Guided Gate Level ATPG for Sequential Circuits using a High Level Test Generation Approach

Bijan Alizadeh, and Masahiro Fujita

VLSI Design and Education Center (VDEC),
University of Tokyo, Japan

January 2010

Outline

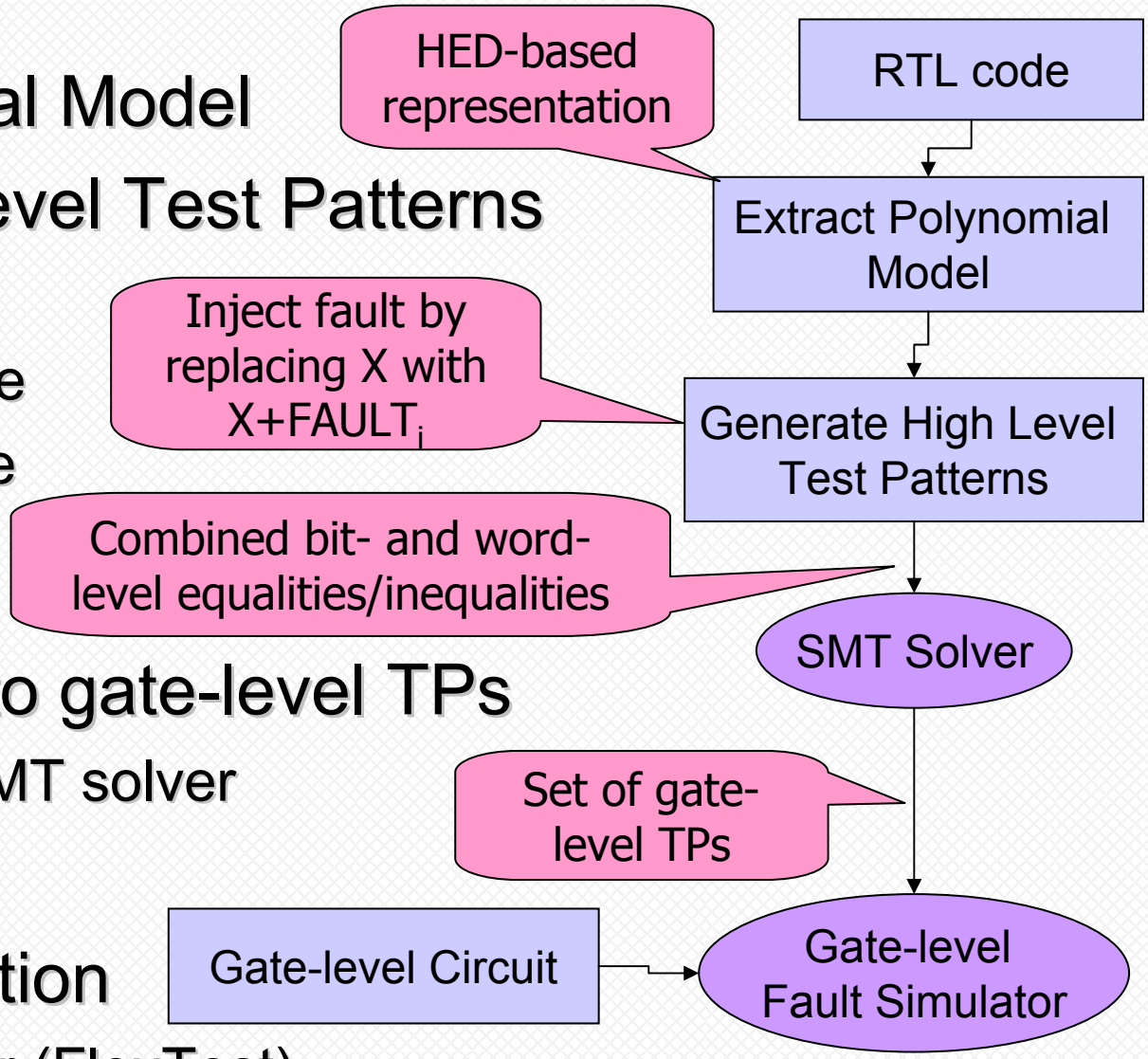
- Introduction
- Proposed high-level test generation
  - Functional test generation for behavioral codes
- How to extend it for covering RTL codes
  - Sequential designs
- Preliminary Experimental Results
- Future Works

Introduction

- Sequential ATPG is becoming important because
  - Scan-based designs have problems in
    - Performance
    - Area overhead
    - Over-testing
    - Power consumptions during testing
- However, the complexity of sequential ATPG is high
  - It is extremely important to make sequential ATPG more powerful

Our Methodology

- Extract Polynomial Model
- Generate High Level Test Patterns (HL-TPs)
  - Propagation Phase
  - Justification Phase
- Convert HL-TPs to gate-level TPs
  - Apply HL-TP to SMT solver
- Coverage Estimation
  - Run fault simulator (FlexTest)



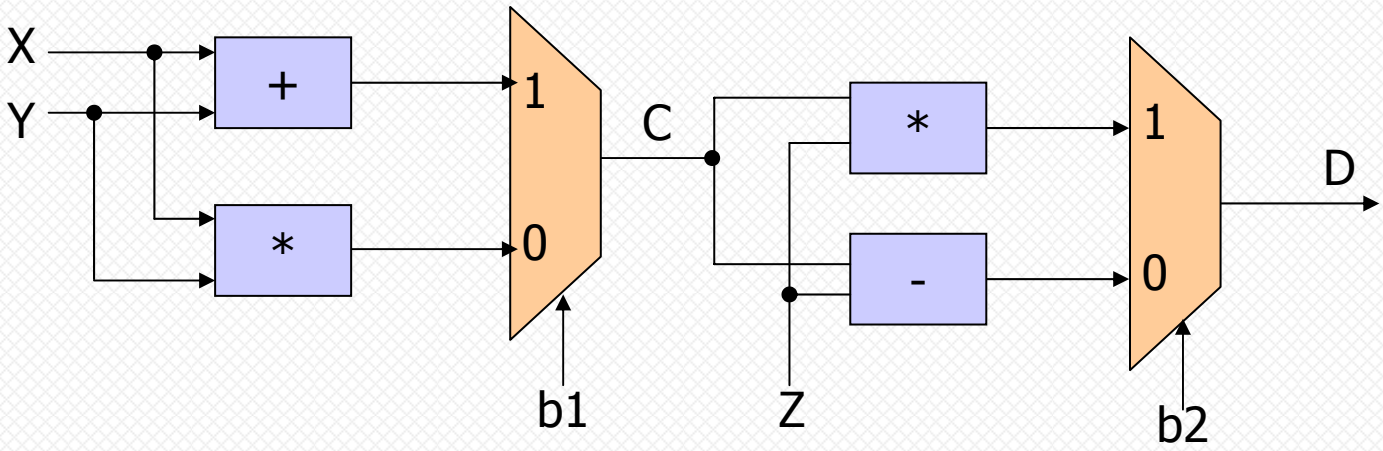
High Level Test Generation

- Suppose a high-level (behavioral) description is given

Primary Inputs: X, Y, Z
Intermediate Variable: C
Primary Output: D

```
1  IF ( b1 )        C = X+Y;
2  ELSE             C = X*Y;

3  IF ( b2 )        D = C*Z;
4  ELSE             D = C-Z;
```



High Level Test Generation

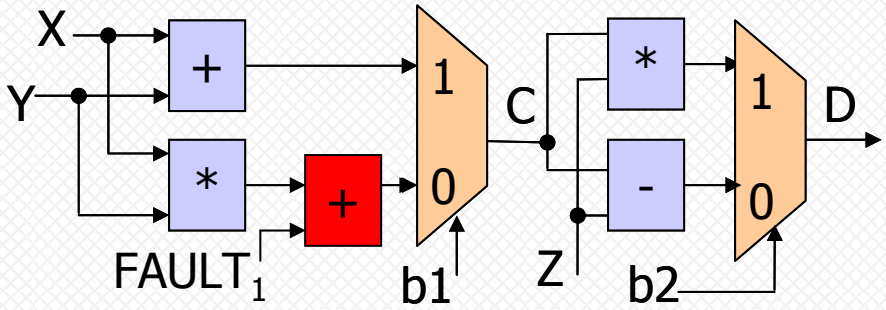
- Fault Injection (based on the following fault model)
  - Bit Failure
    - Stuck-at less than (-FAULT)
    - Stuck-at greater than (+FAULT)
  - Condition Failure
    - Stuck-at TRUE
    - Stuck-at FALSE

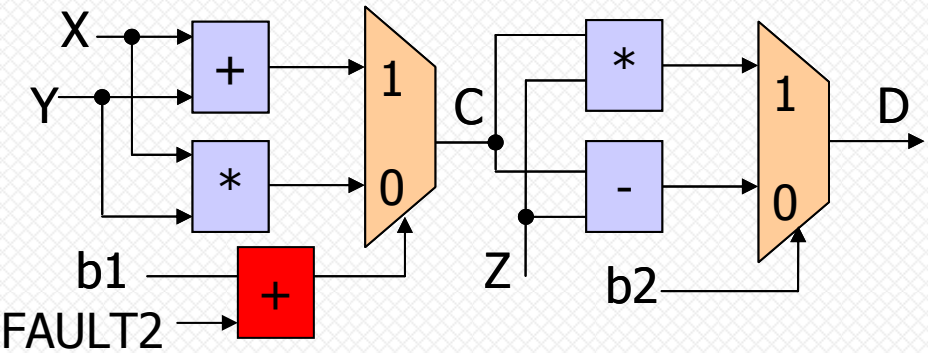
```
1  IF ( b1 )        C = X+Y;
2  ELSE             C = X*Y+FAULT1;

3  IF ( b2 )        D = C*Z;
4  ELSE             D = C-Z;
```

```
1  IF ( b1+FAULT2 )  C = X+Y;
2  ELSE              C = X*Y;

3  IF ( b2 )         D = C*Z;
4  ELSE              D = C-Z;
```



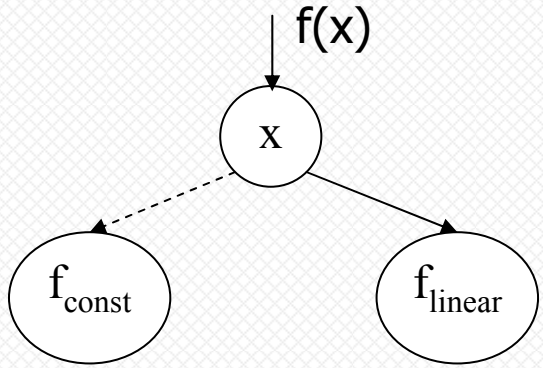


Horner Expansion Diagram (HED)

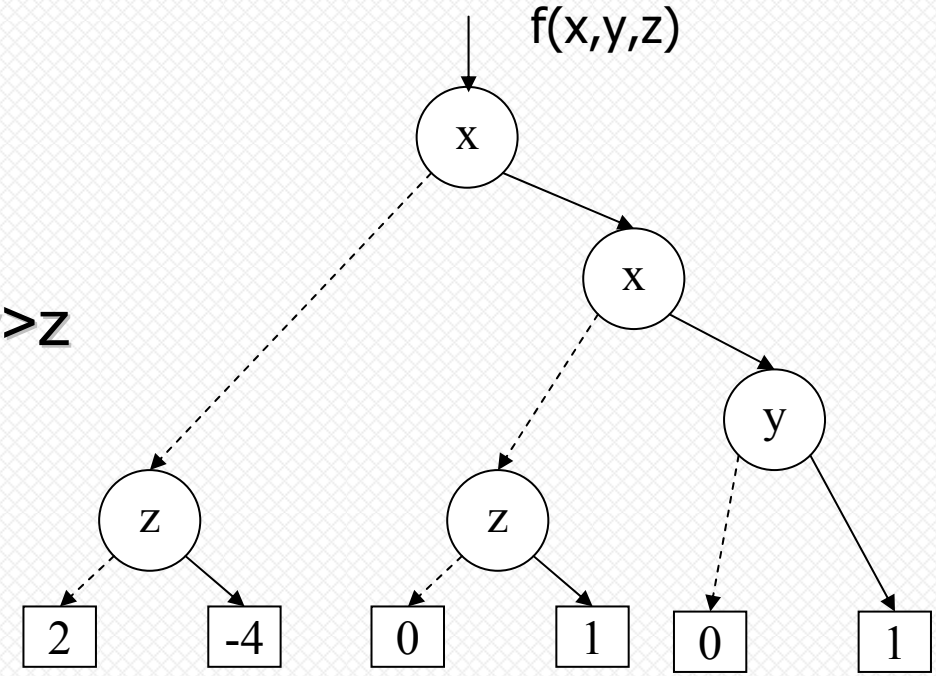
- Horner expansion:

$$f(x, y, \ldots) = f_{const} + x.f_{linear}$$

  - Const (Left) child (dashed line)
  - Linear (Right) child (solid line)

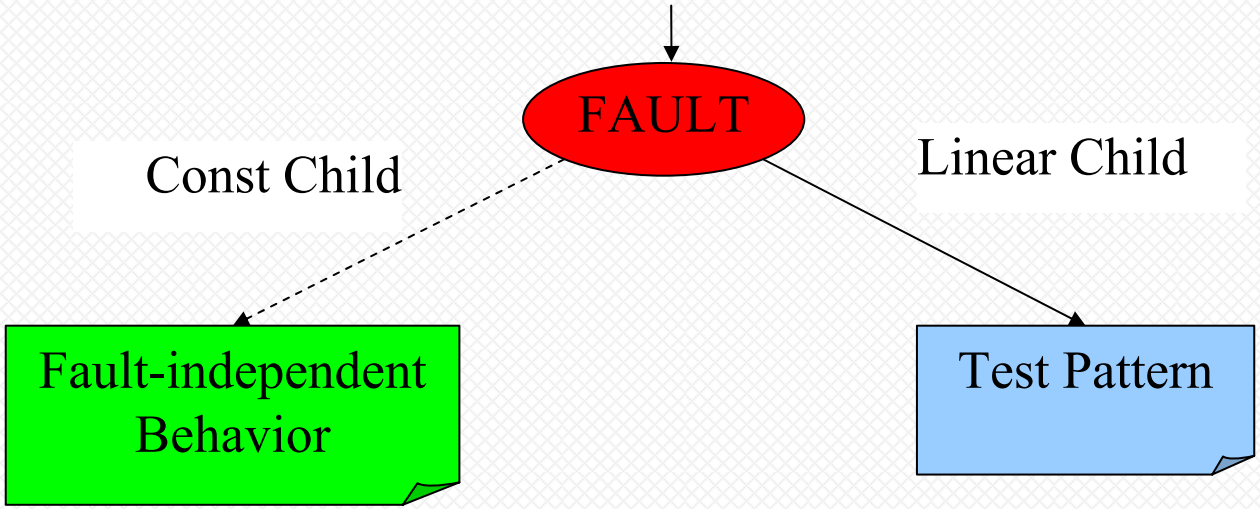


- Example
  - f(x,y,z) = $x^2y+xz-4z+2$
    - Variable ordering: x>y>z
  - Horner form
    - f = x(x(y)+z)+z(-4)+2



Useful Property for Test Generation

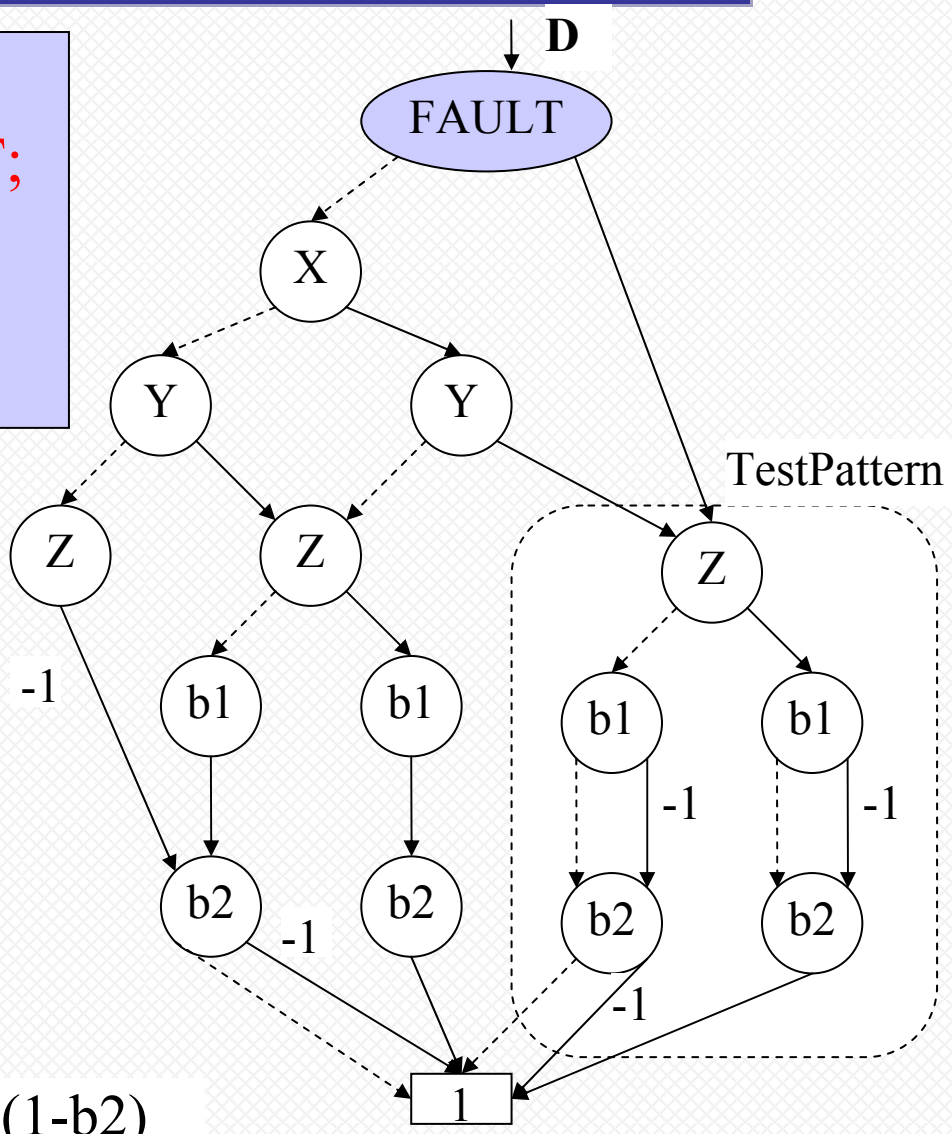
- High-level Test Generation
  - F(X,…) = const + X * linear
  - F(FAULT,…) = const + FAULT * linear



Example 1 - Single Fault


```
1  IF ( b1 )   C = X+Y;
2  ELSE        C = X*Y + FAULT;

3  IF ( b2 )   D = C*Z;
4  ELSE        D = C-Z;
```



(1) 1-b1 ≠ 0 & b2 ≠ 0 & Z ≠ 0

OR

(2) 1-b1 ≠ 0 & 1-b2 ≠ 0

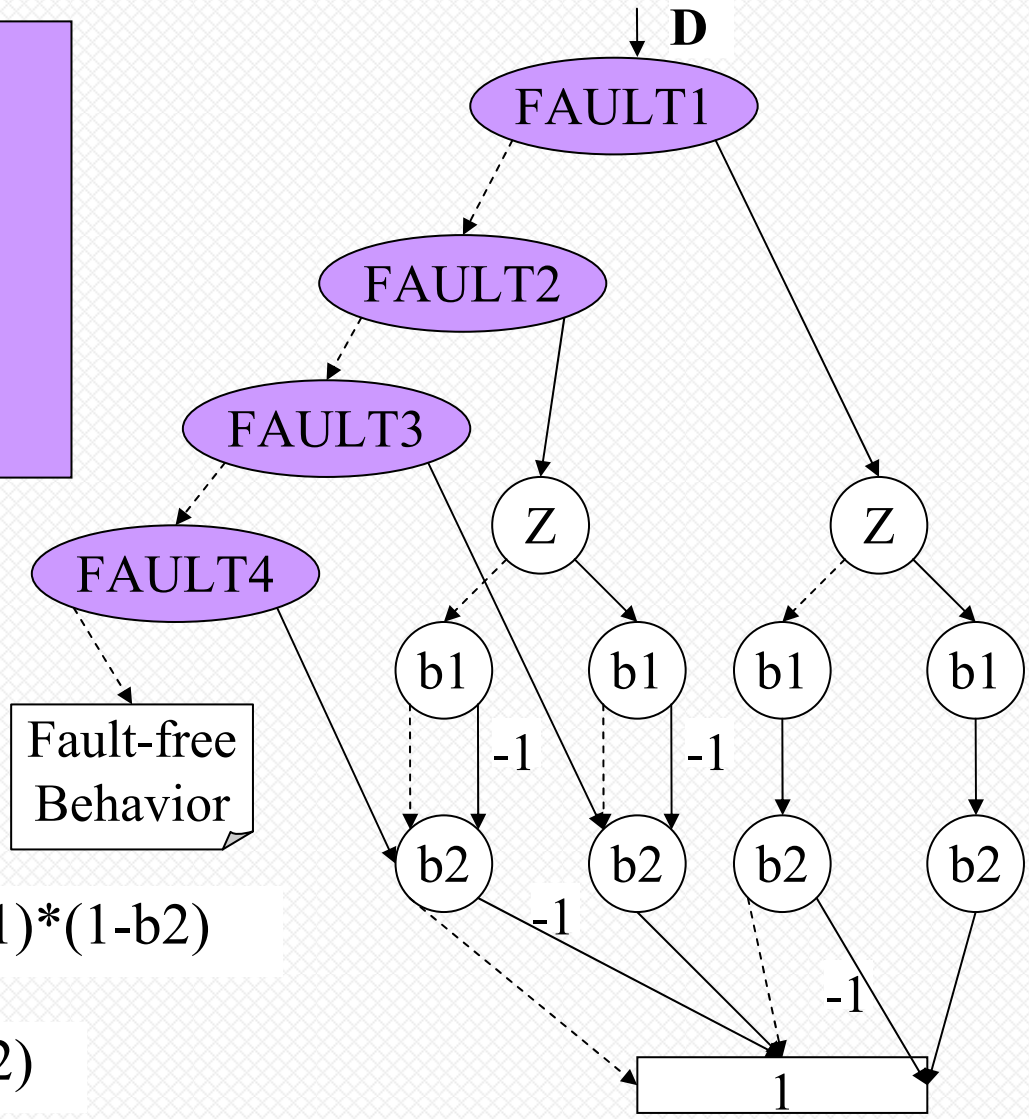
TestPattern ≠ 0

TestPattern = (1-b1)*b2*Z+(1-b1)*(1-b2)

Example 2 - Multiple Faults


```
1  IF ( b1 )      C = X+Y + FAULT1;
2  ELSE           C = X*Y + FAULT2;

3  IF ( b2 )      D = C*Z + FAULT3;
4  ELSE           D = C-Z + FAULT4;
```



TestPattern4 = (1-b2)

TestPattern3 = b2

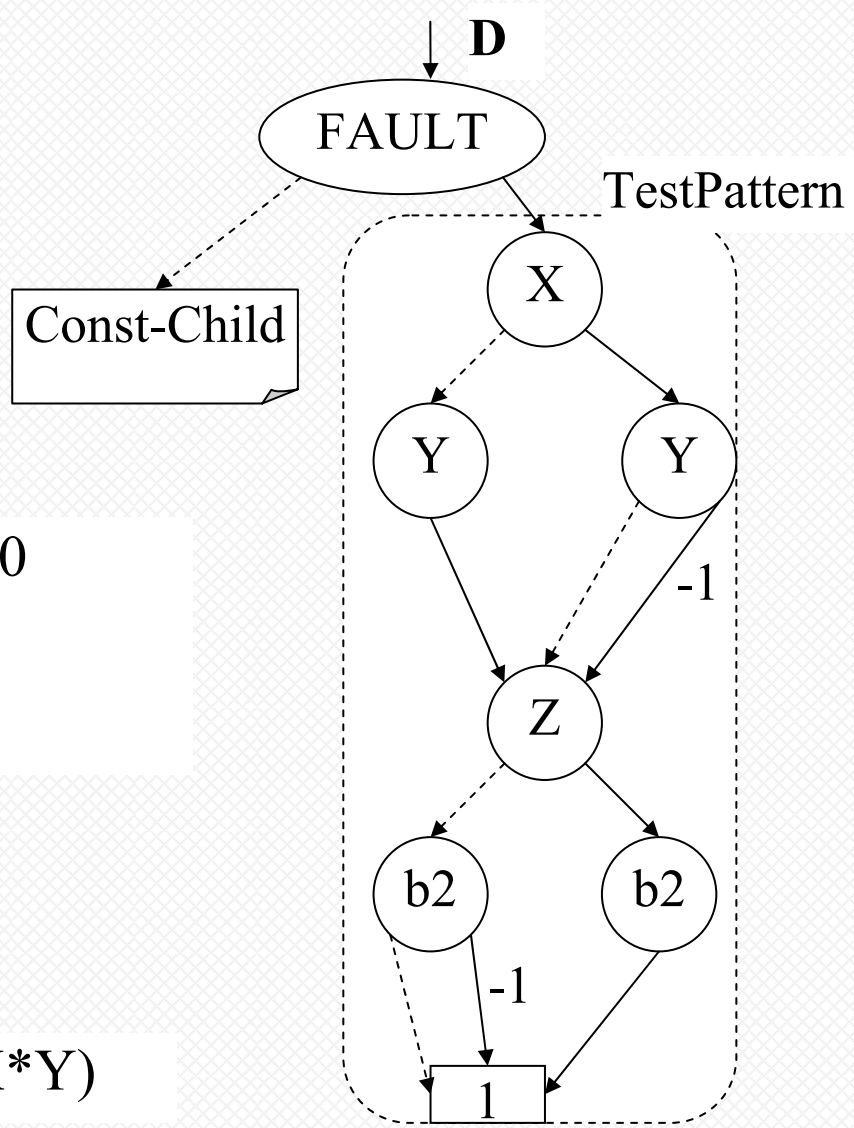
TestPattern2 = (1-b1)*b2*Z+(1-b1)*(1-b2)

TestPattern1 = b1*b2*Z+b1*(1-b2)

Example 3 - Condition Fault


```
1  IF ( b1+FAULT )      C = X+Y;
2  ELSE                 C = X*Y;

3  IF ( b2 )            D = C*Z ;
4  ELSE                 D = C-Z;
```



(1) $b2 \neq 0 \;\&\; Z \neq 0 \;\&\; X+Y-X*Y \neq 0$

OR

(2) $1-b2 \neq 0 \;\&\; X+Y-X*Y \neq 0$

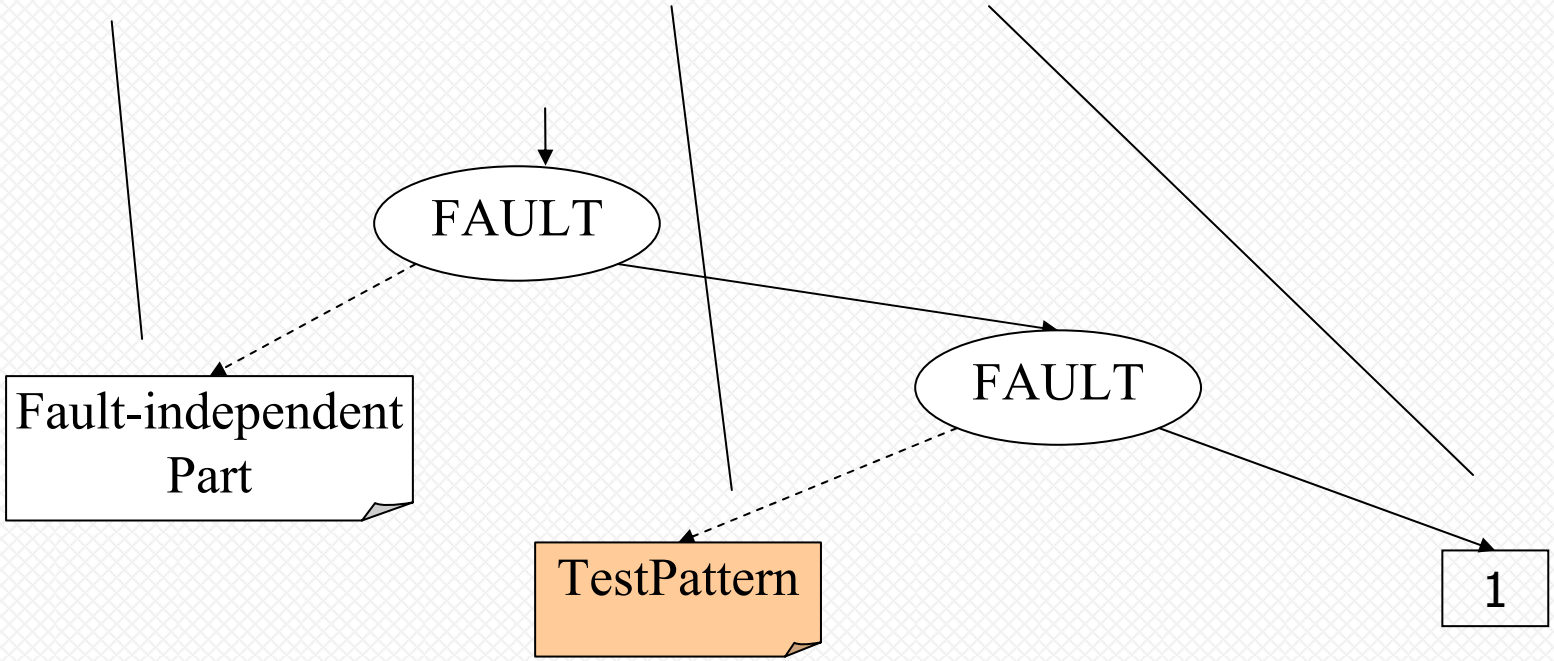
$\Uparrow$

$TestPattern \neq 0$

$TestPattern = ((1-b2)+b2*Z)*(X+Y-X*Y)$

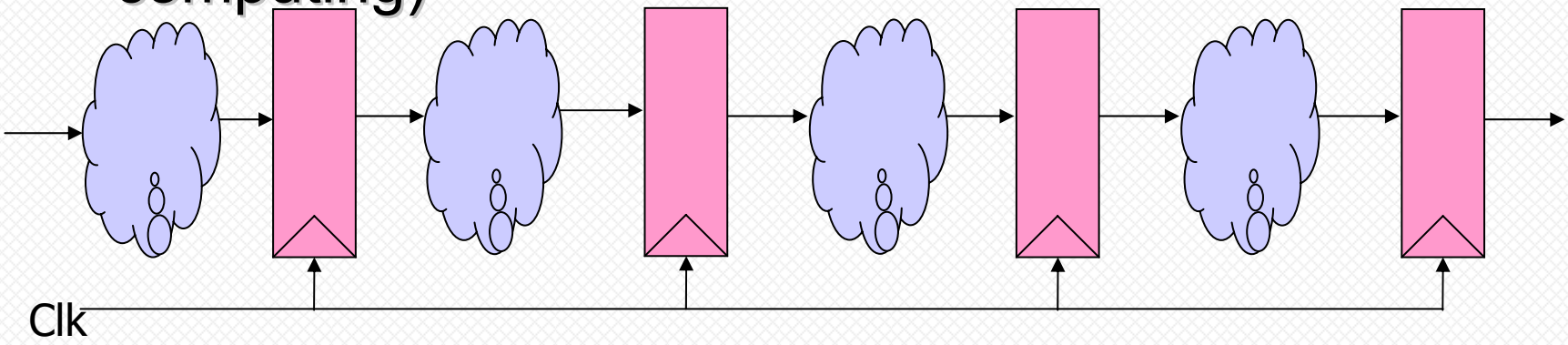
Non-linear term in HED

- If injected fault results in a non-linear term
  - $(f_1 + FAULT)*(f_2 + FAULT)$
    $= f_1*f_2 + (f_1 + f_2)*FAULT + FAULT^2$

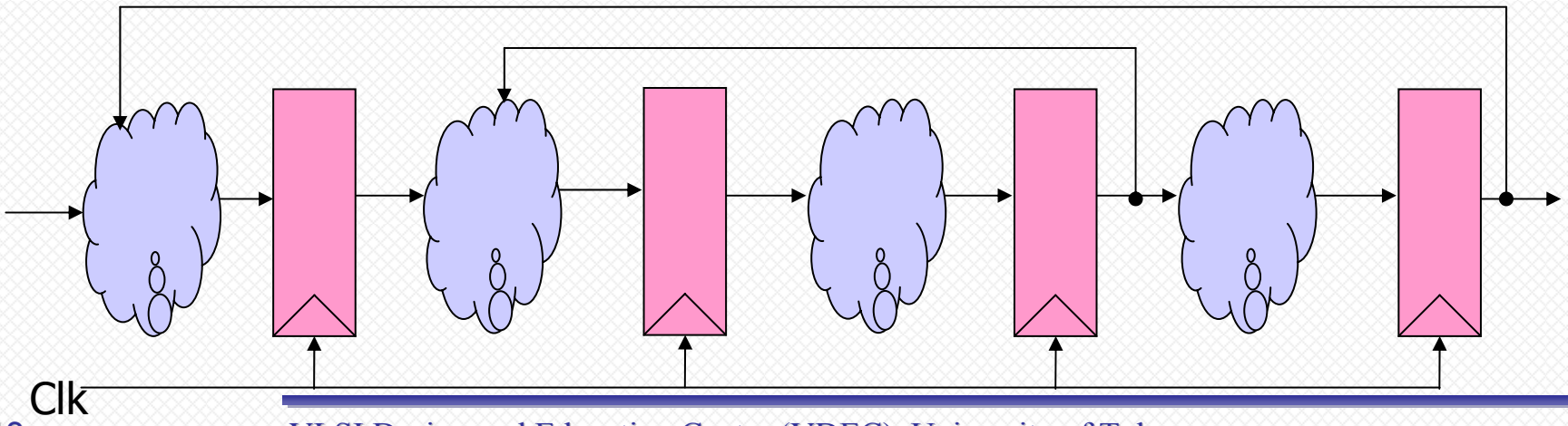


Register Transfer Level Test Generation

- How to extend our technique to RTL designs?
  - Pipelined design without feedback (High performance computing)



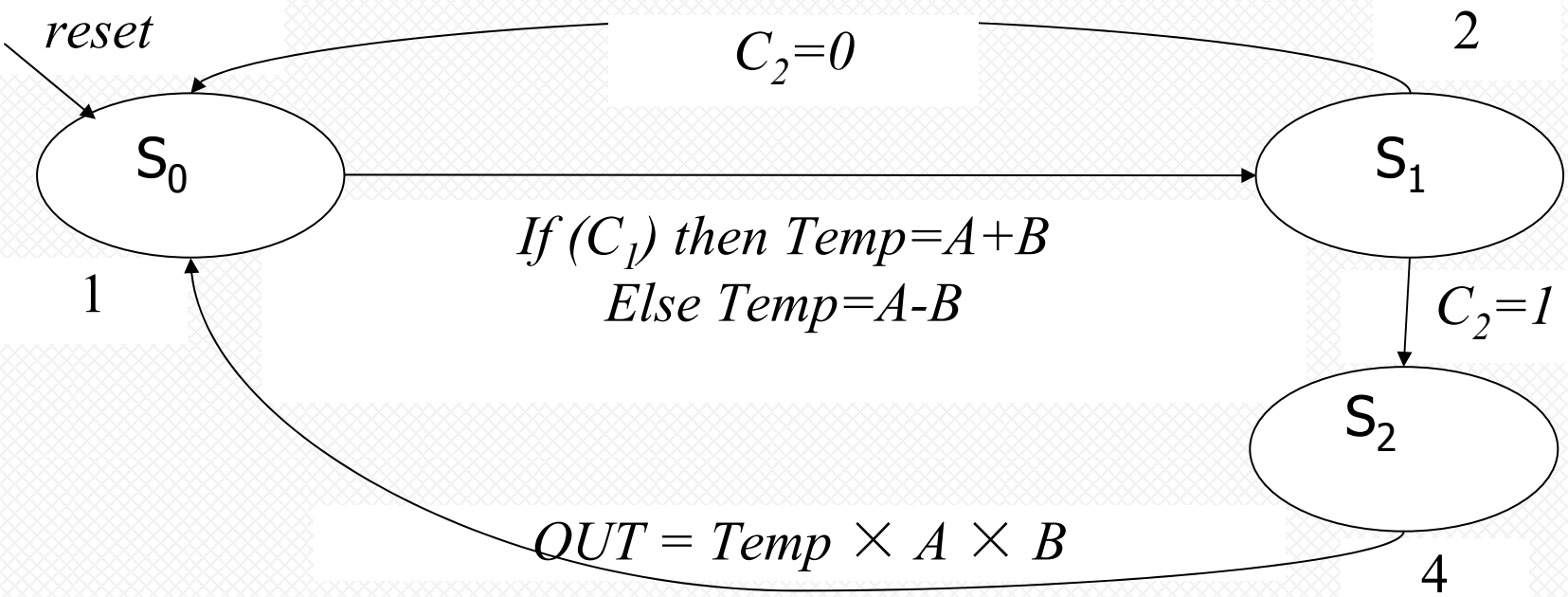
  - Sequential design with feedback



Polynomial Model of Sequential Circuits

- FSM with Datapath (FSMD)
  - I : set of primary inputs
  - S: finite set of control states
  - V: set of storage variables
  - Z: set of primary outputs
  - TRANS: S×AR → S : state transition function
  - OUT: S×AR → U : update function of the output and storage variables
- Where
  - U = {x <= e | x∈ Z ∪ V and e ∈ E }: set of storage and output assignments
  - E = set of arithmetic expressions of input and storage variables
  - AR = set of status signals as arithmetic relations between two expressions from E

Example (FSMD)



I = {A, B, C_1, C_2}; S = {S_0, S_1, S_2};

Z = {OUT}; V = {Temp};

E = {A+B, A-B, Temp×A×B}

AR = {C1=0, C1=1, C2=0, C2=1}

U = {Temp=A+B, Temp=A-B, OUT=Temp×A×B}

Transition and Output functions

- Suppose S = {S_0, S_1, …, S_{n-1}}
  - If PS=2^k, then P_n = 1
  - Else P_n = 0

One-hot Encoding $S_k = 2^k$

- PS = S_k

$$P_n(PS,k) = \frac{\prod_{i=0}^{n-1}(PS - 2^i)}{\prod_{i=0}^{n-1}(2^k - 2^i)} \quad where \quad i \neq k$$

- Transition Function

$$transfunc(PS,I) = \sum_{k=0}^{n-1} P_n(PS,k) \times ftrans(PS,I,k)$$

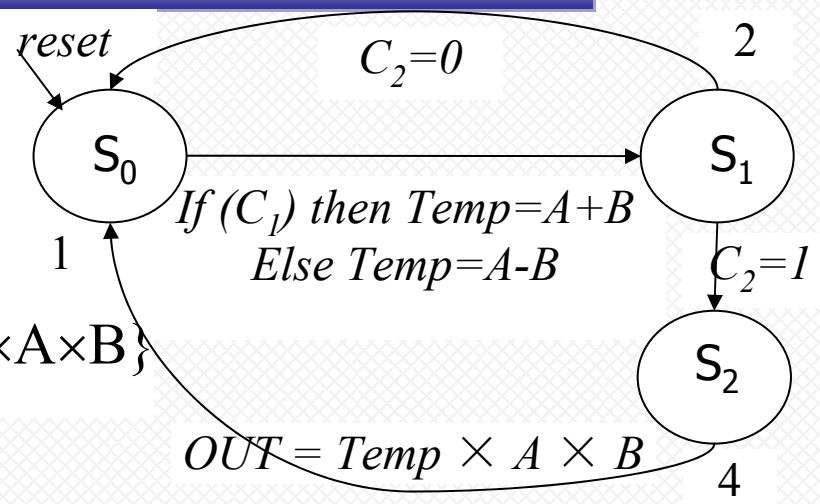
- Output Function

$$outfunc(PS,I) = \sum_{k=0}^{n-1} P_n(PS,k) \times fout(PS,I,k)$$

Example (FSMD)

Computer Aided Design

I = {A, B, C_1, C_2}; S = {S_0, S_1, S_2};
Z = {OUT}; V = {Temp};
E = {A+B, A-B, Temp×A×B}
AR = {C1=0, C1=1, C2=0, C2=1}
U = {Temp=A+B, Temp=A-B, OUT=Temp×A×B}



$$PS \equiv 1: \; P_3(PS,0) = (PS-2)(PS-4)/3$$

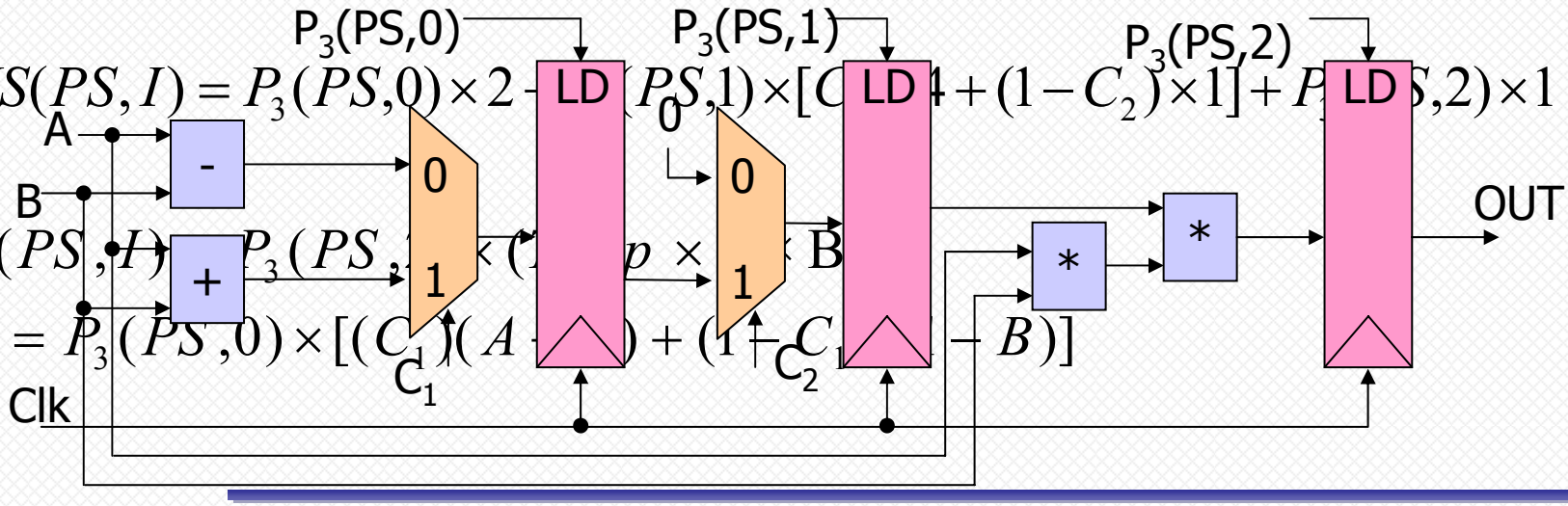
$$PS \equiv 2: \; P_3(PS,1) = (PS-1)(PS-4)/(-2)$$

$$PS \equiv 4: \; P_3(PS,2) = (PS-1)(PS-2)/6$$

$$TRANS(PS,I) = P_3(PS,0)\times 2 + P_3(PS,1)\times[C_1\times 4 + (1-C_2)\times 1] + P_3(PS,2)\times 1$$

$$OUT(PS,I) = P_3(PS,1)\times(Temp \times A \times B)$$

$$Temp = P_3(PS,0)\times[(C_1)(A-B) + (1-C_1)(A+B)]$$



RT Level Test Generation

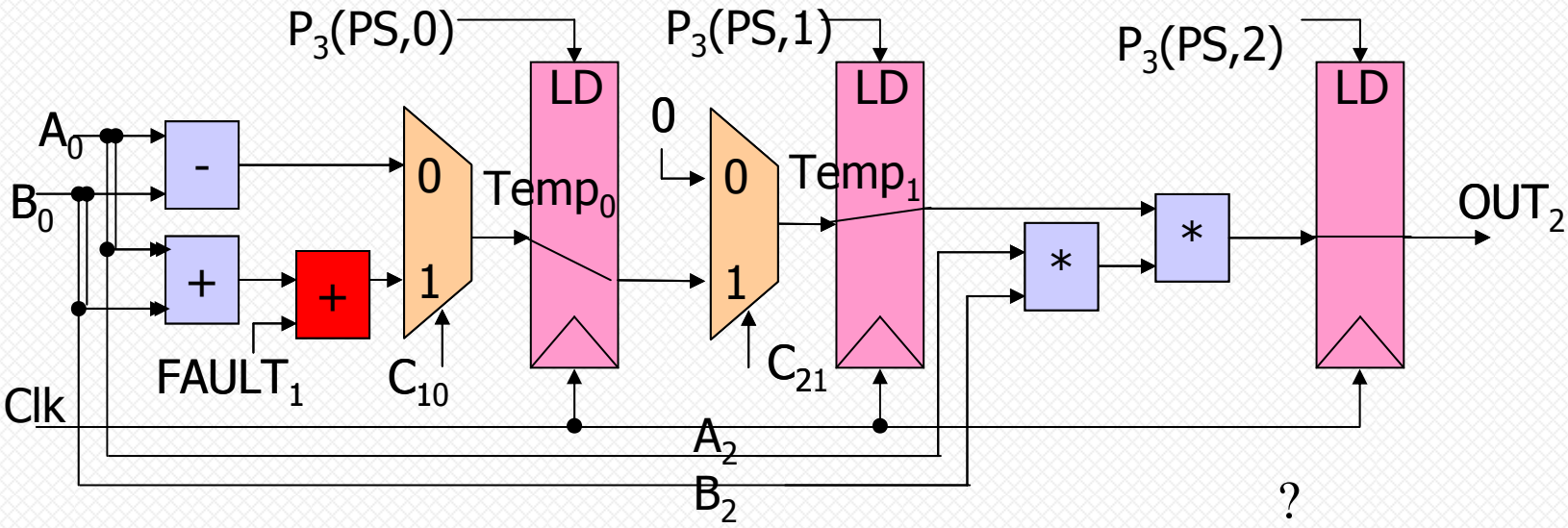
- Propagation Phase
  - Move all registers to the periphery of the design (Time Frame Expansion)
    - by removing state conditions and
    - considering time slices instead
  - Faulty variable needs to be reached to one of the primary outputs
- Justification Phase
  - Determine the status of signals modified by the transition function (not output function)
    - Pre-image is computed until an initial state is reached

Example- Fault Propagation Phase

- A+B unit is affected by $FAULT_1$

$$OUT\,(PS, I) = P_3(PS, 2) \times (Temp \times A \times \mathrm{B})$$

$$Temp = P_3(PS, 0) \times [(C_1)(A + B) + (1 - C_1)(A - B)]$$



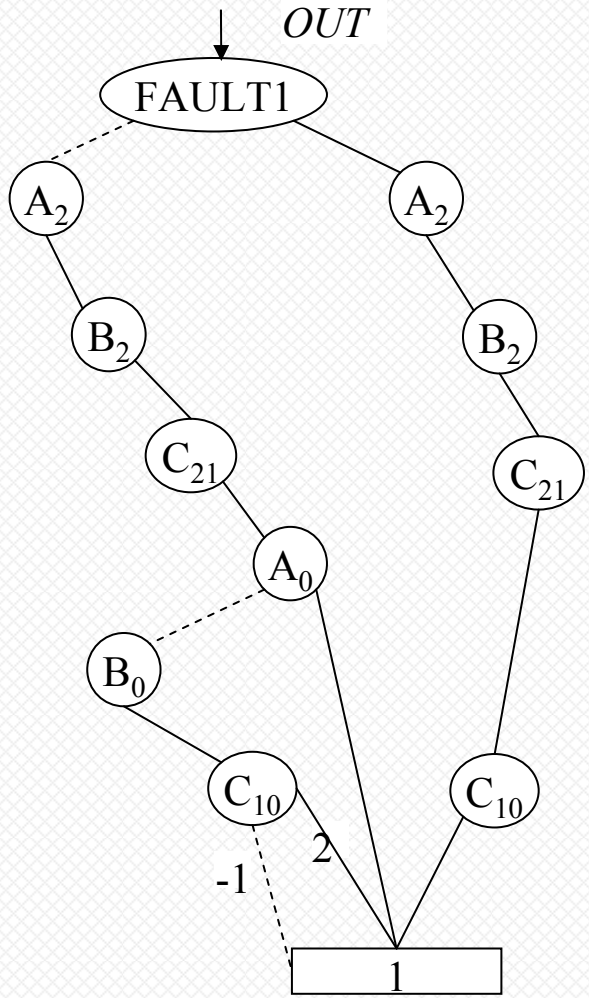
$$OUT = Temp_1 \times A_2 \times \mathrm{B}_2$$

$$Temp_0 = (C_{10})(A_0 + B_0) + (1 - C_{10})(A_0 - B_0)$$

$$Temp_1 \overset{?}{=} f(Temp_0)$$

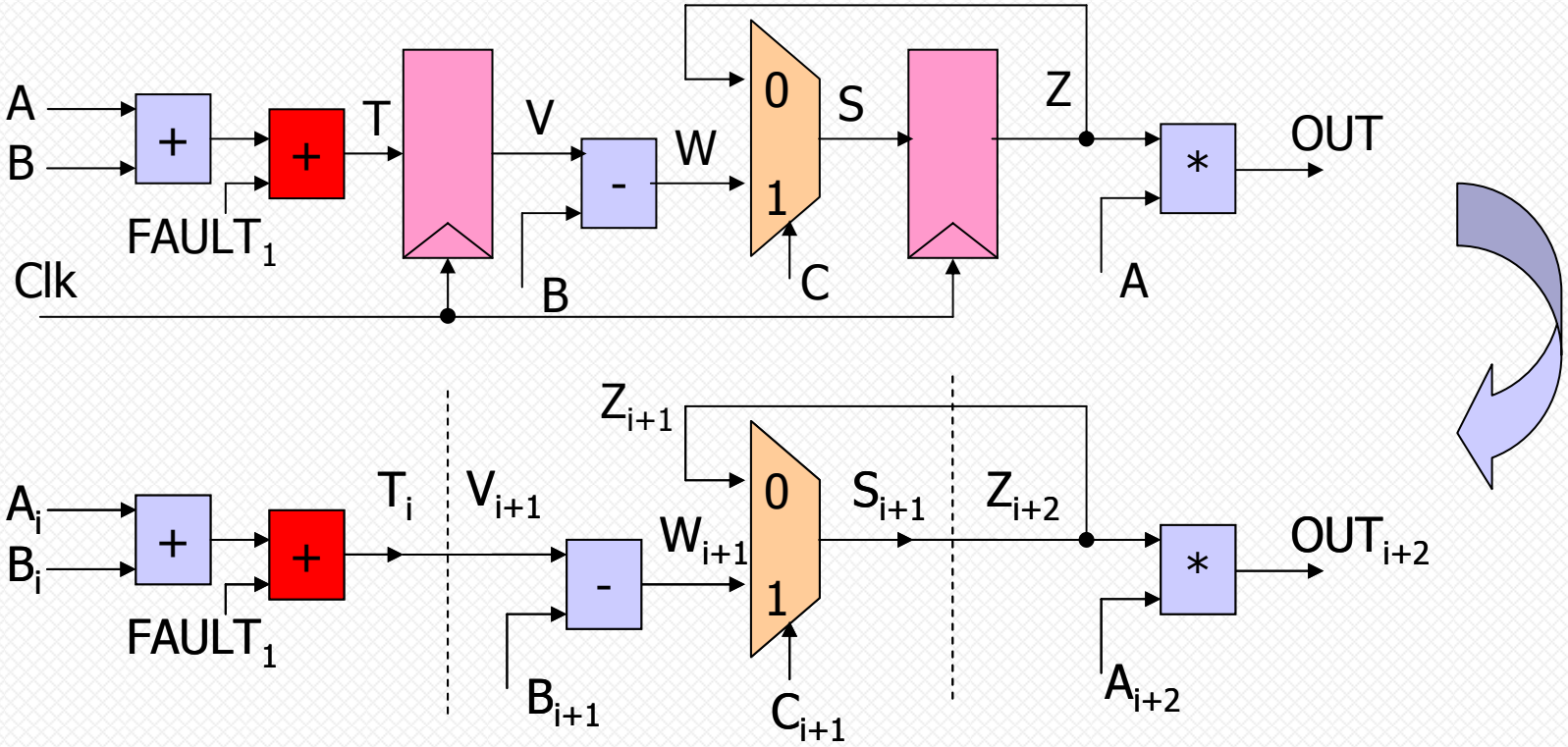
Example-Justification Phase

- Determine the status of signals modified by the transition function not output function
  - Pre-image is computed until an initial state is reached
    - First pre-image computation
      - $P_3(PS,2) \rightarrow P_3(PS,1)$
    - Second pre-image computation
      - $P_3(PS,1) \rightarrow P_3(PS,0)$: if ($C_2$=1)
      - $C_{21} = 1$
    - $Temp_1 = C_{21} * Temp_0$



$$OUT_2 = A_2 \times B_2 \times C_{21} \times [(C_{10})(A_0 + B_0 + FAULT_1) + (1 - C_{10})(A_0 - B_0)]$$

How to deal with sequential loops



$$T_i = A_i + B_i + FAULT_1$$

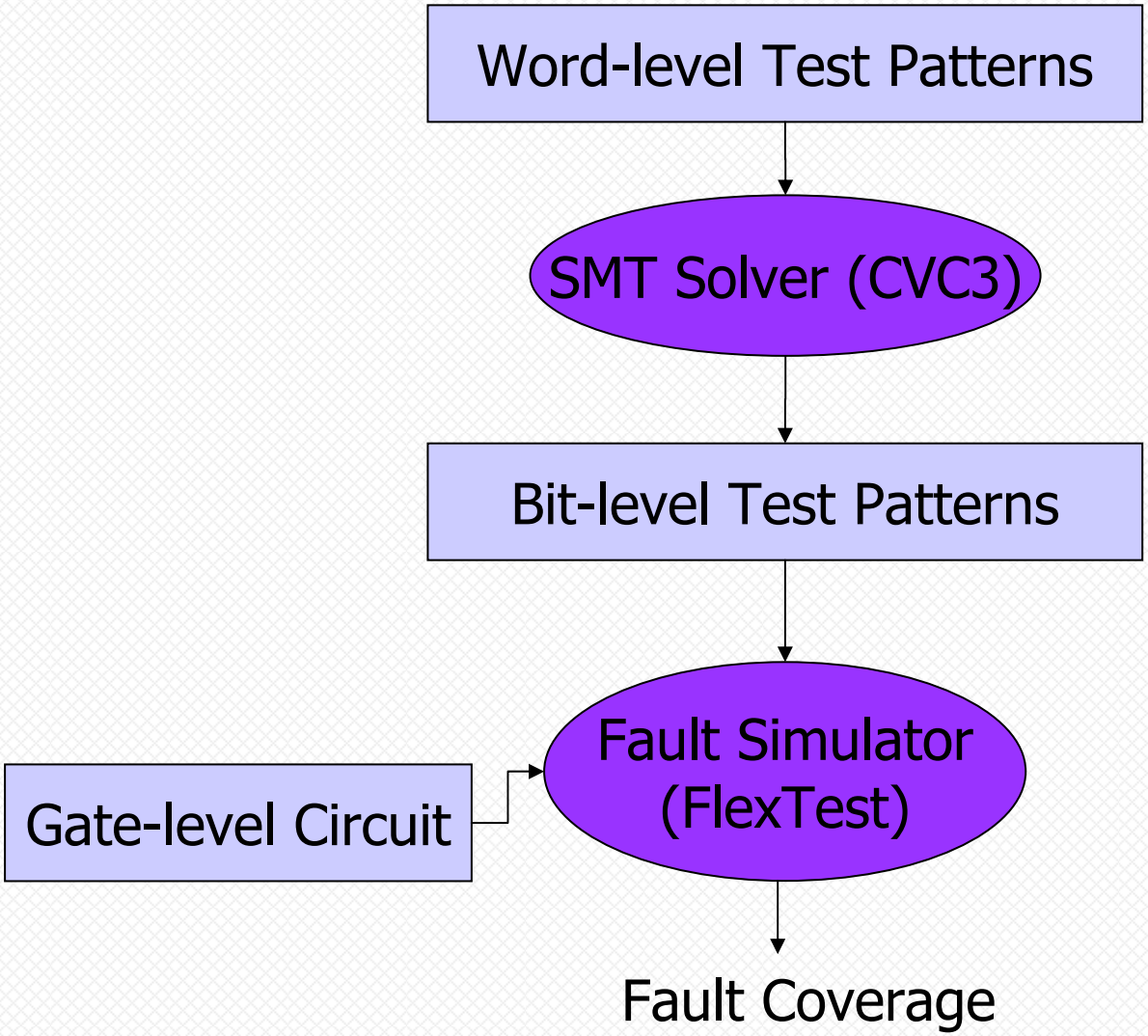
$$V_{i+1} = T_i$$

$$W_{i+1} = V_{i+1} - B_{i+1}$$

$$S_{i+1} = C_{i+1} * W_{i+1} + (1-C_{i+1}) * Z_{i+1}$$

$$Out_{i+2} = Z_{i+2} * A_{i+2}$$

Gate Level Test Generation



Experimental Results

Computer Aided Design

- A sub-set of ITC'99 benchmarks

Bench mark	VHDL Spec	Gate-level Spec					
	#lines	PI	PO	CG	FF	TF	CF
b03	141	4	4	149	30	832	599
b04	102	11	8	797	66	2846	2048
b05	332	1	36	812	34	2970	1759
b06	128	2	6	36	8	358	229
b07	92	1	8	415	41	1942	1313

Experimental Results

Computer Aided Design

- ARTIST: RTL ATPG tool developed at University of Torino, Italy
- After considering platform differences
- Not available for other techniques

Bench mark	ARTIST		HTest		HED-based Approach			
	FC (%)	Time (sec)	FC (%)	Time (sec)	FC (%)	Time (sec)	#TVec	#Paths
b03	74.33	5131	78.0	814	81.04	4.3	112	18
b04	89.42	6905	89.6	768	94.14	4.1	245	12
b05	33.50	33393	48.0	23241	80.0	8.3	812	41
b06	97.02	2315	97.9	535	96.84	3.8	145	29
b07	57.53	2251	70.8	538	88.11	3.9	318	26

Experimental Results

- Large-datapath circuits
  - Modified b03 and b06
    - Added more 32-bit multipliers

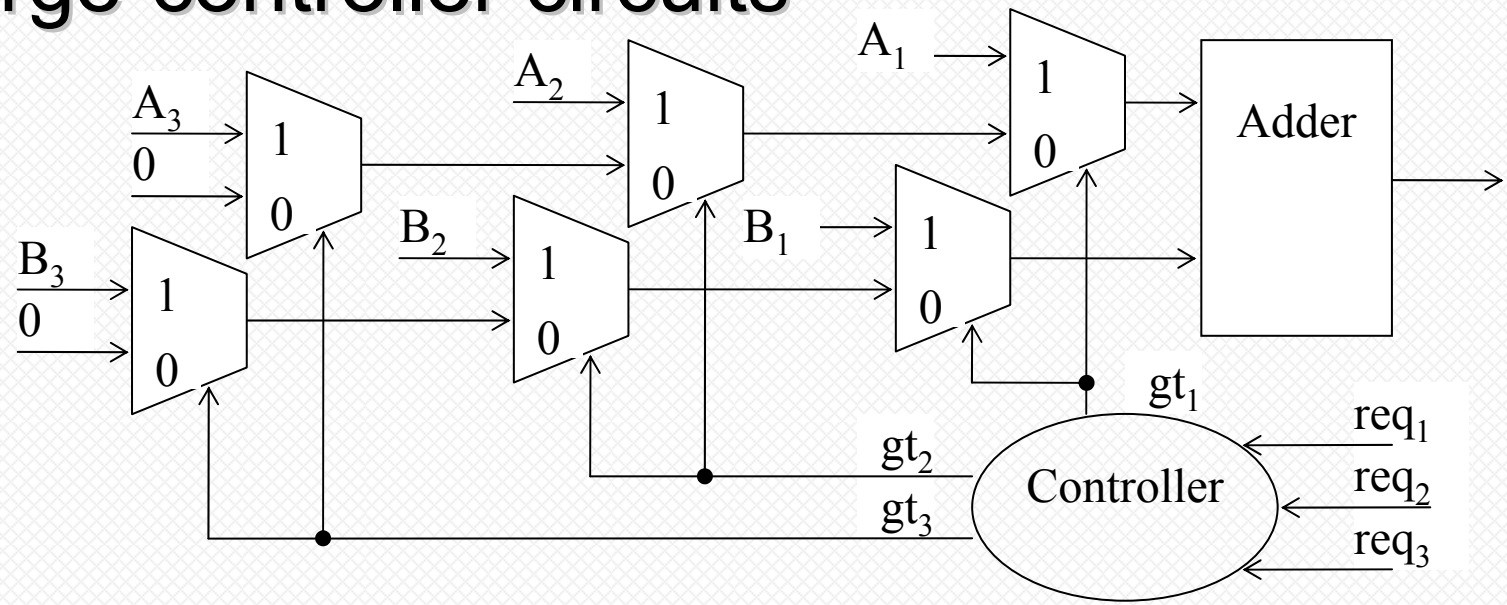
Bench-marks	#Mul	F.C. (%)	Time (Sec)	#TVec	#TF	#TG
b06_1	2	99.30	4.1	512	14890	5313
b06_2	4	99.05	4.2	608	15008	5444
b03_1	2	98.12	4.7	640	15580	5482
b03_2	4	98.04	4.9	704	15700	5619

- Intel@ 933MHz, Main memory: 256MB, OS: WinXP

Experimental Results

Computer Aided Design

- Large-controller circuits



Benchmarks	#gates	#TF	F.C.(%)	Time (Sec)	#TVec
b03_2	237	766	84.45	4.4	240
b03_3	299	1000	89.52	4.4	254
b03_4	350	1181	85.27	4.5	262
b03_5	410	1389	72.66	4.5	271

Conclusion and Future Work

- A high-level test generation technique guided gate-level ATPG
  - Generate high level test patterns using HED
  - Apply them to SMT-solvers as constraints to solve SAT-based ATPG problem more efficient
  - Test generation time
    - Almost 2 orders of magnitude faster than HTest
    - Almost 3 orders of magnitude faster than ARTIST
  - Fault coverage
    - 14% improvement compared to HTest
    - 25% improvement compared to ARTIST
- How to automate all phases in our methodology
- Compare our methodology with commercial ATPG tools at pure gate level
- Investigate the effect of synthesis constraints