Sequential Circuit Testing

Sequential Circuits and Finite State Machines

Priyank Kalla

Professor
Electrical and Computer Engineering, University of Utah
kalla@ece.utah.edu
http://www.ece.utah.edu/~kalla

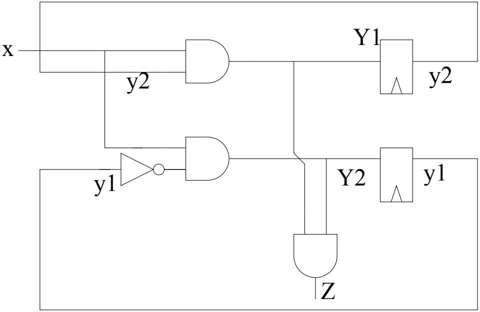
Slides updated Dec 1, 2021

Agenda

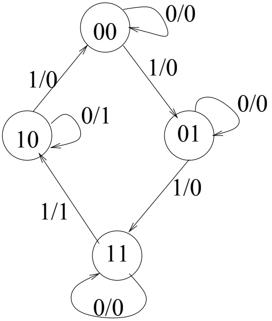
- Review of Sequential Circuits and Mealy Finite State Machines (FSM)
- The concept of FSM equivalence
- Fault excitation and propagation conditions in FSM
- Reset states, synchronizing sequences and redundant states
- Untestable faults in sequential circuits

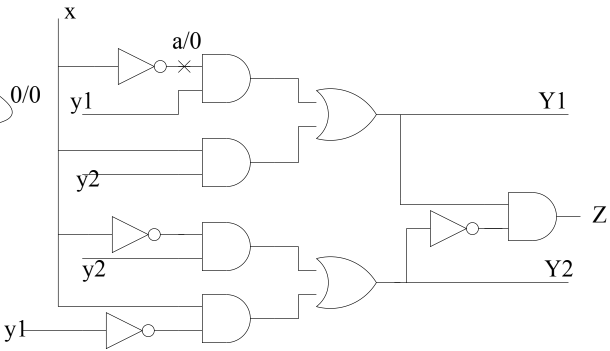
Sequential Circuits

- We consider sequential circuits with edge-triggered D-flip-flops
- Underlying a sequential circuit is a finite state machine (FSM)
- We will consider only FSMs of the Mealy-type (Moore machine ATPG is similar)
- An example of sequential circuit:

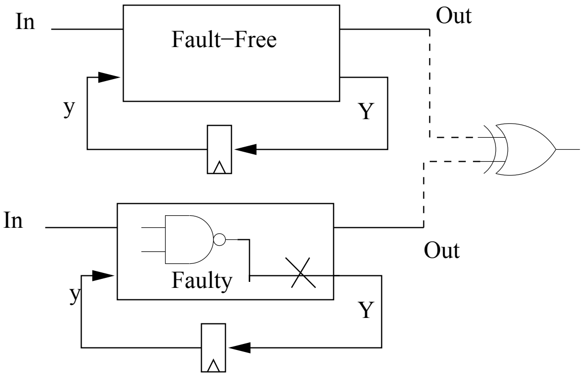


FSM and Circuit example





Sequential Circuit Testing: Sequential Miter



- Find a **sequence of inputs** (or input vectors) that produces an output 1 at the miter output, sometime in the future clock cycles

FSM Equivalence

Two Machines, $\mathcal{M}_1$ and $\mathcal{M}_2$, are equivalent if

- They are identical; or
- They have identical states but different encoding; or
- $\mathcal{M}_1 \subseteq \mathcal{M}_2$ or vice-versa; or
- They have different reachable states but same distinguishable states (same condition as above); or
- Different unreachable states, and unreachable states are a *don't care* condition

Prove that two machines (sequential circuits) produce the same output response on application of all possible input **sequences**

Sequential ATPG is Harder than Combinational

- Given: A sequential circuit, generate ATPG. Sometimes reset (starting) state available, sometimes not!
- Procedure: Activate a fault, and propagate it to PO
- Bottlenecks: Fault activation requires an “activation state”. How to get to the activation state? Fault can be propagated to the next-state line, but not to PO? Unroll the machine, or in other words, traverse to other states.
- How to distinguish between Faulty and Fault-free machines? The concept of state distinguishing seqences.
- In absence of resets, how to synchronize both faulty and fault-free machines to the same states. Concept of synchronizing sequences.

Untestable Faults in Sequential Circuits

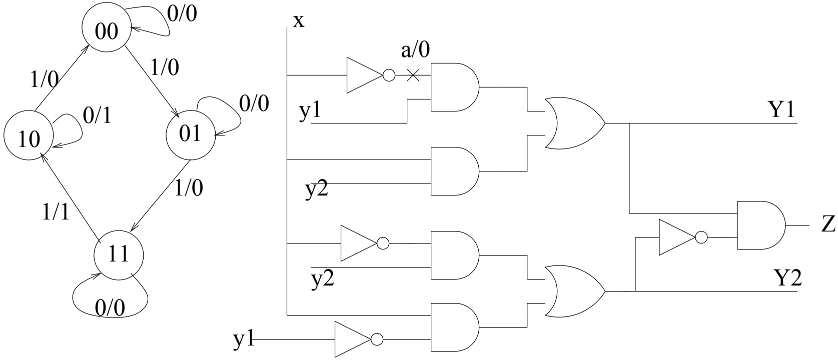
A fault in a sequential circuit can be untestable due to **four reasons**:

- Redundancy in a combinational logic
- Lack of a common synchronizing sequence that can drive both faulty and fault-free machines to a common (starting) state
- Sequential Redundancy: a fault causes a transition to a (faulty) state which may be equivalent to a fault-free state
- Fault excitation or propagation requires getting to an unreachable state

We will look at all these cases!

First example: When reset state exists

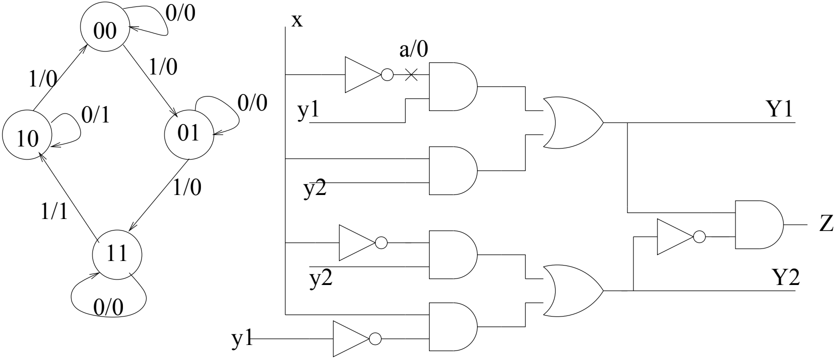
- Start the machine in a reset state (00), traverse FSM
- At some point in the future, faulty and fault-free machines will diverge



Test: $\underline{x_0 = 1}, x_1 = 1, x_2 = 0, x_3 = 1$

First example: When reset state exists

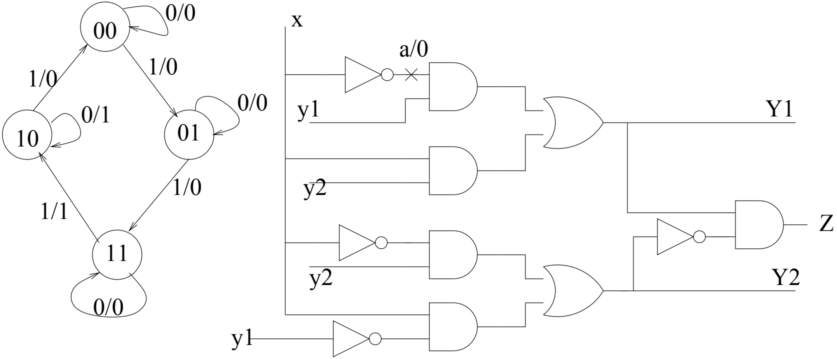
- Start the machine in a reset state (00), traverse FSM
- At some point in the future, faulty and fault-free machines will diverge



Test: $x_0 = 1, \underline{x_1 = 1}, x_2 = 0, x_3 = 1$

First example: When reset state exists

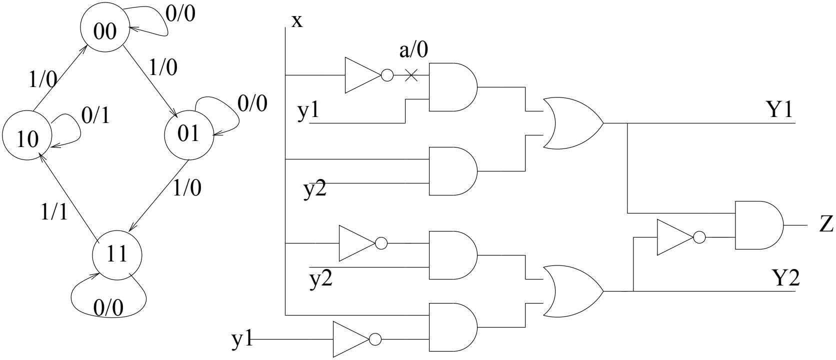
- Start the machine in a reset state (00), traverse FSM
- At some point in the future, faulty and fault-free machines will diverge



Test: $x_0 = 1, x_1 = 1, \underline{x_2 = 0}, x_3 = 1$

First example: When reset state exists

- Start the machine in a reset state (00), traverse FSM
- At some point in the future, faulty and fault-free machines will diverge



Test: $x_0 = 1, x_1 = 1, x_2 = 0, \underline{x_3 = 1}$

Meaning of the above Test

- Fault D *or* $\overline{D}$ gets into next state line.
- Faulty and Fault-free machines go to different states.
- Machine operation diverges....
- Sometime in the future, you can catch that effect.
- You can distinguish between faulty and faulty-free states of the machine.
- How? Using state distinguishing experiments.

When there is no reset state?

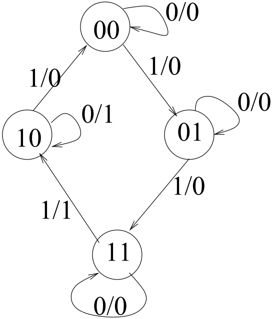
- Power up the machine, it can be in any (unknown) state
- Does there exist a sequence of inputs that can drive the machine to some – singleton – state?
- Some machines have a synchronizing sequence, others do not
- Given an FSM, how to find a synchronizing sequence?
- For ATPG: we are not given the FSM, only the circuit
- For ATPG: We need to find a synchronizing sequence for both the faulty and fault-free machines that can drive both to a common state. This can be hard.

Synchronizing Sequence on an FSM

Table: State Transition Table

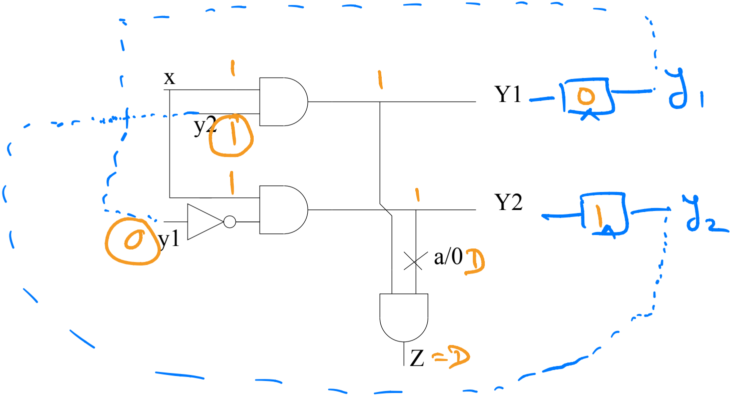
P.S.	Next State, Z	
	$x = 0$	$x = 1$
A	B, 0	D, 0
B	A, 0	B, 0
C	D,1	A,0
D	D,1	C,0

This machine has no Synchronizing Sequence



ATPG without Reset State

TFI: $Y_1=1, Y_2=1$ $Z=D$



Fault excitation state = (0,1)
= current state

No Reset State Example: Test Exists

- No reset state. Start w/ Fault excitation state: $y_1 = 0, y_2 = 1$.
- $x = 1$ propagates the fault effect!
- Problem: How to get to the fault excitation state?
- Answer: Sequential backtrack! Unroll the m/c backwards until you can get unknown values in FFs. This implies there exists a state (backward in time from F.E. state) where you can "control" the FFs (control the state).
- Called Self-Initializing tests! This test implies that both faulty and fault-free machines can be initialized to a "common initial state"; i.e., you can get a common starting point.

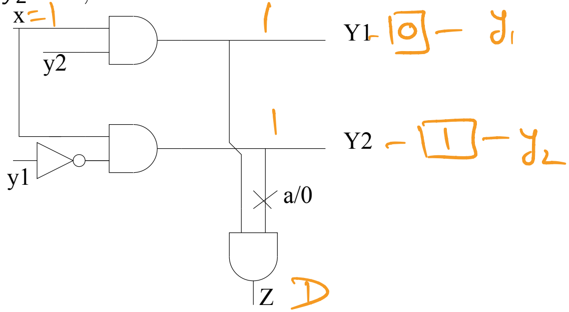
ATPG without Reset State

Y = Next state FF inputs, y = present state FF output

Generate a self initializing test: Start in Time Frame (TF) 1:

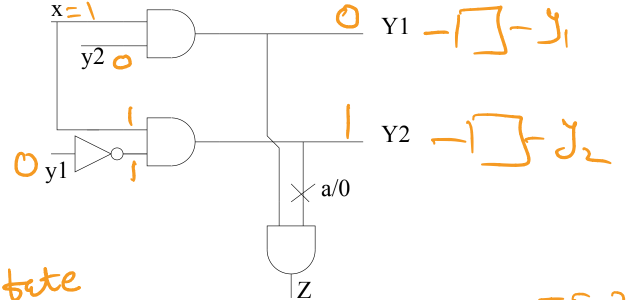
$$Y_2 = 1, Y_1 = 1, Z = D$$

This requires: $y_1 = 0, y_2 = 1, x = 1$



ATPG without Reset State

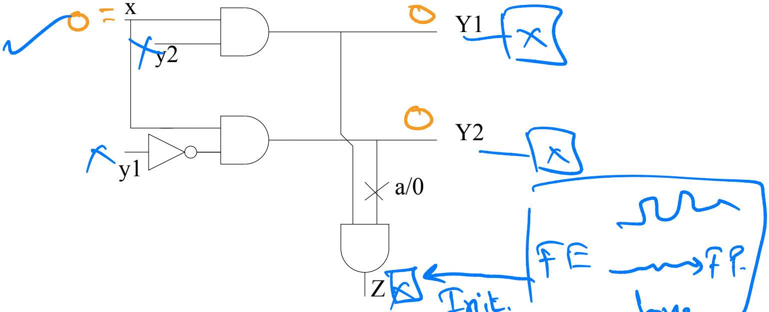
If in TF 1, present states are $y_1 = 0, y_2 = 1$, then in the previous state TF 0: $Y_1 = 0, Y_2 = 1$. Make $x = 1, y_1 = 0, y_2 = 0$



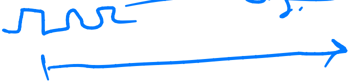
Current state
$y_1=0, y_2=0.$ → In previous state TF-2
$Y_1=0, Y_2=0$

ATPG without Reset State

In TF 0 : $y_1 = 0, y_2 = 0$, then go back in time again TF -1: $Y_1 = 0$, $Y_2 = 0$ is needed, so apply $x = 0$



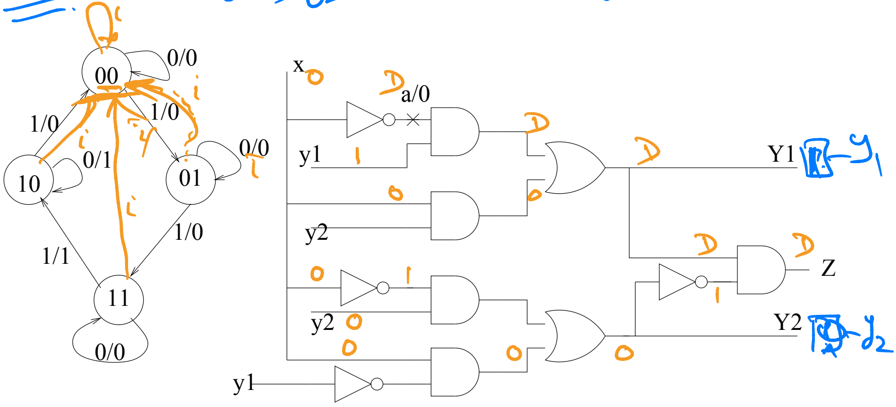
Test: TF -1 $x = 0$, TF 0 $x = 1$, TF 1 $x = 1$.



3-clock sequence.

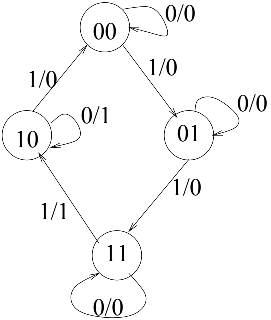
No Self Initializing Test without Reset

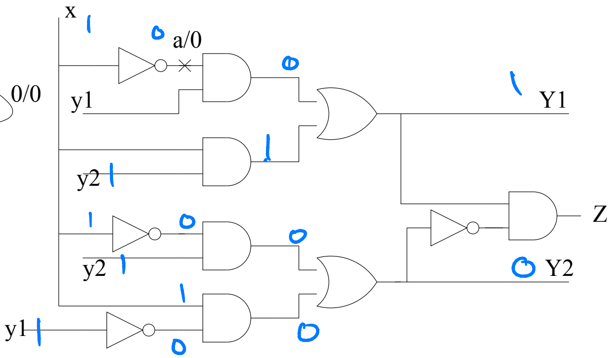
TF1 Need $y_1=1$, $y_2=0$. Go backtrace



No Self Initializing Test without Reset

TFO. $Y_1=1$, $Y_2=0$. Can we control $Y_1=1, Y_2=0$ only using $x=?$



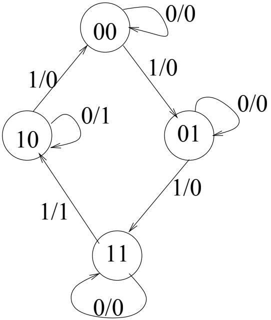


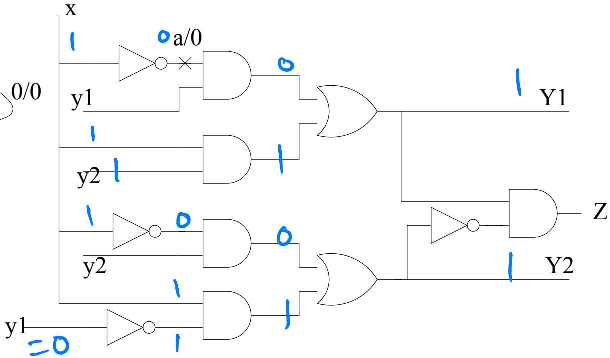
Need $y_2=1$, $y_1=1$ w/ $x=1$ in T.F.O.

Go back.

No Self Initializing Test without Reset

TF -1 $Y_1=1$, $Y_2=1$



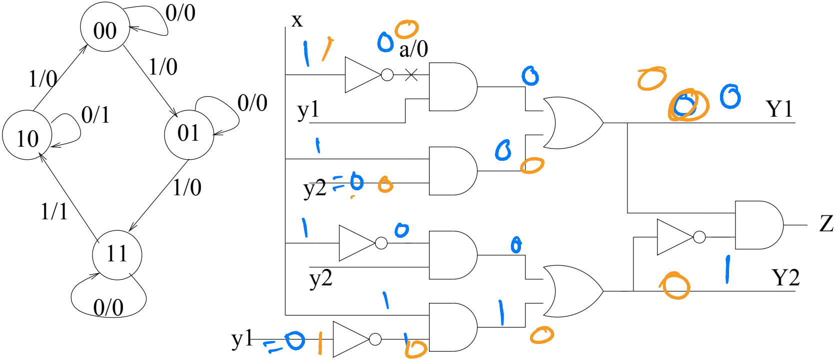


Need $y_2=1$ w/ $x=1$ in previous state TF-2

$y_1=0$

No Self Initializing Test without Reset

TF -2 Need $Y_1=0, Y_2=1$ ⇒ $y_1=0, y_2=0$.

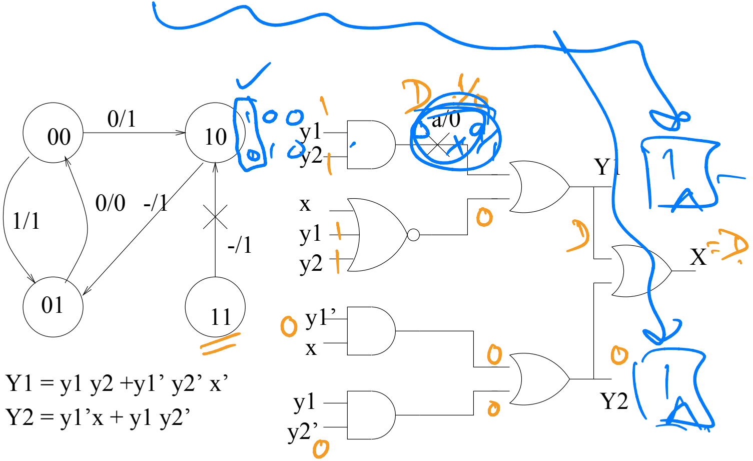


Therefore, no test exists, deduced after 4 clock cycles of sequential backtrace!

TF -3: Need $Y_1=Y_2=0$

$x=1, y_1=1, y_2=0$ → where we started

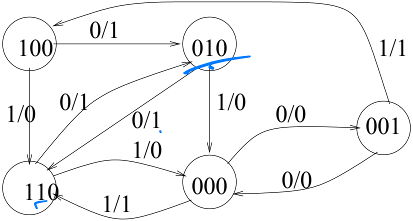
Untestable Faults: Unreachable State



Y1 = y1 y2 +y1' y2' x'

Y2 = y1'x + y1 y2'

Untestable Fault: Sequential Redundancy



$010 \xrightarrow[0]{1} 000$

$010 \xrightarrow[1]{0} 110$

$110 \xrightarrow[0]{1} 000$

$110 \xrightarrow[1]{0} 010$

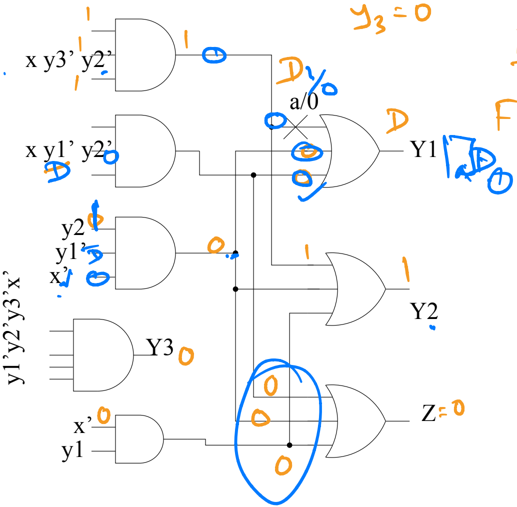
States (110) and (010) are equivalent -> cannot be distinguished!

What if fault takes m/c to 110 (faulty state), instead of 010 (fault free)?

$\boxed{\times} \leftarrow \leftarrow \leftarrow$ FE — FP.

Init. $\longrightarrow$

Redundanct Circuit ATPG



$y_2=0$ $\bar{y}_1=0 \Rightarrow y_1=1$ $x=1$

$y_3=0$

F.E 100

FF. 100

↓

110

↔

Fy 100

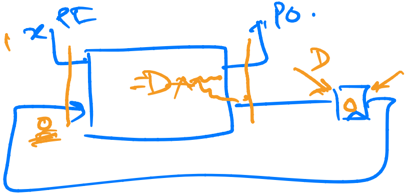
↓

010

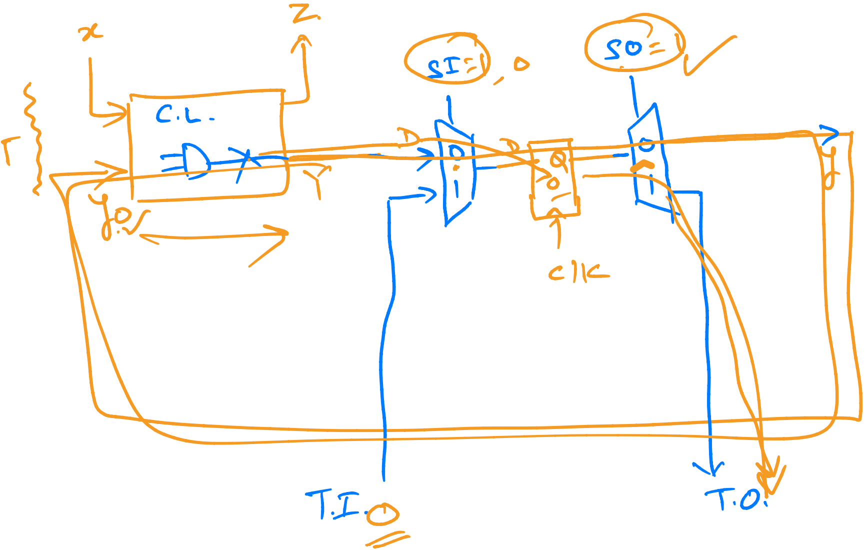
Equiv. states.

Simplify Sequential ATPG using Scan Design

- Primary inputs = controllable, Primary outputs = observable
- Lack of controllability and observability of Flip-Flops
- For testing purposes, make the flip-flops both controllable and observable
- Approach: Introduce Scan-registers
- Make a sequential circuit combinational for Testing



Scan Design



Scan Design