Fault-Tolerant Coding for State Machines

State machines can be rendered immune to single-event upsets.

NASA's Jet Propulsion Laboratory, Pasadena, California

Two reliable fault-tolerant coding schemes have been proposed for state machines that are used in field-programmable gate arrays and application-specific integrated circuits to implement sequential logic functions. The schemes apply to strings of bits in state registers, which are typically implemented in practice as assemblies of flip-flop circuits. If a single-event upset (SEU, a radiation-induced change in the bit in one flip-flop) occurs in a state register, the state machine that contains the register could go into an erroneous state or could "hang," by which is meant that the machine could remain in undefined states indefinitely. The proposed fault-tolerant coding schemes are intended to prevent the state machine from going into an erroneous or hang state when an SEU occurs.

To ensure reliability of the state machine, the coding scheme for bits in the state register must satisfy the following criteria:

1. All possible states are defined.
2. An SEU brings the state machine to a known state.
3. There is no possibility of a hang state.
4. No false state is entered.
5. An SEU exerts no effect on the state machine.

Fault-tolerant coding schemes that have been commonly used include binary encoding and "one-hot" encoding. Binary encoding is the simplest state machine encoding and satisfies criteria 1 through 3 if all possible states are defined. Binary encoding is a binary count of the state machine number in sequence; the table represents an eight-state example. In one-hot encoding, *N* bits are used to represent *N* states: All except one of the bits in a string are 0, and the position of the 1 in the string represents the state. With proper circuit design, one-hot encoding can satisfy criteria 1 through 4. Unfortunately, the requirement to use *N* bits to represent *N* states makes one-hot coding inefficient.

Like one-hot encoding, one of the two proposed schemes satisfies criteria 1 through 4. However, the Hamming distance of 2 encoding is more efficient in that it requires fewer than *N* bits to represent *N* states. The scheme is denoted H2 because it requires a Hamming distance of 2 between any two adjacent legal state representations: that is, the strings of bits representing any two adjacent states must differ in two places. Starting from any legal state representation in H2 encoding, an SEU changes the contents of the register to a defined illegal state representation. Because the illegal representation is defined, it can be recognized automatically and used to prevent the state machine from entering an illegal state and from generating incorrect outputs. An H2 code can be generated by adding one additional bit to a binary encoding scheme, as shown in the table.

State	Binary	One-Hot	Hamming 2	Hamming 3
S0	000	00000001	0000	000000
S1	001	00000010	0011	000111
S2	010	00000100	0101	011001
S3	011	00001000	0110	011110
S4	100	00010000	1001	101010
S5	101	00100000	1010	101101
S6	110	01000000	1100	110011
S7	111	10000000	1111	110100

An **Eight-State Example** shows different encoding schemes.

An extension of the H2 concept to require a Hamming distance of 3 between any two adjacent legal state representations yields the H3 coding scheme, which satisfies all five criteria. Starting from any legal state representation in H3 encoding, an SEU changes the contents of the register to a defined illegal state representation that is unique to the legal state representation. Because the illegal representation is defined and associated with one (and only one) legal state representation, it can be recognized automatically and used to restore the state machine to the correct state. The generation of an H3 code is not as easy as is the generation of an H2 code. An algorithm that generates an H3 code has been devised.

To test these various encodings for fault tolerance, a circuit was devised which could both generate faults and examine the effect of these faults. The circuit was designed to run on an application board containing a Xilinx Spartan II FPGA (field programmable gate array). Hamming-3 (H3) encoding gave by far the best fault tolerance, recording 0 errors in fault injection tests. However, it required the most resources and was the slowest of the encoding methods. Hamming-2 (H2) encoding had fewer errors than binary encoding, but one-hot encoding had the most errors, due to its large number of target flip-flops. It was also the slowest, and showed poor use of resources.

The results from the tests performed showed that for fault-tolerant designs, H2 was the best compromise in terms of size, speed, and fault-tolerance and is preferred over both binary and one-hot state machine encoding. The results also showed H3 encoding to be fault-tolerant to single faults and, therefore, preferred when ultimate reliability is required in a critical application.

This work was done by Stephanie Taft Naegle, Gary Burke, and Michael Newell of Caltech for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1). *NPO-41050*