Test without Fault Model

- Introduction
- Boolean Tests without Fault Model
  - Toggle Test
  - Design Verification
  - <u>Exhaustive Test</u>
    - <u>Checking experiment (1964)</u>
  - Pseudo Exhaustive Test (1984)
- Conclusions

McCluskey and his collection of hats

Exhaustive Tests

- **For combinational circuits with *n* inputs**
  - **Exhaustive test**
    - **all possible 2^n input vectors**
  - ***Super exhaustive test***
    - **all possible input transitions 2^{2n}**
      - **Example: 2 input AND gate**
- **How about sequential circuits?**
  - **Checking experiment**

Checking Experiment

- CE = Input sequence that exhaustively verifies state table of
  - Finite State Machine (FSM)
- CE is high-level, functional testing
  - Does not need implementation of circuit
  - Developed by many [Moore 56] [Poage + McClueksy 64] …
- In this lecture, a general procedure by [Hennie 64]
  - 1. *Synchronizing sequence*: bring FSM to a known state
  - 2. *A sequence*: verify existence of all states
  - 3. *B sequence*: verify all state transitions
- Only control primary inputs (PI), only observe primary outputs (PO)
  - Internal states not observable, not controllable
    - *No scan DFT*

Synchronizing Sequence (SS)

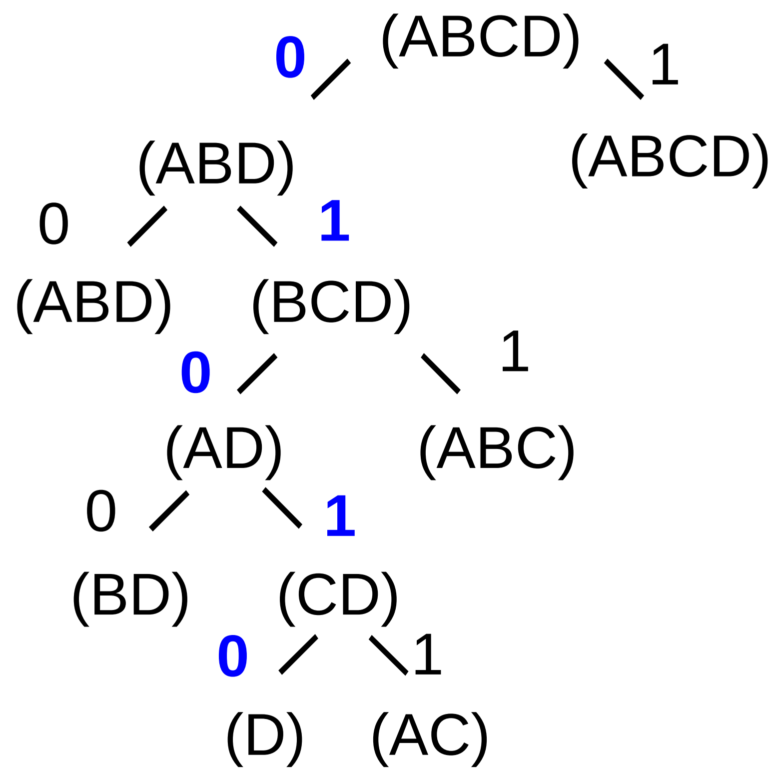
- *Synchronizing Sequence* = Input seq. such that *final state* is fixed
  - Regardless of initial state or output
- Example: FSM #1
  - SS is 01010, final state is D

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

PS=present state; NS=next state
x = input; z=output

NOTE:
1. Not every FSM has SS
2. SS may not be unique

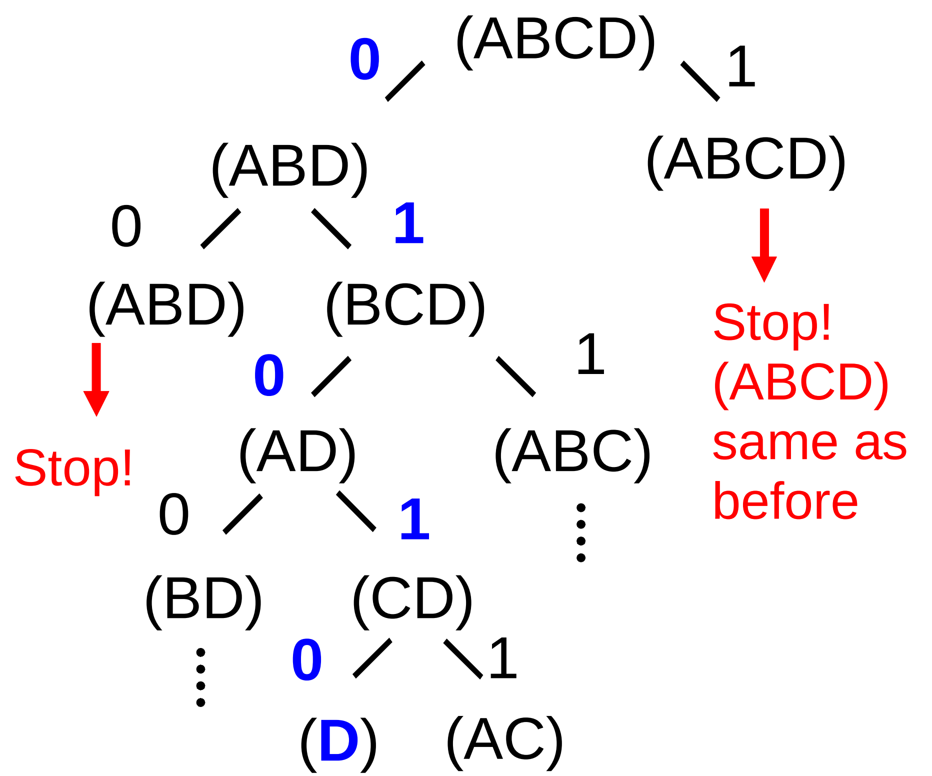
synchronizing tree



How to Derive SS?

- **Initially, root node has all sates together in one parenthesis**
  - **Branch downward, one input combination for each branch**
  - **Group all NS in parenthesis of child node**
- **Stop if same NS as some node in preceding level**
- **Repeat until only one state in parenthesis**

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



NOTE:
Tree size grows exponentially!

Quiz

Q1: Show the synchronizing tree of FSM#2.   What is SS?

Q2: Show that final state after SS is C

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Distinguishing Sequence (DS)

- *Distinguishing sequence* = Input seq. such that
  - Corresponding output sequence is different for each initial state
- Example: FSM#2, 101 is DS

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

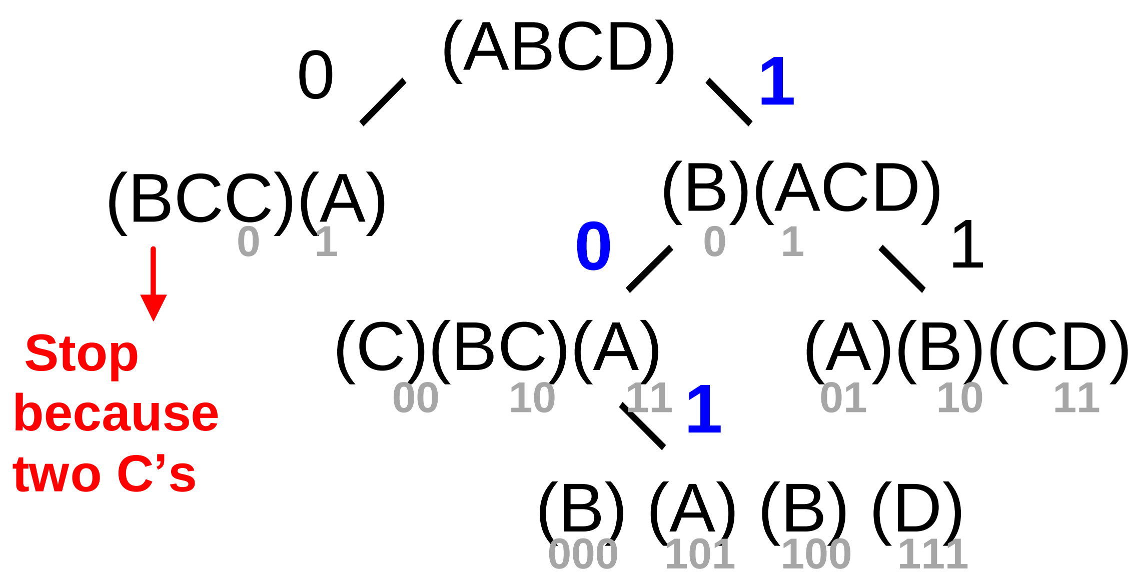
NOTE:
1. Not every FSM has DS
2. DS may not be unique

How to Derive DS?

- Initially, root node has all sates together in one parenthesis
  - Branch downward, one input combination for each branch
  - Group different output sequence in different parenthesis
- Stop if more than one identical NS in a parenthesis
- Repeat until every parenthesis contains only one NS

- Example: DS = 101

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

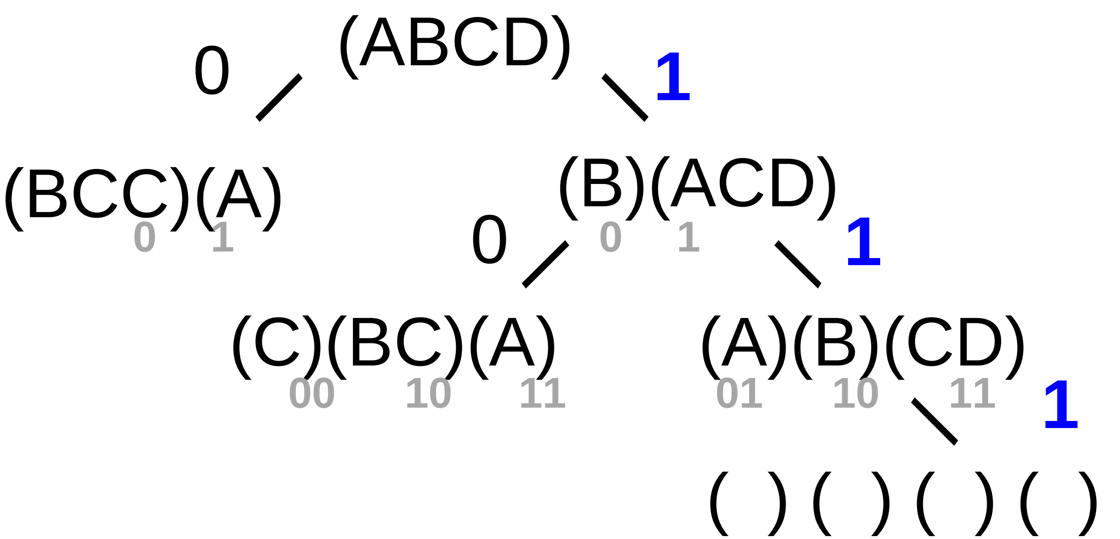


Quiz

Q: Show that 111 is also DS for FSM#2.

(Actually, 100, 101, 110, 111 are all DS.)

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1



A Sequence

- Goal: Verify existence of every state
  - Also verify states before and after DS
- How? Apply two DS to each state continuously
  - Observation of Z_i to identifies S_i
  - Observation of Z_{i+1} to identifies S_{i+1} , which is Q_i
- Notation:
  - S_i & Q_i indicate state before and after DS_i, respectively
  - Z_i indicates outputs when DS_i is applied
  - i = time index

Input		DS_i			DS_{i+1}	
State	S_i		Q_i	S_{i+1}		Q_{i+1}
output		Z_i			Z_{i+1}	

A Sequence (STEP 1)

- **1. Repeatedly apply DS, until**
  - 1A: DS DS has been applied continuously to all states, finish
  - **1B*: $Q_{i+1} = Q_i$* , continue to STEP2**
- 2. DS is applied once more
  - To verify state Q_{i+1}
- 3A: If DS DS applied to all states, finish
- 3B. Else, apply *Transfer Sequence* (TS)
- 4. Goto 1

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

X		101		101		101		1		101		101		1		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

Step 1 Step 2 Step 3 TS Step 1 Step 2 Step 3 TS Step 1 Step 2

A Sequence (STEP 2)

- 1. Repeatedly apply DS, until
  - 1A: DS DS has been applied continuously to all states, finish
  - 1B: $Q_{i+1} = Q_i$, continue to STEP2
- **2. DS is applied once more**
  - **To verify state Q_{i+1}**
- 3A: If DS DS applied to all states, finish
- 3B. Else, apply *Transfer Sequence* (TS)
- 4. Goto 1

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

X		101		101		101		1		101		101		1		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

Step 1 — **Step 2** — Step 3 TS — Step 1 — Step 2 — Step 3 TS — Step 1 — Step 2

A Sequence (STEP 3)

- 1. Repeatedly apply DS, until
  - 1A: DS DS has been applied continuously to all states, finish
  - 1B: $Q_{i+1} = Q_i$, continue to STEP2
- 2. DS is applied once more
  - To verify state Q_{i+1}
- 3A: If DS DS applied to all states, finish
- 3B. Else, apply *Transfer Sequence* (TS)
- 4. Goto 1

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

X		101		101		101		1		101		101		1		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

Step 1 | Step 2 | Step 3 TS | Step 1 | Step 2 | Step 3 TS | Step 1 | Step 2

A Sequence (STEP 4)

- 1. Repeatedly apply DS, until
  - 1A: DS DS has been applied continuously to all states, finish
  - 1B: $Q_{i+1} = Q_i$, continue to STEP2
- 2. DS is applied once more
  - To verify state Q_{i+1}
- 3A: If DS DS applied to all states, finish
- 3B. Else, apply *Transfer Sequence* (TS)
- 4. Goto 1

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

X		101		101		101		**1**		101		101		**1**		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

Step 1 | Step 2 | Step 3 TS | Step 1 | Step 2 | Step 3 TS | Step 1 | Step 2

A Sequence (FINISH)

- 1. Repeatedly apply DS, until
  - 1A: DS DS has been applied continuously to all states, finish
  - 1B: $Q_{i+1} = Q_i$, continue to STEP2
- 2. DS is applied once more
  - To verify state Q_{i+1}
- **3A: If DS DS applied to all states, finish**
- 3B. Else, apply *Transfer Sequence* (TS)
- 4. Goto 1

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

X		101		101		101		**1**		101		101		**1**		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

Step 1 | Step 2 | Step 3 TS | Step 1 | Step 2 | Step 3 TS | Step 1 | Step 2

Why Two DS Continuously?

- Because we need to verify not only initial state
  - But also final state after applying DS

X		101		101		101		**1**		101		101		**1**		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

C verified

B verified

also verify after DS C→B

after DS B→B

A verified

after DS A→A

D verified

But if stop here, don't know what state it is now

Need one more DS. after DS D→D

Quiz

Q: Find A sequence for the same FSM. Use DS = '111'. Starts from state C, end of synchronizing sequence.

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 1 1
A	z= 1 1 0 s= DCB
B	z= 1 1 1 s= ADC
C	z= 0 1 1 s= BAD
D	z= 1 0 1 s= CBA

X		111															
S	C																
Z		011															

B Sequence

- **Goal: Verify state transition**

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 0 1
A	z= 1 0 1 s= DBA
B	z= 1 0 0 s= ACB
C	z= 0 0 0 s= BCB
D	z= 1 1 1 s= CAD

- **Example: Use DS='101'**
  - **Starts from D, end of A seq.**
  - **Notation: *N(S, X) = Q* means next state for *S* with input *X* is *Q***

X		**0**		101		**0**		101		0		**0**		101		**0**		101		11		**1**		101		0		**1**		101
S	D		B		B		C		B		C		A		A		C		B		D		C		B		C		B	
Z		0		100		0		000				1		101		0		000				1		000				0		100
		N(D, 0)=B				N(B, 0)=C						N(C, 0)=A				N(A, 0)=C						N(D,1)=C						N(C,1)=B		

6 / 8 Transitions Verified

How about Other Transitions?

- ***N(B*,1)=*A*, *N(A*,1) =*D*** **already verified in A sequence**
  - **No need to verify again in B sequence**

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

X		101		101		101		**1**		101		101		**1**		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

N(B,1)=A **N(A,1)=D**

8 / 8 Transitions Verified

Whole Checking Experiment

- This example uses DS = 101
- Checking experiment is NOT unique
  - Other CE is OK, as long as all states and transitions verified
  - The shorter, the better

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

	Synchronizing Sequence	A-sequence	B-sequence
X:	01010	101 101 101 1 101 101 1 101 101	0 101 0 101 0 0 101 0 101 11 1 101 01 101
Expected Output:	Don’t care	000 100 100 1 101 101 1 111 111	0 100 0 000 0 1 101 0 000 11 1 000 00 100

- NOTE: The introduced procedure
  - 1. Does NOT guarantee shortest CE
  - 2. Need distinguishing sequence

Quiz

Q: (cont'd from last quiz)

Find B sequence for the same FSM.

Use DS = '111'.

Starts from state D, end of A sequence

(FSM#2) PS	NS, z	
	x = 0	x = 1
A	C, 0	D, 1
B	C, 0	A, 1
C	A, 1	B, 0
D	B, 0	C, 1

Init. State	apply DS x= 1 1 1
A	z= 1 1 0 s= DCB
B	z= 1 1 1 s= ADC
C	z= 0 1 1 s= BAD
D	z= 1 0 1 s= CBA

X																					
S	D																				

X																						
S																						

Summary

- Checking experiment exhaustively verify FSM
  - Independent of circuit implementation
- General procedure of checking experiment
  - Synchronizing sequence: fixed final state
  - A sequence: verify all states
  - B sequence: verify all state transitions
- Assumptions of checking experiment:
  - 1. No equivalent states (*i.e.* reduced FSM)
  - 2. Strongly connected FSM
  - 3. Defect Does not increase state of circuit

FFT

- **Q: at end of A sequence, how do we verify last state is indeed D?**

X		101		101		101		**1**		101		101		**1**		101		101	
S	C		B		B		B		A		A		A		D		D		D
Z		000		100		100		1		101		101		1		111		111	

C verified

B verified

we also verify after DS C→B

after DS B→B

A verified

after DS A→A

D verified

But if stop here, don't know what state it is now

after DS D→D