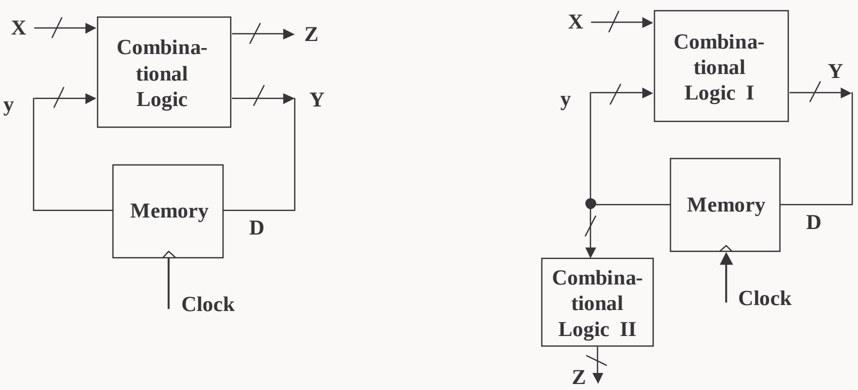
DIGITAL SYSTEM TESTING
COE -545

Lecture – 14

Test Generation for Sequential Circuit

Test Generation for Sequential Circuit



Mealy Machine
Z= F(x, y)
Y = Y(x,y)

Moore Machine
Z= F(y)
Y = Y(x,y)

COE – KFUPM *Dr. Alaaeldin Amin (COE 545)* *Slide Number 2*

Introduction

- Test Generation Problem is More Complicated for Seq. Circuits, where output is function of past history → Temporal Diemension

Characteristics of Sequential Testing

- Much harder to test than combinational circuits
- Tests are *Sequences* of input vectors
- Two circuit types: asynchronous and synchronous (Only Synchronous is Considered here)

COE – KFUPM   Dr. Alaaeldin Amin (COE 545)   Slide Number 3

Introduction

Basic Approaches

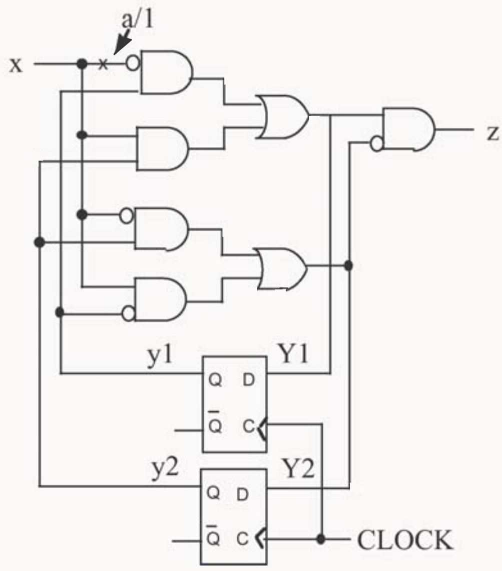
- State table analysis
- Machine identification (checking sequence) method
- *Time-Frame Expansion*: combinational ATPG methods such as PODEM) are applied to an *iterative logic array* (ILA) model of the target sequential circuit
- Simulation-based methods

Key Assumptions

- Unknown initial state
- Known Initial State

COE – KFUPM   Dr. Alaaeldin Amin (COE 545)   Slide Number 4

State Table Method



y1y2		x = 0	x = 1
0 0	A	A,0	B,0
0 1	B	B,0	C,0
1 1	C	C,0	D,1
1 0	D	D,1	A,0

Fault-free state table

y1y2		x = 0	x = 1
0 0	A	A,0	B,0
0 1	B	B,0	C,0
1 1	C	B,0	D,1
1 0	D	A,0	A,0

State table with fault a/1

State Table Method (Diagnostic/Decision Tree)

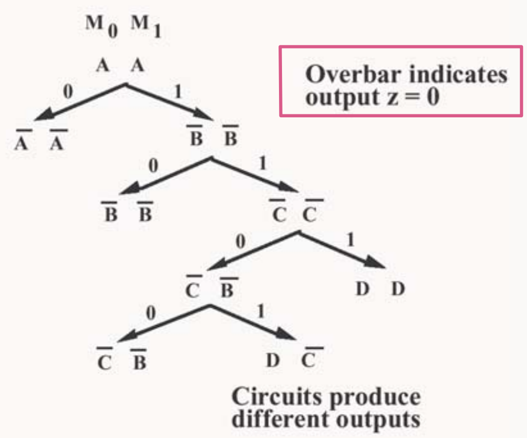
	x = 0	x = 1
A	A,0	B,0
B	B,0	C,0
C	C,0	D,1
D	D,1	A,0

Fault-free machine M_0

	x = 0	x = 1
A	A,0	B,0
B	B,0	C,0
C	B,0	D,1
D	A,0	A,0

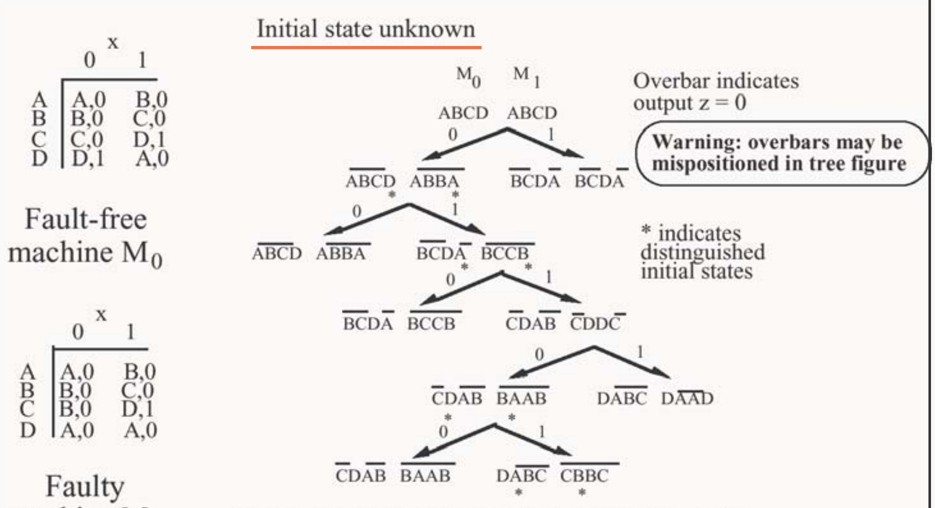
Faulty machine M_1

Initial state A known



Minimum length test for a/1 with initial state A: x(t) = 1 1 0 1

State Table Method (Diagnostic/Decision Tree)



Minimal test for a/1 with unknown initial state: $x(t) = 0\ 1\ 1\ 0\ 1$

Useful Sequences

Homing Sequence

- *Definition*: An input sequence *S* that takes ***M*** to a final state that is uniquely determined by the output response to *S*
- *Every reduced **M** has a homing sequence [length < n(n–1)/2]*

Synchronizing Sequence

- *Definition*: An input sequence *S* that takes ***M*** to a unique final state, independent of the initial state
- *Not every machine has a synchronizing sequence*

Distinguishing Sequence

- *Definition*: An input sequence *S* that yields a unique output response for every possible initial state
- *Not every **M** has a distinguishing sequence, but very **M** has a complete set of partial distinguishing sequences called characterizing sequences*

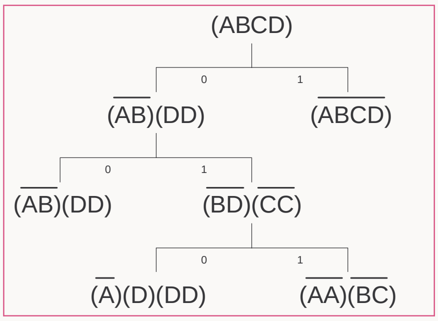
Homing Sequence

Example

- Build Successor Tree, Terminate When
  - New Uncertainty Vector Identical to a Previous Level Uncertainty Vector
  - Uncertainty Vector Resolved (No 2 Different States in the same Subcomponent)
- Shortest Homing Sequence is 010

Initial State	*Response to 010*	*Final State*
A	000	A
B	001	D
C	101	D
D	101	D

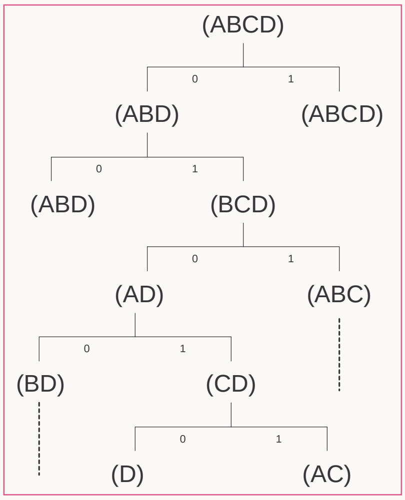
PS	NS, Z *x*=0	*x*=1
A	B, 0	D,0
B	A, 0	B, 0
C	D, 1	A, 0
D	D, 1	C, 0



Synchronizing Sequence

Example

- Build Successor Tree, Terminate When
  - New Uncertainty Vector Identical to a Previous Level One
  - A Node is Associated with an Uncertainty of Just One State
- Shortest Synchronizing Sequence is **01010**



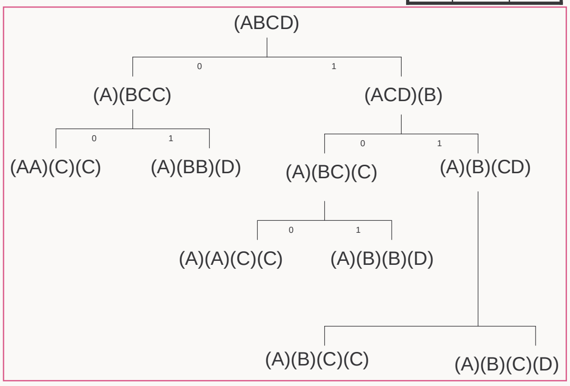
PS	NS, Z *x*=0	*x*=1
A	B, 0	D,0
B	A, 0	B, 0
C	D, 1	A, 0
D	D, 1	C, 0

Distinguishing Sequence

PS	NS, Z x=0	NS, Z x=1
A	C, 0	D,1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Example

- Build Successor Tree, Terminate When
  - New Uncertainty Vector whose non-individual components appear at a preceding level
  - One or more component of the Uncertainty Vector has repeated state occurrence
  - Node has components of individual States
- Shortest Synchronizing Sequence is 01010



Machine Identification

Problem

- Given an n-state sequential machine ***M*** described by a state table T (or equivalent), construct a test called a *checking sequence* that will determine whether any given (potentially faulty) machine ***M**** has the same state table as ***M*** .

Assumptions

- ***M*** is Completely Specified
- ***M*** is reduced
- ***M*** is strongly connected (every state reachable from every other state)
- *In general*, the initial state before testing begins is unknown

Fault Model

- Any behavioral change that does not increase the number of states *n*

Fault Detection Experiments

Two Parts

1. *Initialization*: Transfer the Ckt., into a a Prespecified State .
2. *Transition Tests*: Take the CUT through all possible transitions
   - *Verify Current State, e.g. Using a Distinguishing Sequence*
   - *Perform actual transition by applying a Transfer sequence from the current to the desired state .*

***M* has a distinguishing sequence (DS)**

- *Let the States of M be* $S_1, S_2, \ldots S_n$
- *Let* X_0 *be a* ***DS*** *of M*
- *Let* $T(S_i, S_j)$ *be a transfer Sequence that takes M from state* S_i *to state* S_j
- $X_k\ S_i$, *is the final state reached when the sequence* X_k *is applied to State* S_i

COE – KFUPM Dr. Alaaeldin Amin (COE 545) Slide Number 13

Fault Detection Experiments

- *Let* Q_i *(=* $X_0\ S_i$ *) denote the State M reaches when the Sequence* X_0 *is applied to M while in state* S_i

First Part

- Apply a Homing/Synchronizing Sequence to reach some known initial state, Say S_1
- *Applying the Sequence:*

$$X_0\ T(Q_1, S_2)\ X_0\ T(Q_2, S_3)\ X_0\ T(Q_3, S_4) \ldots\ldots X_0\ T(Q_n, S_1)\ X_0$$

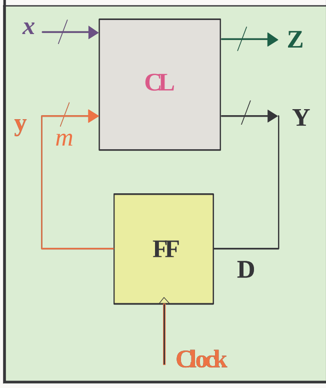
Takes M through each of its states → Experiment Leaves M in state Q_1

- *Verify Each State Transition, e.g. to Verify the 0-Transition out of State Si when M is in Qj, use the Sequence* $X_0\ T(Q_j, S_{i-1})\ X_0\ T(Q_{i-1}, S_i) 0\ X_0$

COE – KFUPM Dr. Alaaeldin Amin (COE 545) Slide Number 14

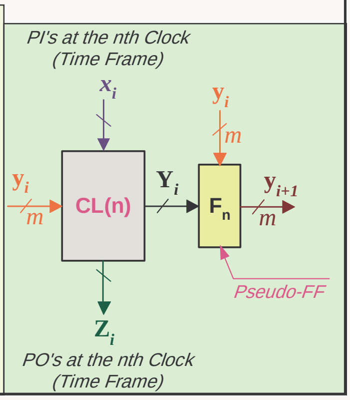
Time Frame Expansion

- *Turn the Temporal Relationship between Machine (CUT) States into a Spatial one →*
- *Convert the Seq CUT into a 1-Dimensional Iterative Array of Identical Combinational CUT →*
- *Each CUT Copy Represents a Time Frame which is Fed information about its State from the Previous Stage (Copy).*



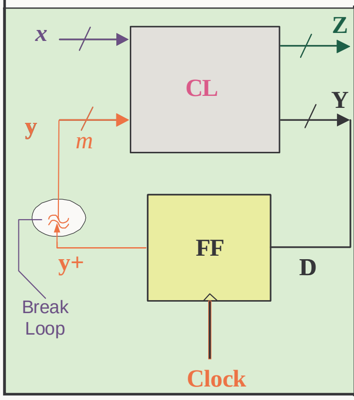
FFs are Represented by Equivalent Combinational Logic → Pseudo-FF with

- In addition To Regular Excitation Inputs PFFs have additional Inputs Representing PS
- In addition To Regular Outputs, PFFs have additional Outputs Representing NS



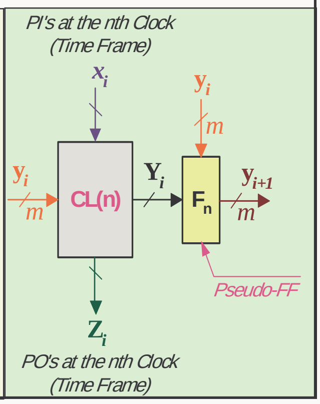
Time Frame Expansion

- *Turn the Temporal Relationship between Machine (CUT) States into a Spatial one →*
- *Convert the Seq CUT into a 1-Dimensional Iterative Array of Identical Combinational CUT →*
- *Each CUT Copy Represents a Time Frame which is Fed information about its State from the Previous Stage (Copy).*



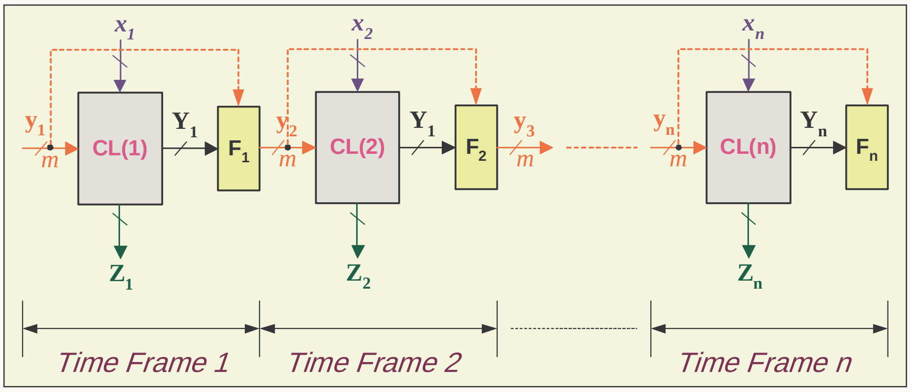
FFs are Represented by Equivalent Combinational Logic → Pseudo-FF with

- In addition To Regular Excitation Inputs PFFs have additional Inputs Representing PS
- In addition To Regular Outputs, PFFs have additional Outputs Representing NS



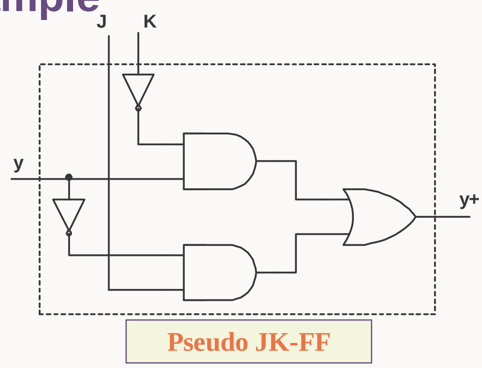
Time Frame Expansion

- *The Pseudo-FF is Purely Combinational CKT with*
  - *No Clock Input*
  - *Extra Input Representing the Present State*
  - *Extra Output Representing the Next State*

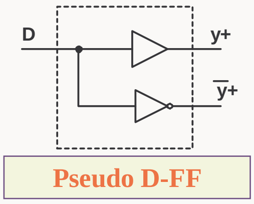


Example

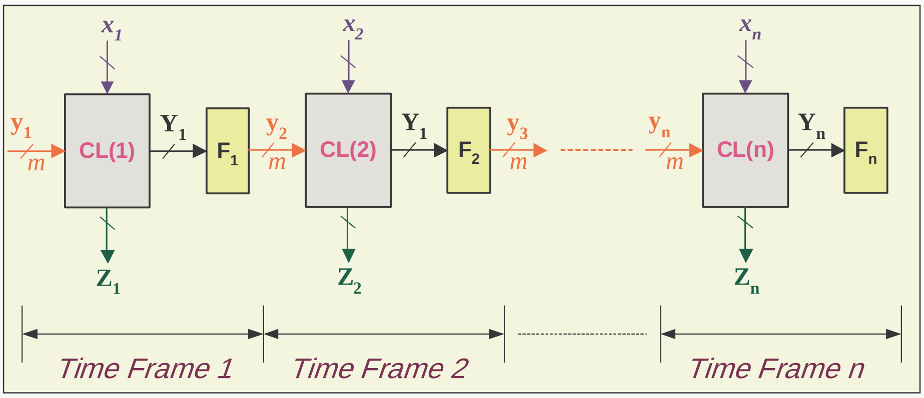
- JK - FF
  - **y+ = J y' + K'y**



- D-FF
  - **y+ = D**

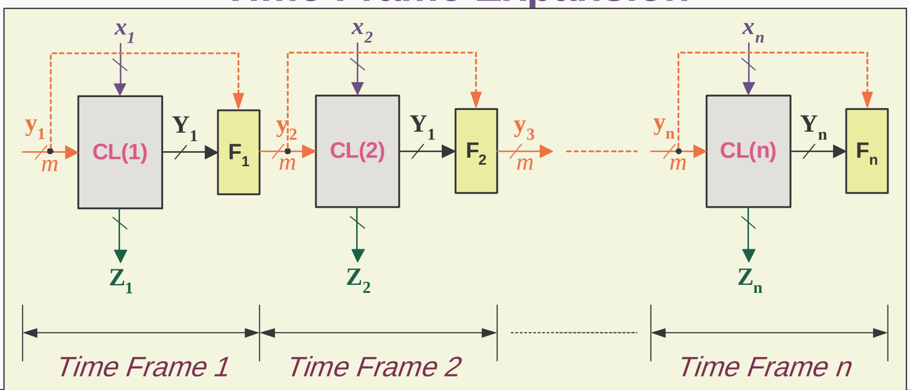


Iterative Array Model for D-FFs



COE – KFUPM Dr. Alaaeldin Amin (COE 545) Slide Number 19

Time Frame Expansion



- *FFs and Clock Line are Fault-Free*
- *Same Clock Signal Gating all FFs*
- *Fault doesn't Increase the Number of States*
- *Resulting Iterative Model is Purely Combinational → D-Alg, PODEM FAN May be Used.*

- *All Time Frames Have Identical Ckts*
  - *A Single Stuck Fault is Mapped into **n** Multiple Faults*
  - *TG Algorithm need not Construct **n**-Copies of the Same Ckt → One is Enough.*
  - *Signal Values in Different Time Frames Must be Maintained*

Time Frame Expansion

- *Attempt is first made to propagate the fault (**D or DB**) to a PO → If Successful a TV is generated*
- *If not Successful, attempts are made to Propagate the Fault to the Next State Variables which Requires another Time Frame.*
- *A Signal Value Assigned to **y+** Must be **Propagated** into the Next Time-Frame*
- *A Signal Value Assigned to **y** Must be **Justified** Backwards into the Previous Time-Frame*
- *Number of Time Frames is Unknown → Complexity*
- *If Initial States of the Fault-Free and the Faulty Circuits are not known → Added Complexity.*
- *Even if Initial State is Known, it May not be Activatable in the first Frame → A Sequence may be Required to Activate the Fault*

COE – KFUPM | Dr. Alaaeldin Amin (COE 545) | Slide Number 21

Time Frame Expansion

- *With Multiple Faults in the Multiple Time-Frames, Propagating Signal Values ∈* **{0, 1, D, DB}** onto a Faulty Line Follows the Shown Table

Propagated Value	Line Fault	Resulting Line Value
0	S_a_0	0
1	S_a_0	D
0	S_a_1	DB
1	S_a_1	1
D	S_a_0	D
DB	S_a_0	0
D	S_a_1	1
DB	S_a_1	DB

COE – KFUPM | Dr. Alaaeldin Amin (COE 545) | Slide Number 22

Sequential ATPG

- Initial state is assumed to be known

```
While r < fmax do
begin
    Build model with r time frames;
    Ignore primary outputs in first r – 1 time frames;
    Ignore secondary (Y) outputs in last time frame;
    Set initial state to Y(0);
    if (test is found) return(SUCCESS);
    /* No solution exists with r time frames */
    r = r + 1
end
return(FAILURE);
```

Where: fmax: maximum number of frames,

Test Generation Procedure

1. **In time-frame model of the CUT set t = 0 and the flip-flops to an initial state Y(0) by**
   - **Resetting to a known state, or else**
   - **Setting state values to unknown X**
2. **Apply combinational ATPG to frame corresponding to current t. Try to drive D/D' to primary output Z. If successful, exit.**
3. **Otherwise, try to drive D/D' to secondary output Y. If unsuccessful exit with no test and go to step 2.**
4. **Perform justification. If successful and D/D' is at Z, exit with a test. Otherwise, increment t to t + 1 and go to step 2.**

Example

- Line a: s-a-1 fault. Known state (0, 0).

<u>Time frame 1</u>: Fault activation: $(y_1,y_2) = (0,0) \rightarrow$ **y_2 =DB**

- Fault propagation I(1) =1 → **y_2^+ = DB** , Z= 1

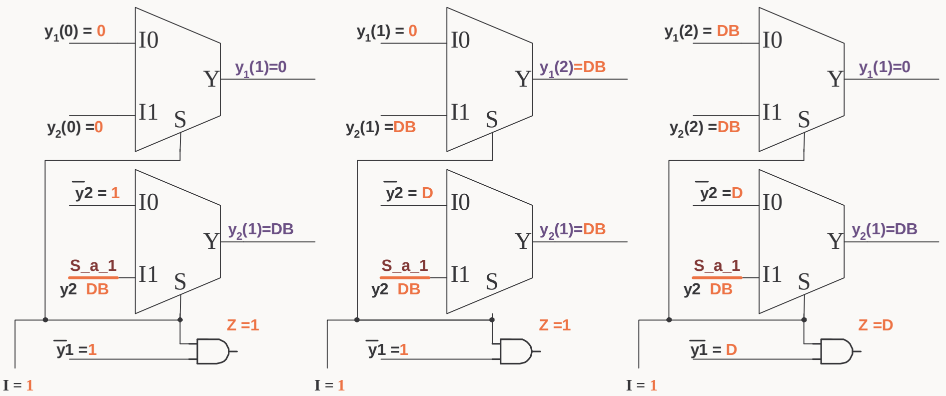
<u>Time frame 2:</u> **Depending on the Chosen D-Frontier Gate**

Either I(2)=1 → **(y_2^+ , y_1^+)=(DB, DB) → $y_2(2)$ = DB & $y_1(2)$ = D**

OR I(2)=0 → **(y_2^+ , y_1^+)=(D , 0) → $y_2(2)$ = D & $y_1(2)$ = 0**

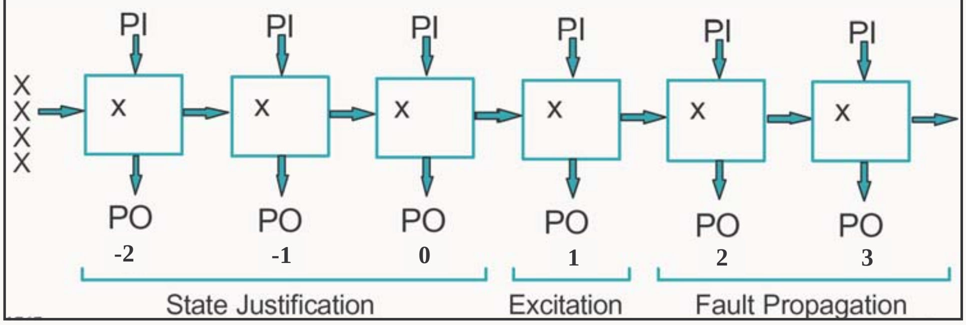
<u>Time frame 3:</u> First Choice **(y_2^+ , y_1^+)=(DB, DB) Puts Z on the D-Frontier**

→ I(3)=1 → Z = DB



Sequential ATPG

- **Initial state is Unknown**
- *Initial State = xx…..x*
- Three main components of sequential circuit test
- generation (not in order):
  - **Excite the fault in one Time Frame (Labeled 1 Say)**
  - **Propagate the fault effects Forwards (in Time) to a PO if possible or to a State (Secondary) Variable → Requires r ≥ 1 Frames**
  - **Justify the state with backward propagation Using p Frames (Frames 0, -1, -2, …-(p-1))**
  - **Stop when All State Variables are all *x*'s**



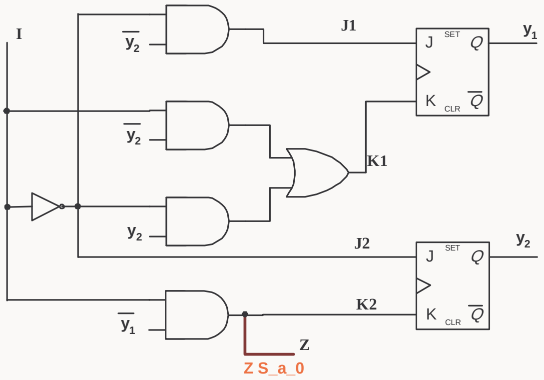
Sequential ATPG

- **Initial state is Unknown**

r = 1
p = 0
repeat
  begin
    Build model with *p*+*r* time frames
    Ignore the POs in the first *p*+*r*-1 frames
    Ignore the q+ outputs in the last frame
    If (test generation success) and every q input in the first frame has value *x*) **then return** *SUCCESS*
    Increment *r* or *p*
  end
until (*r* + *p* = *f*max)
return *FAILURE*

Example

- Consider the Fault Z/0
- Use Pseudo JK-FF (PJKFF)
- Singular Covers of the PJKFF and Propagation D-Cubes need to be Derived

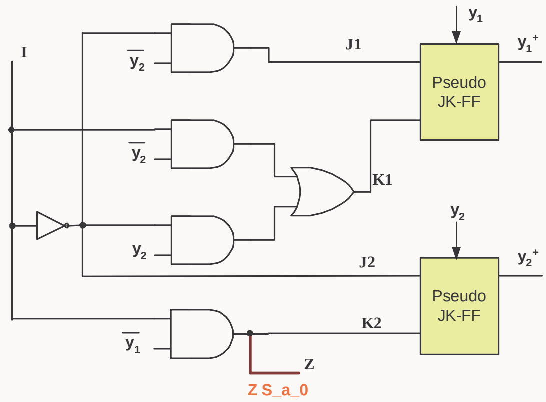


Example

- Pseudo JK-FF Implements the Function
  - ➢ y+ = J y' + K'y

Time Frame 1

- Only Test to Detect Fault is: **(I, y_1)= 10**
- Only one Frame (r =1) is Thus Needed to Propagate the Fault to the PO (Z)
- Need to Justify **y_1=0 → (p > 0)**



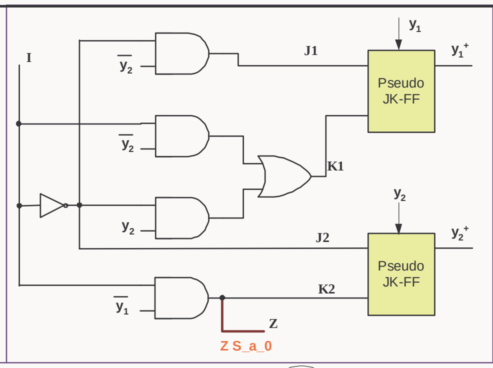
COE – KFUPM *Dr. Alaaeldin Amin (COE 545)* *Slide Number 29*

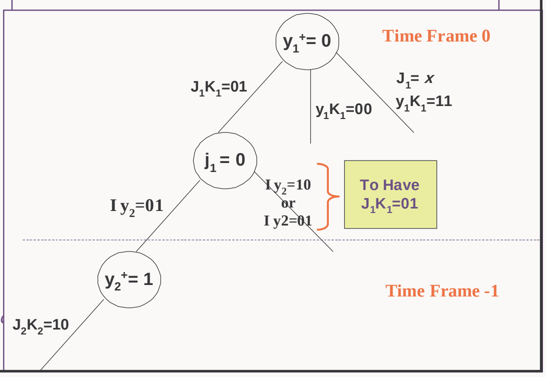
Example

- Pseudo JK-FF Implements the Function
  - ➢ y+ = J y' + K'y

Time Frame 0

- To Justify **y_1^+=0** → (J_1K_1 =01) is Thus Needed to Propagate the Fault to the PO (Z)
- Need to Justify **y_1=0 → (p > 0)**
- **For All options to set** (J_1K_1 =01), **y_2** Must be Assigned a Value → Needs to be Justified in YET ANOTHER Time Frame → **(p > 1)**





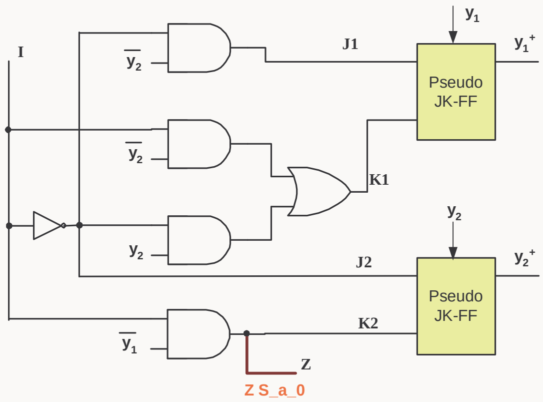
COE – KFUPM *Dr. Ala*

Example

Time Frame -1

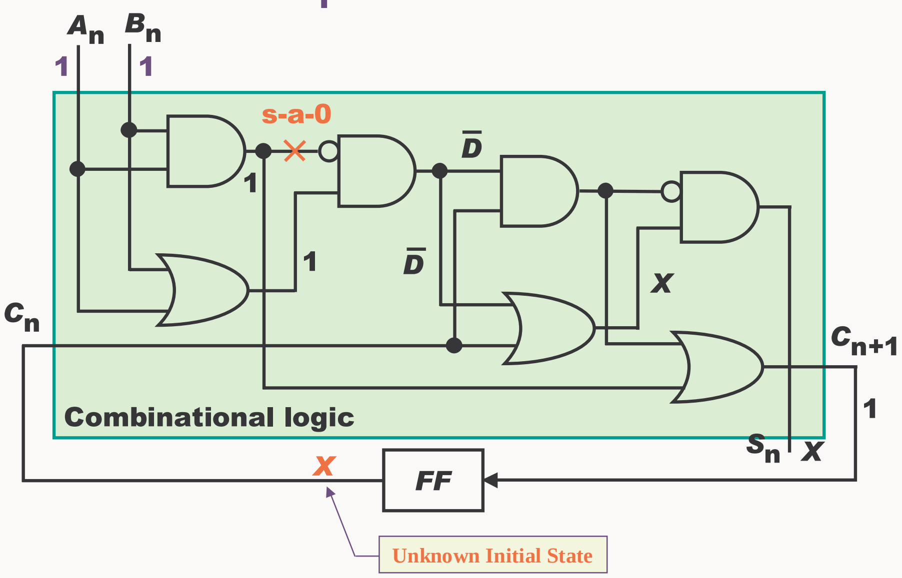
- Justify $y_2^+=1$ → (J_2K_2 =10) → Both Satisfied by **I= 0**
- This Represents a Self-Initializing Test Sequence Since
  - $y_2\,y_1 = xx$
  - *All Lines are Justified*
- *The Test Sequence Becomes:*

  I=001



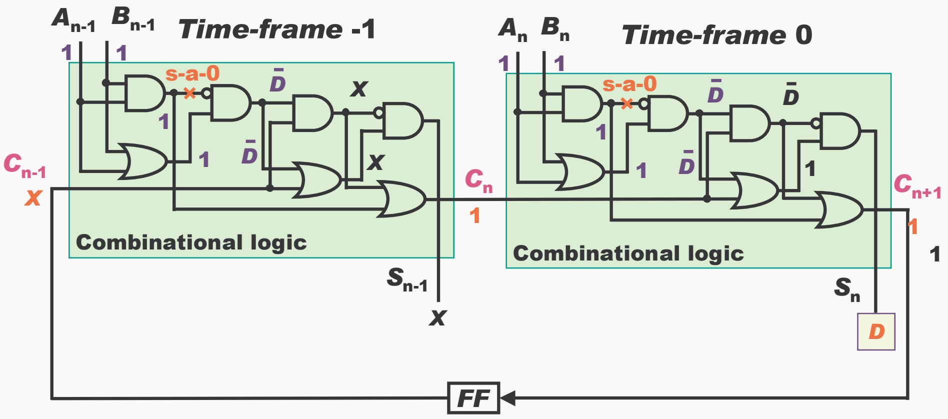
COE – KFUPM | Dr. Alaaeldin Amin (COE 545) | Slide Number 31

Example: A Serial Adder



COE – KFUPM | Dr. Alaaeldin Amin (COE 545) | Slide Number 32

Time-Frame Expansion



COE – KFUPM   Dr. Alaaeldin Amin (COE 545)   Slide Number 33