Week 9

Structured and Testable Design

References

• STRUCTURED DESIGN

Objective: to reduce the global amount of effort involved in the design and layout, and to allow for direct extension and alteration

• HIERARCHY
• REGULARITY
• MODULARITY
• LOCALITY

• TESTABLE DESIGN (Design for testability)

Objective: to reduce the burden of exhaustively testing the fabricated VLSI circuit for structural faults and other non-idealities

• OBSERVABILITY
• CONTROLLABILITY

(The above requirements for structured design are useful here as well)
STRUCTURED DESIGN

a) Hierarchy and Regularity

Hierarchy: tree-based structured decomposition towards higher detail in the description

Regularity: parallel instances of identical cells are used throughout for the modules and sub-modules, at different levels in the hierarchy

\[ R = \frac{\text{total # of instances}}{\text{# actually designed}} \approx \text{average # instances/cell} \]

\[ \Rightarrow \text{dramatically reduces the "explicit" layout and design work} \]

Fabrication Cost (for given performance)

Design Cost (yr)
b) Modularity and Locality

**Modularity**: Each block in the hierarchical decomposition has well-defined internal functionality and a suitable external interface (self-contained externally) typically:
- **voltage**: input: $H1-2$, output: $L0-2$
- **currents**: input: $L0-2$, output: $H1-2$

examples:

![Example 1](image1)

Counterexample:

![Counterexample](image2)

**Locality**: Modules (or sub-modules) are constructed as homogeneous "clusters" of internally self-contained functionality in order to reduce interconnections between modules to a minimum level.

- mostly local signals
- a few global signals for module interconnect and for bus/interrow generation and control

example: pipe-line / systolic array

$$A_{in}(k) = \sum_{i=1}^{(k-1)} (A_{in}(i) \cdot B_{in}(k-i))$$

systolic cell

$$B_{in}(k)$$

SYSTOLIC ARRAY
NOTE: restriction on locality does NOT exclude global computation schemes which exploit a DISTRIBUTED REPRESENTATION, nor does the requirement of modularity exclude cases where boundary values are influenced by other values in the same class, sharing the global distributed computational function.

EXAMPLES:

1) Vector quantizer:
   a) Summing node on the output of the distance cells:
      \[
      \text{out} = d(A_j, o_{i,j})
      \]
      \[
      \text{summing node: CURRENT (continuous-time)}
      \]
      \[
      \text{SUMMING node: CHARGE (sampled in time)}
      \]

2) Common "competition" line in winner-take-all
   \[
   \text{Input} \quad \begin{array}{c}
   Y_{in}
   \\
   Y_{in} + I
   \\
   Y_{in} + I
   \end{array} \rightarrow \begin{array}{c}
   \text{Common}
   \\
   \text{Common}
   \end{array} \quad \rightarrow \begin{array}{c}
   I_1
   \\
   I_2
   \\
   I_3
   \end{array}
   \]
EXAMPLES (continued):

2) follower (or) (attenuation...):

a) modularity violation: (chaining)

\[
Z_{\text{out}} = \frac{1}{j} = \frac{24}{V_{\text{bias}}}
\]

\[
\begin{align*}
\text{IN} & \quad \text{OUT} \\
V_{\text{in}} & \quad V_{\text{out}}
\end{align*}
\]

\[
\begin{align*}
\text{IN} & \quad \text{OUT} \\
V_{\text{in}} & \quad V_{\text{out}}
\end{align*}
\]

b) possible locality violation: (chaining)

\[
\begin{align*}
V_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
\end{align*}
\]

reduction of local circuitry:

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]

\[
\begin{align*}
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \} \\
I_{\text{in}} & \quad \{ I_{\text{out}}, V_{\text{out}} \}
\end{align*}
\]
**DESIGN FOR TESTABILITY**

Ideally: make it possible to test all the individual functions in isolated fashion.

Test for:
1. Physical defects (SHORTS; OPENS, open or less)
2. Mismatches; offsets (in analog circuitry)
3. Functionality (global; in principle exhaustive conditions)

\[
\text{input pads} \rightarrow I \xrightarrow{\text{IC}} O \xrightarrow{\phi} \text{output pads}
\]

\[N \text{ (internal nodes)}\]

For a complete and efficient check, following requirements will do:

- **CONTROLLABILITY**: ability to set all (relevant) internal nodes and branches to (more or less) arbitrary values

- **OBSERVABILITY**: ability to (directly or indirectly) obtain the values of internal nodes & branches

- **INDUCTIVE CAUSALITY**: ability to decompose the circuit into individually testable parts, each part being accessible individually through induction (assuming under correct operation a given model is in effect)

\[\text{input pads} \rightarrow \begin{array}{c} \text{(sub module)} \\ \end{array} \rightarrow \text{output pads} \]

\[\text{m (internal nodes)} \quad (m \in N) \quad \text{assumes a given model for correct operation}
\]

\[\text{indirect access neighbors} \quad \phi \]

\[\text{controllable} \quad \text{observable} \]
Recursive modular testing assuming induction in causality:

Objective: reduce the combinatorial explosion for testing the flat structure

\[ \text{inputs} \rightarrow \text{N} \rightarrow \text{outputs} \]

\[ i \ll I \]

\[ \text{decomposition} \]

\[ i \ll o \]

\[ m \ll N \]

\[ \alpha \text{ test patterns needed} \rightarrow \sum \alpha_i \ll \alpha I \]

\( \alpha = \# \text{ levels of inputs (cascading depth)} \)

\( \alpha = 2 \) for binary inputs

Equivalence is only guaranteed if the decomposition satisfies MODULARITY and if the model is correctly implemented (except for the faults under investigation).

Extension to \underline{sequential} systems rather than \underline{combinatorial}:

\[ \text{inputs} \rightarrow \text{N internal modes} \rightarrow \text{I+F} \rightarrow \text{outputs} \]

\[ \text{decomposition (along internal state variables also)} \]

\( \alpha \) I+F patterns needed \( \rightarrow \sum \alpha_{I+F} \ll \alpha \) I+F all (sub)modules

\[ \Rightarrow \text{TESTABILITY also requires \{MODULARITY, LOCALITY\}} \]
example: PARITY TREE (8-bit, say)

\[ A_1 \quad A_2 \quad A_3 \quad A_4 \quad A_5 \quad A_6 \quad A_7 \quad A_8 \]
\[ X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \quad X_6 \quad X_7 \]
\[ B_1 \quad B_2 \quad B_3 \quad B_4 \quad B_5 \quad B_6 \]
\[ \quad \quad \quad \quad \quad \quad \quad \quad \quad C \]

Controllability:

to make \( B_5 = 1 \): \( B_1 \oplus B_2 = 1 \) \( \Rightarrow (A_1, A_2, A_3, A_4) \text{ odd parity even} \)

etc...

Observability:

to observe \( B_2 \): set \( B_1 \) and \( B_6 \) (to whatever) \( \Rightarrow B_2 = B_1 \oplus B_6 \oplus C \)

etc...

Inductive causality:

7 x 4 test vectors for all individual \( X_i \) \( i = 1 \) through 7

28 test cycles rather than 256 for exhaustive test \((2^8)\)
BOUNDARY SCAN REGISTERS FOR TESTABILITY IN GENERAL ARCHITECTURES

\[ B_i \rightarrow \Delta \rightarrow B_o \]
\[ B_o(i) = B_i(i-\Delta) \]

or:

\[ B_i \rightarrow \square \rightarrow B_o \]
\[ B_o(i) = \sum B_i(i+\delta) \]

\( B_i \); \( B_o \): BOUNDARY REGISTER
TRANSFER NODES FOR

1. Interconnect
2. Internal state variables

Example:

\[ i j \]
\[ i'j' \]
\[ i'j' \]
\[ i'j' \]

\( \Phi_c \) and then return \( \Phi_c \), Low

\( \Phi_c \) pulse on \( \Phi_c \): CONTROL \( B_i \)
\( \Phi_c \) pulse on \( \Phi_o \): OBSERVE \( B_o \)
\( \Phi_{shift} \) and \( \Phi_{shift} \): scan in \( B_i \)'s and scan out \( B_o \)'s
RANDOM-ACCESS TESTABILITY

See Week 3 - memory architecture

Address decoders/demux can be used for other purposes! (e.g., I/O multiplexing, programming etc.)

1) current/voltage input:

2) voltage buffer:

3) current replica: