June 1966

Brief 66-10264

NASA TECH BRIEF

Binary Sequence Detector Uses Minimum Number of Decision Elements

The problem:

To provide a detector of an n bit binary sequence code within a serial binary data stream, using a minimum number of elements in the detector. To date there is no known algorithm for assigning states or combinations of states to the memory elements of a code sequence detector that yields combinatorial logic of minimum complexity.

The solution:

A procedure that, given the linear recursion relationship employed by the sequence generator, will employ the same ordering of states for the sequence detector as that of the sequence generator.

How it's done:

The selection of the initial state assignment is performed by assigning p initial states to the sequence detector, where p is the smallest integer greater than or equal to the logarithm (base 2) of n. Then the Boolean equations are written and minimized for each of the p initial state cases. Of the p cases, the case that results in the combinatorial logic of minimum complexity is implemented for the sequence detector. The resulting sequence detector consists of p memory elements interconnected by combinatorial logic employing decision elements.

The sequence detector must assume at least n states to detect an n bit sequence. The detector will require a sufficient number of memory elements for each of the n states. The minimum number of two-state or binary memory elements required is $p = [\log_2 n]$.

The successive combination of states the sequence generator (shift register) assumes provides an ordering of states for the detector that leads to a relatively small number of logic elements for the detector. Therefore, the successive states of the detector are made to correspond to the successive states of the generator shift register.

A state table is constructed that contains, for each of the n states corresponding to the incoming serial code stream, the next state that the detector should assume. For the valid code sequence the detector should successively assume states 1, 2, . . . n, and on the nth input bit should produce a signal signifying that the valid code sequence has occurred. Also, at this point the detector should revert to the initial state. For nonvalid code sequences the detector should successively assume states for as long as the nonvalid code stream matches the valid code sequence. At the first code bit that does not match the valid code, the detector should revert to an earlier state.

For instance, assume the detector is designed to have 31 states for detecting a 31 bit code sequence and the detector is in state 18, having received 17 bits that corresponded bit-by-bit with the first 17 bits of the valid code. Should the 18th input bit not correspond to the 18th bit of the valid sequence, the detector would not progress to the 19th state but would instead revert to an earlier state. The state to which the detector reverts would take into consideration the code prefix possibility. For example, if the detector has reached its 18th state before encountering an incoming bit that does not match the valid code, the state to which the detector would revert is not necessarily the first state. The next state is a function of the match of the most recently received bits to the beginning of the valid code. Assuming the most recently received 4 bits match the first 4 bits of the valid code, the detector would revert to state 5 rather than state 1.

(continued overleaf)

For each of the p candidate initial states, the set of combinatorial logic equations is written. Their implementation would cause the detector state sequence to correspond to the generator state sequence upon receipt of the valid code sequence. Following minimization of the sets of equations by standard techniques, the set that requires the minimum number of logic elements is selected for implementation. The design procedure avoids extensive or exhaustive evaluation of the n! possible state sequences which could be implemented in the detector.

As an example of the logic element requirement for a 31 bit code sequence detector, the method outlined above results in logic consisting of only five memory elements in the form of RS flip–flops and 17 NAND gates. This compares quite favorably with an alternative straightforward implementation of a 31 stage shift register as a code sequence detector, where the logic required totals 31 memory elements (flip–flops) and one 31 input decision element (and circuit).

Note:

Inquiries concerning this invention may be directed to:

Technology Utilization Officer
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California, 91103
Reference: B66-10264

Patent status:

Inquiries about obtaining rights for the commercial use of this invention may be made to NASA, Code GP, Washington, D.C., 20546.

Source: Marvin Perlman
(JPL-673)