all the things that have to happen between the synchronizer clock edge and the next state clock changing edge ($T_{psyncFFmax}$, $TXsVDV_{max}$, $T_{setup min}$). In this example, the first concern will lead to an equation giving limits for $T_{clk}$ and the second to an equation giving limits on $T_{clk}/2$.

2 Output Hazards

Next we concern ourselves with the question of output timing. Here again, it will be helpful to refer to the structure of the general state machine in Figure 5. Figure 11 shows timing for the various kinds of outputs that can be found in state machines. Remember

![State Machine Output Timing Diagram](image)

that state machines can have two kinds of outputs. Moore outputs depend only on the present state, while Mealy outputs depend both on the present state and the current
input². Figure 11 shows the transition from state a to state b in a general state machine. A Moore output to be asserted in state b would not be valid until after the flip-flop outputs have settled and propagated through the combinational logic to the output, that is $T_{PFFmax} + T_{QVOVmax}$. To be valid, a Mealy output not only needs a valid present state, but also needs a valid input ($Xs$). Changes in the output due to changes in the present state and changes due to changes in $Xs$ can be considered separately (just as we did when considering changes in the next state inputs). Changes in a Mealy output resulting from present state changes alone are the same as for a Moore output. Figure 11 shows the timing for changes in Mealy outputs for SST and AST inputs. These diagrams include output changes resulting from both present state changes and changes in $Xs$. The intermediate timing showing changes in the output resulting only from changes in $Xs$ is not shown. Just when a synchronous input is valid depends upon how it was synchronized. In Figure 11 we have shown an SST input that comes valid after the present state information settles. In this case it is the input that changes last and delays a valid output. A smaller value of $T_{CXsVmax}$ or a larger value of $T_{PFFmax}$ than that shown could result in changes in Mealy outputs due to the flip-flop propagation delay lasting longer than output changes resulting from the synchronous input. Similar arguments can be made for the case of a Mealy output based on an AST input. Notice in Figure 11 how the period of time during which the AST Mealy output is valid is severely restricted. Part of this is due to the scale of the timing diagram. If $T_{CLK}$ were made much larger, then the period of time during which the output was valid would increase.

Figure 11 also shows the period of time during which the previous output remains valid. This depends on $T_{PFFmin}$ (the minimum state flip-flop propagation delay) which is often not specified, and therefore is usually treated as 0. Taking $T_{PFFmin}$ to be 0, moves the the beginning of the period of time during which the output is invalid back to the state changing clock edge.

During the period of time in which the output is invalid, we can have unwanted outputs. Another way to say this is to say that we can have output hazards. To illustrate this, consider the example shown in Figure 12. When the machine is in state a, and the machine is clocked, the flip-flop outputs must change from 00 (state a) to 11 (state b). In Figure 11 this change occurs during the criss-crossed period on the flip-flop output line. Since it is unlikely that both flip-flops have exactly the same propagation delay, it is unlikely that the flip-flop outputs will change from 0 to 1 at exactly the same time. This means that the flip-flop outputs actually pass temporarily through other states. In the example of Figure 12, if flip-flop B changes to 1 before flip-flop A changes to 1, the sequence of flip-flop outputs following the clock edge is 00 – 01 – 11. That is, the flip-flop outputs temporarily pass through the unused state 01. If, however, flip-flop A has a shorter propagation delay than flip-flop B so its output changes from 0 to 1 before flip-flop B changes to 1, the sequence of flip-flop outputs following the clock edge is 00 – 10 – 11. That is, the flip-flop outputs temporarily pass through state c. If this state exists long enough (just longer than the propagation delay of the AND gate), we will temporarily have output F asserted. Since we only want to assert F when we actually are in state c, this temporary output is a hazard.

²A machine is a Moore machine if it has only Moore outputs, otherwise it is a Mealy machine.
2.1 Hardware Solutions

There are two possible hardware based solutions to the output hazard problem. In one solution, called clock suppression, a signal, usually derived from the clock, is ANDed with the state machine output. For this to work, the signal must only be asserted during the time the output is valid. From Figure 11 we can see that the inverse of the state clock delayed slightly would work as a clock suppression signal for Moore style inputs and the SST Mealy style inputs. However, the inverse of the state clock would not work as a suppression signal for the AST Mealy output in Figure 11 since the period of time the output is valid is much shorter than half of the clock signal. It is important to carefully consider timing when using clock suppression. Clock suppression done improperly can increase the number of hazards. Clock suppression has the disadvantage that a clock suppressed output can only be asserted for half the clock period. If an output needs to be asserted over an entire clock period, then a different solution is needed.

Another hardware solution involves using flip-flops to sample the state machine outputs when they are stable. These flip-flops are referred to as output holding registers. As drawn in Figure 11, the Moore outputs could be sampled using a negative edge triggered flip-flop clocked by the state clock since the output is stable when the clock line goes low. Note that this depends on the actual propagation delays and the clock period. If $TQVOV_{\text{max}}$ or $TpFF_{\text{max}}$ were larger than shown, then the holding register clock might have to be delayed from the state clock. As drawn in Figure 11, the SST Mealy outputs could be sampled using an edge triggered flip-flop clocked by an inverted and delayed state clock. But we could not do this for the AST Mealy outputs. We could sample the AST output with a positive edge triggered flip-flops driven by the state clock, but if we did this, we would have the somewhat confusing situation of having an output for state b actually asserted in the following state. It is possible to make this work by changing the state graph so that the output we actually desire asserted in a given state is setup during the previous state(s).

Figure 13 shows how clock suppression, or an output holding register can be used to prevent output hazards in a general state machine. The AND gate used for clock
suppression might actually be combined with logic already present in the combinational logic block.

![Figure 13: Hardware Solutions for Output Hazards](image)

### 2.2 State Assignment Solutions

It is possible to prevent output hazards in Moore outputs without adding extra hardware, through proper choice of state assignment. Consider again the example in Figure 12. Let us change the state assignment for state b from 11 to 01. Now when the machine moves from 00 (state a) to 01 (state b), there is no way to temporarily pass through an other state. The machine simply moves from state a to b, never passing through state d. This will always be true when only one flip-flop has to change as the state machine moves from one state to the next.

Let us reconsider the machine in Figure 12. Figure 14 gives a K-map showing the state assignment and the machine outputs. By examining this we can discover a way to create state assignments that do not lead to output hazards. The arrows on the map show the temporary states the flip-flop outputs can pass through when the machine transitions from state a to b. The machine passes through the unused state if flip-flop B changes before A. The machine passes through state c, possibly generating a hazard in output F, if flip-flop A is faster than B. In general, it is possible to easily determine the states that may be passed through, and hence the outputs that may have hazards, for every transition from present state to next state for any machine. First, create a present state map (a map showing the assignment of state variables to states) and include outputs in the map. Then, for each
transition in the machine's state graph, visualize the smallest prime implicant loop that contains the present and next states. The states that may be passed through during the transition will be all the other states in the loop and the outputs that may have hazards will be the outputs associated with those states. Figure 15 shows a five state state machine and two possible state assignments.

![Figure 15: A Five State Machine and Two Possible State Assignments](image)

For assignment (a), the transition from state a to b or from state b to a, could produce a hazard in output F. The transition from a to c causes no problem since a and c are logically adjacent. The transition from c to d can generate a hazard in output W. The transition from state d to e or from d to a causes no problems since d is adjacent to both a and e. Finally, the transition from state e to a could generate a hazard in output F.

The state assignment shown in part (b) of Figure 15 shows a hazard free assignment. That this is so can be verified by testing the transition paths for each present-state next-state transition in the state graph. Note that it was not possible in this case to solve the hazard problems by making all states that transition to one another logically adjacent since all four other states would have to be adjacent to state a and in a three variable map, each cell has only three adjacent cells. Instead, the engineer who designed the state assignment found a way to make sure that all transitions between non-adjacent states pass through states that won't generate output hazards. In assignment (b), the only non-
adjacent transitions are the transitions c to d and e to a. The transition from state e to a could pass through either state b or d, neither of which have an output, so this transition will not generate any output hazards. The transition from state c to d can pass through state a or the unused state 110. State a doesn't have an output and it is possible to avoid assigning an output to state 110. However, as shown in Figure 15 (b), we could assign output F to the unused states in order to minimize the hardware required to implement F. Since state c already has output F, if F were also asserted in state 110, a transition from state c to d that passed through 110 would not generate a hazard in F. Similarly, we can assign output W to unused state 111 to help minimize it. Since neither of states 111 and 101 are ever passed through on any transition, the assignment of outputs to these states will not generate output hazards. On the other hand, assigning output W to unused state 110 could create a hazard in W on a transition from state c to a. Note that there may be other considerations for the unused states involving the self-correcting behavior of the machine. Also note that we not only have to worry about hazards arising from the state assignment, but also we should take care that the combinational logic that implements the output doesn't have any inherent hazards. This can be done by making sure that the K-maps used to design the output logic have any needed hazard covers.

Assignment (b) in Figure 15 makes use of an unused state to provide a hazard free transition path. In certain situations it may be impossible to come up with a hazard free assignment. In this case, output hazards may be controlled by clock suppression or by adding an output holding register. Another approach is to introduce additional adjacencies and unused states by adding an extra state flip-flop to the state machine.

State assignment solutions will not work for Mealy style outputs since the output is a function of the input in addition to the present state. Consider again Figure 11. Note that the SST input may change while the old flip-flop outputs are still stable or the SST input may change after the state flip-flops are stable. Precisely when the input changes will depend on the relationship between the state clock and the synchronizer clock. Similarly, the AST input will change long after the present state has stabilized. One way to deal with hazards in Mealy outputs is to use output holding registers or clock suppression. Taking another approach, Mealy outputs could be converted to Moore outputs. This usually requires the addition of states to the state machine. Depending on the number of states already present, the conversion to a Moore output may require less hardware than would be required for clock suppression or the use of holding registers.

3 Asynchronous Inputs

Let us consider what would happen if we did not synchronize an input. Again it is helpful to visualize a general state machine (see Figure 5). Since the next state (the inputs to the flip-flops) is a function of the present state and the inputs, it is possible that an input changing at just the wrong time would cause the flip-flop inputs to be changing as the flip-flops were clocked. Consider the ASM chart shown in Figure 16. The shading in the bottom of the X decision diamond is sometimes used in ASM charts to indicate that the input X is asynchronous. First assume that the state assignment shown in map (a) is