Reading Assignment

- This lecture: 4.1, 4.2
- Next lecture: 4.2 – 4.4
Homework 2

- Due: 5:00pm 02/25 Chicago time
- Submit on Blackboard.
Outline

Logical Fault Models

Fault Detection for Combinational Circuits

Detectability and Redundancy
Physical Faults and Logical Faults

- Physical faults affect the systems in different ways.
  - Affect functionality (logical).
  - Affect timing (delay).
- We are interested in logical faults, which represent the effect of physical faults on system functionality in terms of logic.
- Advantages of considering logical faults
  - Reduce complexity compared to physical faults.
  - Independent of technology.
  - Applicable when the underlying physical fault models are not well-understood.
Physical Faults and Logical Faults

- Physical faults affect the systems in different ways.
  - Affect functionality (logical).
  - Affect timing (delay).
- We are interested in logical faults, which represent the effect of physical faults on system functionality in terms of logic.
- Advantages of considering logical faults
  - Reduce complexity compared to physical faults.
  - Independent of technology.
  - Applicable when the underlying physical fault models are not well-understood.
Physical faults affect the systems in different ways.
  ▶ Affect functionality (logical).
  ▶ Affect timing (delay).

We are interested in logical faults, which represent the effect of physical faults on system functionality in terms of logic.

Advantages of considering logical faults
  ▶ Reduce complexity compared to physical faults.
  ▶ Independent of technology.
  ▶ Applicable when the underlying physical fault models are not well-understood.
Logical Faults and System Models

- Logical fault modeling is closely related to system models.
  - The logic function of a system is determined by the system model and a given logical fault.

- Structural faults.
  - Modifications of interconnects.

- Functional faults.
  - Modifications of components (logic operations).
Logical fault modeling is closely related to system models.
  ▶ The logic function of a system is determined by the system model and a given logical fault.

Structural faults.
  ▶ Modifications of interconnects.

Functional faults.
  ▶ Modifications of components (logic operations).
Logical Faults and System Models

- Logical fault modeling is closely related to system models.
  - The logic function of a system is determined by the system model and a given logical fault.
- Structural faults.
  - Modifications of interconnects.
- Functional faults.
  - Modifications of components (logic operations).
Occurrence of Logical Faults

- We consider permanent faults instead of intermittent and transient faults.
- Single-fault assumption: one logical fault at a time.
  - Assume it is very unlikely to have multiple faults between two consecutive tests by the frequent testing strategy.
  - Even if multiple faults happen at the same time, they may be detected by the tests designed for the individual faults.
- A multiple fault may happen,
  - A set of faults happen at the same time.
  - A single physical fault may manifest itself as a multiple logical fault.
  - The first testing need to handle multiple faults at the same time.
  - Usually, a test cannot detect every single fault so a multiple fault may happen later.
Occurrence of Logical Faults

- We consider permanent faults instead of intermittent and transient faults.
- Single-fault assumption: one logical fault at a time.
  - Assume it is very unlikely to have multiple faults between two consecutive tests by the frequent testing strategy.
  - Even if multiple faults happen at the same time, they may be detected by the tests designed for the individual faults.
- A multiple fault may happen,
  - A set of faults happen at the same time.
  - A single physical fault may manifest itself as a multiple logical fault.
  - The first testing need to handle multiple faults at the same time.
  - Usually, a test cannot detect every single fault so a multiple fault may happen later.
Occurrence of Logical Faults

- We consider permanent faults instead of intermittent and transient faults.

- Single-fault assumption: one logical fault at a time.
  - Assume it is very unlikely to have multiple faults between two consecutive tests by the frequent testing strategy.
  - Even if multiple faults happen at the same time, they may be detected by the tests designed for the individual faults.

- A multiple fault may happen,
  - A set of faults happen at the same time.
  - A single physical fault may manifest itself as a multiple logical fault.
  - The first testing need to handle multiple faults at the same time.
  - Usually, a test cannot detect every single fault so a multiple fault may happen later.
Structural Fault Models

- Assume components are fault-free.
  - We assume faults in components can be modeled by faults on interconnects.
- Interconnects are affected.
  - Shorts: connecting points that shouldn’t.
  - Opens: breaking of a connection.
- Stuck faults: signal being stuck at a value.
  - s-a-0: shorts between signal and GND.
  - s-a-1: shorts between signal and VDD.
- Bridging faults: shorts between signals.
  - AND bridging fault: every signal is set the AND of all.
  - OR bridging fault: every signal is set the OR of all.
- Opens are modeled as stuck-at faults.
Structural Fault Models

- Assume components are fault-free.
  - We assume faults in components can be modeled by faults on interconnects.
- Interconnects are affected.
  - Shorts: connecting points that shouldn’t.
  - Opens: breaking of a connection.
- Stuck faults: signal being stuck at a value.
  - s-a-0: shorts between signal and GND.
  - s-a-1: shorts between signal and VDD.
- Bridging faults: shorts between signals.
  - AND bridging fault: every signal is set the AND of all.
  - OR bridging fault: every signal is set the OR of all.
- Opens are modeled as stuck-at faults.
Structural Fault Models

- Assume components are fault-free.
  - We assume faults in components can be modeled by faults on interconnects.
- Interconnects are affected.
  - Shorts: connecting points that shouldn’t.
  - Opens: breaking of a connection.
- Stuck faults: signal being stuck at a value.
  - s-a-0: shorts between signal and GND.
  - s-a-1: shorts between signal and VDD.
- Bridging faults: shorts between signals.
  - AND bridging fault: every signal is set the AND of all.
  - OR bridging fault: every signal is set the OR of all.
- Opens are modeled as stuck-at faults.
Structural Fault Models

- Assume components are fault-free.
  - We assume faults in components can be modeled by faults on interconnects.
- Interconnects are affected.
  - Shorts: connecting points that shouldn’t.
  - Opens: breaking of a connection.
- Stuck faults: signal being stuck at a value.
  - s-a-0: shorts between signal and GND.
  - s-a-1: shorts between signal and VDD.
- Bridging faults: shorts between signals.
  - AND bridging fault: every signal is set the AND of all.
  - OR bridging fault: every signal is set the OR of all.
- Opens are modeled as stuck-at faults.
Structural Fault Models

- Assume components are fault-free.
  - We assume faults in components can be modeled by faults on interconnects.
- Interconnects are affected.
  - Shorts: connecting points that shouldn’t.
  - Opens: breaking of a connection.
- Stuck faults: signal being stuck at a value.
  - s-a-0: shorts between signal and GND.
  - s-a-1: shorts between signal and VDD.
- Bridging faults: shorts between signals.
  - AND bridging fault: every signal is set the AND of all.
  - OR bridging fault: every signal is set the OR of all.
- Opens are modeled as stuck-at faults.
A single physical fault may result in many stuck faults.
Logical Fault Models

Fault Detection for Combinational Circuits

Detectability and Redundancy
For a combinational circuit $N$, let $Z(x)$ be its logic function.
- $x$ is the input vector and $Z(x)$ could also be a vector.
- The presence of a fault $f$ transforms $N$ into a new circuit $N_f$.
- Assume $N_f$ is combinational with function $Z_f(x)$.
- $N_f$ could be sequential, though we ignore that possibility for now.
For a combinational circuit $N$, let $Z(x)$ be its logic function.

- $x$ is the input vector and $Z(x)$ could also be a vector.

The presence of a fault $f$ transforms $N$ into a new circuit $N_f$.

- Assume $N_f$ is combinational with function $Z_f(x)$.
  - $N_f$ could be sequential, though we ignore that possibility for now.
Combinational Circuits

- For a combinational circuit $N$, let $Z(x)$ be its logic function.
  - $x$ is the input vector and $Z(x)$ could also be a vector.
- The presence of a fault $f$ transforms $N$ into a new circuit $N_f$.
- Assume $N_f$ is combinational with function $Z_f(x)$.
  - $N_f$ could be sequential, though we ignore that possibility for now.
Test and Fault Detection

- A test contains multiple test vectors.
  - $t_1, t_2, \ldots, t_m$.

- A test vector detect a fault if the faulty system behaves differently.
  - The test vector $t$ detects a fault $f$ if and only if $Z(t) \neq Z_f(t)$.
  - If the output is a single bit, that’s $Z(t) \oplus Z_f(t) = 1$.
  - If the output contains multiple bits, we can apply bit-wise XOR and then OR the results.

- To find a test vector detecting a given fault is as hard as SAT.
A test contains multiple test vectors.

- $t_1, t_2, \ldots, t_m$.

A test vector *detect* a fault if the faulty system behaves differently.

- The test vector $t$ detects a fault $f$ if and only if $Z(t) \neq Z_f(t)$.
- If the output is a single bit, that's $Z(t) \oplus Z_f(t) = 1$.
- If the output contains multiple bits, we can apply bit-wise XOR and then OR the results.

To find a test vector detecting a given fault is as hard as SAT.
A test contains multiple test vectors.

- $t_1, t_2, \ldots, t_m$.

A test vector *detect* a fault if the faulty system behaves differently.

- The test vector $t$ detects a fault $f$ if and only if $Z(t) \neq Z_f(t)$.
- If the output is a single bit, that’s $Z(t) \oplus Z_f(t) = 1$.
- If the output contains multiple bits, we can apply bit-wise XOR and then OR the results.

To find a test vector detecting a given fault is as hard as SAT.
Example: Detecting Bridging Fault

▶ Assume a OR bridging fault $f$.
  
  ▶ $Z(x_1, x_2, x_3) = (x_1x_2, x_2x_3)$.  
  ▶ $Z_f(x_1, x_2, x_3) = (x_1 + x_2, (x_1 + x_2)x_3)$.  
  
▶ 011 detects $f$.  

(Fig. 4.2, Abramovici et al., 1990)
Example: Detecting Bridging Fault

Assume a OR bridging fault $f$.

$Z(x_1, x_2, x_3) = (x_1 x_2, x_2 x_3)$.

$Z_f(x_1, x_2, x_3) = (x_1 + x_2, (x_1 + x_2)x_3)$.

011 detects $f$. 

(Fig. 4.2, Abramovici et al., 1990)
How to detect s-a-0 at $x_4$?

- Solve $Z \oplus Z_f = 1$.
  - $Z = (x_2 + x_3)x_1 + x'_4x_4$ and $Z_f = (x_2 + x_3)x_1$.
- Any test vector such that $x'_4x_4 = 1$.
  - 0001, 0011, 0101, 0111.
Example: Detecting Stuck Fault

How to detect s-a-0 at \( x_4 \)?

Solve \( Z \oplus Z_f = 1 \).

\[ Z = (x_2 + x_3)x_1 + x'_1x_4 \text{ and } Z_f = (x_2 + x_3)x_1. \]

Any test vector such that \( x'_1x_4 = 1 \).

0001, 0011, 0101, 0111.
Example: Detecting Stuck Fault

How to detect s-a-0 at $x_4$?

Solve $Z \oplus Z_f = 1$.

- $Z = (x_2 + x_3)x_1 + x'_1x_4$ and $Z_f = (x_2 + x_3)x_1$.

- Any test vector such that $x'_1x_4 = 1$.
  - 0001, 0011, 0101, 0111.

(Fig. 4.3, Abramovici et al., 1990)
To detect s-a-1 at $G_2$, a test vector should output 0 at $G_2$.

The test vector must *activate* the stuck fault.

Moreover, there is a path from $G_2$ to $Z$ such that the signals along it are different.

The test vector must *propagate* the error.

A line is *sensitized* to a fault for a test if its value changes for the fault under the test.

A path of sensitized lines is a *sensitized* path.
Sensitization

To detect s-a-1 at $G_2$, a test vector should output 0 at $G_2$.

- The test vector must *activate* the stuck fault.

Moreover, there is a path from $G_2$ to $Z$ such that the signals along it are different.

- The test vector must *propagate* the error.

A line is *sensitized* to a fault for a test if its value changes for the fault under the test.

- A path of sensitized lines is a *sensitized path*.

(Fig. 4.3, Abramovici et al., 1990)
Sensitization

To detect s-a-1 at $G_2$, a test vector should output 0 at $G_2$.

- The test vector must activate the stuck fault.

Moreover, there is a path from $G_2$ to $Z$ such that the signals along it are different.

- The test vector must propagate the error.

A line is sensitized to a fault for a test if its value changes for the fault under the test.

- A path of sensitized lines is a sensitized path.

(Fig. 4.3, Abramovici et al., 1990)
For the output of a AND/OR/NAND/NOR gate to be sensitized because of one or more inputs,

- All sensitized inputs should have the same value.
- All inputs not sensitized should not control the output.
- The output value is based on the sensitized input and the inversion of the gate.
Logical Fault Models

Fault Detection for Combinational Circuits

Detectability and Redundancy
Detectability

- Recall to detect a fault $f$ we need to solve $Z(t) \oplus Z_f(t) = 1$ for a test vector $t$
- $f$ is *detectable* if we can find such a $t$.
- Some faults are NOT detectable.
  - They are NOT harmless since they invalidate the signal-fault assumption.
Detectability

- Recall to detect a fault $f$ we need to solve $Z(t) \oplus Z_f(t) = 1$ for a test vector $t$.
- $f$ is *detectable* if we can find such a $t$.
- Some faults are NOT detectable.
  - They are NOT harmless since they invalidate the signal-fault assumption.
Detectability

- Recall to detect a fault $f$ we need to solve $Z(t) \oplus Z_f(t) = 1$ for a test vector $t$
- $f$ is *detectable* if we can find such a $t$.
- Some faults are NOT detectable.
  - They are NOT harmless since they invalidate the signal-fault assumption.
The output is $C + b'(A' + a')$

- If there is no fault, $b = a' = B$ so the output is $C + A'B'$.
- $b$ s-a-0 is detectable for $C + A'B' \neq C + A' + B$.
  - e.g with test vector 1101.
- $a$ s-a-1 is undetectable as the output remains $C + A'B'$.
The output is \( C + b'(A' + a') \)

- If there is no fault, \( b = a' = B \) so the output is \( C + A'B' \).
- \( b \) s-a-0 is detectable for \( C + A'B' \neq C + A' + B \).
  - e.g. with test vector 1101.
- \( a \) s-a-1 is undetectable as the output remains \( C + A'B' \).
Detectable and Undetectable Faults

- The output is $C + b'(A' + a')$
  - If there is no fault, $b = a' = B$ so the output is $C + A'B'$.
- $b$ s-a-0 is detectable for $C + A'B' \neq C + A' + B$.
  - e.g with test vector 1101.
- $a$ s-a-1 is undetectable as the output remains $C + A'B'$. 

(Fig. 4.5 (a), Abramovici et al., 1990)
What if both faults happen?

The output is \( C + A' \).

- Cannot be detected by 1101.
- You may detect \( b \) s-a-0 whether or not \( a \) s-a-1 presents.
  - e.g. 0101 for the system without fault \((C + A'B')\) vs. those with faults \((C + A' + B\) or \(C + A')\).
- However, that’s not always possible. We also need to consider a multiple fault including the undetectable ones.

(Fig. 4.5 (b), Abramovici et al., 1990)
What if both faults happen?

The output is $C + A'$.
- Cannot be detected by 1101.
- You may detect $b$ s-a-0 whether or not $a$ s-a-1 presents.
  - e.g. 0101 for the system without fault ($C + A'B'$) vs. those with faults ($C + A' + B$ or $C + A'$).
  - However, that’s not always possible. We also need to consider a multiple fault including the undetectable ones.
A combinational circuit is *redundant* if there is any undetectable stuck fault.

- In the sense that one can use this information to simplify the circuit by removing gates/interconnects.
- Other simplifications may be possible when undetectable stuck faults are not presented. However, our definition of redundancy solely depends on whether such faults exist.
- If all stuck faults are detectable, then the circuit is *irredundant*.

- One can permanently place the stuck-at value at the line with the undetectable stuck fault.
- Constant may be propagated through the fanouts of the line and multiple gates/lines may be removed recursively.
- Gates not lead to circuit outputs can be removed further.
Redundancy

- A combinational circuit is *redundant* if there is any undetectable stuck fault.
  - In the sense that one can use this information to simplify the circuit by removing gates/interconnects.
  - Other simplifications may be possible when undetectable stuck faults are not presented. However, our definition of redundancy solely depends on whether such faults exist.
  - If all stuck faults are detectable, then the circuit is *irredundant*.

- One can permanently place the stuck-at value at the line with the undetectable stuck fault.
  - Constant may be propagated through the fanouts of the line and multiple gates/lines may be removed recursively.
  - Gates not lead to circuit outputs can be removed further.
A combinational circuit is *redundant* if there is any undetectable stuck fault.

- In the sense that one can use this information to simplify the circuit by removing gates/interconnects.
- Other simplifications may be possible when undetectable stuck faults are not presented. However, our definition of redundancy solely depends on whether such faults exist.
- If all stuck faults are detectable, then the circuit is *irredundant*.

One can permanently place the stuck-at value at the line with the undetectable stuck fault.

- Constant may be propagated through the fanouts of the line and multiple gates/lines may be removed recursively.
- Gates not lead to circuit outputs can be removed further.
Redundancy

A combinational circuit is *redundant* if there is any undetectable stuck fault.

- In the sense that one can use this information to simplify the circuit by removing gates/interconnects.
- Other simplifications may be possible when undetectable stuck faults are not presented. However, our definition of redundancy solely depends on whether such faults exist.
- If all stuck faults are detectable, then the circuit is *irredundant*.

One can permanently place the stuck-at value at the line with the undetectable stuck fault.

- Constant may be propagated through the fanouts of the line and multiple gates/lines may be removed recursively.
- Gates not lead to circuit outputs can be removed further.
Redundancy in Circuits

Redundancy does not necessarily denote an inefficient or undesirable implementation.

- e.g. in triple modular redundancy (TMR), which is a basic technique for fault-tolerant design.
  - Each module has to be tested individually.

\[\text{Figure 4.7 TMR configuration}\]

(Abramovici et al., 1990)
Redundancy in Circuits

Redundancy does not necessarily denote an inefficient or undesirable implementation.

- e.g. in triple modular redundancy (TMR), which is a basic technique for fault-tolerant design.
  - Each module has to be tested individually.

**Figure 4.7** TMR configuration

(Abramovici et al., 1990)
Discussions

- Detectable faults may become undetectable when undetectable faults present.
- Two undetectable single faults may become detectable if simultaneously present.
- Practically, we don’t need to worry about undetectable faults.
  - For large circuits, it is computationally prohibitive to generate a set of test vectors to cover all detectable faults, and to show the circuit is irredundant.
  - You cannot distinguish an undetectable fault and a detectable one that is not detected by the test vector set.
Discussions

- Detectable faults may become undetectable when undetectable faults present.
- Two undetectable single faults may become detectable if simultaneously present.
- Practically, we don’t need to worry about undetectable faults.
  - For large circuits, it is computationally prohibitive to generate a set of test vectors to cover all detectable faults, and to show the circuit is irredundant.
  - You cannot distinguish an undetectable fault and a detectable one that is not detected by the test vector set.
Discussions

- Detectable faults may become undetectable when undetectable faults present.
- Two undetectable single faults may become detectable if simultaneously present.
- Practically, we don’t need to worry about undetectable faults.
  - For large circuits, it is computationally prohibitive to generate a set of test vectors to cover all detectable faults, and to show the circuit is irredundant.
  - You cannot distinguish an undetectable fault and a detectable one that is not detected by the test vector set.
Logical faults provide an abstraction of physical faults and their models depend on system models.

We assume a single fault when considering testing.

Logical fault can be detected by choosing a proper test vector.

Undetectable faults in redundant circuits complicate testing, though to decide whether a circuit is redundant is as hard as to generate test vectors to cover all detectable faults.