報を記憶します。計算機基礎を取っている方は機能面の働きは理解していると思います。ここでは内部構造、STA(Static Timing Analysis)をやります。思い出して関連付けてください。

最も簡単な記憶回路

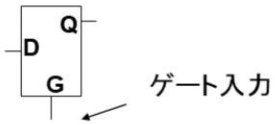


セットの状態　　リセットの状態

一度その状態になったら保持し続ける→　記憶
このままだと切り替える手段がない→記憶用のデータとタイミング入力

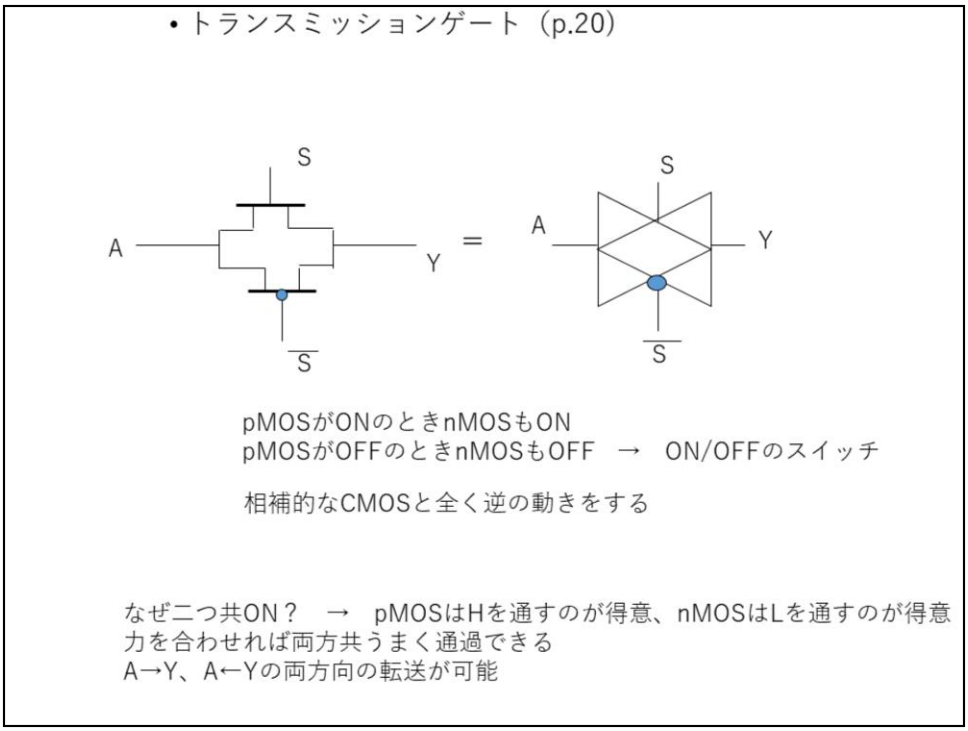
最も簡単な記憶回路は、NOTゲートを2つ用意して、出力を互いの入力に繋ぎます。片方の出力がHの時はもう片方はL、片方がLならばもう片方はHになります。片方の出力にQという名前を付けると、もう片方はQバーになります。QがHの状態をセット、QバーがHの状態をリセットと呼びます。この回路は一度セットになるとずっとセットの状態を、リセットになるとずっとリセット状態を維持します。つまり記憶をすることができるのですが、これではデータを記憶する形になっておらず使いにくいです。

D-ラッチ



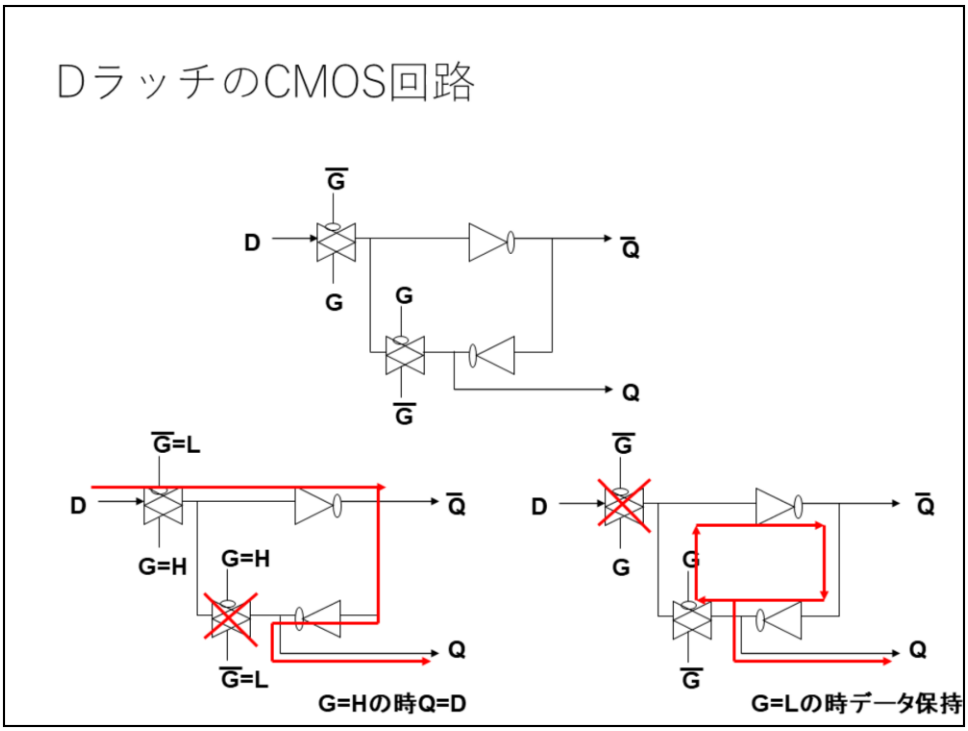
G=Hの時のD入力を記憶して
Qから出力

データを記憶するためには、記憶するデータを入れるための入力が必要ですが、それ以外にも、記憶するタイミングを示す入力が必要です。D-ラッチは、データを入力するDとタイミングを示すG、出力のQを持っています。基本動作は、G=Hの時のD入力を記憶してQに出力します。

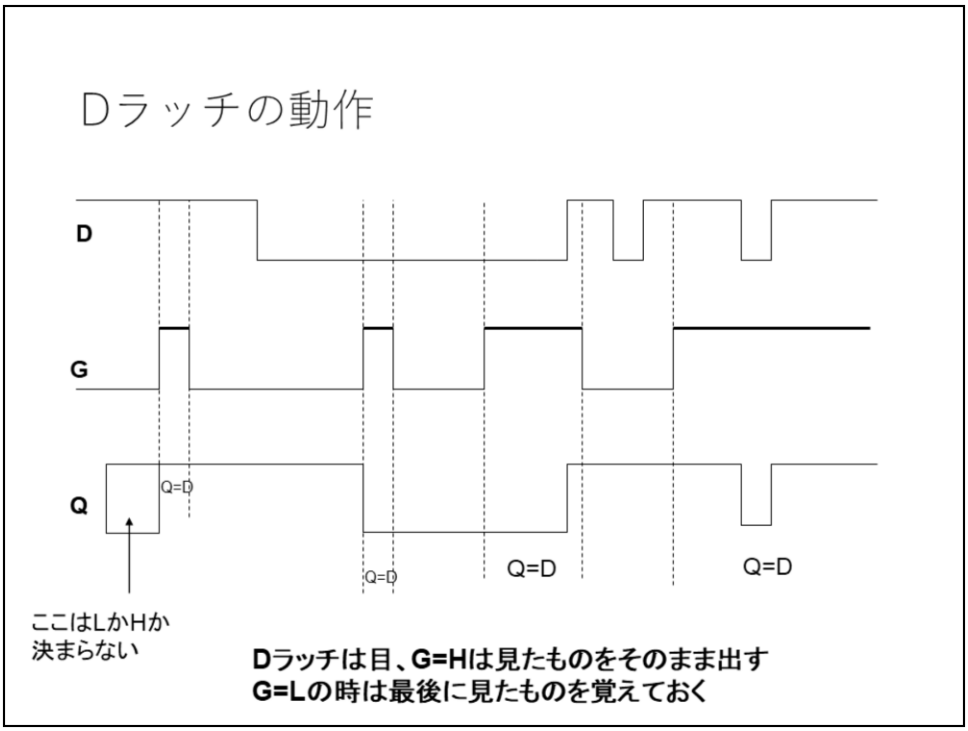


データの入力と記憶を切り替えるためには、スイッチの役割をする簡単な素子があると便利です。以前紹介したトランスミッションゲート（トランスファーゲート）が、この目的で使えます。

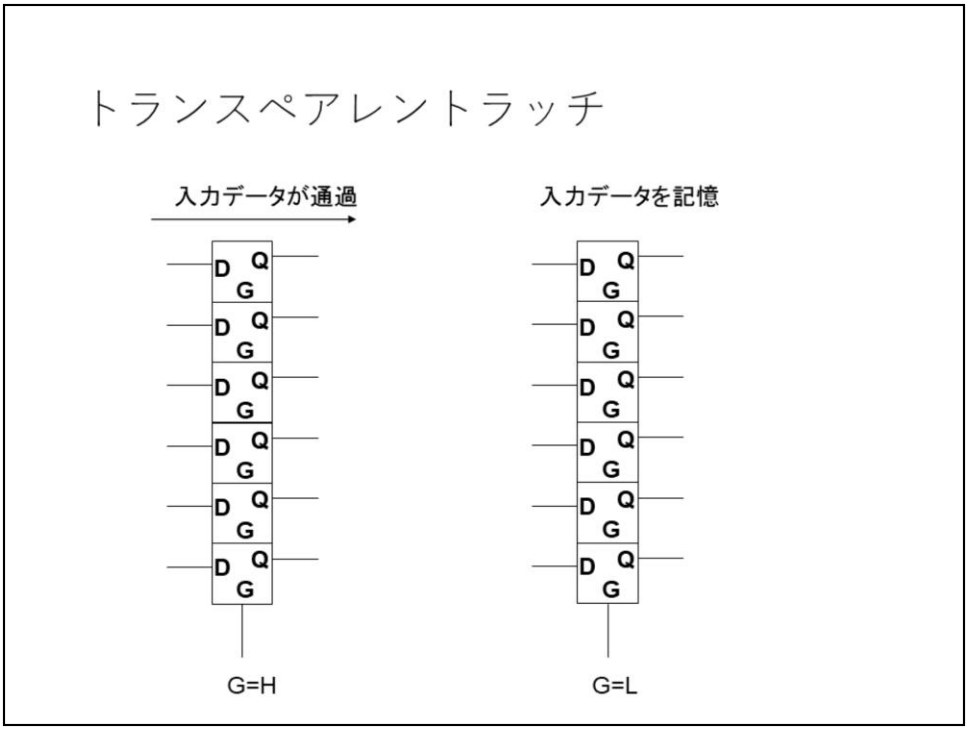
トランスミッションゲートでは、nMOSとpMOSのソースードレイン同士を接続し、ゲートには片方にSを与えたとすると、もう片方にはSの反転信号を与えます。この場合、pMOSがONのときはnMOSもONになり、nMOSがOFFのときはpMOSもOFFになります。つまり、両方が必ず同じ状態になるのです。このことにより、ONになった時はAとYが接続され、OFFの時は、AとYが切り離されます。これをトランスミッションゲート、あるいはトランスファーゲートと呼びます。なぜnMOSとpMOSの両方必要か？というと、以前解説したとおり、それぞれ通すのが得意なレベルが違うからです。Hレベルは pMOSが、Lレベルは nMOSが主に動作することで、両方のレベルを高速に通すことができます。トランスミッションゲートは、双方向である点にご注意ください。この双方向性を利用してFPGAの配線用スイッチとして使うことができます。



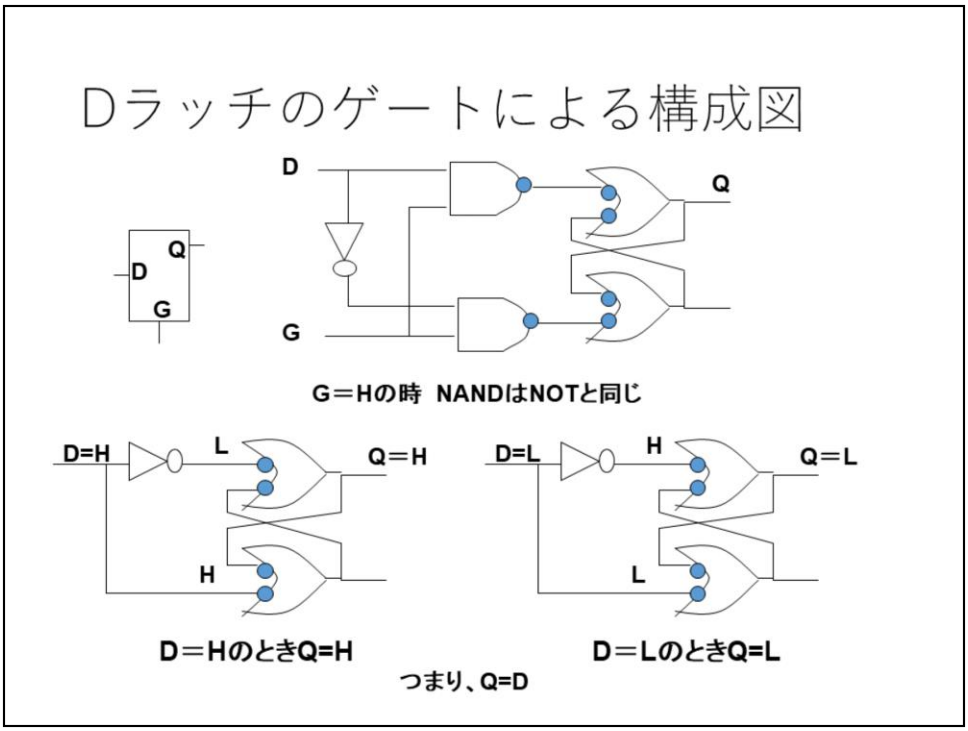
DラッチはCMOSのトランスミッションゲートを使うと簡単に実現できます。トランスミッションゲートはONになると接続、OFFになると切断するスイッチとして使えることを思い出しましょう。G＝Hの時は、入力のトランスミッションゲートがON、フィードバックのトランスミッションゲートがOFFになり、入力Dが2つのNOTを介してQに筒抜けになります。G＝Lにすると、入力側が切れる一方、フィードバック側がONになります。このことで、NOTゲートの8字つなぎが実現され、状態が保存されます。



Dラッチのタイミングチャートを示します。計算機基礎を取っている方は、習ったと思います。Dラッチの動作はDを目で見ている動作にたとえられます。G＝Hの時は目を開きます。この時は見たものを全てがQに筒抜けになります。G=Lにすると目を閉じるのですが、この時最後に見たものを覚えておきます。



Dラッチを必要なビット数並べてGを共通にします。G＝Hにすると、入力の情報がそのまま通過して出力されます。ここでG=Lにすると入力データが記憶されます。これは一種のレジスタの役割をしますが、データの通過を許すことからトランスペアレント(透過)ラッチと呼ばれます。

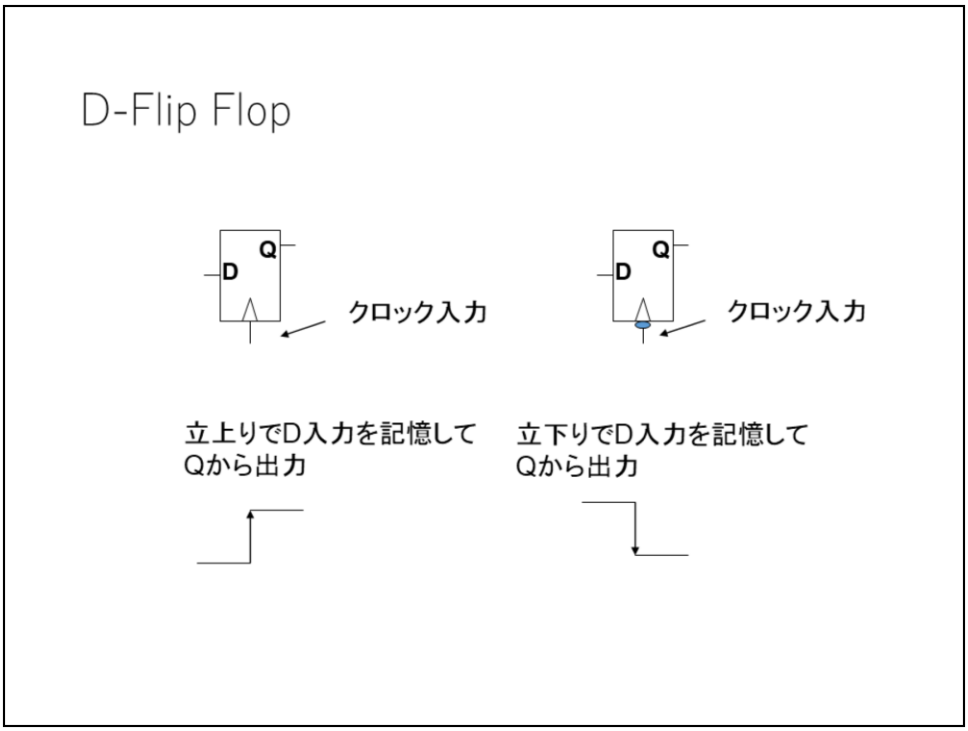


計算機基礎ではDラッチは、RSラッチの入力に切り替え回路を付けた構成で習ったと思います。論理的には全く同じですが、実際にLSIの内部では最初に述べた構造が使われます。

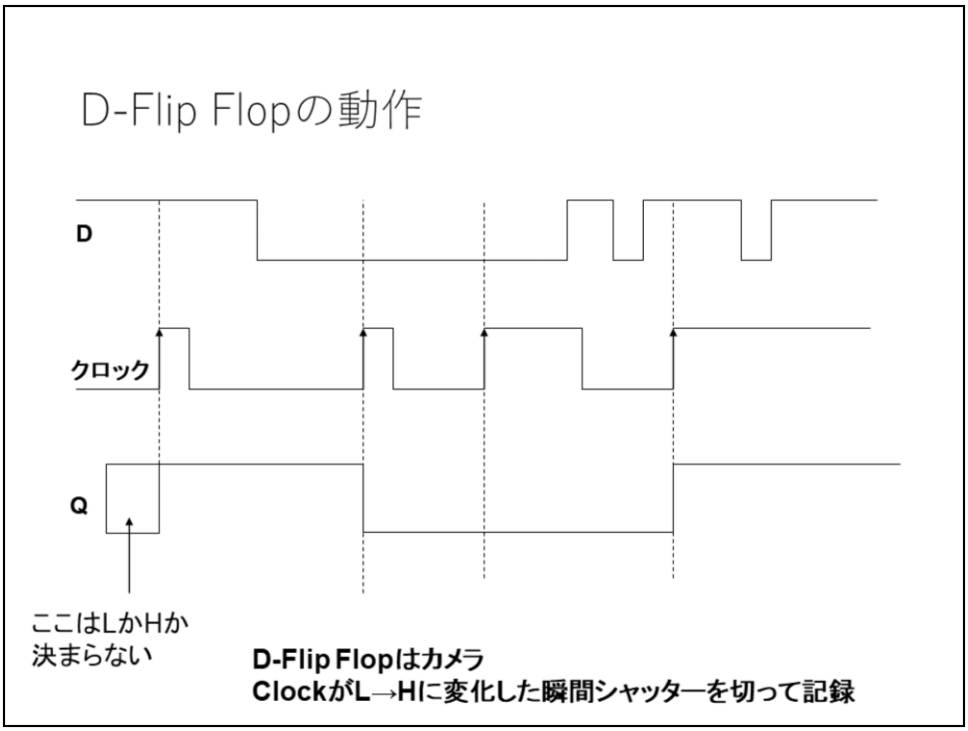
狭義のフリップフロップ（Flip Flop）

- Dラッチもレベル動作
  - 入力レベルによって記憶操作を行う
- フリップフロップはエッジ動作
  - クロック入力の変化に応じて記憶操作を行う
  - 状態遷移を行うためにはエッジ動作が必要
  - 記憶素子のほとんどはエッジ動作→同期回路
- 言葉の意味
  - 広義のフリップフロップ
    - 1ビットの記憶素子全般を指す
  - 狭義のフリップフロップ
    - エッジ動作のものに限定
    - レベル動作のものをラッチと呼んで区別する

今まで紹介したDラッチもレベル動作、すなわち入力信号のレベルによって記憶操作を行います。これに対して真のフリップフロップはクロック入力の変化に応じて記憶操作を行います。このことをエッジ動作と呼びます。なぜエッジ動作が必要なのでしょうか？大規模なディジタル回路をきちんと動作させるためには、クロックの変化（エッジ）に同期して状態を変えることが必要なのです。このため、世の中でよく使われるのは、ラッチではなくフリップフロップです。言葉の意味をはっきりさせておきましょう。広い意味でフリップフロップとは1ビットの記憶装置全般を指します。しかし、狭い意味ではエッジ動作をするものに限定され、レベル動作のものはラッチと呼んで区別します。

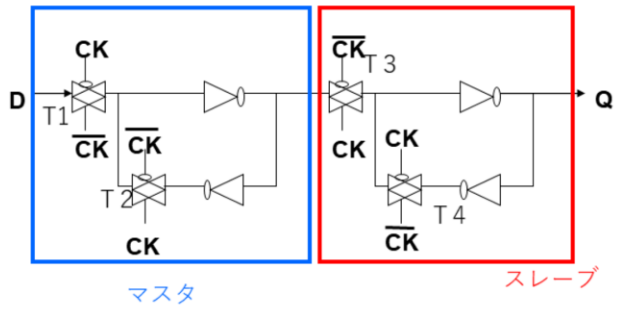


まず最も良く使われるD－Flip Flopを紹介しましょう。D－FlipFlopはD入力とクロック入力を持っています。クロックはちょっと変わった入力なので目立つために△印をつけて表します。ただの△印はクロックの立ち上がり（L→H）で動作することを示し、△印に〇が付いているとクロックの立下り（H→L)で動作することを示します。



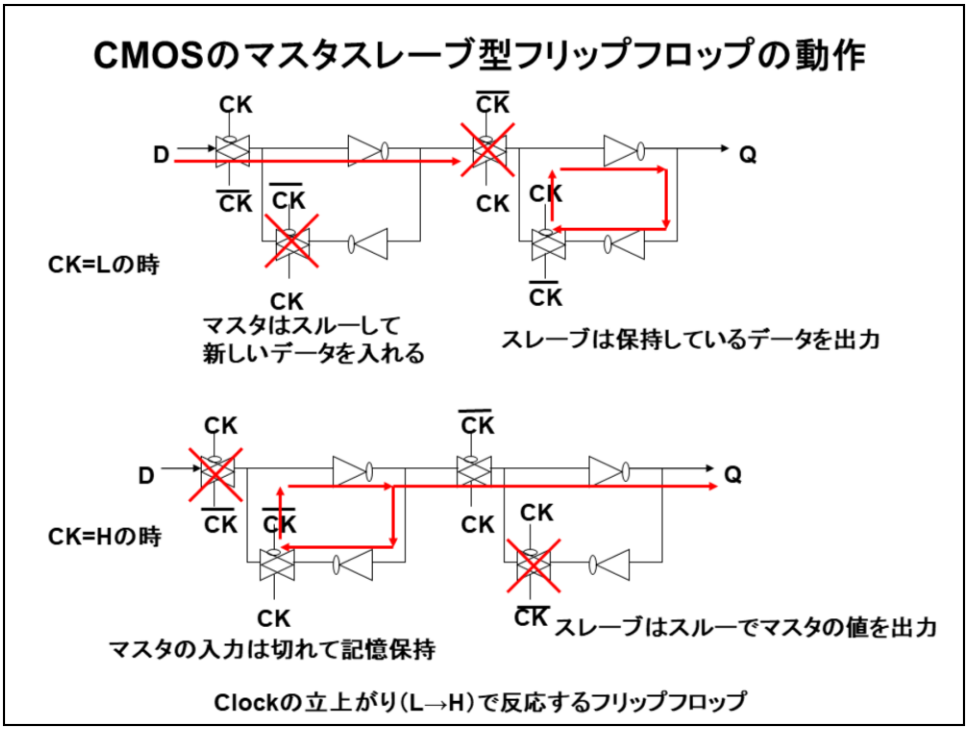
D－Flip Flopの動作をDラッチと同じ波形で示します。Dラッチが目ならばD-Flip Flopはカメラです。クロックがL→Hに変化した瞬間の写真を撮って記録します。このため、クロックの立ち上がり以外ではQは変化しません。

D-フリップフロップの構造：マスタスレーブ型



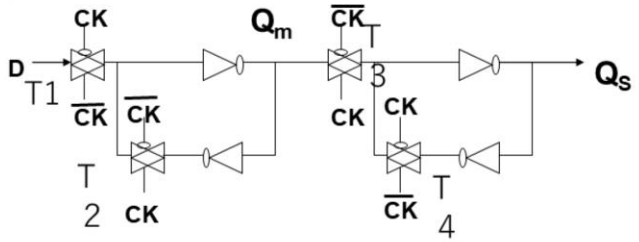
マスタ：通常のラッチとT1,T2の制御（CK,$\overline{CK}$)が逆
スレーブ：通常のラッチとQ、$\overline{Q}$が逆

Dフリップフロップには、マスタスレーブ型とエッジトリガ型があります。エッジトリガ型はCMOSが普及する以前は74シリーズに利用されましたが、最近は単純な構造でトランスミッションゲートを使って簡単に作られるマスタスレーブ型が利用されます。マスタスレーブ型は、2つのDラッチを接続して作ります。最初のDラッチをマスターと呼び、次のDラッチをスレーブと呼びます。マスタは先に紹介したラッチとT1,T2の制御が逆です。スレーブはラッチそのものですが、出力Qは、ラッチと反転したものを取り出します。ちなみに、この二つのラッチの動作はマスタ（主人）とスレーブ（奴隷）とは思えないのですが、歴史的にこのように呼ばれています。



マスタとスレーブは対称的な動作をします。クロックがLの時は、スレーブは筒抜けになり、マスタはNOTの8の字構造ができてここにデータを記憶します。クロックがHに切り替わると、スレーブが8の字構造を作ってデータを蓄え、マスタは筒抜けになります。ここで外から見た場合、Qの値が変化するのは、CKがLからHに変化し、スレーブで記憶されているデータが外に直接出てきた瞬間です。一方、CKがHからLに変化すると、データはスレーブからマスタに移動しますが、外側から見ると変化はありません。

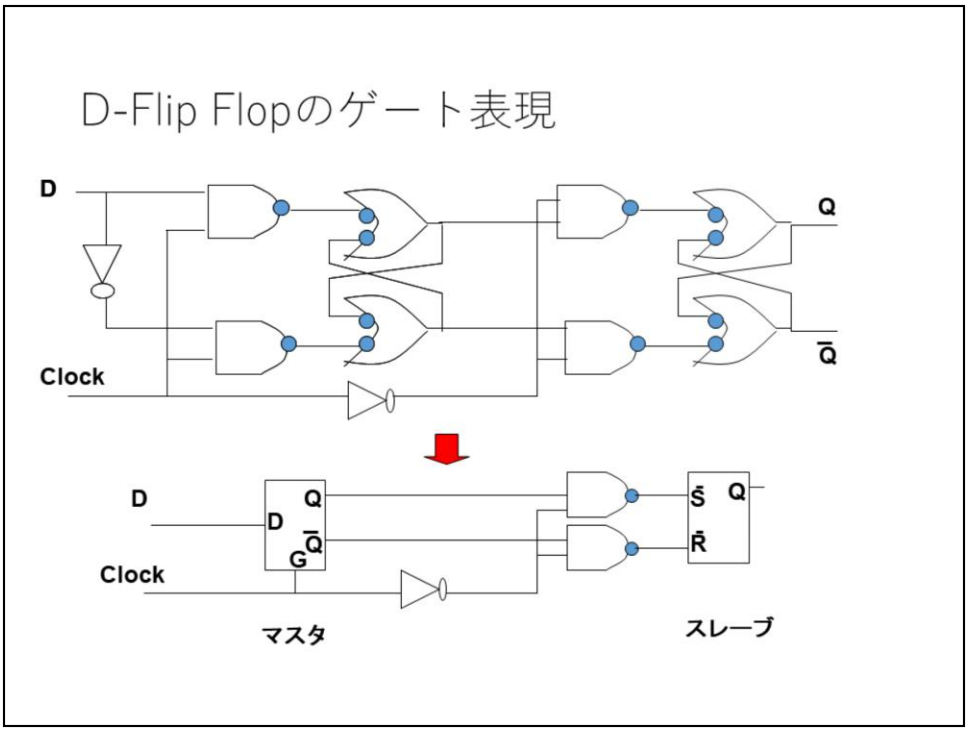
演習1



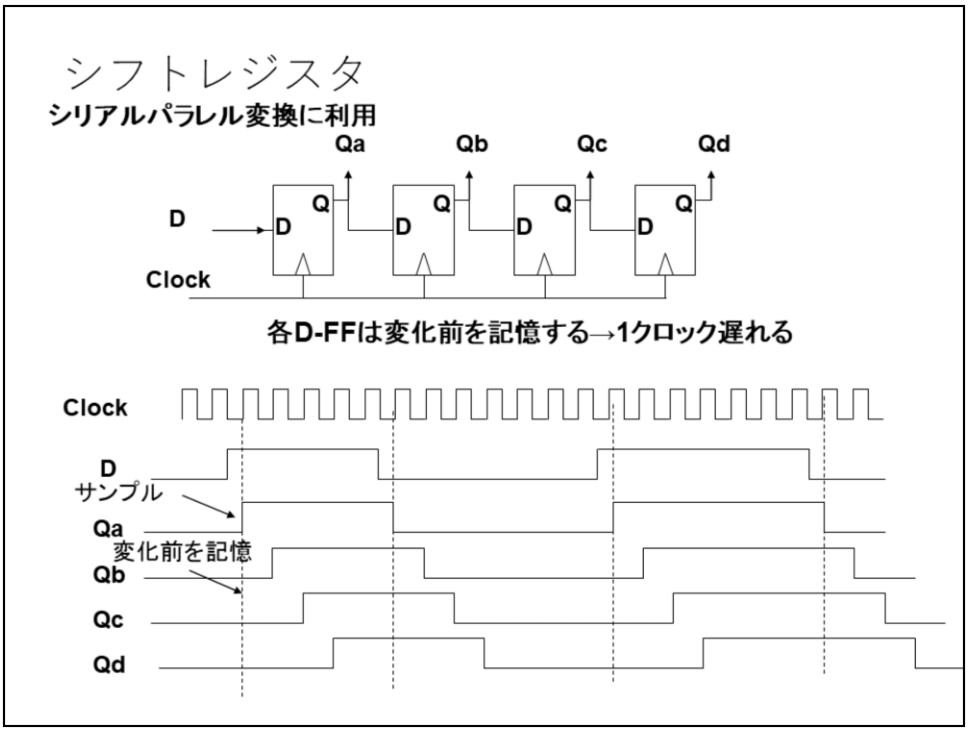
最初の状態でQs＝L、CK=H、D=Lであった。順に以下が変化した。
1．D=Hになった
2．CK=Lになった
3．CK=Hになった
それぞれの変化後のT1,-T4のON/OFF,Qm,QsのL/Hを表で示せ。

	T1	T2	Qm	T3	T4	Qs
1						
2						
3						

では内部動作の理解を深めるための演習をやってみましょう。

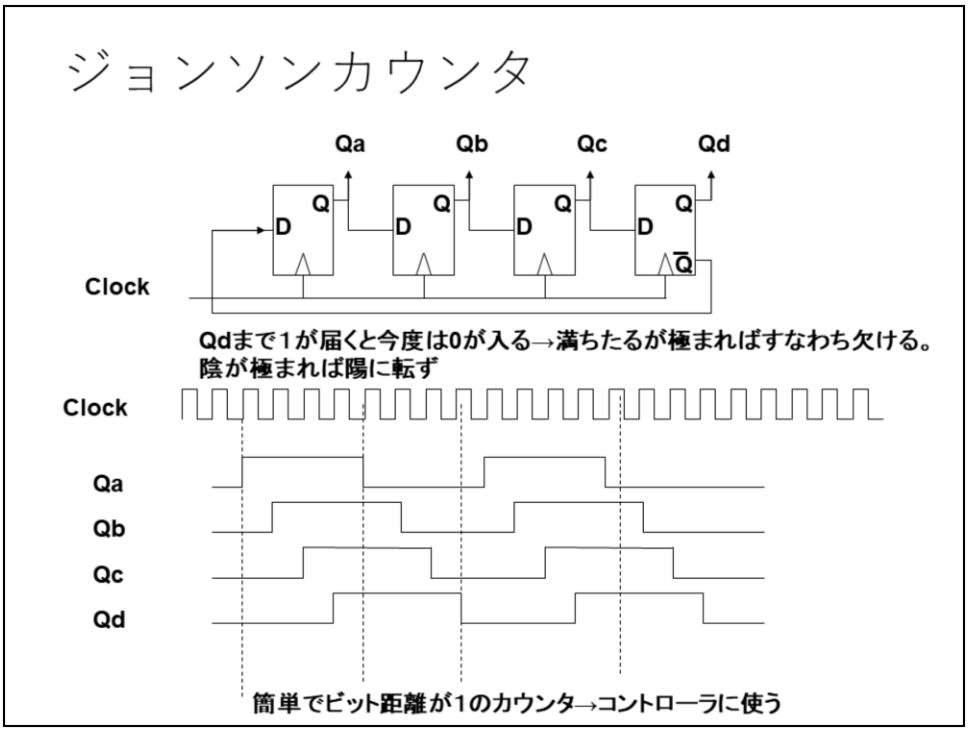


Dフリップフロップをゲートで表すと上記のようになります。こちらの方が分かりやすいかもしれません。

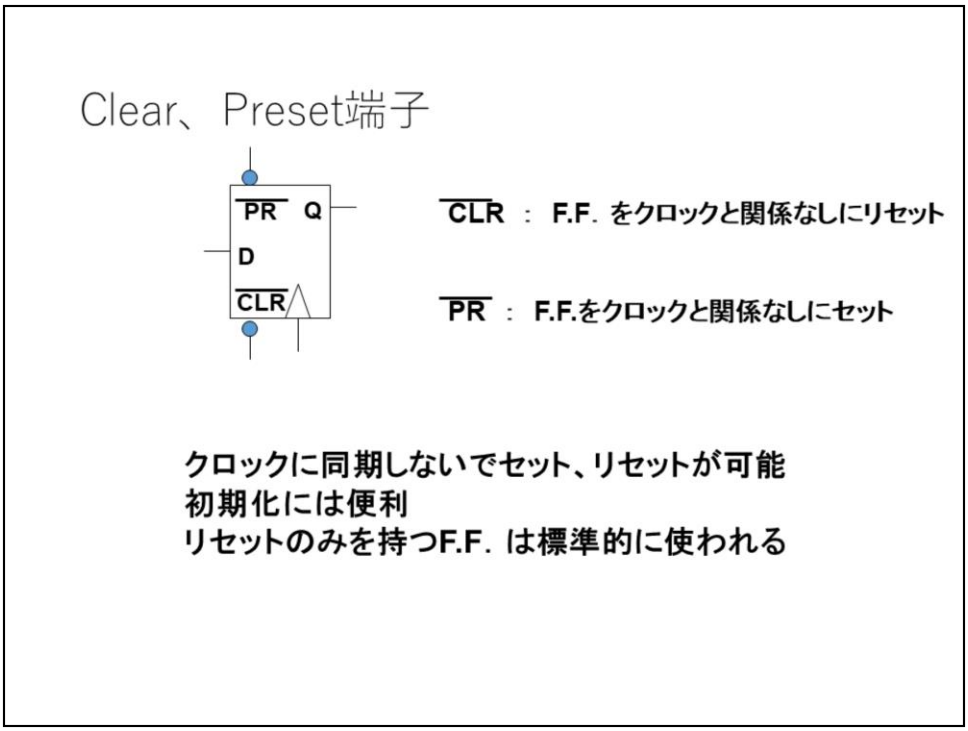


D-Flip FlopはD-ラッチ同様、レジスタとして使います。クロックを共通にして、入力データを記憶します。これがD-Flip Flopの主要な役割です。

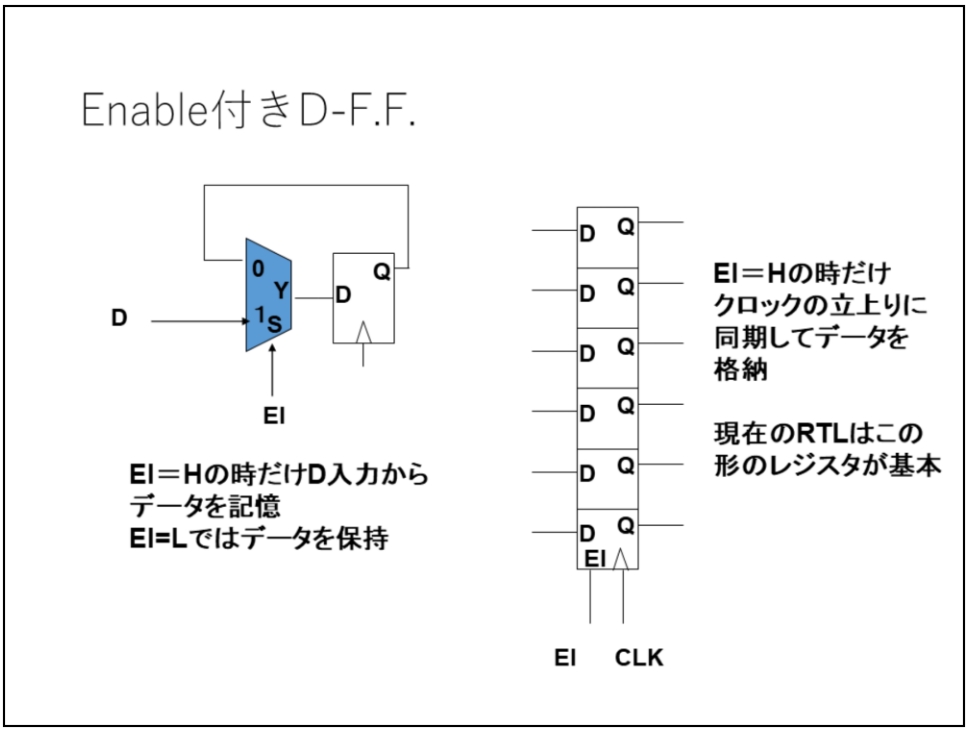
その他にもD－Flip Flopは数珠繋ぎにすると、データを1クロック遅らせる働きがあります。あるFlip Flopの出力はクロックの立ち上がりに同期して変化しますが、同じクロックの立ち上がりでその変化は保持できないため、変化は1クロック分遅れて伝わります。この働きを使うと、逐次的に入力したデータを並列に取り出すことができます。



この図は、ジョンソンカウンタと言ってシフトレジスタの応用です。Qdのみ反転出力が最初の入力になっていることに注目してください。全てがLの状態から始まると考えると、順番にHになっていき、QdまでHになると、今度はLになっていきます。満ちたるが極まればすなわち欠け、陰が極まれば陽に転ずというのがこのカウンタの動作です。コントローラなどに使います。



D-Flip Flopはクロックに同期しないと動作しません。これば場合によっては不便なので、クロックとは関わらず状態を切り替える端子を付ける場合があります。これがClear端子、Preset端子です。Clear端子は、Lにすると Flip Flopをリセットします。Preset端子は、LにするとFlip Flopをセットします。この端子は電源投入直後に初期化する時など、例外的な動作をさせるときに使います。

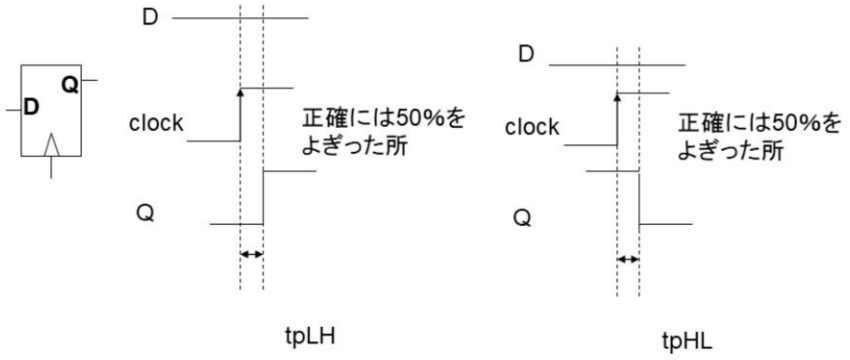


D-F.F.にはもう一つ問題があります。逆に、クロックの立ち上がりで常にデータを蓄えてしまうことです。本当に蓄えるべきときだけ蓄えるためには、入力にマルチプレクサを付けてやり、EI=Lの時は現在の値を蓄えさせ、EI=Hの時だけD入力がF.F.の入力に入るようにします。このマルチプレクサはトランスミッションゲートで簡単に実現できます。これがEnable付きF.F.です。EI=1の時のクロックの立ち上がりでデータを記憶します。現在のハードウェア記述言語によるRTL（Register Transfer Level)設計では、このEnable付きF.F.を主に使います。

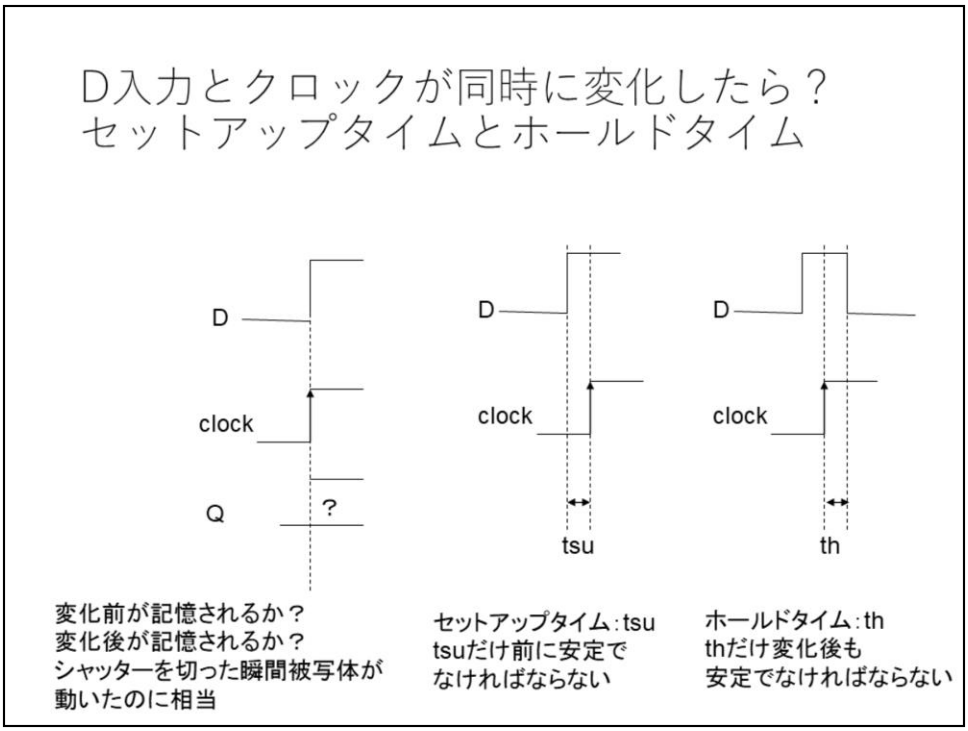
D-Flip Flopの動特性

静特性（スレッショルドレベル、ファンアウトなど）は通常のゲートと同じ

伝搬遅延時間tpdは、クロックの変化から測る →テキストp.115
この表はtpHL=tpLHとしてある



D-F.F.もゲートの組み合わせで作るので、ゲートと同じ静特性を持っています。つまりスレッショルドレベル、ファンアウトなどは通常のゲートと同じです。動特性のうち伝搬遅延時間もゲートと同じ考え方に基づいています。クロックがL→Hに変化してから（正確には50%のレベルに達してから）QがDを記憶した結果Hレベルに変化するまで（正確には50%のレベルに達するまで）の時間をtpLHと呼び、クロックがL→Hに変化してから（正確には50%のレベルに達してから）、QがDを記憶した結果Lレベルに変化するまで（正確には50%のレベルに達するまで）をtpHLと呼びます。先に紹介したようにD-F.F.はゲートを一定の段数使っているので、単純なNANDゲートなどに比べると伝搬遅延時間は大きくなります。



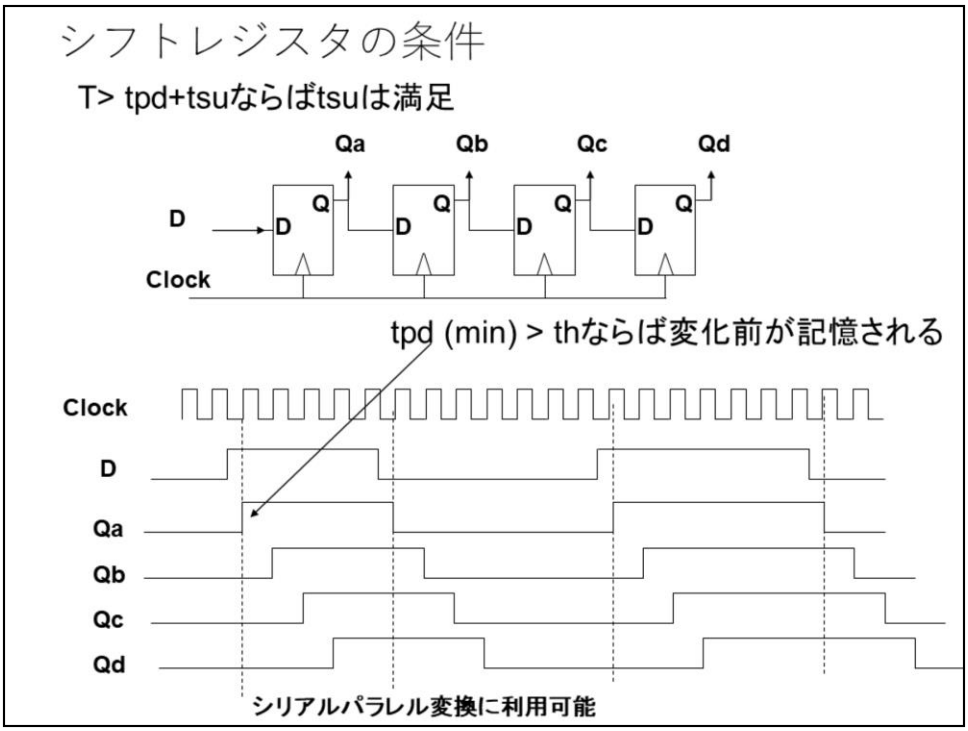
フリップフロップ特有の動特性は、記憶を確実に行わせるための条件です。クロックがL→Hに変化したのと全く同時にD入力が変化したらどうなるでしょう？結果としてQはLかHかどちらかの状態になる（メタステーブルといってLとHの中間レベルにしばらく留まる現象があり、ディジタル回路の動作の不安定性の原因になります）のですが、どちらになるかは保証されません。これはちょうどシャッターを切った瞬間に被写体が動いてしまうことに相当します。きちんとした写真を撮るため、つまりきちんとデータを記憶させるためには、クロックをL→Hに変化する際にD入力に安定してもらわなければなりません。クロックが変化する前に安定しなければならない最小時間をセットアップタイムtsuと呼び、クロックが変化した後に安定していなければならない最小時間をホールド時間thと呼びます。フリップフロップに確実にデータを蓄えるためには、入力データはクロック変化前にtsu、変化後にthの時間変化しないことが必要です。

フリップフロップの動特性
tsu, th, tpd

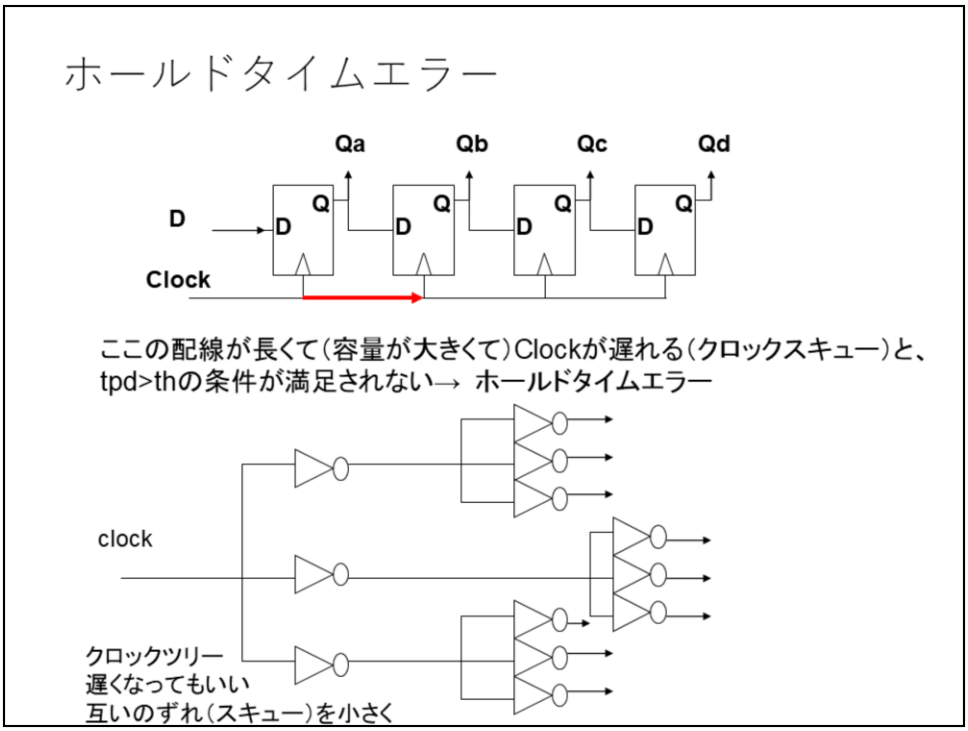
- 74AC74の5Vにおける動特性

伝搬遅延時間	tpd(tpHL, tpLH)	10.5nsec
セットアップ時間	tsu	3nsec
ホールド時間	th	0nsec

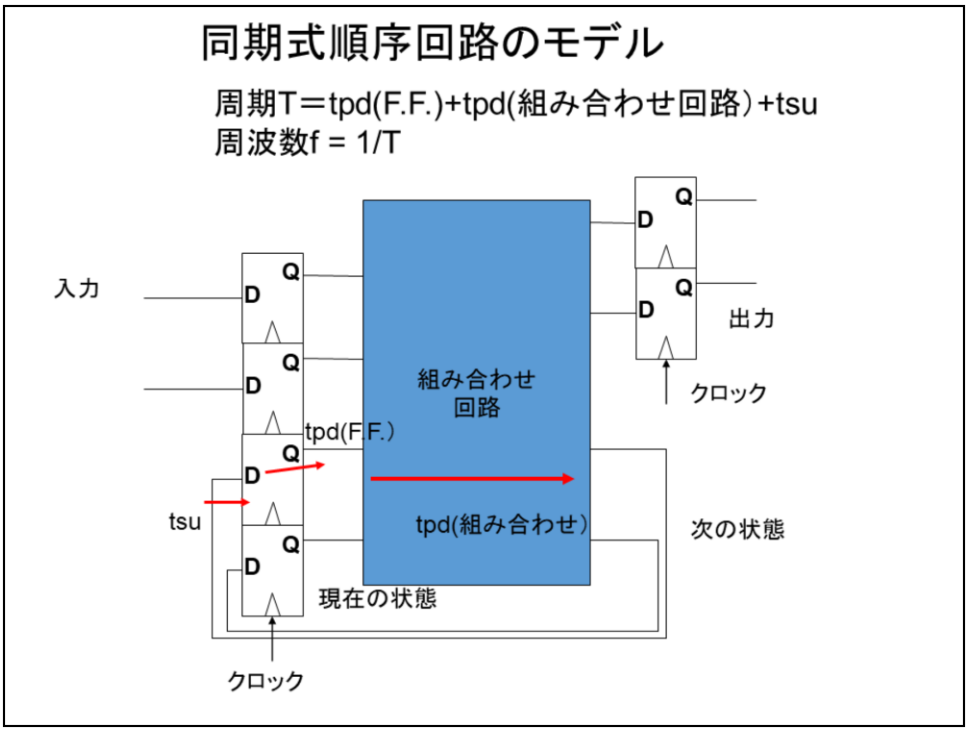
一例として74AC74の電源電圧5Vにおける動特性を示します。



では図のシフトレジスタが正しく動作するかを検討しましょう。最初のF.F.を除いたF.F.は、前段のF.F.の変化前の値を記憶します。tsuは、クロック周期T>tsu＋tpdならば満足します。あとはtpdの最小値＞thを満足していれば動作します。tpdは最大値しか書いていない場合が多いですが、規格表によるとth=0ですので、tpdが短くともこの条件は満足します。

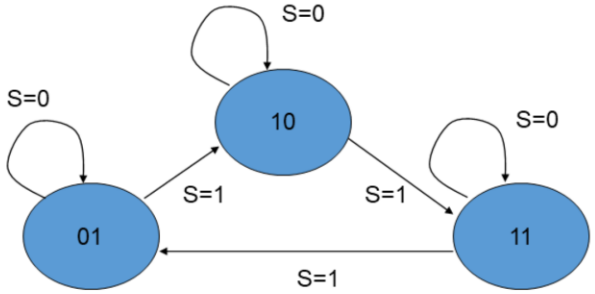


さて、ホールドタイムを満たす条件は多くの場合、th=0なので自動的に満足されます。しかし、図中の赤で示された配線距離が長く、容量負荷によって信号の伝搬が遅れると、D入力の変化の方が早く次のF.F.に届いてしまう可能性があります。これがホールドタイムエラーです。ホールドタイムエラーはそれぞれのフリップフロップにクロックが届く時間にずれがある（クロックスキューと呼びます）と生じます。クロックスキューをなくすため、クロックはツリー状にゲートを組み合わせて、末端に届く遅延時間が同じになるように工夫します。これをクロックツリーと呼びます。



シフトレジスタでは、T>tpd+tsuが満足すればデータを確実に蓄えることができましたが、一般の同期式順序回路では、これに組み合わせ回路の遅延が加わります。クロックが変化してから、フリップフロップの遅延tpd(F.F.)に次の状態を作ってやるための組み合わせ回路の遅延tpd(組み合わせ回路）が加わった遅延でフリップフロップに蓄えるべき次の状態が決まります。これが次のクロックが立ち上がるtsuだけ前に決まっていなければなりません。このようにT>tpd(F.F)+tpd(組み合わせ回路)+tsuが満足されれば、同期式順序回路は確実に動作します。

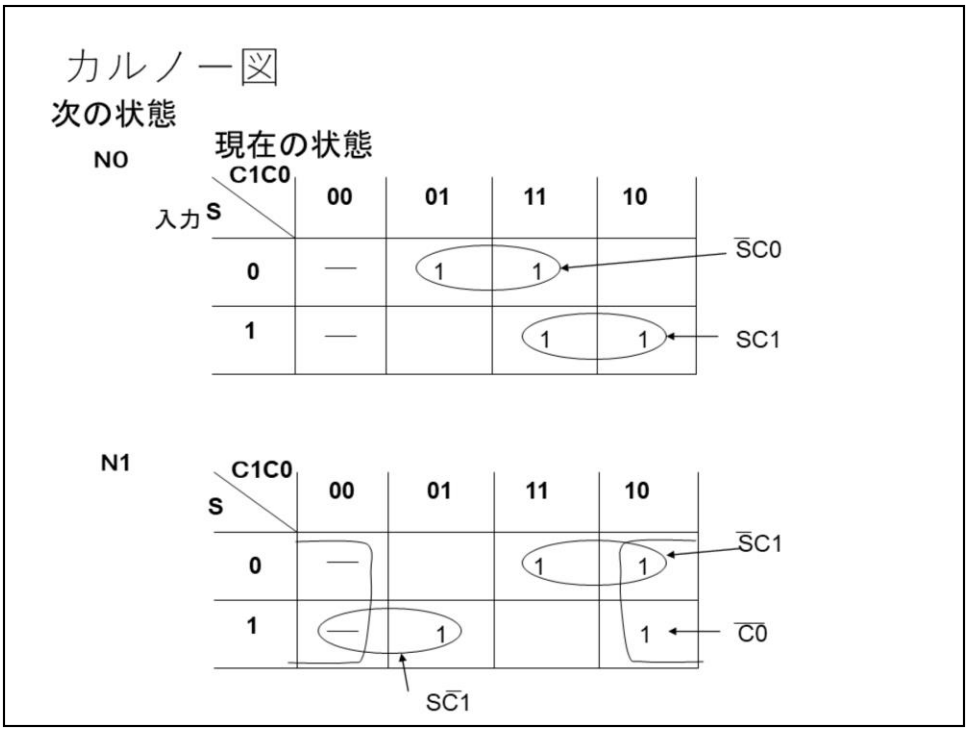
例題：S=Hのとき1→2→3とカウントし、
S=Lの時は停止するカウンタの設計
（テキストp.13)



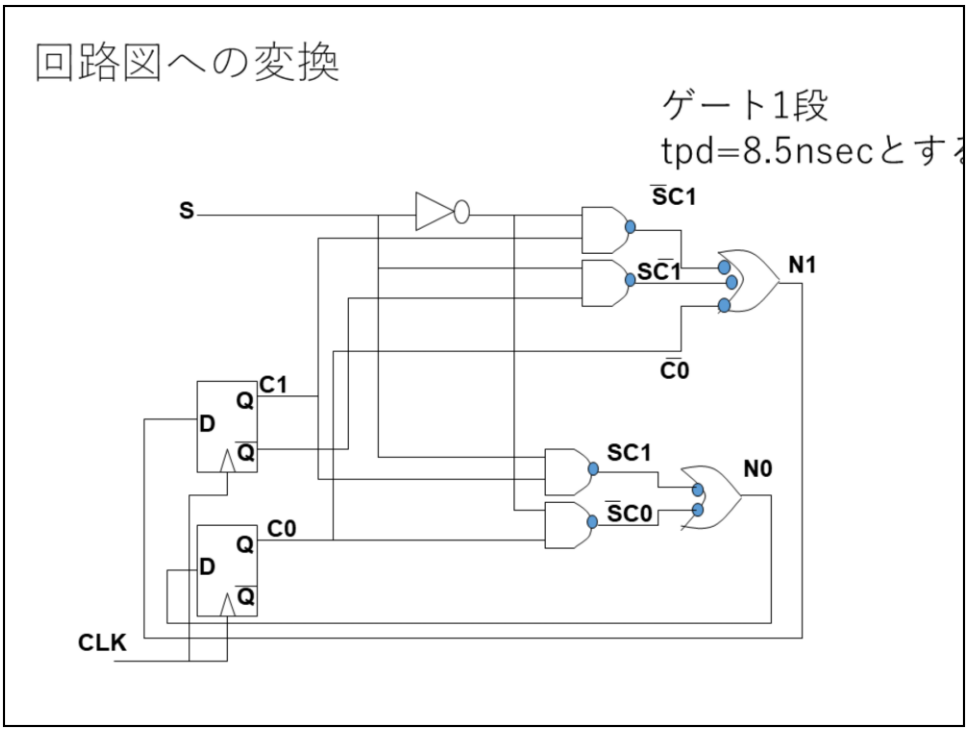
状態遷移図　状態番号＝出力とする

現在の状態C1C0　→　次の状態N1N0

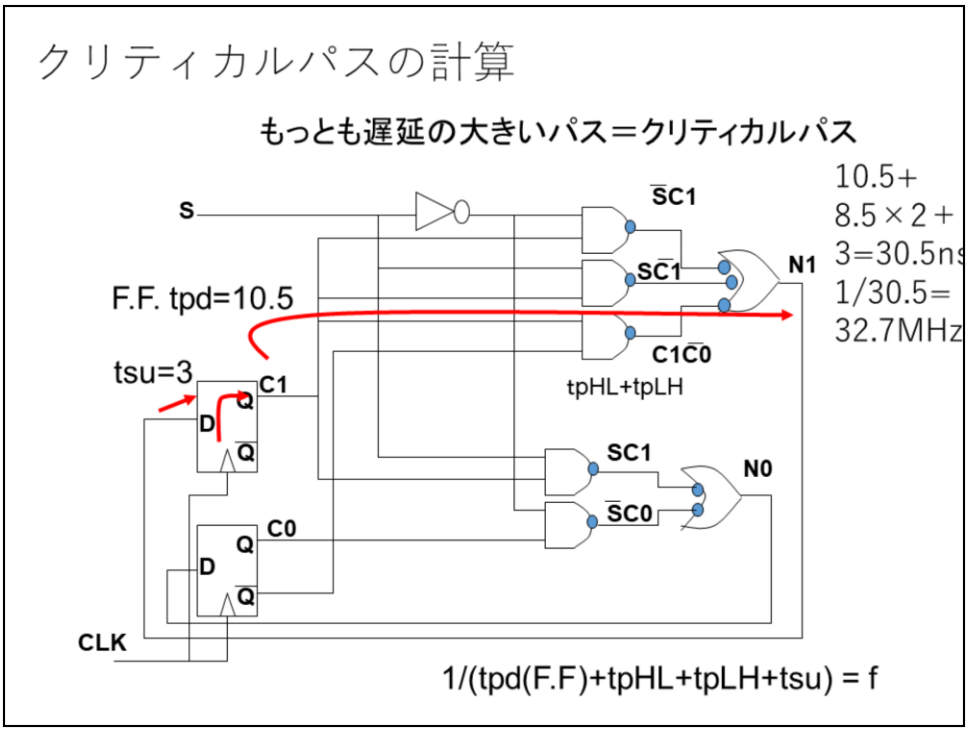
それでは、S=Hの時1→2→3とカウントし、S=Lの時には停止するカウンタの最大動作周波数を求めてみましょう。このカウンタの状態遷移図はここに示す3つの状態で表されます。それぞれの状態の番号がそのまま出力となるようにします。



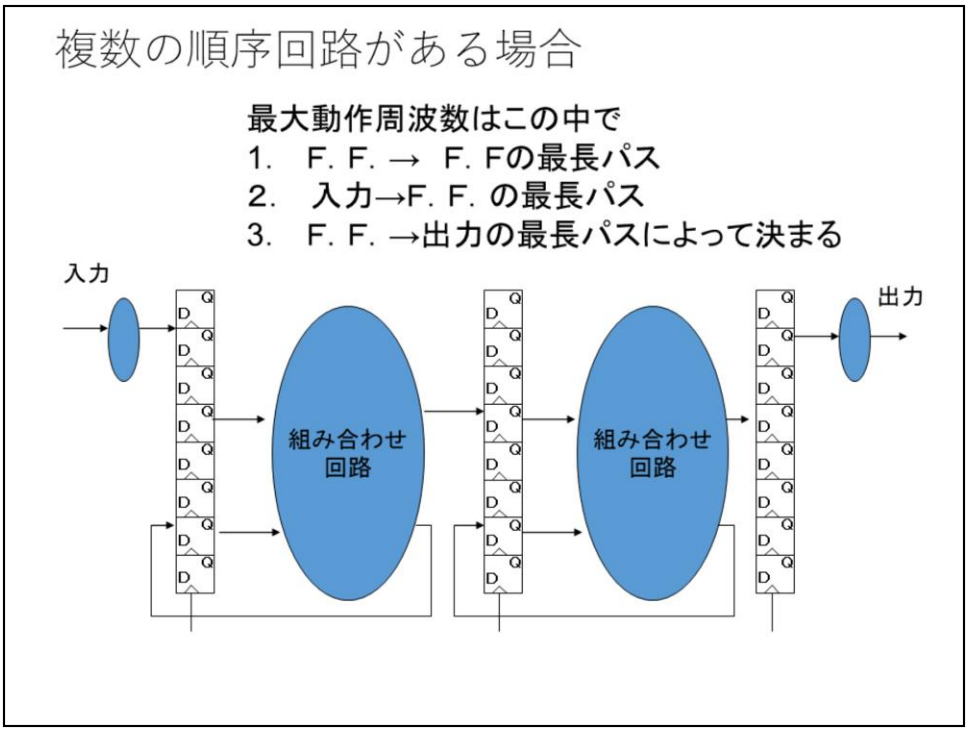
現在の状態をC1C0として、次の状態N1N0を決めてやります。N1,N0のカルノー図をそれぞれ示します。この図より、同期式順序回路を設計することができます。



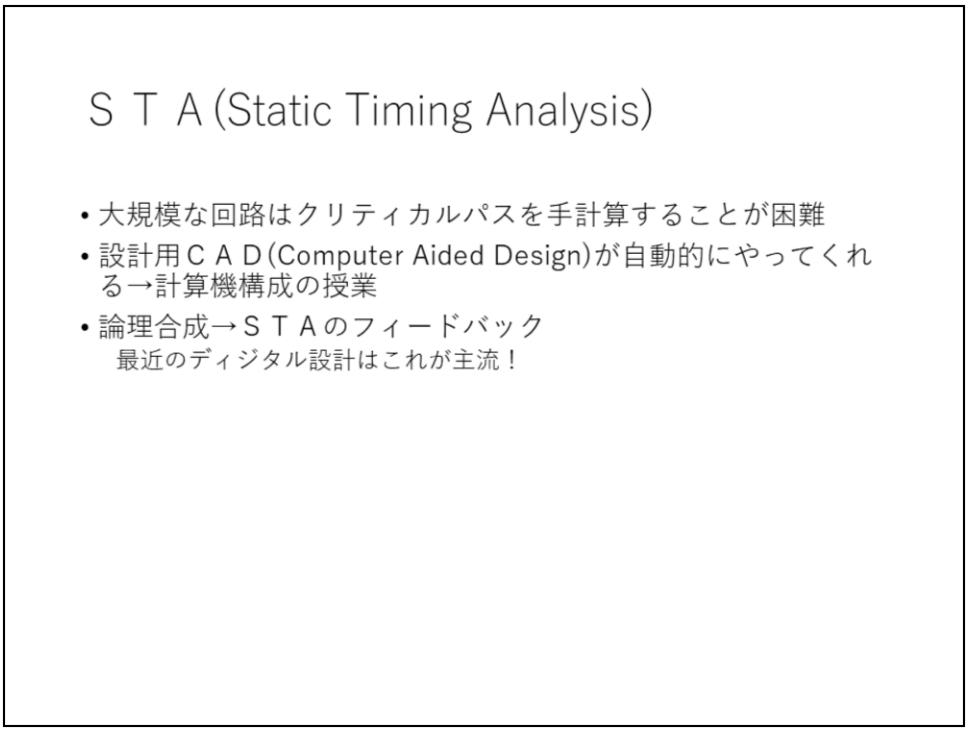
カルノー図に従って回路図を描いた結果です。今、それぞれのゲートの遅延時間をtpLH=tpHL=8.5nsecとしましょう。



F.F.に74AC74を用いるとすると、tpd=10.5nsec、tsu=3nsecです。回路の最大動作周波数はF.F.間の最も遅延時間の長いディジタル信号の通り道（パス）によって決まります。このパスのことをクリティカルパスと呼びます。ここではゲート1段の遅延を8.5nsecとしたので、クリティカルパスは10.5＋8.5×2＋3となります。この値は30.5nsecになります。逆数を取るとGHzでの周波数が出てくるのでこれをMHzに直すと32.7MHzになります。



先の回路は単一の順序回路だったため、F.F.から出たパスが同じF.F.に戻ってきましたが、一般的に大規模なディジタル回路は、複数のF.F.間に組み合わせ回路が存在し、それらの間に複雑なパスが構成されます。ただし、クロックは単一のものを用います。この場合、最大動作周波数は、F.F.間の最長パス、入力からF.F.までの最長パス、F.F.から出力までの最長パスのうちもっとも長いものの逆数を取って決めます。この解析は結構大変です。



通常、設計用CAD（Computer Aided Design)が自動的にこれをやってくれます。これをStatic Timing Anaylsis (STA)と呼びます。これは計算機構成の授業で演習します。現在のディジタル回路の設計はハードウェア記述言語で設計した結果を論理合成してSTAを行って最長パスを計算し、もしもこれが長すぎたら短くするように設計を変更したり、論理合成の条件を変更します。この段階が設計の質に最も影響を与えます。

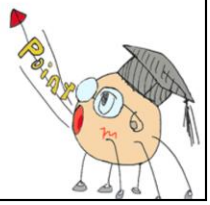
今日のポイント

フリップフロップは1ビットの記憶素子

- DラッチはG=Hの時は目を開き、G=Lで目を閉じ最後に見たものを覚えておく。目を開いている時は見たものはQに筒抜け
- Dフリップフロップはクロックの立ち上がり（立ち上がり）に同期してシャッターを切ってD入力を記憶してQに出力

クロックの立ち上がりのtsu以前とth以後までデータが安定していないとデータを正しく取り込めない

順序回路の動作周波数は、
1／（tpd(F.F)+tpd(組み合わせ回路)+tsu)

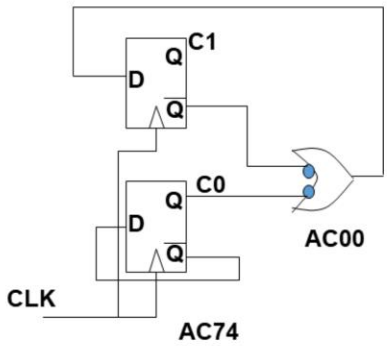


今日のポイントをインフォ丸が示します。

演習2

以下の回路の最大動作周波数を計算せよ。

D.FFのtpd=10.5ns
tsu = 3ns
ゲートのtpd=8.6ns



では演習をやってみましょう。例題を見ながらやればすぐできるはずです。単位を忘れないでください。