Design for testability (DFT) refers to those design practices that allow us to answer the above questions in the affirmative.

Electronic systems contain three types of components: (a) digital logic, (b) memory blocks, and (c) analog or mixed-signal circuits. There are specific DFT methods of each type of component. In this chapter, we discuss DFT techniques for digital logic. *Built-in self-test* (BIST), which is also used for digital logic as well as for memory blocks, is the subject of Chapter 15. A system (a printed circuit board or a multi-chip module) consists of an interconnect of the three types of components. Thus, the component-level DFT methods, as discussed here and in the next chapter, are not sufficient for producing a testable system. Special techniques, known as *boundary-scan* and *analog test bus*, provide test access to components embedded in a system. These are discussed in Chapters 16 and 17, respectively. Finally, it all comes together in Chapter 18 where we discuss systems test.

14.1 Ad-Hoc DFT Methods

Keeping in line with current practices, we will focus on DFT techniques that aim at improving the testability of stuck-at faults. An interested reader may review recent literature on DFT methods for other fault models such as delay faults [371] and physical faults [557].

Logic DFT takes one of two possible routes: *ad-hoc* and structured. The ad-hoc DFT relies on "good" design practices learned from experience. Some of these are [7, 721]:

- *Avoid asynchronous logic feedbacks.* A feedback in the combinational logic can give rise to oscillation for certain inputs. This makes the circuit difficult to verify and impossible to generate tests for by automatic programs. This is because test generation algorithms are only known for acyclic combinational circuits.

- *Make flip-flops initializable.* This is easily done by supplying *clear* or *reset* signals that are controllable from primary inputs.

- *Avoid gates with a large number of fan-in signals.* Large fan-in makes the inputs of the gate difficult to observe and makes the gate output difficult to control.

- *Provide test control for difficult-to-control signals.* Signals such as those produced by long counters require many clock cycles to control and hence increase the length of the test sequence. Long test sequences are harder to generate.

These and many other similar situations cause poor controllability and observability of signals and usually result in long test sequences and low fault coverage. Experienced designers could easily spot the problem areas on a logic schematic. For very large circuits, approximate testability measures have been used (see Chapter 6.)