

Few digital circuits perform the same series of steps repeatedly. How can we arrange to stop the sequence after the six steps have been executed? The solution lies in a fundamental concept: *feedback*. Feedback lies at the heart of complex digital circuits because it allows the results of processing to affect the way a circuit behaves. In the computer startup sequence, feedback is needed for each of the steps. If the disk cannot be started, for example, the boot sector cannot be read from the disk.

To see a trivial example of feedback, consider how we might use the final output, call it F , from the decoder to stop the process. An easy way consists of using the value of F to prevent clock pulses from reaching the counter. That is, instead of connecting the clock output directly to the counter input, we insert logic gates that only allow the counter pulses to continue when F has the value 0. In terms of Boolean algebra, the counter input should be:

$$\text{CLOCK and (not } F\text{)}$$

That is, as long as F is false, the counter input should be equal to the clock; when F is true, however, the counter input changes to (and remains) zero; Figure 2.16 shows how two inverters and a nand gate can be used to implement the necessary function.

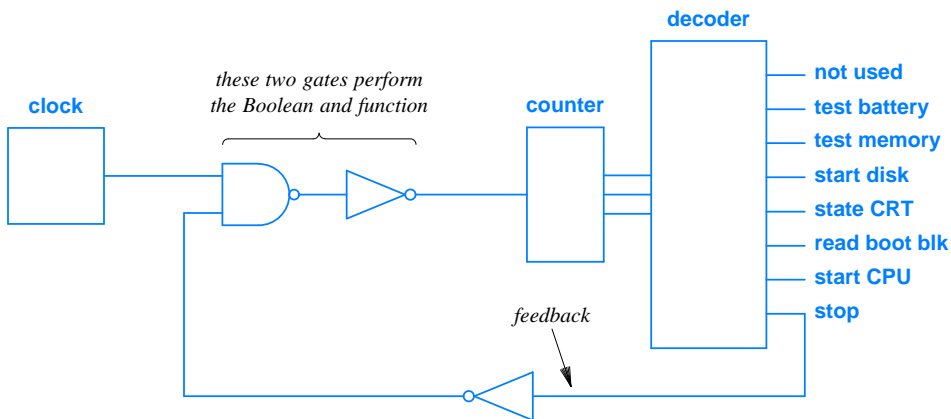


Figure 2.16 A modification of the circuit in Figure 2.15 that includes feedback to stop processing after one pass through each output.

The feedback in Figure 2.16 is fairly obvious because there is an explicit physical connection between the last output and the combinatorial circuit on the input side. The figure also makes it easy to see why feedback mechanisms are sometimes called *feedback loops*[†].

[†]A feedback loop is also present among the gates used to construct a flip-flop.