Unit-3

Introduction

→ In combinational circuits, the o/p is a function of present inputs.

→ In sequential circuits, the o/p depends on previous as well as current inputs.

→ FSM and pipelines are two examples of sequential circuits

→ Sequential circuits are designed with flipflops or latches, which are sometimes called memory elements that hold data called tokens.

→ The purpose of elements is not really memory; instead it is to enforce sequence, to distinguish the current token from the previous or next token. Therefore, we will call them sequencing elements. Without sequencing elements, the next token might catch up with the previous token, garbling both.

→ This chapter considers sequencing both static & dynamic circuits.

→ Static circuits refer to gates that have no clock i/p, such as complementary CMOS, pseudo nMOS, or pass transistor logic.

→ Dynamic circuits refer to gates that have a clock input especially domino logic.

→ Sequencing elements can be either static or dynamic circuits.

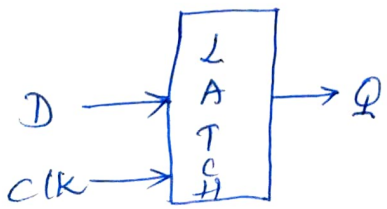
- A sequencing element with static storage employs some sort of feedback to retain its o/p value indefinitely.
- An element with dynamic storage generally maintains its value as charge on a capacitor that will leak away if not refreshed for a long period of time.
- A periodic clock is commonly used to indicate the timing of a sequence.

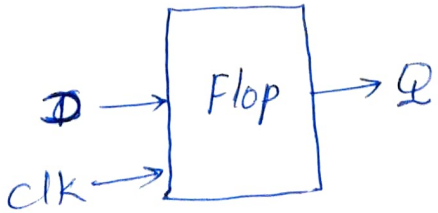
Sequencing Static Circuits

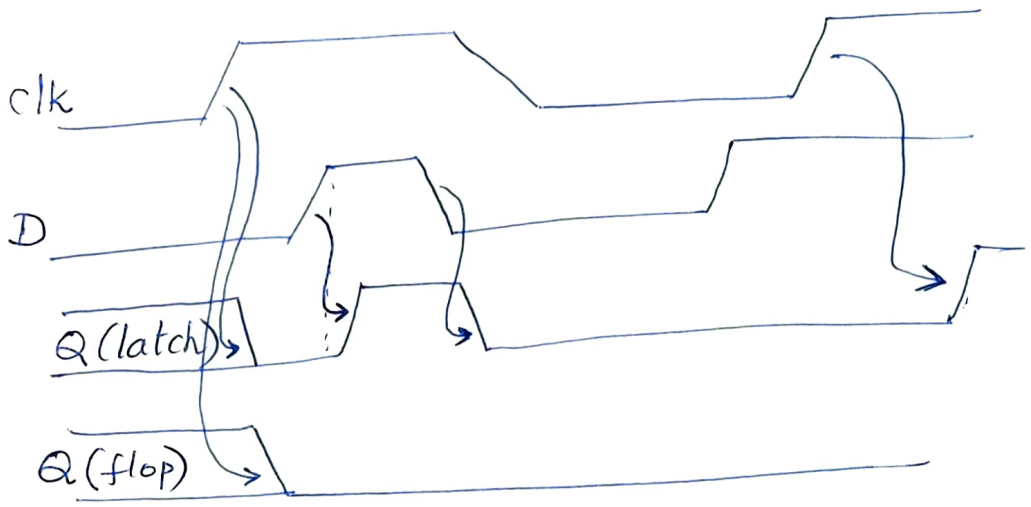
→ Latches & flip flops are the two most commonly used sequencing elements. Both are 3 terminals: D, clk, & Q.

→ The latch is transparent when the clock is high & opaque when the clock is low; In otherwords, when clock is high, D flows through to Q as if the latch were just a buffer, but when the clock is low, the latch holds its present Q o/p even if D changes.

→ The f/f is an edge triggered device that copies D to Q on the rising edge of the clock & ignores D' at all other times.







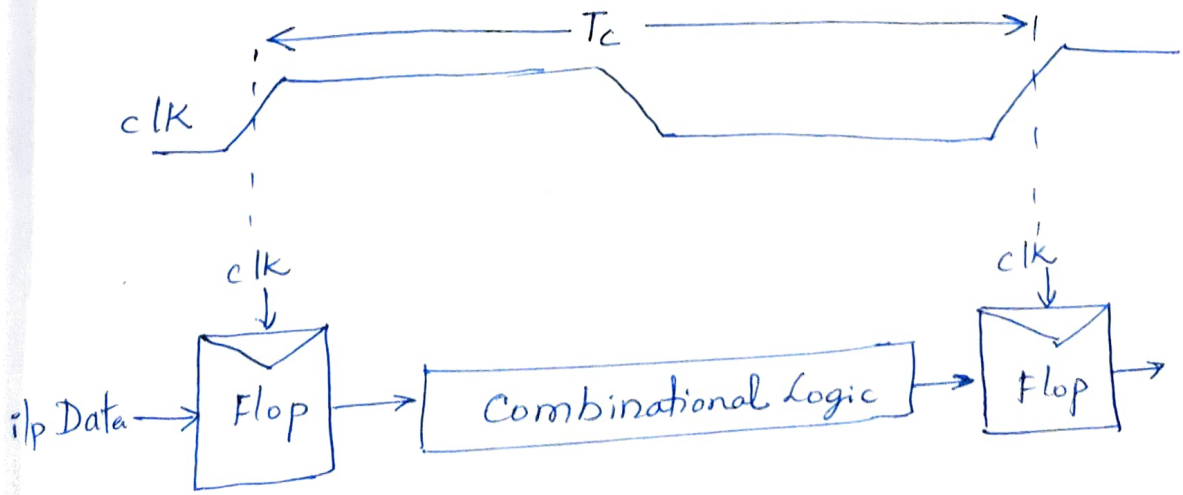
fig(1): Waveforms of latch & flop.

→ The three most widely used methods of sequencing static circuits ~~with~~ are

(a) flop-flops
(b) 2-phase transparent latches
(c) Pulsed latches.

Sequencing Methods

- Fig(2) illustrates three methods of sequencing blocks of Combinational logic. In each case, the clock waveforms, sequencing elements, & Combinational logic are shown.



fig(2) Flipflops.

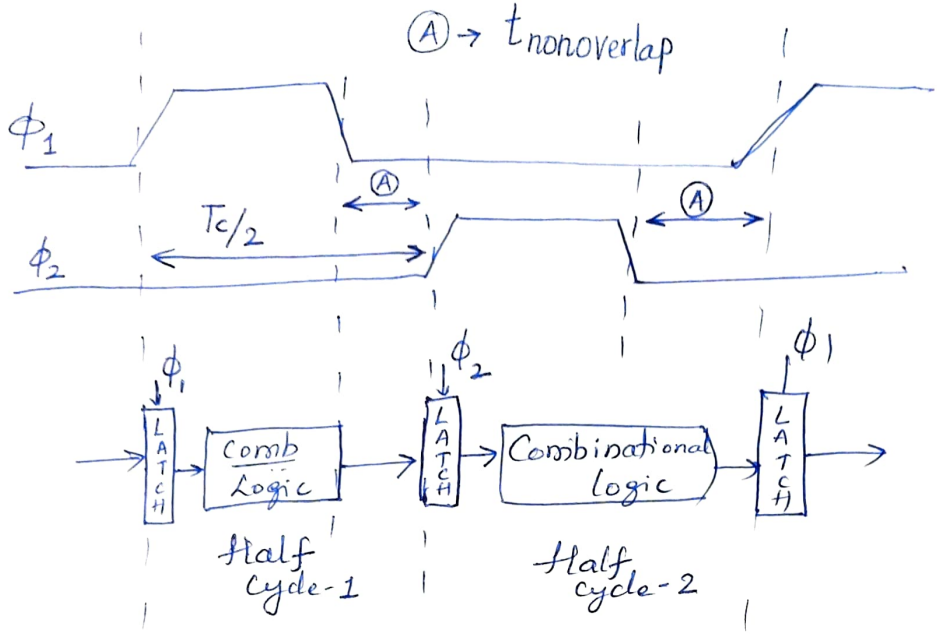


fig (2). 2-phase transparent Latches.

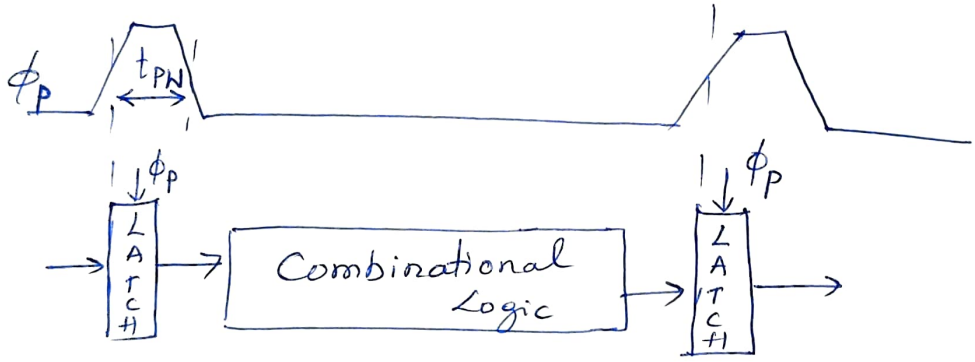


fig (2) Pulsed Latches.

- In the above fig, the horizontal axis corresponds to the time at which a token reaches a point in the circuit.

For ex, the token is captured in the first flip-flop on the first rising edge of the clock. It propages through the combinational logic & reaches the second flipflop on the rising edge of the clock.

- The clock period is T_c
- In a 2-phase system, the phases may be separated by $t_{nonoverlap}$.

- In a pulsed system, the pulse width is t_{pw}.

a) Flip-flop based system

This method uses one F/F on each cycle boundary. Tokens advance from one cycle to next on the rising edge. If a token arrives too early, it waits at the F/F until next clock cycle.

b) In 2-phase System

clock & its complement are represented as ϕ_1 and ϕ_2. At any given time, at least one clock is low & the corresponding latch is opaque, preventing one token from catching up with another.

c) Pulsed latch

In this, one of the latch is eliminated from each cycle. A brief pulse is applied to the remaining latch. If the pulse is shorter than the delay through the Combinational logic, we can still expect that a token will only advance through one clk cycle on each pulse.

Delay & timing Constraints: Combinational logic & Sequencing elements.

1. t_{pd} - Logic propagation delay
2. t_{cd} - Logic contamination delay
3. t_{pcq} - Latch/flop clock-to-Q propagation delay
4. t_{ccq} - Latch/flop clock-to-Q Contamination delay
5. t_{pdq} - Latch D-to-Q propagation delay.
6. t_{cdq} - Latch D-to-Q Contamination delay

7. t_{setup} - Latch/flop setup time

8. t_{hold} - Latch/flop hold time.

Fig.(3) illustrates these delays in a timing diagram. In a timing diagram, the horizontal axis indicates time & the vertical axis indicates logic level.

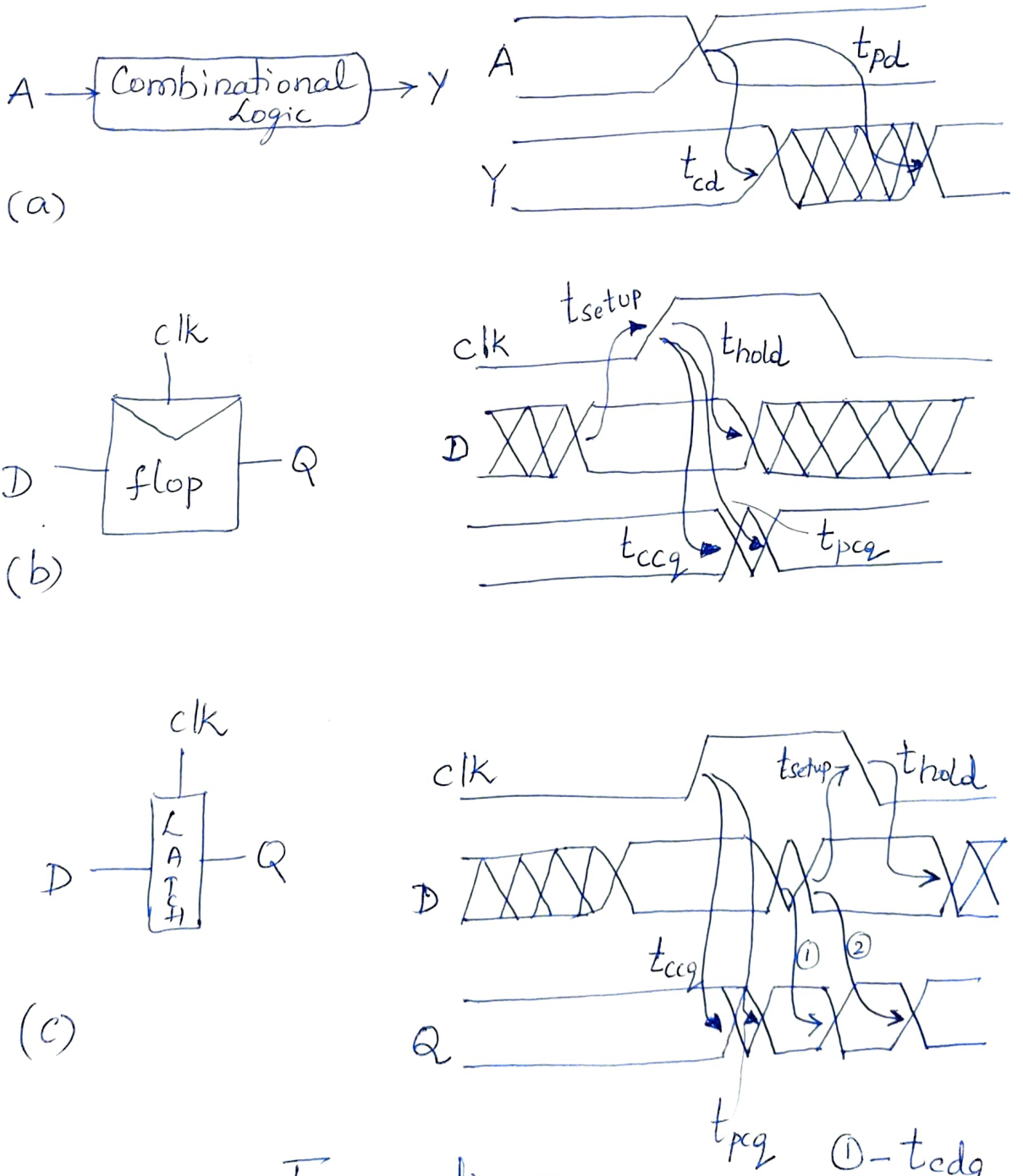


fig (3) Timing diagrams

- In fig.(3), Criss-Crossed line indicate that the signal might change at that time.
- A pair of lines with Cross-hatching indicates that the signal may change once or more over an interval of time.
- fig 3(a) shows the response of Combinational logic to the input A changing from one arbitrary value to another. The o/p Y Cannot change instantaneously. After the t_{cd}, Y may begin to change or glitch. After the t_{pd}, Y must have settled to a final value.
- The t_{cd} and t_{pd} may be very different because of multiple paths through the Combinational logic.
- fig. 3(b) shows the response of a flipflop. The data i/p must be stable for some window around the rising edge of the flop if it is to be reliably sampled.
- Specifically, the i/p D must have settled by some setup time t_{setup} before the rising edge of the clk & shouldnot change again until a hold time t_{hold} after the clock edge.
- The o/p begins to change after a clock-to-Q Contamination delay t_{ccq} & Completely settles after a clock-to-Q propagation delay t_{pcq}.

- Fig 3(c) shows the response of a latch. Now the i/p D must set up & hold around the falling edge that defines the end of the sampling period. The o/p initially changes t_{ccq} after the latch becomes transparent on the rising edge of the clock & settles by t_{pcq}. While the latch is transparent, the o/p will continue to track the i/p after some D-to-Q delay t_{dq} and t_{pdq}.

Max-Delay Constraints

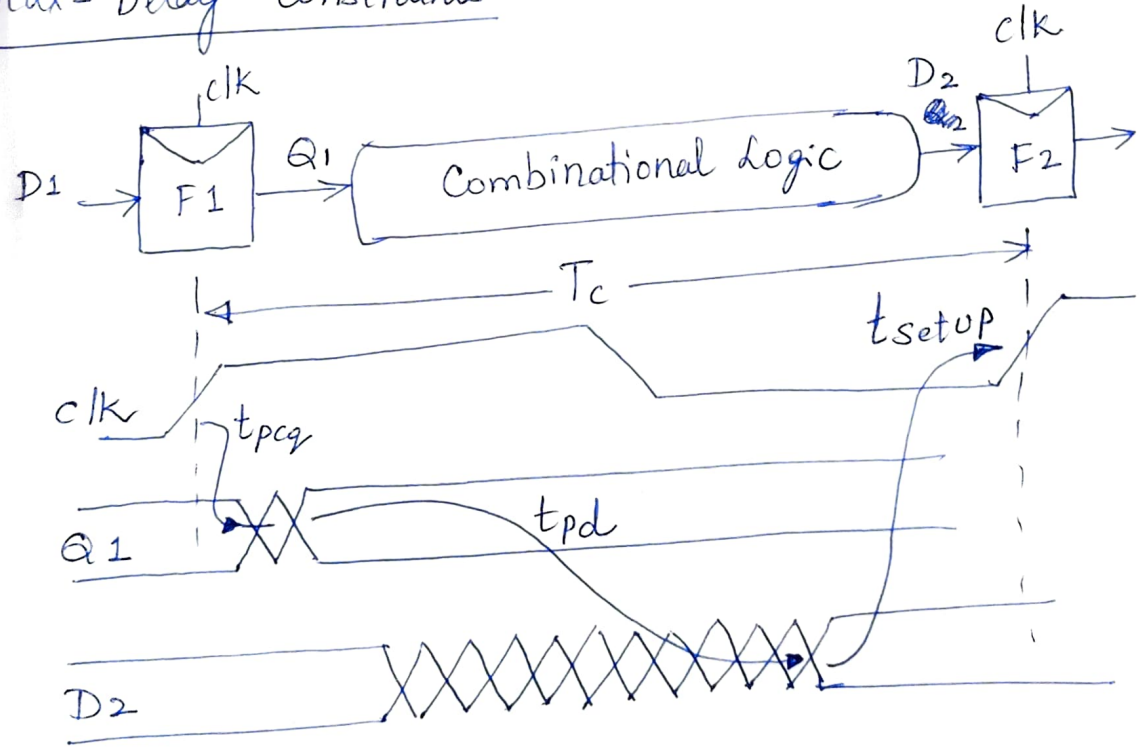


fig 4 Flip-flop max-delay constraint

- fig (4) shows the max-delay timing constraints on a path from one F/F to the next, assuming ideal clocks with no skew.

D2

- The Path begins with the rising edge of the clock triggering F1. The data must propagate to the o/p of the F/F Q_1 and thro the Combinational logic to D_2, setting up at F2 before the next rising-clk edge. This implies that the clock period must be atleast

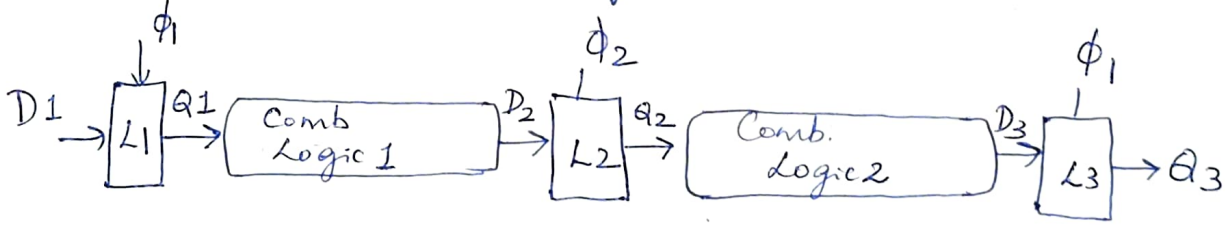
$$T_c \geq t_{pcq} + t_{pd} + t_{setup} \quad \text{—(1)}$$

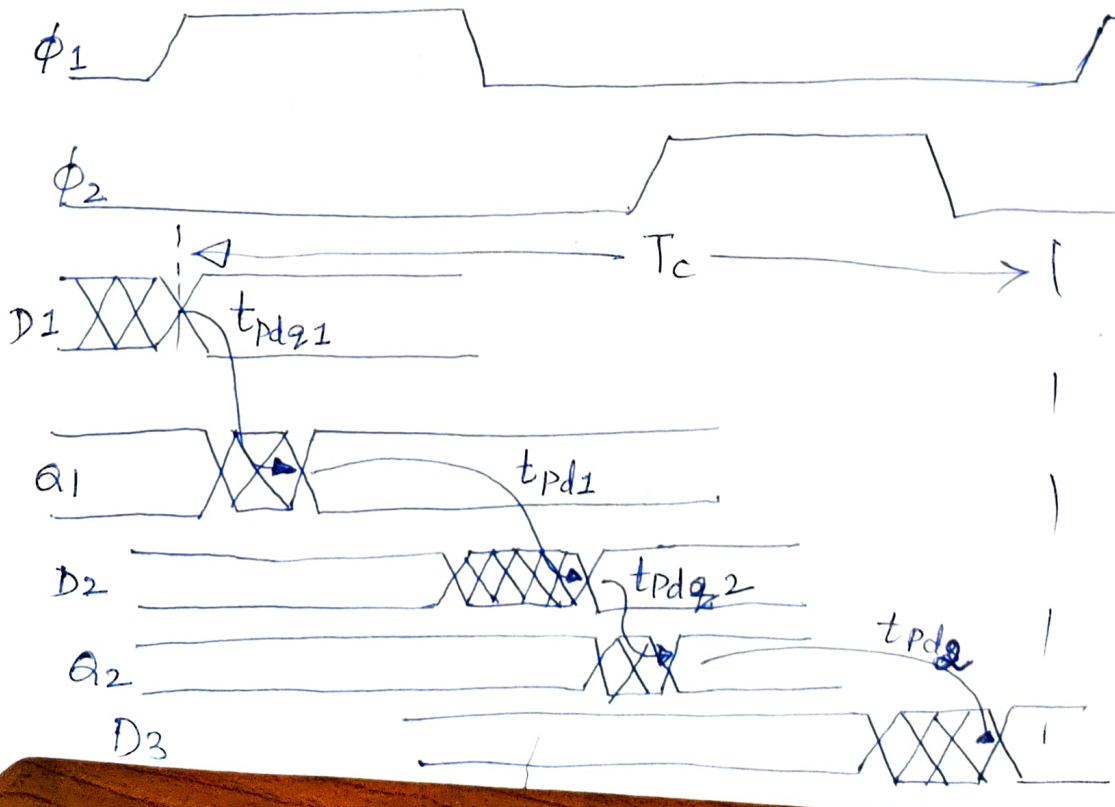
Alternatively, [rearranging (1)]

$$t_{pd} \leq T_c - (t_{setup} + t_{pcq}) \quad \text{—(2)}$$

Sequencing overhead

2-phase latch max-delay Constraint





$$T_c \geq t_{pdq_1} + t_{pd_1} + t_{pdq_2} + t_{pdq} \quad \text{— (A)}$$

Rearranging (A),

$$t_{pd} = t_{pd_1} + t_{pd_2} \leq T_c - \underbrace{(2\, t_{pdq})}_{\text{Sequencing overhead}} \quad \text{— (B)}$$

Pulsed latch max-delay constraint

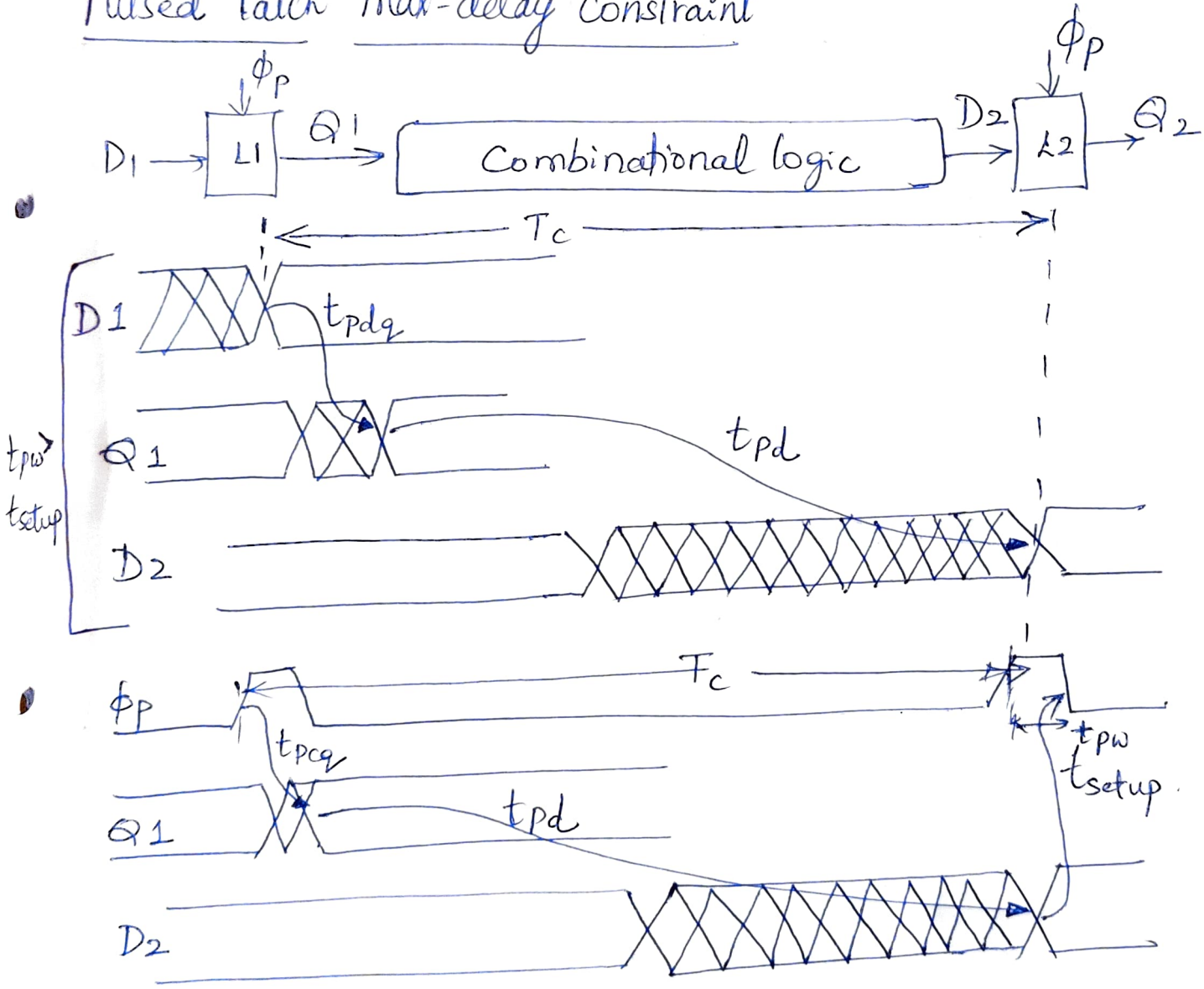


fig. Pulsed Latch max-delay constraint.

$$T_c \geq \max\left(t_{pdq} + t_{pd},\ t_{pcq} + t_{pd} + t_{setup} - t_{pw}\right)$$

$$t_{pd} \leq T_c - \max\left(t_{pdq},\ t_{pcq} + t_{setup} - t_{pw}\right)$$

Min-Delay constraints

Ideally, sequencing elements can be placed back-to-back without intervening combinational logic & still function correctly.

For ex, a pipeline can use back-to-back registers to sequence along an instruction opcode without modifying it. However, if the hold time is large & the contamination delay is ~~large~~ small, data can incorrectly propagate through two successive elements on one clock edge, corrupting the state of the system. This is called a race condition, hold-time failure or min-delay failure. It can only be fixed by redesigning the logic, not by slowing the clock.

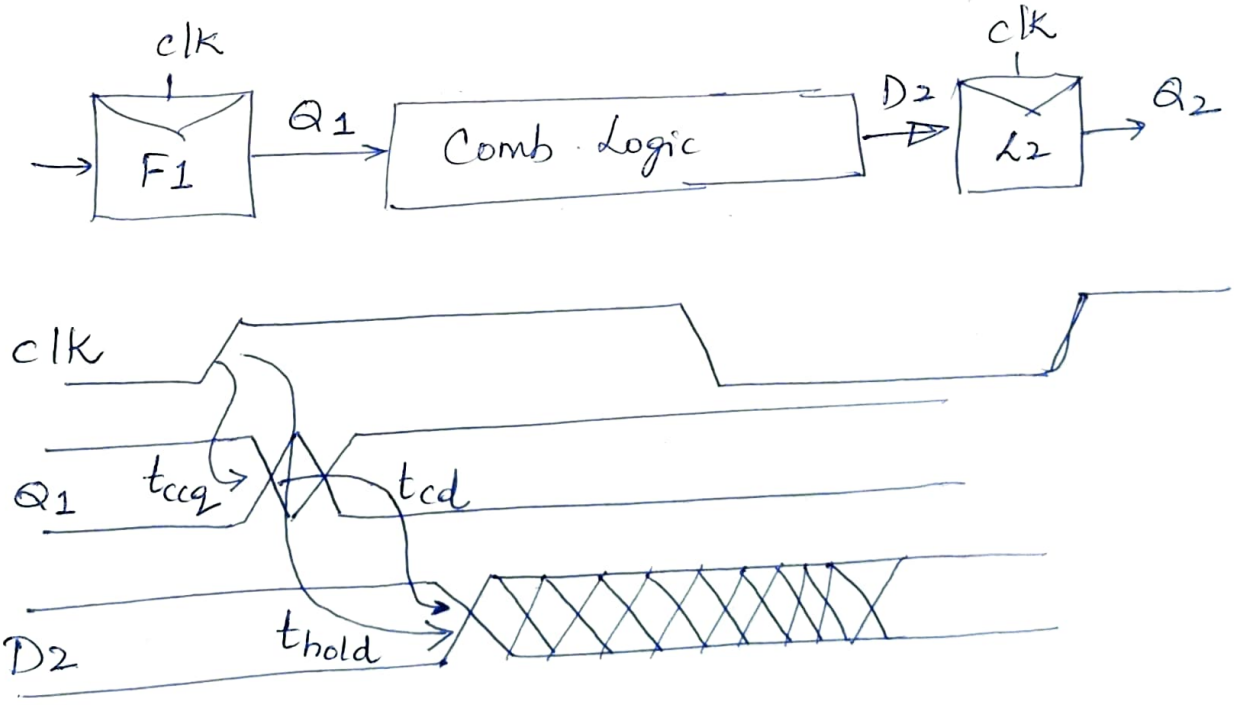
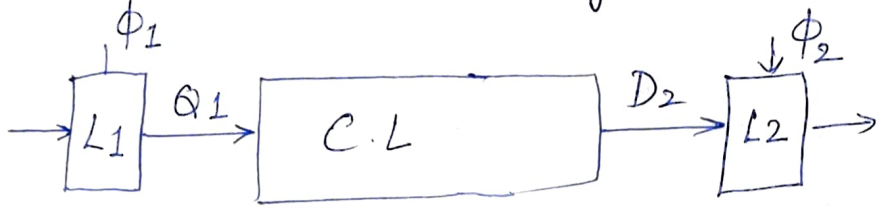
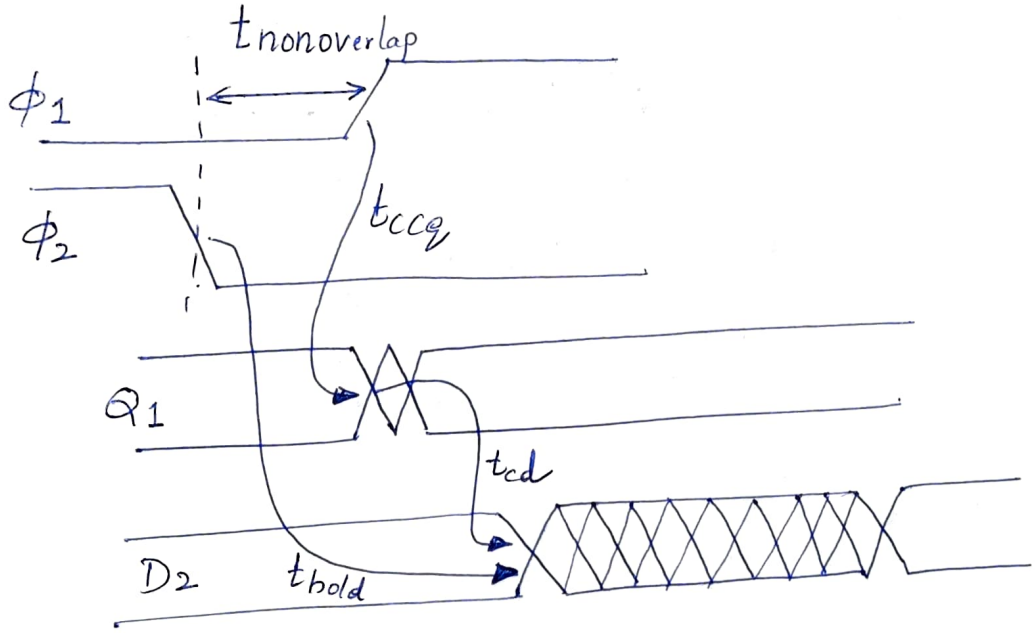


fig. F/F latch min-delay constraint.

$$t_{cd} \geq t_{hold} - t_{ccq}$$

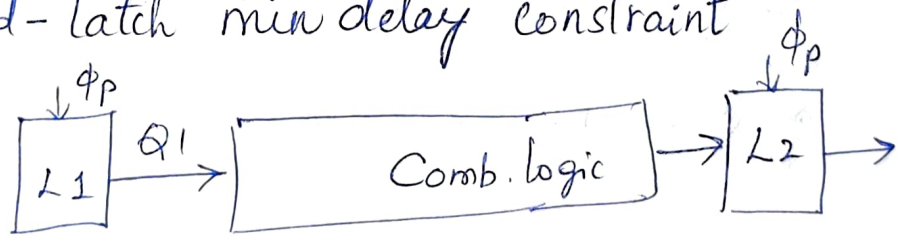
(2) 2-phase Latch min-delay constraint

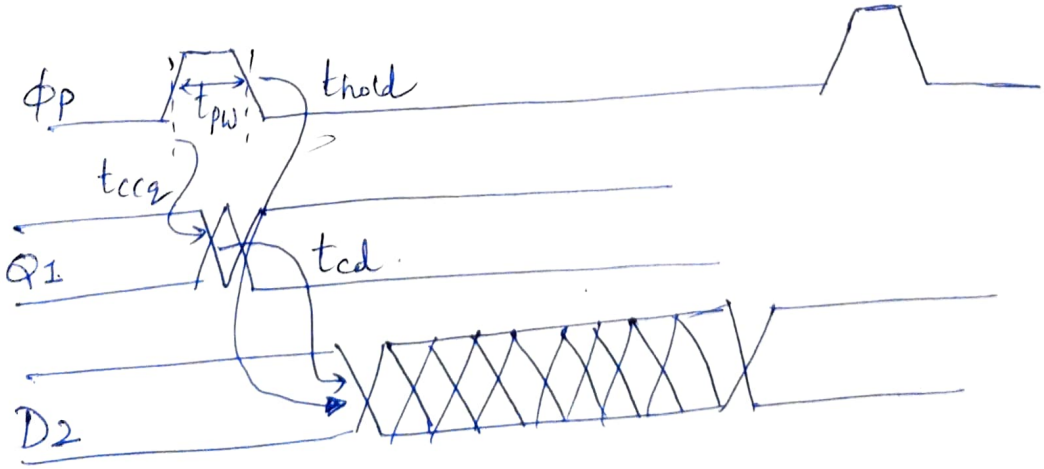




$$t_{cd1},\ t_{cd2} \geq t_{hold} - t_{ccq} - t_{nonoverlap}$$

(3) Pulsed-latch min delay constraint





$$t_{cd} \geq t_{hold} - t_{ccq} + t_{pw}.$$

CMOS Latches

(a)

This shows a simple latch built from a single transistor. It is compact & fast but suffers 4 limitations.

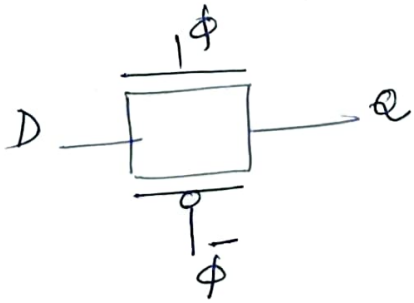
(a) o/p doesnot swing from rail-to-rail; it never rises above $V_{DD} - V_t$.

(b) The o/p is dynamic; o/p floats when latch is opaque.

(c) If it floats long enough, it can be disturbed by leakage.

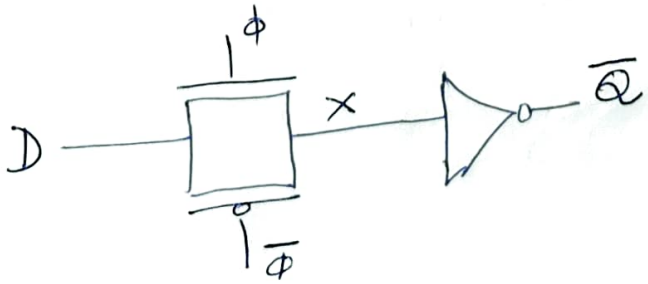
(d) 'D' drives the diffusion i/p of a pass transistor directly, leading to potential noise issues & making the delay harder to model with static timing analyzers.

(b)



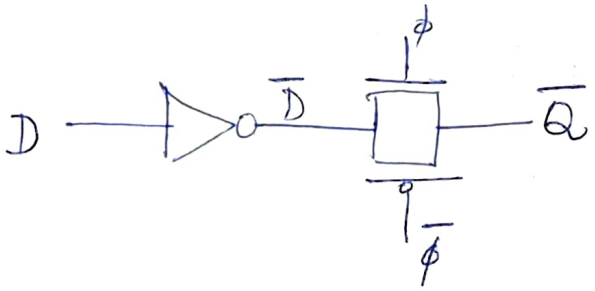
This topology uses a TG. TG offers a rail-to-rail o/p swing. It requires an additional clk ($\bar{\Phi}$) input.

(c)



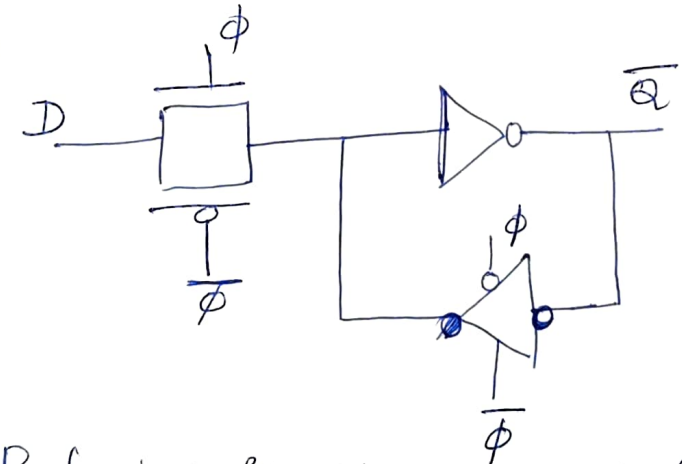
In fig.(c) adds an o/p inverter so that the state node X is isolated from noise on the o/p. of course, this creates an inverting latch.

(d)



fig(d) behaves as an inverting latch with a buffered i/p but unbuffered o/p. The inverter followed by a TG is essentially equivalent to a tristate inverter.

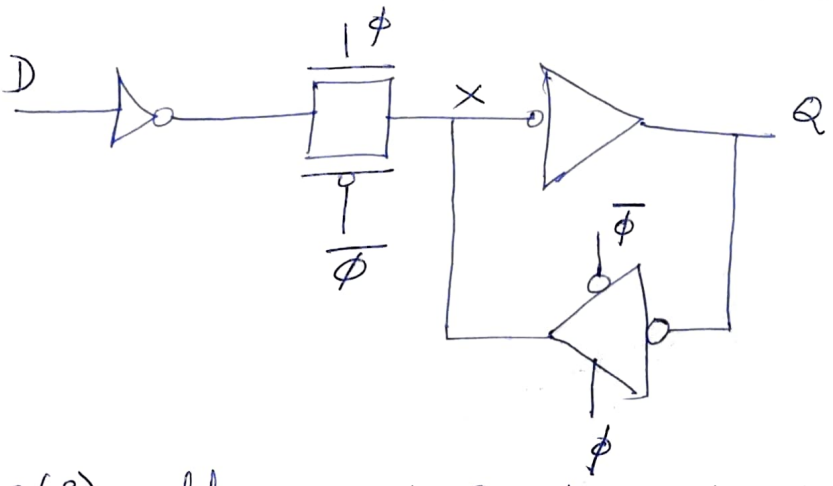
(e)



Refering fig.(e), when clk(ϕ)=1, i/p TG is ON, the feedback Tristate is off, & the latch is transparent.

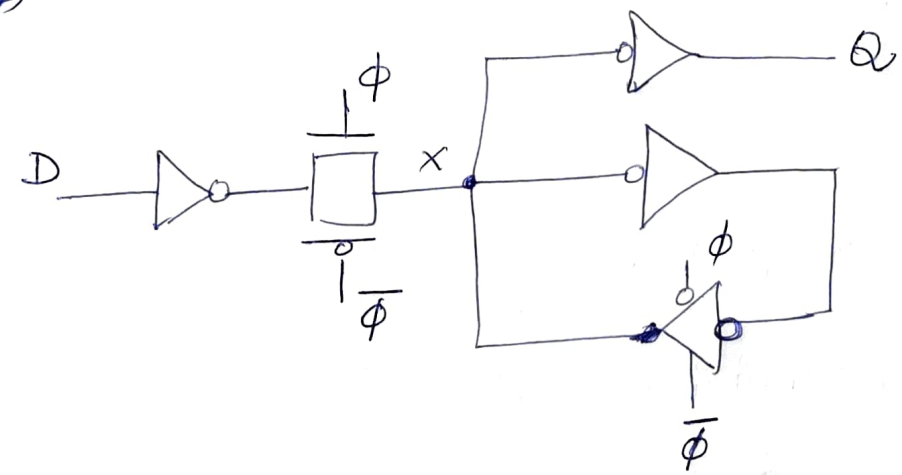
When clk(ϕ)=0, i/p TG is OFF, however the f/b tristate turns ON, holding X at the correct level.

(f)

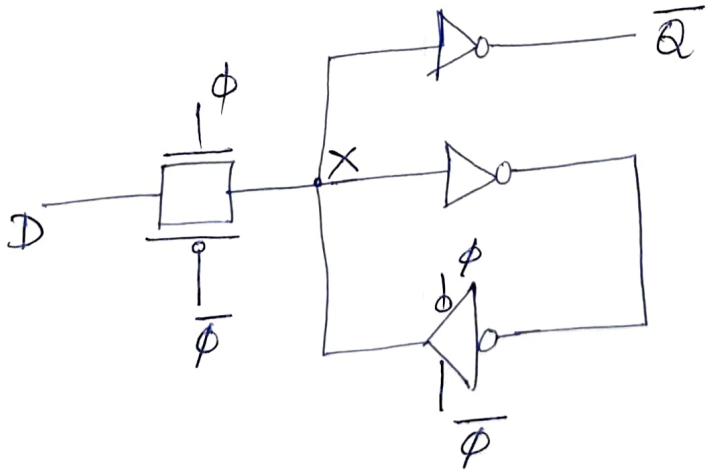


fig(f) adds an i/p inverter so the i/p is a transistor gate rather than unbuffered diffusion. (e) & (f) are o/p noise sensitive circuits.

(g)



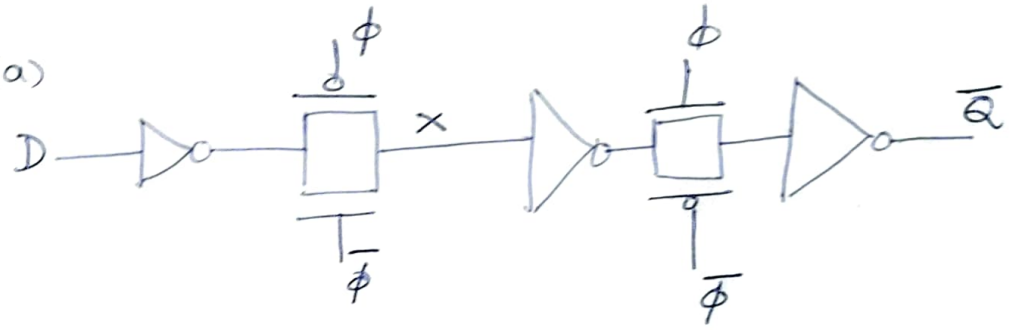
(h)



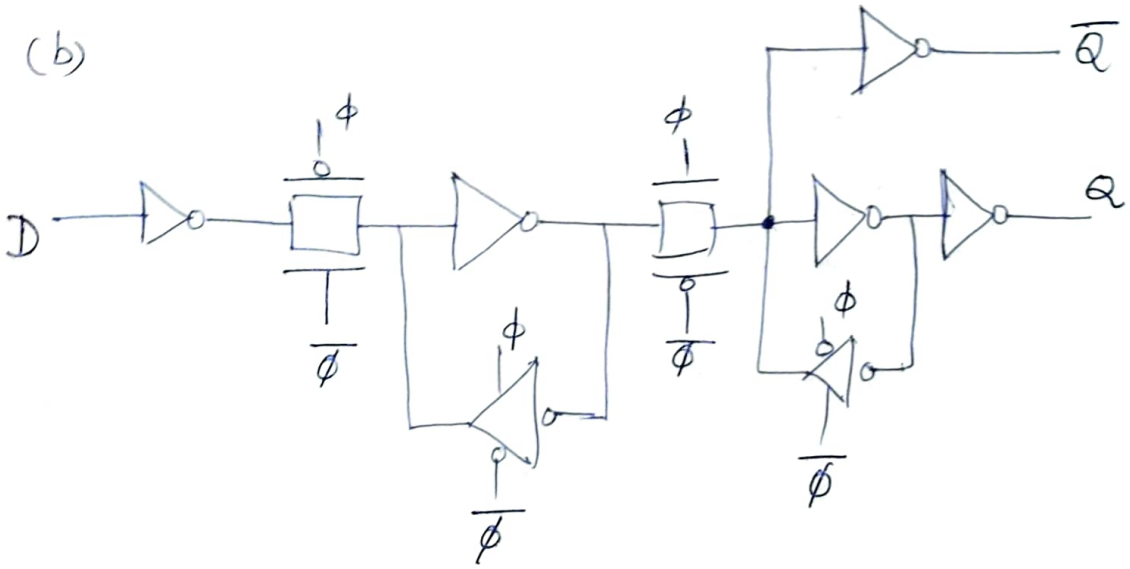
fig(g) is a robust latch that addresses all of the deficiencies discussed so far. The latch is static & all nodes swing rail-to-rail, the state noise is isolated from o/p noise.

CMOS Flipflops

(a)



(b)

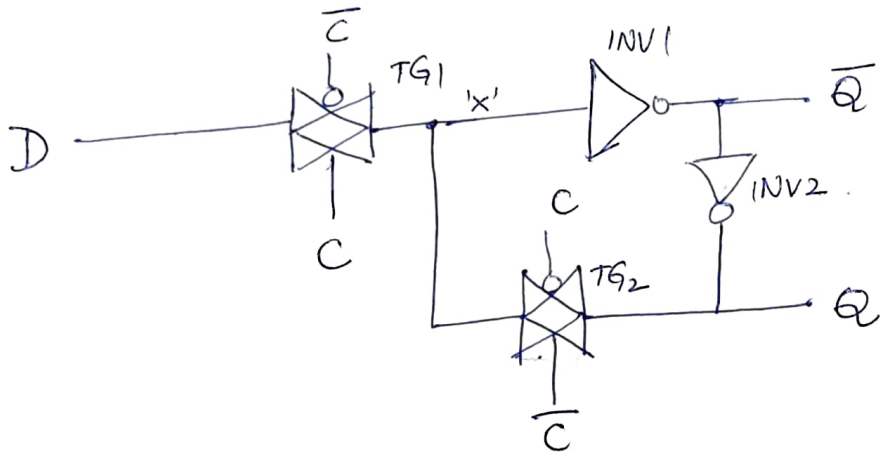


→ fig(a) shows a dynamic inverting F/F built from a pair of back-to-back dynamic latches. Either the first or the last inverter can be removed to reduce the delay at the expense of noise sensitivity on the unbuffered i/p or o/p.

→ fig(b) adds f/b & another inverter to produce a non-inverting static flip flop. Most Std. cell libraries employ this design because it is simple, robust, compact, & energy-efficient.

→ F/F usually take a single clock (ϕ) & locally generate its complement ($\overline{\phi}$).

D Latch using TG



(a) when $C=0$ $(\overline{C}=1)$, TG1 acts as open switch The circuit holds the values of Q & $\overline{Q}$ at the o/p.

(b) when $C=1$ $(\overline{C}=0)$, TG1 is 'ON', allowing the i/p bit 'D' to be transferred to the latching circuit.

During this time, $X=D$

$Q=D$

$\overline{Q}=\overline{D}$

Master- Slave D flipflop

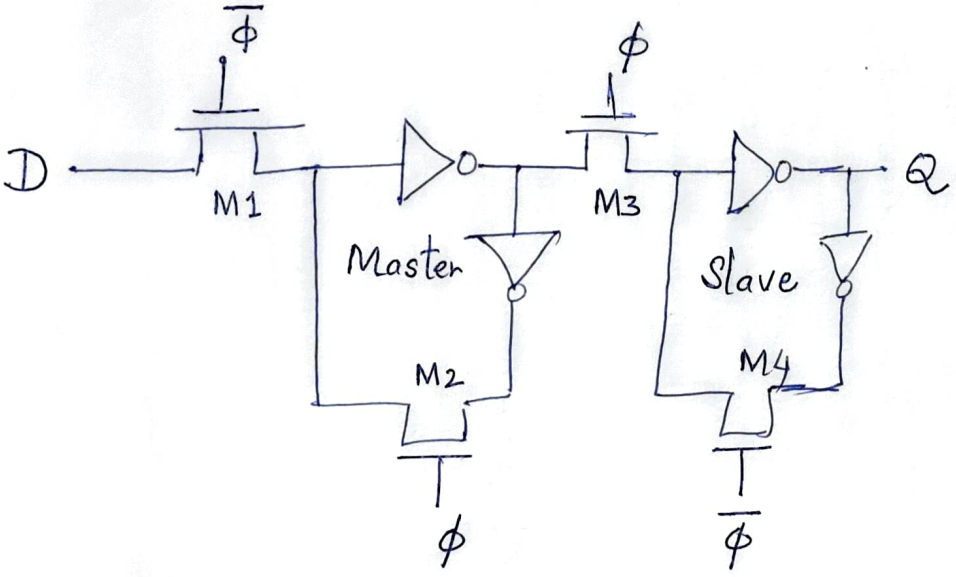
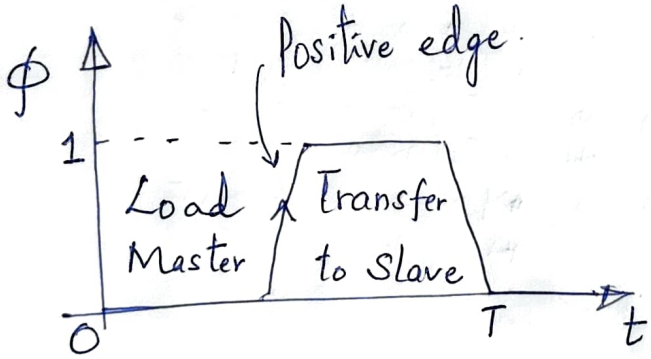


fig. Master - slave D - Flipflop.



→ ϕ → Synchronization & controls the operation.

→ $\phi = 0$ ($\bar{\phi} = 1$)
M1 → ON (closed switch), M2 & M3 → Open circuits.
Master Latch allows the i/p 'D'

→ $\phi = 1$, M2 & M3 → closed switch
transfer the bit to the slave.

→ M,S D F/F → +ve edge triggered D FF

Clock distribution

→ clock tree

→ clock Mesh

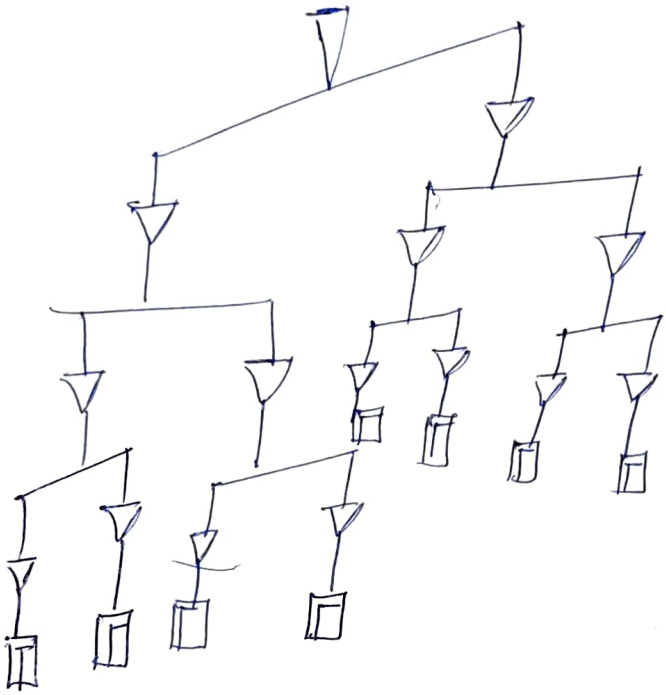
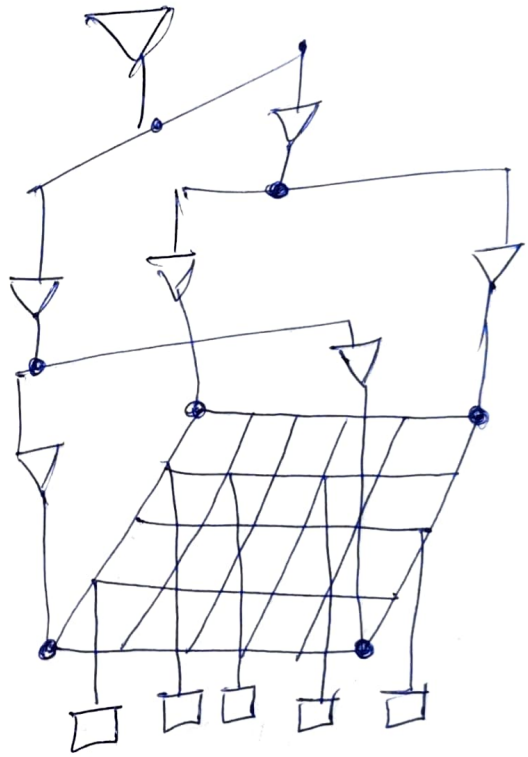


fig. clock tree



clock Mesh

CT → Single path from Source to Sink.

CM → Mesh is a horizontal & Vertical Connections of wire segments.

For the Sinks, clk can reach thro any of the paths.

In CT → there is a unique path from Source to Sink Whereas in CM there is no unique path.

There are sub topologies in clock tree

→ Unconstrained tree

→ H- tree (Balanced tree)

clock generation & distribution

→ Crystal oscillators are used to generate accurate, low-jitter clocks with a freq. range from 10's of MHz to 200 MHz.

→ Crystal oscillator will have tank circuit which is responsible for oscillations.

→ In order to generate higher frequencies (>200MHz) then Phase-locked loop is used.

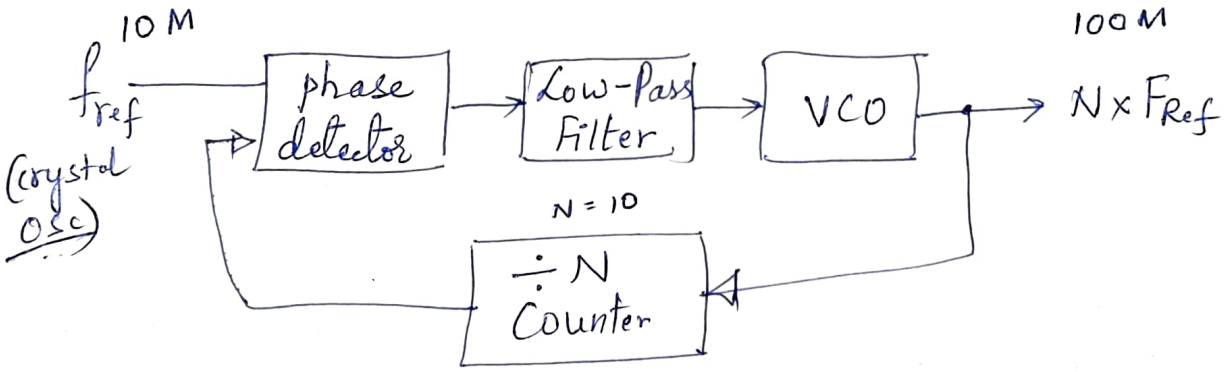


fig. PLL

phase detector → 2 i/p Ex-OR gate

LPF → removes the noise
Provides Pulse to the VCO.

Quartz Crystal are naturally available.

CLASS

STUDENTS NAME

→ In unconstrained tree, there is no structural matching,
clk is taken from the source & the delays are balanced

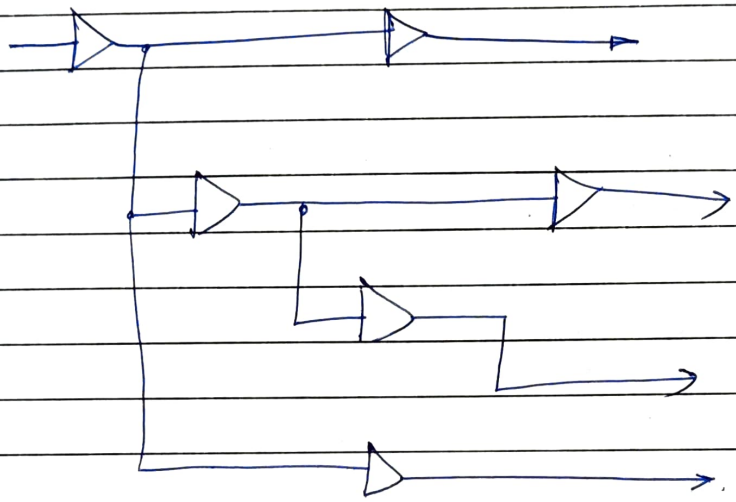
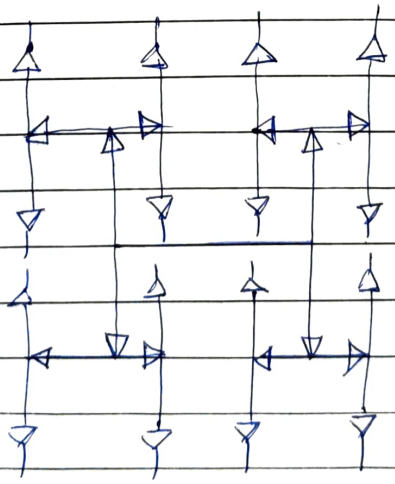


fig. Unconstrained tree

Balanced tree (H- tree)



→ B.t is a tree with structural symmetry & exhibits identical delay, buffer & interconnect segments from the root of the distribution to all branches.

→ Length of the wire is same
→ No of buffers is also same.

clock tree / Binary tree)

Binary tree is a tree intended to delives the clock in a balanced manner in either the vertical or horizontal dimension.

This is different from a balanced tree which is designed to span the entire die in both the horizontal & vertical dimensions.

→ Common structures of clock distribution networks including a trunk, tree, mesh and H-tree