Appendix A

Further Memory Fault Modeling

Extended Listing

The fault models in this appendix include those not described in chapter eight. It is recommended that the reader first become familiar with the faults covered in that chapter as they describe real manufacturing defects which must be covered during memory testing. The fault models covered here will help in the understanding of nomenclature and will aid readers as they review other literature on memory testing. Some of the fault models covered here are only mathematical curiosities while others provide helpful insight.

1. LINKED FAULTS

Since more than one defect can exist in a memory, the multiple faults can interact. As these interact they are referred to as linked fault models. Fault models that are linked can be of similar or dissimilar types [207]. They can also work in such a manner that one fault can mask the behavior of another fault. The occurrence of this kind of a fail must be rare in order to get reasonable chip yield. The probability of having two fails needs to be rarer still. The probability of having two fails, that in fact interact, must be exceedingly rare. Therefore linked faults should normally be of little concern. The only possibility for linked faults, which would be of concern, is if one defect activates two faults, which in turn are linked.

When multiple faults do not interact they are said to be *unlinked.* This is the normal case with multiple faults.

2. COUPLING FAULT MODELS

2.1 Inversion coupling fault

An inversion coupling fault involves an aggressor and a victim cell where the aggressor causes the victim cell's data to invert. A Markov diagram of this fault model is shown in figure A-1. This figure illustrates an example whereby any transition in cell "j" from a "0" to a "1" causes cell "i" to invert. No known defects will cause this type of defective operation nor is it known if such a fault has ever been seen.

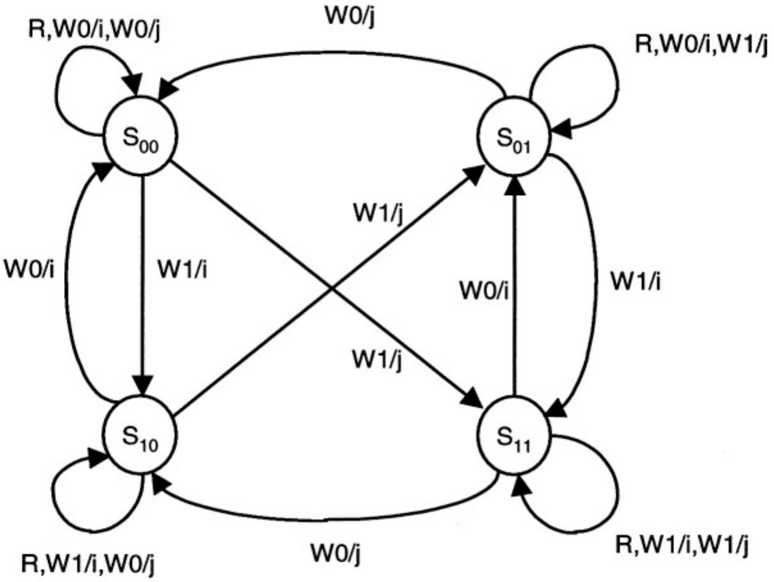


Figure A-1. Markov diagram of an inversion coupling fault.

2.2 Idempotent coupling fault

An idempotent coupling fault describes an event where writing a specific state in an aggressor cell causes a victim cell to go to a specific unintended state. Figure 8-8 shows an idempotent coupling fault where writing cell "j" to a "1" erroneously forces cell "i" to a "1" as well [208]. The source of the name for this fault is curious, although idempotent is a well understood mathematical term [209]. After some searching and dialog with others in this field, it seems that the reason for the name "idempotent" and its source have been lost in the annals of time. Since idempotent can refer to having the same singular result, even if an operation is repeated multiple times, it is possible that the idempotent coupling fault was so named in converse to the inversion coupling fault. The inversion coupling fault gets different results in the victim cell based on the number of times the aggressor cell is written. In contrast is the idempotent coupling fault, where a single erroneous value is maintained in the victim cell, as long as the victim cell itself is not purposely re-written.

2.3 Complex coupling fault

A complex coupling fault describes a defect where multiple cells must be in a specific state in order to sensitize a victim cell to a certain aggressor. A k-complex coupling fault is one where k is a value describing the number of cells which must interact [210]. If k is 5, then k-1 or 4 cells must be in a specific state in order for a 5th cell's transition to cause a failure in the victim cell.

2.4 State coupling fault

A state coupling fault does not require an operation to be performed nor a transition to occur. If an aggressor cell is in a specific state then the victim cell is forced to an erroneous state.

2.5 V coupling fault

A fault which requires two aggressor cells to transition, thereby forcing more charge into a victim cell and causing it to flip, is described by the V-type coupling fault model [211]. If only one of the aggressor cells transitions then the victim cell will not go to an erroneous state. Having two cells transition in direct adjacency to a victim cell can only happen in certain circumstances. If all of the bits in a word are written into a single sub array, then a cell in one row can be immediately adjacent to one cell and diagonally adjacent to another. If there is a two-to-one column decode, then a victim cell can be wedged between the two aggressor cells. Further, if a multi-port memory is employed then two ports can be writing to two cells adjacent to the victim cell and thus cause it to flip. In the case of a multi-port memory, the adjacencies can be from any pair of cells in the eight cells surrounding the base cell.

3. NEIGHBORHOOD PATTERN SENSITIVE FAULT MODELS EXPANDED

3.1 Pattern Sensitive Fault Model

The pattern sensitive fault model is the all encompassing form of the neighborhood pattern sensitive fault model (NPSF). When the neighborhood becomes all of the cells in the memory the fault model is referred to as the pattern sensitive fault model (PSF) and is sometimes called an unrestricted PSF [212, 213,214].

3.2 Active Neighborhood Pattern Sensitive Fault Model

The active or dynamic neighborhood pattern sensitive fault model [215, 216] refers to a defect that is exhibited when one of the neighborhood cells transitions causing a deleterious impact on the base or victim cell. The other cells in the neighborhood must be in specific states as is the case for a NPSF.

3.3 Passive Neighborhood Pattern Sensitive Fault Model

A passive neighborhood pattern sensitive fault model describes a defect where, as all of the neighborhood cells are in specific states, the base or victim cell is unable to transition. When active and passive neighborhood pattern sensitive faults are collectively discussed they sometimes are referred to by the acronym APNPSF.

3.4 Static Neighborhood Pattern Sensitive Fault Model

A static neighborhood pattern sensitive fault model describes a defect where, as all of the neighborhood cells are in specific states, the base or victim cell is forced to a specific erroneous state.

4. RECOVERY FAULT MODELS

4.1 Sense amplifier recovery fault model

A sense amplifier can saturate due to numerous reads of one data type [217]. When the opposite data type is read the sense amplifier has a defective tendency toward the previous data type and thus does not read correctly. This type of defective operation is described by the sense amplifier recovery fault model.

4.2 Write recovery fault model

When a write is performed, the address decoder can be slow to recover. This slowness can prevent a new address from being correctly read or written. The subsequent incorrect read or write is described as by the write recovery fault model.

4.3 Slow write recovery fault model

A slow write recovery fault model is similar to the previous write recover fault model [218]. The difference is that in a slow write recovery fault a subsequent read is correct but is delayed in time.

5. STUCK OPEN FAULT MODELS

5.1 Stuck open cell fault model

A cell can have a stuck off transfer device. The transfer device can be open or the gate can simply be stuck-at a zero [219]. In either case the result is a cell, which effectively has a stuck open connection to a bit line. In the case of an SRAM, since a true and a complement bit line are utilized the stuck-open defect may only affect one data type.

5.2 Stuck open bit line fault model

A column bit line can be open part way along its path to the cells [220,221] causing some of the cells can be connected to the bit line and others, beyond the break, are inaccessible. If the sense amplifier and the write driver circuitry are on the same side of the memory array then the cells up to the point of the break can function normally.

6. IMBALANCED BIT LINE FAULT MODEL

Defective cells along a DRAM bit line can be leaking onto the bit line and inducing an erroneous potential. This kind of a defect causes unintended cells to be “accessed” and they in turn drive the bit line into an imbalanced state resulting in incorrect reads. Thus, the name of an imbalanced bit line fault model is utilized. An SRAM can also be impacted by this type of fault if leakage occurs more on one of the bit lines in a pair than on the other.

7. MULTI-PORT MEMORY FAULTS

Faults, which can be sensitized by exercising a single port, are logically referred to as single-port faults [222] and are often use the notation "1PF". Faults, which require two ports to be exercised in order to sensitize a defect, are referred to as two-port faults. These are referred to by the notation "2PF". Faults can impact one or more cells. A single-port fault that impacts one cell or two cells is referred to by the notation "1PF1s" and "1PF2s", respectively. This notation is clearly extensible [223].

A strong fault is one that can be sensitized by performing a single port operation. A weak fault is one that creates insufficient disturbance to result in a fail when only one port is exercised. When multiple ports are exercised, multiple weak faults can cumulatively impact the operation and result in a memory failure. This is similar to the V-type coupling fault.

Appendix B

Further Memory Test Patterns

Extended Listing

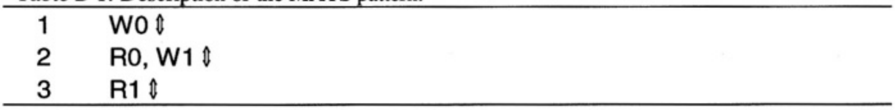
The memory test patterns listed in this appendix are a supplement to those described in chapter nine of this text. The reader should first examine chapter nine and familiarize themselves with the patterns covered there. The patterns included in chapter nine are utterly essential to successfully producing high quality memories and thereby good chips. The patterns contained in this appendix are for reference and are especially helpful as one reads literature on memory testing. Some of the patterns discussed here are only mathematically interesting while others provide very helpful insight.

1. MATS PATTERNS

1.1 MATS

The modified algorithmic test sequence, also known as MATS, is a 4N pattern [224,225]. It is focused on finding stuck-at faults as well as detecting some address decoder faults. MATS has the same length as the Zero-One pattern but is far superior.

Table B-1. Description of the MATS pattern.



1	W0 ⇕
2	R0, W1 ⇕
3	R1 ⇕

1.2 MATS+

The MATS+ pattern requires 5N operations and is considered optimal for unlinked stuck-at faults [226].

Table B-2. Description of the MATS+ pattern.

1	W0 ⇕
2	R0, W1 ⇑
3	R1, W0 ⇓

1.3 MATS++

The MATS++ pattern is an improvement on the Marching 1/0 pattern (covered next in this appendix). It is a 6n pattern and eliminates certain redundancies [227]. (A *redundancy* is a repeat of an operation that does not allow any further faults to be detected. A pattern without redundancies is said to be *irredundant.*) This pattern detects some address decoder faults, stuck-at faults, and transition faults, along with some coupling faults.

Table B-3. Description of the MATS++ pattern.

1	W0 ⇕
2	R0, W1 ⇑
3	R1, W0, R0 ⇓

1.4 Marching 1/0

The Marching 1/0 pattern detects the same faults as the MATS++, but is longer. It is a 14n pattern.

Table B-4. Description of the Marching 1/0 pattern.

1	W0 ⇑
2	R0, W1, R1 ⇑
3	R1, W0, R0 ⇓
4	W1 ⇑
5	R1, W0, R0 ⇑
6	R0, W1, R1 ⇓

2. LETTERED MARCH PATTERNS

2.1 March A

The March A pattern is a 15n pattern. It focuses on detecting linked idempotent coupling faults. It also detects address decoder faults, stuck-at faults, and transition faults not linked with idempotent coupling faults.

Table B-5. Description of the March A pattern.

1	W0 ⇕
2	R0, W1, W0, W1 ⇑
3	R1, W0, W1 ⇑
4	R1, W0, W1, W0 ⇓
5	R0, W1, W0 ⇓

2.2 March B

The March B pattern can detect linked transition and idempotent coupling faults as well as detecting address decoder faults and stuck-at faults [228]. It is a 17n pattern.

Table B-6. Description of the March B pattern.

1	W0 ⇕
2	R0, W1, R1, W0, R0, W1 ⇑
3	R1, W0, W1 ⇑
4	R1, W0, W1, W0 ⇓
5	R0, W1, W0 ⇓

2.3 March C

The March C pattern [229] had the March C- pattern derived from it. The March C is not irredundant, as can be seen in the fourth march element. Chapter nine can be examined for a detailed discussion of the March C- pattern.

Table B-7. Description of the March C pattern.

1	W0 ⇕
2	R0, W1 ⇑
3	R1, W0 ⇑
4	R0 ⇕
5	R0, W1 ⇓
6	R1, W0 ⇓
7	R0 ⇕

2.4 March X

The March X pattern takes 6n cycles and is focused on finding unlinked inversion coupling faults.

Table B-8. Description of the March X pattern.

1	W0 ⇕
2	R0, W1 ⇑
3	R1, W0 ⇓
4	R0 ⇕

2.5 March Y

The March Y pattern enables testing of linked transition and inversion coupling faults. It also detects address decoder faults and stuck-at faults. It is an 8n pattern.

Table B-9. Description of the March Y pattern.

1	W0 ⇕
2	R0, W1, R1 ⇑
3	R1, W0, R0 ⇓
4	R0 ⇕

2.6 March C+, C++, A+, A++ Patterns

The March C+ and March C++ patterns are based on the March C pattern [230]. In the March C+, each read from the March C pattern is replaced with three reads to detect disconnected pull-up and pull-down paths inside a cell. The March C++ pattern includes two delay elements that look for retention type defects. Some refer to a different pattern that is called by the same name. This March C+ pattern is like the PMOVI pattern but with a read "0" element added at the end.

The March A+ and March A++ have the same changes, i.e. the triple reads and the added delay elements, as described in the preceding paragraph.

2.7 March LA

March LA is a 22n pattern with three consecutive writes in each of the key march elements [231]. It can detect all simple faults and many linked faults.

Table B-10. Description of the March LA pattern.

1	W0 ⇕
2	R0, W1, W0, W1, R1 ⇑
3	R1, W0, W1, W0, R0 ⇑
4	R0, W1, W0, W1, R1 ⇓
5	R1, W0, W1, W0, R0 ⇓
6	R0 ⇓

2.8 March SR+

The march test for simple realistic faults, also known as March SR+, is an 18n pattern [232]. It detects stuck-at faults, transition faults, coupling faults, and numerous other faults as well. Further, the double back-to-back read detects deceptive destructive reads.

Table B-11. Description of the March SR+ pattern.

1	W0 ⇓
2	R0, R0, W1, R1, R1, W0, R0 ⇑
3	R0 ⇓
4	W1 ⇑
5	R1, R1, W0, R0, R0, W1, R1 ⇓
6	R1 ⇑

This pattern was further enhanced by the inclusion of a delay to detect subtle retention faults. This is referred to as a March SRD+ pattern.

Table B-12. Description of the March SRD+ pattern.

1	W0 ⇓
2	R0, R0, W1, R1, R1, W0, R0 ⇑
3	Pause
4	R0 ⇓
5	W1 ⇑
6	R1, R1, W0, R0, R0, W1, R1 ⇓
7	Pause
8	R1 ⇑

3. IFA PATTERNS

3.1 9N Linear

Some patterns have become known simply by their numbers. One such pattern is the 9N linear test algorithm [233]. It is also often referred to as the inductive fault analysis-9 pattern or IFA-9. This pattern is commonly used and employs a pause for retention testing.

Table B-13. Description of the 9N linear test algorithm.

1	W0 ⇕
2	R0, W1 ⇑
3	R1, W0 ⇑
4	R0, W1 ⇓
5	R1, W0 ⇓
6	Pause
7	R0, W1 ⇑
8	Pause
9	R1 ⇑

3.2 13N

The 13N pattern detects coupling faults for bits within the same word [234,235]. This pattern is also referred to as the inductive fault analsysis-13 or IFA-13 pattern. Multiple background data types are required. Chapter nine can be examined for a discussion on background data types, which are needed in some patterns. The 13N pattern was developed to detect stuck-open cell errors.

Table B-14. Description of the 13N pattern.

1	W0 ⇕
2	R0, W1, R1 ⇑
3	R1, W0, R0 ⇑
4	R0, W1, R1 ⇓
5	R1, W0, R0 ⇓
6	Pause
7	R0, W1 ⇑
8	Pause
9	R1 ⇑

4. OTHER PATTERNS

4.1 MovC

The movC was developed to ease BIST pattern implementation [236]. It is not, however, irredundant. The pattern requires 33N for each data background type.

Table B-15. Description of the movC pattern.

1	W0 ⇕
2	R0, W0, W1, R1 ⇓
3	R1, W1, W0, R0 ⇓
4	R0, W0, W1, R1 ⇓
5	R1, W1, W0, R0 ⇓
6	R0, W0, W1, R1 ⇑
7	R1, W1, W0, R0 ⇑
8	R0, W0, W1, R1 ⇑
9	R1, W1, W0, R0 ⇑

4.2 Moving Inversion

The moving inversion or MOVI pattern [237] was the precursor to the PMOVI pattern discussed in chapter nine. It involved all of the elements of the PMOVI pattern but repeated them based on the number of address inputs to the memory.

4.3 Butterfly

The butterfly pattern is quite complicated but does take fewer steps than the galloping pattern. The essence of the butterfly pattern is that the base cell is modified following a walking algorithm. After each write of a base cell, the four cells adjacent to it are read. These are in the north, south, east, and west directions. Once these four cells are read the base cell is again read. The butterfly pattern may be continued by reading the next farther out cells in these four directions, followed again by the base cell. The distance continues to be doubled until the edge of the memory or sub-array is reached. It can be seen that this pattern is rather convoluted and only DRAMs have been helped uniquely through this test.

5. SMARCH

The SMARCH pattern was described in chapter nine. The pseudo-code for this pattern is described below.

Table B-16. Pseudo-code description of the SMARCH pattern.

```
// Initialize memory to all 0s
for i = 0..N {
    for j = 0..m { Rx, W0 }
    for j = 0..m { W0, R0 }
}
// like a R0, W1, R1, W1 ⇑ march pattern element
for i = 0..N {
    for j = 0..m { R0, W1 }
    for j = 0..m { R1, W1 }
}
// like a R1, W0, R0, W0 ⇑ march pattern element
for i = 0..N {
    for j = 0..m { R1, W0 }
    for j = 0..m { R0, W0 }
}
// like a R0, W1, R1, W1 ⇓ march pattern element
for i = N..0 {
    for j = 0..m { R0, W1 }
    for j = 0..m { R1, W1 }
}
// like a R1, W0, R0, W0 ⇓ march pattern element
for i = N..0 {
    for j = 0..m { R1, W0 }
    for j = 0..m { R0, W0 }
}
// Read all the cells at 0
for i = 0..N {
    for j = 0..m { R0, W0 }
    for j = 0..m { W0, R0 }
}
```

6. PSEUDO-RANDOM

Pseudo-random patterns were described in chapter nine. In table B-16 a five-bit pseudo-random sequence is provided for reference. Note that the first and 32nd entries match as do the second and 33rd entries. After 31 cycles the pseudo-random sequence has re-started. Note also that the all zeros state is not present.

Table B-17. Example five-bit pseudo-random sequence.

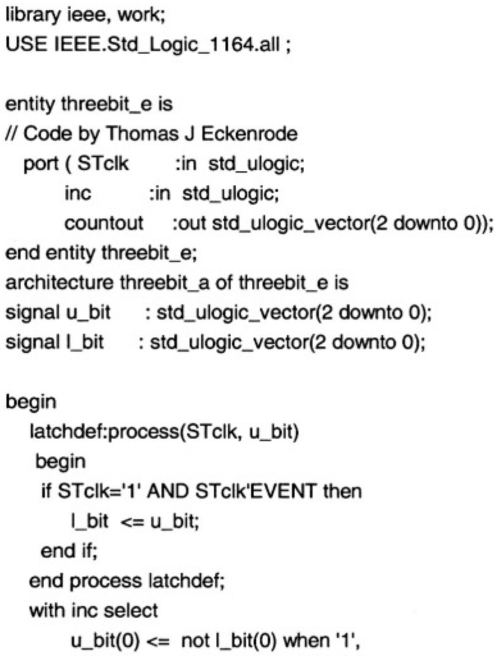
Pass	X[0]	X[1]	X[2]	X[3]	X[4]
1	1	0	0	0	0
2	0	1	0	0	0
3	1	0	1	0	0
4	0	1	0	1	0
5	1	0	1	0	1
6	1	1	0	1	0
7	1	1	1	0	1
8	0	1	1	1	0
9	1	0	1	1	1
10	1	1	0	1	1
11	0	1	1	0	1
12	0	0	1	1	0
13	0	0	0	1	1
14	1	0	0	0	1
15	1	1	0	0	0
16	1	1	1	0	0
17	1	1	1	1	0
18	1	1	1	1	1
19	0	1	1	1	1
20	0	0	1	1	1
21	1	0	0	1	1
22	1	1	0	0	1
23	0	1	1	0	0
24	1	0	1	1	0
25	0	1	0	1	1
26	0	0	1	0	1
27	1	0	0	1	0
28	0	1	0	0	1
29	0	0	1	0	0
30	0	0	0	1	0
31	0	0	0	0	1
32	1	0	0	0	0
33	0	1	0	0	0

Appendix C

State Machine HDL

Example Code

In chapter 11 a discussion was included of state machine BISTs. A series of interrelated counters define the stimulus to the memory. Table 11-2 included some VHDL code for a three-bit ripple counter. The complete VHDL follows in figure C-1.



```
library ieee, work;
USE IEEE.Std_Logic_1164.all ;

entity threebit_e is
// Code by Thomas J Eckenrode
  port ( STclk       :in  std_ulogic;
         inc         :in  std_ulogic;
         countout    :out std_ulogic_vector(2 downto 0));
end entity threebit_e;
architecture threebit_a of threebit_e is
signal u_bit     : std_ulogic_vector(2 downto 0);
signal l_bit     : std_ulogic_vector(2 downto 0);

begin
   latchdef:process(STclk, u_bit)
    begin
     if STclk='1' AND STclk'EVENT then
         l_bit  <= u_bit;
     end if;
   end process latchdef;
   with inc select
         u_bit(0) <=  not l_bit(0) when '1',
                    l_bit(0) when others;
   with inc select
        u_bit(1) <= (l_bit(0) XOR l_bit(1)) when '1',
                  l_bit(1) when others;
   with inc select
        u_bit(2) <= ((l_bit(0) AND l_bit(1)) XOR l_bit(2)) when '1',
                  l_bit(2) when others;
   countout <= l_bit;

                              end architecture threebit_a;
```

Figure C-1. VHDL deck for a three-bit ripple counter.

A key portion of a memory BIST is the read/write controller. It generates the read-enable and write-enable signals. A segment of code was included in Table 11-4. The complete Verilog code for generating the read-enable, write-enable, and data signals for the first two elements of the March C- pattern are given in figure C-2.

```
module RWGEN( clock, start, max_addr, read, write, data );
// Code by Garret S. Koch
input  clock, start, max_addr;
output read, write, data;

reg  next_state0,
   next_state1,
   next_state2,
   next_state3;

reg  state0,
   state1,
   state2,
   state3;

initial
 begin
  state0 = 1 ;
  state1 = 0 ;
  state2 = 0 ;
  state3 = 0 ;
 end
```

```
    always @ (posedge clock)
     begin
      state0 = next_state0 ;
      state1 = next_state1 ;
      state2 = next_state2 ;
      state3 = next_state3 ;
     end

    always @ (start or state0 or state3 or max_addr)
        if      ( start    )         next_state0 = 0 ;
        else if ( max_addr & state3 ) next_state0 = 1 ;
        else                         next_state0 = state0;

    always @ (start or state1 or max_addr)
        if      ( start    )         next_state1 = 1 ;
        else if ( max_addr )         next_state1 = 0 ;
        else                         next_state1 = state1;

    always @ (state1 or state3 or max_addr)
        if      ( max_addr & state1 ) next_state2 = 1 ;
        else if (!max_addr & state3 ) next_state2 = 1 ;
        else                         next_state2 = 0 ;

    always @ (state2 or max_addr)
        if      (           state2 ) next_state3 = 1 ;
        else                         next_state3 = 0 ;

    assign write = state1  | state3;
    assign read  = state2;
    assign data  = !(state1 | state2);

endmodule
```

Figure C-2. Sample Verilog for read/write generator.

The BIST address counter should be able to increment or decrement addresses and perform these operations on either a ripple word or ripple bit basis. When decrementing, the maximum address must first be loaded into the counter. All of these features are included in the counter shown in Figure C-3 [238].

```
module address_counter (a, b, c, f, inc_dec, increment, word_limit,
   bit_limit, w_limit, b_limit, word_address, bit_address,
   next_word_r, next_bit_r, scan_in, scan_out,add_comp, word_match);

// Code by R. Dean Adams, Robert M. Mazzarese, Esq.

   input a, b, c, f, scan_in, inc_dec, add_comp, word_match, increment;
   input [7:0] word_limit, bit_limit, w_limit, b_limit;

   output [7:0] word_address, bit_address, next_word_r, next_bit_r;
   output scan_out ;

   reg [7:0] next_word_r, next_bit_r, word_address, bit_address;
   reg [7:0] tempword, tempbit;

   initial begin
       next_word_r = 8'b00000000; next_bit_r = 8'b00000000;
       word_address = 8'b00000000; bit_address = 8'b00000000;
   end

   //TRANSFER pipeline values to output

   always @ (posedge f)  begin
       #1;
       word_address <= next_word_r;
       bit_address <= next_bit_r;
   end

//CALCULATE new address when increment signal received
   always @ (posedge increment) begin #1;
       casez ({add_comp,inc_dec,word_match})
           3'b10z: begin
               next_word_r = 8'b00000000;
               next_bit_r = 8'b00000000; end
           3'b11z: begin
               next_word_r = w_limit;
               next_bit_r = b_limit; end
           3'b000: begin
               next_word_r = next_word_r+1; end
           3'b010: begin
               next_word_r = next_word_r - 1; end
```

```
            3'b001: begin
                next_bit_r = next_bit_r + 1;
                next_word_r = 8'b00000000; end
            3'b011: begin
                next_bit_r = next_bit_r - 1;
                next_word_r = w_limit; end
            default: begin
                $display("address counter default");
                next_word_r = 8'b00000000;
                next_bit_r = 8'b00000000; end
        endcase
    end

/* RELOAD COUNTER whenever inc_dec changes */
    always @(inc_dec) begin #1;
        case (inc_dec)
            1'b1: begin
                next_word_r = 0; next_bit_r = 0;
                next_bit_r <= bit_limit; next_word_r <= word_limit;
                end
            1'b0: begin
                next_word_r = 1; next_bit_r = 1;
                next_bit_r <= 8'b00000000; next_word_r <= 8'b00000000;
                end
        endcase
    end
endmodule
```

Figure C-3. Address counter example.

References

[1] G.E. Moore, “Cramming more components onto integrated circuits,” Electronics, Vol. 38 No. 8, 4/19/65.

[2] G. Bell, “Some thoughts by Gordon Bell,” DOE Clusters Workshop, http://www.scl.ameslab.gov/workshops/Talks/Bell/sld001.htm, 4/10/97.

[3] G.E. Moore, “The continuing Silicon technology evolution inside the PC platform,” http://developer.intel.com/update/archive/issue2/feature.htm.

[4] P.P. Gelsinger et al., “Microprocessors circa 2000,” IEEE Spectrum, 10/89, pp. 43-7.

[5] G. Vandling, “Modeling and testing the Gekko microprocessor, an IBM PowerPC derivative for Nintendo,” International Test Conference 2001, pp. 593-9.

[6] K. Thompson, International Test Conference 1995 Keynote, Electronic Engineering Times, 10/30/95, p. 1.

[7] P. Gelsinger, “The challenges of design and test for the world wide web,” International Test Conference 2001 Keynote, p. 12.

[8] S. Parihar, et al., “A high density 0.10um CMOS technology using low K dielectric and copper interconnect,” International Electron Devices Meeting 2001, pp. 249-52.

[9] International Technology Roadmap for Semiconductors, 2001 Edition, http://public.itrs.net/Files/2001ITRS/Home.htm.

[10] K. Osada, “Universal-Vdd 0.65-2.0 V 32-kB cache using a voltage-adapted timing-generation scheme and a lithographically symmetrical cell,” IEEE Journal of Solid-State Circuits Vol. 36, No. 11,11/2001, pp. 1738-43.

[11] K. Furumochi et al., “A 500 M Hz 288kb CMOS SRAM macro for on-chip cache,” International Solid State Circuits Conference 1996, pp. 156-7.

[12] H. Pilo, R.D. Adams et al., “Bitline Contacts in High-Density SRAMS: Design for Testability and Stressability,” International Test Conference 2001, pp. 776-82.

[13] A.J. Bhavnagarwala, X. Tang, J.D. Meindl, “The impact of intrinsic device fluctuations on CMOS SRAM cell stability,” IEEE Journal of Solid-State Circuits, Vol. 36, No. 4, 4/2001, pp. 658-65.

[14] M. Tamjidi, personal communication August 2000.

[15] D. Draper et al., “Circuit techniques in a 266-MHz MMX-Enabled processor,” IEEE Journal of Solid-State Circuits, Vol. 32, No. 11,11/1997, pp. 1650-64.

[16] N. Shibata, M. Watanabe, Y. Tanabe, "A current-sensed high-speed and low-power first-in-first-out memory using a wordline/bitline-swapped dual-port SRAM cell," IEEE Journal of Solid-State Circuits, Vol. 37, No. 6, 6/2002, pp. 735-50.

[17] K. Yokomizo, K. Naito, "Design techniques for high-throughput BiCMOS self-timed SRAM's," IEICE Trans. Electron, Vol. E76-C, No. 5, 5/1993, pp. 824-9.

[18] K.J. Schultz, P.G. Gulak, "Fully-parallel multi-megabit integrated CAM/RAM design," IEEE Int. Workshop on Memory Technology, 8/1994, pp. 46-51.

[19] N. Shibata et al., "A 2-V 300-MHz 1-Mb current-sensed double-density SRAM for low-power 0.3 um CMOS/SIMOX ASICs," IEEE Journal of Solid-State Circuits, Vol. 36, No. 10, 10/2001, pp. 1524-37.

[20] N. Shibata et al., "A 2-V 300-MHz 1-Mb current-sensed double-density SRAM for low-power 0.3 um CMOS/SIMOX ASICs," IEEE Journal of Solid-State Circuits, Vol. 36, No. 10, pp. 1524-37.

[21] K. Sasaki, "A 7 nanosecond 140 milliwatt 1 Megabit CMOS SRAM with current sense amplifier," International Solid State Circuits Conference 1992, pp. 208-209.

[22] R.D. Adams, E.S. Cooley, P.R. Hansen, "A self-test circuit for evaluating memory sense-amplifier signal," International Test Conference 1997, pp. 217-25.

[23] B.S. Amutur, M.A. Horowitz, "Fast low-power decoders for RAMs," IEEE Journal of Solid-State Circuits, Vol. 36, No. 10,10/2001, pp. 1506-15.

[24] R.D. Adams et al., "An 11-ns 8k x 18 CMOS static RAM," International Solid State Circuits Conference 1988, pp. 242-3.

[25] H. Pilo et al., "A 300 M Hz, 3.3V 1Mb SRAM," International Solid State Circuits Conference 1996, pp. 148-9.

[26] R. Mookerjee, "Segmentation: a technique for adapting high-performance logic ATE to test high-density, high-speed SRAMs," IEEE Workshop on Memory Test 1993, pp. 120-4.

[27] K. Noda et al., "An ultrahigh-density high-speed loadless four-transistor SRAM macro with twisted bitline architecture and triple-well shield," IEEE Journal of Solid-State Circuits, Vol. 36, No. 3, 3/2001, pp. 510-5.

[28] M. Redeker, B.F. Cockburn, D.G. Elliott, "An investigation into crosstalk noise in DRAM structures," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 123-9.

[29] R. Flaker, IBM senior technical staff member, 1995 personal communication.

[30] IEEE Memory Technology, Design, & Test Workshop 2001, Panel on Memory Redundancy & Repair Challenges.

[31] R.D. Adams et al., "A 5 nanosecond Store Barrier Cache with Dynamic Prediction of Load / Store Conflicts in Superscalar Processors," IEEE International Solid-State Circuits Conference 1997, pp. 414-5.

[32] J.L. Hennessy, D.A. Patterson, *Computer Architecture: A Quantitative Approach,* Morgan Kaufmann, 1996.

[33] T.P. Haraszti, *CMOS Memory Circuits,* Kluwer, 2000.

[34] P. Lin, J. Kuo, "A 1-V 128-kv four-way set-associative CMOS cache memory using wordline-oriented tag-compare (WLOTC) structure with the content-addressable-memory (CAM) 10-transistor tag cell," IEEE Journal of Solid-State Circuits, Vol. 36, No. 4, 4/2001, pp. 666-75.

[35] R. Gibbins, R.D. Adams et al., "Design and Test of a 9-port SRAM for a 100Gb/s STS-1 Switch," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 83-7.

[36] S. Wood, et al., "A 5Gb/s 9-port application specific SRAM with built-in self-test," IEEE Memory Technology, Design, and Test Workshop 1995, pp. 68-73.

[37] M. Canada et al., "A 580 MHz RISC microprocessor in SOI," IEEE International Solid-State Circuits Conference 1999, pp. 430-1.

[38] S.K. Mathew et al., "Sub-500-ps 64-b ALUs in 0.18-um SOI/Bulk CMOS: Design and scaling trends," IEEE Journal of Solid-State Circuits, Vol. 36, No. 11, 11/2001, pp. 1636-46.

[39] K. Bernstein, N. Rohrer, *SOI Circuit Design Concepts,* Kluwer, 2000.

[40] J. Colinge, *Silicon-On-Insulator Technology: Materials to VLSI,* Kluwer, 1997.

[41] D.A. Johns, K. Martin, *Analog Integrated Circuit Design,* Wiley, 1997.

[42] M.S.L. Lee et al., "A physically based compact model of partially depleted SOI MOSFETs for analog circuit simulation," IEEE Journal of Solid-State Circuits, Vol. 36, No. 1, 1/2001, pp. 110-21.

[43] M. Wood, G. Smith, J. Pennings, "Converting a SRAM from bulk Si to partially depleted SOI," IEEE Custom Integrated Circuits Conference 1999, pp. 227-230.

[44] T. Ohsawa et al., "Memory design using one-transistor gain cell on SOI," International Solid State Circuits Conference 2002, pp. 152-3.

[45] R.D. Adams, P. Shephard III, "Silicon on insulator technology impacts on SRAM testing," VLSI Test Symposium 2000, pp. 43-47.

[46] R.J. Sung, et al. "Design of an embedded fully-depleted SOI SRAM," IEEE Memory Technology, Design, and Test Workshop 2001, pp. 13-8.

[47] A.G. Aipperspach et al., "A 0.2um, 1.8-V, SOI, 550-MHz, 64-b PowerPC microprocessor with copper interconnects," IEEE Journal of Solid-State Circuits, Vol. 34, No. 11, 11/1999, pp. 1430-5.

[48] A. Marshall, S. Natarajan, *SOI Design: Analog, Memory, and Digital Techniques,* Kluwer, 2002.

[49] E. MacDonald, N.A. Touba, "Delay testing of SOI circuits: Challenges with the history effect," International Test Conference 1999, pp. 269-75.

[50] E. MacDonald, "Testability and fault modeling of partially depleted silicon-on-insulator integrated circuits," Ph.D. Dissertation, University of Texas at Austin, 5/2002.

[51] F. Shafai et al., "Fully parallel 30-MHz, 2.5-Mb CAM," IEEE Journal of Solid-State Circuits, Vol. 33, No. 11,11/1999, pp. 1690-6.

[52] J. Handy, *The Cache Memory Book,* Academic Press, 1993.

[53] J. Podaima, G. Gulak, "A self-timed, fully-parallel content addressable queue for switching applications," IEEE Custom Integrated Circuits Conference 1999, pp. 239-42.

[54] D. Bradley, P. Mahoney, B. Stackhouse, "The 16kB single-cycle read access cache on a next-generation 64b Itanium microprocessor," International Solid State Circuits Conference 2002, pp. 110-1.

[55] H. Miyatake, M. Tanaka, Y. Mori, "A design for high-sped low-power CMOS fully parallel content-addressable memory macros," IEEE Journal of Solid-State Circuits, Vol. 36, No. 6, 6/2001, pp. 956-68.

[56] V. Lines et al., "66MHz 2.3M ternary dynamic content addressable memory," IEEE Memory Technology, Design, and Test Workshop 2000, pp. 101-5.

[57] L. Ternullo, R.D. Adams et al., "Deterministic self-test of a high-speed embedded memory and logic processor subsystem," International Test Conference 1995, pp. 33-44.

[58] International Technology Roadmap for Semiconductors, 2001 Edition, http://public.itrs.net/Files/2001ITRS/Home.htm.

[59] H. Yoon, et al., "A 2.5-V, 333-Mb/s/pin, 1-Gbit, double-data-rate synchronous DRAM," IEEE Journal of Solid-State Circuits, Vol. 34, No. 11,11/1999, pp. 1589-99.

[60] B. Prince, *High Performance Memories,* Wiley, 1999.

[61] B. Keeth, R.J. Baker, *DRAM Circuit Design: A Tutorial,* IEEE Solid-State Circuit Society, 2001.

[62] M. Redeker, B. Cockburn, et al., "Fault modeling and pattern-sensitivity testing for a multilevel DRAM," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 117-22.

[63] T. Sekiguchi, et al., "A low-impedance open-bitline array for multigigabit DRAM," IEEE Journal of Solid-State Circuits, Vol. 37, No. 4, 4/2002, pp. 487-98.

[64] D. Takashima, H. Nakano, "A cell transistor scalable DRMA array architecture," IEEE Journal of Solid-State Circuits, Vol. 37, No. 5, 5/2002, pp. 587-91.

[65] H. Hoenigschmid, et al., "A $7F^2$ cell and bitline architecture featuring tilted array devices and penalty-free vertical BL twists for 4-Gb DRAMs," IEEE Journal of Solid-State Circuits, Vol. 35, No. 5, 5/2000, pp. 713-18.

[66] T. Takahashi, et al., "A multigigabit DRAM technology with $6F^2$ open-bitline cell, distributed overdriven sensing and stacked-flash fuse," IEEE Journal of Solid-State Circuits, Vol. 36, No. 11, pp. 1721-7.

[67] C. Chang, J. Wang, C. Yang, "Low-power and high-speed ROM modules for ASIC applications," IEEE Journal of Solid-State Circuits, Vol. 36, No. 10, 10/2001, pp. 1516-23.

[68] B. Yand, L. Kim, "A low-power ROM using charge recycling and charge sharing," International Solid State Circuits Conference 2001, pp. 108-9.

[69] T.P. Haraszti, *CMOS Memory Circuits,* Kluwer, 2000.

[70] W.D. Brown, J.E. Brewer, Nonvolatile Semiconductor Memory Technology, IEEE Press, 1997.

[71] T. Jung, et al., "A 3.3-V single power supply 16-Mb nonvolatile virtual DRAM using a NAND flash memory technology," IEEE Journal of Solid-State Circuits, Vol. 32, No. 11, 11/1997, pp. 1748-57.

[72] J. Lee, et al., "A 1.8V 1Gb NAND flash memory with 0.12um STI process technology," International Solid State Circuits Conference 2002, pp. 104-5.

[73] K. Takeuchi, T. Tanaka, "A dual-page programming scheme for high-speed multigigabit-scale NAND flash memories," IEEE Journal of Solid-State Circuits, Vol. 36, No. 5, 5/2001, pp. 744-51.

[74] S. Atsumi, et al., "A channel-erasing 1.8-V-only 32-Mb NOR flash EEPROM with a bitline direct sensing scheme," IEEE Journal of Solid-State Circuits, Vol. 35, No. 11, 11/2000, pp. 1648-54.

[75] P. Cappelletti, et al., *Flash Memories,* Kluwer, 1999.

[76] J. Tsouhlarakis, et al., "A flash memory technology with quasi-virtual ground array for low-cost embedded applications," IEEE Journal of Solid-State Circuits, Vol. 36, No. 6, 6/2001, pp. 969-78.

[77] G. Campardo, et al., "40-mm2 3-V-only 50-MHz 64-Mb 2-b/cell CHE NOR flash memory," IEEE Journal of Solid-State Circuits, Vol. 35, No. 11, 11/2000, pp. 1655-67.

[78] M. Borgatti, et al., "A 64-min single-chip voice recorder/player using embedded 4-b/cell flash memory," IEEE Journal of Solid-State Circuits, Vol. 36, No. 3, 3/2001, pp. 516-21.

[79] J.M. Portal, et al., "An automated design methodology for EEPROM cell (ADE)," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 137-42.

[80] M. Mohammad, K.K. Saluja, "Flash memory disturbances: modeling and test," VLSI Test Symposium 2001, pp. 218-24.

[81] A. Sheikholeslami, "Ferroelectric memory design," International Solid State Circuits Conference Tutorial, 2/3/2002.

[82] B. Prince, *Emerging Memories: Technology and Trends,* Kluwer, 2002.

[83] S. Kawashima, et al., "Bitline GND sensing technique for low-voltage operation FeRAM," IEEE Journal of Solid-State Circuits, Vol. 37, No. 5, 5/2002, pp. 592-8.

[84] D. Takashima, et al., "A sub-40-ns chain FRAM architecture with 7-ns cell-plate-line drive," IEEE Journal of Solid-State Circuits, Vol. 34, No. 11, 11/1999, pp. 1557-63.

[85] M. Choi, et al., "A 0.25um 3.0V 1T1C 32Mb nonvolatile ferroelectric RAM with address transition detector(ATD) and current forcing latch sense amplifier(CFKSA) Scheme," International Solid State Circuits Conference 2002, pp. 162-3.

[86] B.F. Cockburn, "Advanced embedded memory technologies tutorial," IEEE Memory Technology, Design, and Test Workshop 2002.

[87] J. Kang, et al., "A hierarchy bitline boost scheme for sub-1.5V operation and short precharge time on high density FeRAM," International Solid State Circuits Conference 2002, pp. 158-9.

[88] G. Braun, et al., "A robust $8F^2$ ferroelectric RAM cell with depletion device(DeFeRAM)," IEEE Journal of Solid-State Circuits, Vol. 35, No. 5, 5/2000, pp. 691-6.

[89] B. Jeon, et al., "A 0.4-um 3.3V 1T1C 4-Mb nonvolatile ferroelectric RAM with fixed bitline reference voltage scheme and data protections circuit," IEEE Journal of Solid-State Circuits, Vol. 35, No. 11, 11/2000, pp. 1690-4.

[90] M. Clendenin, "Common memories turn exotic," Electronic Engineering Times, 4/29/2002, p. 60.

[91] S. Tehrani, et al., "Progress and outlook for MRAM technology," IEEE Transactions on Magnetics, Vol. 35, No. 5, 9/1999, pp. 2814-9.

[92] R.A. Sinclair, et al., "A practical 256K GMR NV memory for high shock applications," International Non-Volatile Memory Technology Conference 1998, pp. 38-42.

[93] R. Zhang, W.C. Black, M.W. Hassoun, "Windowed MRAM sensing scheme," IEEE Memory Technology, Design, and Test Workshop 2000, pp. 47-52.

[94] C. Brown, "MTJ memory seen as possible DRAM, SRAM replacement," Electronic Engineering Times, 7/23/2001, p. 65.

[95] R. Scheuerlein, et al., "A 10ns read and write non-volatile memory array using a magnetic tunnel junction and FET switch in each cell," International Solid State Circuits Conference 2000, pp. 128-9.

[96] P.K. Naji, et al., "A 256kb 3.0V 1T1MTJ nonvolatile magnetoresistive RAM," International Solid State Circuits Conference 2001, pp. 122-3.

[97] S. Lai, T. Lowrey, "OUM – a 180 nm nonvolatile memory cell element technology for stand alone and embedded applications," International Electron Devices Meeting 2001, pp. 803-6.

[98] M. Gill, T. Lowrey, J. Park, "Ovonic unified memory – a high-performance nonvolatile memory technology for stand-alone memory and embedded applications," International Solid State Circuits Conference 2002, pp. 202-3.

[99] M. Brehob, et al., "The potential of carbon-based memory systems," IEEE Memory Technology, Design, and Test Workshop 1999, pp. 110-4.

[100] R. McConnell, U. Moller, D. Richter, "How we test Siemens' embedded DRAM cores," International Test Conference 1998, pp. 1120-5.

[101] R.L. Geiger, P.E. Allen, N.R. Strader, *VLSI Design techniques for analog and digital circuits,* McGraw-Hill, 1990.

[102] A.J. van de Goor, *Testing Semiconductor Memories: Theory and Practice,* ComTex Publishing, Gouda, The Netherlands, 1998.

[103] John Barth, personal communication April 3, 2002.

[104] Bryan Robbins, personal communication 4Q 1997.

[105] R. David, A. Fuentes, B. Courtois, "Random pattern testing versus deterministic testing of RAMs," IEEE Transactions on Computers, Vol. 38, No. 5, May 1989, pp. 637-50.

[106] R. Rajsuman, "Algorithms to test PSF and coupling faults in random access memories," IEEE Int. Workshop on Memory Testing 1993, pp. 49-54.

[107] M. Franklin, K.K. Saluja, K. Kinoshita, "Design of a RAM with row/column pattern sensitive fault detection capability," International Test Conference 1989, pp. 327-36.

[108] S. Murray, "A user's approach to characterization and test of commercially available SRAMs," IEEE Memory Technology, Design, and Test Workshop 1998, p. 68.

[109] R.D. Adams, E.S. Cooley, "Analysis of a deceptive destructive read memory fault model and recommended testing," IEEE North Atlantic Test Workshop 1996.

[110] R.D. Adams, E.S. Cooley, "False write through and un-restored write electrical level fault models for SRAMs," IEEE Memory Design, Technology, and Test Workshop 1997, pp. 27-32.

[111] D. Niggemeyer, M. Redeker, J. Otterstedt, "Integration of non-classical faults in standard march tests," International Test Conference 1998, pp. 91-6.

[112] Jon Lachman, personal communication August 6, 2001.

[113] S. Hamdioui, A.J. van de Goor, "Address decoder faults and their tests for two-port memories," IEEE Memory Technology, Design, and Test Workshop 1998, pp. 97-103.

[114] S. Al-Harbi, S.K. Gupta, "An efficient methodology for generating optimal and uniform march tests," VLSI Test Symposium 2001, pp. 231-7.

[115] J. Otterstedt, D. Niggemeyer, T. Williams, "Detection of CMOS address decoder open faults with march and pseudo random memory tests," International Test Conference 1998, pp. 53-62.

[116] M. Sachdev, "Test and testability techniques for open defects in RAM address decoders," International Test Conference 1996, pp. 428-34.

[117] R.D. Adams, E.S. Cooley, "The limits of digital testing for dynamic circuits," VLSI Test Symposium 1999, pp. 28-32.

[118] P. Nagaraj, et al. "Defect analysis and a new fault model for multi-port SRAMs," IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems 2001, pp. 366-74.

[119] F. Karimi, et al., "A parallel approach for testing multi-port static random access memories," IEEE Memory Technology, Design, and Test Workshop 2001, pp. 73-81.

[120] J. Zhao, et al., "Detection of inter-port faults in multi-port static RAMs," VLSI Test Symposium 2000, pp. 297-302.

[121] C.A. Dean, Y. Zorian, "Do you practice safe test? What we found out about your habits," International Test Conference 1994, pp. 887-92.

[122] H. Bonges III, R.D. Adams, et al., "A 576K 3.5ns access BiCMOS ECL static RAM with array built-in self-test," IEEE Journal of Solid-State Circuits, Vol. 27, No. 4, 4/1992, pp. 649-56.

[123] A.J. van de Goor, *Testing Semiconductor Memories: Theory and Practice*, ComTex Publishing, Gouda, The Netherlands, 1998.

[124] H. Maeno, S. Iwade, S. Kayano, "Embedded RAM test using flag-scan register," Electronics and Communications in Japan, Part 3, Vol. 77, No. 4,1994, pp. 110-18.

[125] R. Treuer, V.K. Agarwal, "Built-in self-diagnosis for repairable embedded RAMs," IEEE Design & Test of Computers, 6/1993, pp. 24-32.

[126] Y. Matsuda, et al., "A new array architecture for parallel testing in VLSI memories," International Test Conference 1989, pp. 322-6.

[127] L.Ternullo, R.D. Adams, et al., "Deterministic self-test of a high-speed embedded memory and logic processor subsystem," International Test Conference 1995, pp. 33-44.

[128] A.J. van de Goor, et al., "March LR: A test for realistic linked faults," VLSI Test Symposium 1996, pp. 272-80.

[129] A.J. van de Goor, "Using march tests to test SRAMs," IEEE Design & Test of Computers, 3/1993, pp. 8-13.

[130] J. Sosnowski, "In system testing of cache memories," International Test Conference 1995, pp. 384-93.

[131] B. Nadeau-Dostie, A. Silburt, V.K. Agarwal, "Serial interfacing for embedded-memory testing," IEEE Design & Test of Computers, 4/1990, pp. 52-63.

[132] A.J. van de Goor, "Automatic computation of test length for pseudo-random memory tests," IEEE International Workshop of Memory Technology 1995, pp. 56-61.

[133] A. Krasniewski, K. Gaj, "Is there any future for deterministic self test of embedded RAMs," IEEE Computer Society Press, 4/1993, pp. 159-68.

[134] Y. Zorian, "BIST for embedded memories," EE-Evaluation Engineering, 9/1995, pp. 122-3.

[135] W.K. Al-Assadi, A.P. Jayasumana, Y.K. Malaiya, "On fault modeling and testing of content-addressable memories," IEEE Memory Technology, Design, and Test Workshop 1994, pp. 78-83.

[136] K. Lin, C. Wu, "Functional testing of content-addressable memories," IEEE Memory Technology, Design, and Test Workshop 1998, pp. 70-5.

[137] P.R. Sidorowicz, "Modeling and testing transistor faults in content-addressable memories," IEEE Memory Technology, Design, and Test Workshop 1999, pp. 83-90.

[138] R.D. Adams, P. Shephard III, "Silicon on insulator technology impacts on SRAM testing," VLSI Test Symposium 2000, pp. 43-47.

[139] Y. Wu, L. Calvin, "Shadow write and read for at-speed BIST of TDM SRAMs," International Test Conference 2001, pp. 985-94.

[140] K. Zarrineh, R.D. Adams, et al., "Self test architecture for testing complex memory structures," International Test Conference 2000, pp. 547-56.

[141] P. Mazumder, K. Chakraborty, *Testing and Testable Design of High-Density Random-Access Memories*, Kluwer, 1996.

[142] S. Hamdioui, "Testing multi-port memories: Theory and practice," Ph.D. Dissertation, Delft University, 2001.

[143] J. Zhao, et al., "Detection of inter-port faults in multi-port static RAMs," VLSI Test Symposium 2000, pp. 297-302.

[144] M.L. Bushnell, V.D. Agrawal, *Essentials of Electronic Testing: For Digital, Memory & Mixed Signal VLSI Circuits*, Kluwer, 2000.

[145] S.K. Jain, C.E. Stroud, "Built-in self-testing of embedded memories," IEEE Design & Test, 10/86, pp. 27-37.

[146] Y. Zorian, "BIST for embedded memories," EE-Evaluation engineering, 9/1995, pp. 122-3.

[147] C.E. Stroud, *A Designer's Guide to Built-In Self-Test*, Kluwer, 2002.

[148] B.F. Cockburn, Y.F. Nicole Sat, "Transparent built-in self-test scheme for detecting single V-coupling faults in RAMs," IEEE Memory Technology, Design, and Test Workshop 1994, pp. 119-24.

[149] K. Thaller, "A highly-efficient transparent online memory test," International Test Conference 2001, pp. 230-9.

[150] H. Bonges III, R.D. Adams, et al., "A 576K 3.5ns access BiCMOS ECL static RAM with array built-in self-test," IEEE Journal of Solid-State Circuits, Vol. 27, No. 4, 4/1992, pp. 649-56.

[151] R. Mookerjee, "Segmentation: a technique for adapting high-performance logic ATE to test high-density, high-speed SRAMs," IEEE Workshop on Memory Test 1993, pp. 120-4.

[152] P.H. Bardell, W.H. McAnney, J. Savir, *Built-In Test for VLSI,* Wiley, 1987.

[153] R. David, A. Fuentes, B. Courtois, "Random pattern testing versus deterministic testing of RAMs," IEEE Trans. on Computers, Vol. 38, No. 5, 5/89, pp. 637-50.

[154] W.W. Peterseon, E.J. Weldon, *Error-Correcting Codes,* MIT Press, 1972.

[155] A.J. van de Goor, "Automatic computation of test length for pseudo-random memory tests," IEEE International Workshop on Memory Technology 1995, pp. 56-61.

[156] V.N. Yarmolik, M. Nicolaidis, O. Kebichi, "Aliasing free signature analysis for RAM BIST," International Test Conference 1994, pp. 368-77.

[157] M. Franklin, K.K. Saluja, K. Kinoshita, "Design of a RAM with row/column pattern sensitive fault detection capability," International Test Conference 1989, pp. 327-36.

[158] J. van Sas, et al., "BIST for embedded static RAMs with coverage calculation," International Test Conference 1993, pp. 339-47.

[159] X. Li, F. Yang, "A cost-effective BIST scheme for embedded RAMs," International Conference on CAD and Computer Graphics 1993, Vol. 2, pp. 647-51.

[160] R.D. Adams, et al., "A 370-MHz memory built-in self-test state machine," European Design and Test Conference 1995, pp. 139-41.

[161] C. Hunter, et al., "The PowerPC 603 microprocessor: an array built-in self test mechanism," International Test Conference 1994, pp. 388-94.

[162] P. Camurati, et.al., "Industrial BIST of embedded RAMs," IEEE Design & Test of Computers, Fall 95, pp. 86-95.

[163] M. Franklin, K.K. Saluja, "Embedded RAM testing," IEEE Memory Technology, Design, and Test Workshop 1995, pp. 29-33.

[164] R. Dekker, F. Beenker, A. Thijssen, "Fault modeling and test algorithm development for static random access memories," International Test Conference 1988, pp. 353-61.

[165] V.D. Agrawal, C.R. Kime, K.K. Saluja, "A tutorial on built-in self-test," IEEE Design & Test of Computers, 6/1993, pp. 69-77.

[166] Thomas J. Eckenrode, personal communication July 2002.

[167] Garrett S. Koch, personal communication July 2002.

[168] R.D. Adams, R. Mazzarese, "A high-speed pipelined memory built-in self-test state machine," Dartmouth – Thayer School of Engineering HDL-Based System Design Project, May 1996.

[169] M. Nicolaidis, O. Kebichi, V. Castor Alves, "Trade-offs in scan path and BIST implementations for RAMs," Journal of Electronic Testing: Theory and Applications, Vol. 5, No. 4, 5-6/1994, pp. 147-57.

[170] R. Gibbins, R.D. Adams et al., "Design and test of a 9-port SRAM for a 100Gb/s STS-1 switch," IEEE Memory Technology, Design, and Test Workshop 2002.

[171] V. Castro Alves, et al., "Built-in self-test for multi-port RAMs," International Conference on Computer Aided Design 1991, pp. 248-51.

[172] Y. Wu, S. Gupta, "Built-in self-test for multi-port RAMs," Asian Test Symposium 1997, pp. 398-403.

[173] J. Zhu, "An SRAM built-in self-test approach," EE-Evaluation engineering, 8/1992, pp. 90-3.

[174] H. Koike, T. Takeshima, M. Takada, "A BIST scheme using microprogram ROM for large capacity memories," International Test Conference 1990, pp. 815-22.

[175] M.H. Tehranipour, Z. Navabi, S.M. Fakhraie, "An efficient BIST method for testing of embedded SRAMs," International Symposium on Circuits and Systems 2001, pp. 73-6.

[176] K. Zarrineh, R.D. Adams et al., "Self test architecture for testing complex memory structures," International Test Conference 2000, pp. 547-556.

[177] J. Dreibelbis, et al., "Processor-based built-in self-test for embedded DRAM," IEEE Journal of Solid-State Circuits, Vol. 33, No. 11,11/1998, pp. 1731-40.

[178] A. Benso, et al., "A programmable BIST architecture of clusters of multiple-port SRAMs," International Test Conference 2000, pp. 557-66.

[179] D. Youn, T. Kim, S. Park, "A microcode-based memory BIST implementing modified march algorithm," Asian Test Symposium 2001, pp. 391-5.

[180] J. Barth, et al., "Embedded DRAM design and architecture for the IBM 0.11 um ASIC offering," IBM Journal of Research and Development, to be published.

[181] P. Jakobsen, et al., "Embedded DRAM built in self test and methodology for test insertion," International Test Conference 2001, pp. 975-84.

[182] R. Gibbins, R.D. Adams et al., "Design and Test of a 9-port SRAM for a l00Gb/s STS-1 Switch," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 83-7.

[183] IEEE Memory Technology, Design, & Test Workshop 2001, Panel on Memory Redundancy & Repair Challenges.

[184] E. Rondey, Y. Tellier, S. Borri, "A silicon-based yield gain evaluation methodolgy for embedded-SRMAs with different redundancy scenarios," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 57-61.

[185] H. Kikukawa, et al., "0.13-um 32-Mb/64-Mb embedded DRAM core with high efficient redundancy and enhanced testability," IEEE Journal of Solid-State Circuits, Vol. 37, No. 7, 7/2002, pp. 932-40.

[186] T. Namekawa, et al., "Dynamically shift-switched dataline redundancy suitable for DRAM macro with wide data bus," IEEE Journal of Solid-State Circuits, Vol. 35, No. 5, 5/2000, pp. 705-12.

[187] J. Barth, et al., "A 300MHz multi-banked eDRAM macro featuring GND sense, bit-line twisting and direct reference cell write," International Solid State Circuits Conference 2002, pp. 156-7.

[188] M. Yamaoka, et al., "A system LSI memory redundancy technique using an ie-Flash (inverse-gate-electrode flash) programming circuit," IEEE Journal of Solid-State Circuits, Vol. 37, No. 5, 5/2002, pp. 599-604.

[189] M. Ouellette, et al., "On-chip repair and an ATE independent fusing methodology," International Test Conference 2002.

[190] J. Wee, et al., "A post-package bit-repair scheme using static latches with bipolar-voltage programmable anti-fuse circuit for high-density DRAMs," IEEE Journal of Solid-State Circuits, Vol. 37, No. 2, 2/2002, pp. 251-4.

[191] T. Takahashi, et al., "A multigigabit DRAM technology with $6F^2$ open-bitline cell, distributed overdriven sensing and stacked-flash fuse," IEEE Journal of Solid-State Circuits, Vol. 36, No. 11, pp. 1721-7.

[192] K. Chakraborty, "A physical design tool for built-in self-repairable RAMs," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 9, No. 2, 4/2001, pp. 352-64.

[193] S. Tanoi, "On-wafer BIST of a 200Gb/s failed-bit search for 1Gb DRAM," International Solid State Circuits Conference 1997, pp. 70-1.

[194] Y. Wu, S. Gupta, "Built-in self-test for multi-port RAMs," Asian Test Symposium 1997, pp. 398-403.

[195] Y. Nagura, "Test cost reduction by at-speed BISR for embedded DRAMs," International Test Conference 2001, pp. 182-7.

[196] D.K. Bhavsar, "An algorithm for row-column self-repair of RAMs and its implementation in the Alpha 21264," International Test Conference 1999, pp. 311-8.

[197] A. Meixner, J. Banik, "Weak write test mode: An SRAM cell stability design for test technique," International Test Conference 1996, pp. 309-18.

[198] H. Pilo, R.D. Adams, et al., "Bitline contacts in high density SRAMs: Design for testability and stressability," International Test Conference 2001, pp. 776-82.

[199] J. Brauch, J. Fleischman, "Design of cache test hardware on the HP PA8500," International Test Conference 1997, pp. 286-93.

[200] Y. Wu, L. Calin, "Shadow write and read for at-speed BIST of TDM SRAMs," International Test Conference 2001, pp. 985-94.

[201] R. Gibbins, R.D. Adams et al., "Design and Test of a 9-port SRAM for a l00Gb/s STS-1 Switch," IEEE Memory Technology, Design, and Test Workshop 2002, pp. 83-7.

[202] H. Pilo, et al., "Design-for-test methods for stand-alone SRAMs at 1Gb/s/pin and beyond," International Test Conference 2000, pp. 436-43.

[203] K. Takeda, et al., "Quasi-worst-condition built-in-self-test scheme for 4-Mb loadless CMOS four-transistor SRAM macro," Symposium on VLSI Circuits 2001, pp. 229-30.

[204] J. Yamada, et al., "A 128-kb FeRAM macro for contact/contactless smart-card microcontrollers," IEEE Journal of Solid-State Circuits, Vol. 37, No. 8, 8/2002, pp. 1073-9.

[205] H. Kikukawa, et al., "0.13-um 32-Mb/64-Mb embedded DRAM core with high efficient redundancy and enhanced testability," IEEE Journal of Solid-State Circuits, Vol. 37, No. 7, 7/2002, pp. 932-40.

[206] K. Zarrineh, S.J. Upadhyaya, V. Chickermane,"System-on-chip testability using LSSD scan structures," IEEE Design & Test of Computers, 5-6/2001, pp. 83-97.

[207] A.J. van de Goor, G.N. Gaydadjiev, "An analysis of (linked) address decoder faults," IEEE Memory Technology, Design, and Test Workshop 1997, pp. 13-9.

[208] D. Niggemeyer, E.M. Rudnick, "Automatic generation of diagnostic march tests," VLSI Test Symposium 2001, pp. 299-304.

[209] http://www.tuxedo.org/~esr/jargon/html/entry/idempotent.html

[210] V. Castro Alves, et al., "Built-in self-test for multi-port RAMs," International Conference on Computer Aided Design, 11/1991, pp. 248-51.

[211] B.F. Cockburn, Y.F. Nicole Sat, "Transparent built-in self-test scheme for detecting single V-coupling faults in RAMs," EEE International Workshop on Memory Technology, Design, and Testing 1994, pp. 119-24.

[212] J.P. Hayes, "Detection of pattern-sensitive faults in random-access memories," IEEE Trans. on Computers, Vol. C-24, No. 2, 1975, pp.150-7.

[213] J.P. Hayes, "Testing memories for single-cell pattern-sensitive faults in semiconductor random-access memories," IEEE Trans, on Computers, Vol. C-29, No. 3, 1980, pp. 249-54.

[214] M.G. Karpovsky, V.N. Yarmolik, "Transparent memory testing for pattern sensitive faults," International Test Conference 1994, pp. 860-869.

[215] D.S. Suk, S.M. Reddy, "Test procedures for a class of pattern-sensitive faults in semiconductor random-access memories," IEEE Trans. on Computers, Vol. C-29, No. 6, pp. 419-29.

[216] K.K. Saluja, K. Kinoshita, "Test pattern generation for API faults in RAM," IEEE Trans. on Computers, Vol. C-34, No. 3, 1985, pp. 284-7.

[217] P. Camurati, et al., "Industrial BIST of embedded RAMs," IEEE Design & Test of Computers, Fall 1995, pp. 86-95.

[218] M.S. Abadir, H.K. Reghbati, "Functional testing of semiconductor random access memories," Computing Surveys, Vol. 15, No. 3, 9/83, pp. 175-98.

[219] B. Nadeau-Dostie, A. Silburt, V.K. Agarwal, "Serial interfacing for embedded-memory testing," IEEE Design & Test of Computers, 4/1990, pp. 52-63.

[220] S. Griep, et al., "Application of defect simulation as a tool for more efficient failure analysis," Quality and Reliability Engineering Int., Vol. 10, 1994, pp. 297-302.

[221] Y. Lai, "Test and diagnosis of microprocessor memory arrays using functional patterns," M.S. Thesis, MIT, May 1996.

[222] S. Hamdioui, A. J. van de Goor, et al., "Realistic fault models and test procedure for multi-port SRAMs," IEEE Memory Technology, Design, and Test Workshop 2001, pp. 65-72.

[223] S. Hamdioui, A. J. van de Goor, et al., "Detecting unique faults in multi-port SRAMs," Asian Test Symposium 2001, pp. 37-42.

[224] J. Knaizuk, C.R.P. Hartmann, "An optimal algorithm for testing stuck-at faults in random access memories," IEEE Trans. on Computers, Vol. C-26, No. 11, 1977, pp. 1141-1144.

[225] R. Nair, "Comments on 'An optimal algorithm for testing stuck-at faults in random access memories,'" IEEE Trans. on Computers, Vol. C-28, No. 3, 1979, pp. 258-61.

[226] M.S. Abadir, J.K. Reghbati, "Functional testing of semiconductor random access memories," ACM Computing Surveys, Vol. 15, No. 3, 1983, pp. 175-98.

[227] A.J. van de Goor, *Testing Semiconductor Memories: Theory and Practice,* Wiley, 1991.

[228] M. Nicolaidis, O. Kebichi, V. Castor Alves, "Trade-offs in scan path and BIST implementations for RAMs," JETTA, Vol. 5, 1994, pp. 273-83.

[229] M. Marinescu, "Simple and efficient algorithms for functional RAM testing," IEEE Test Conference 1982, pp. 236-9.

[230] K. Zarrineh, S.J. Upadhyaya, "A new framework for automatic generation, insertion and verification of memory built-in self-test units," VLSI Test Symposium 1999, pp. 391-6.

[231] A.J. van de Goor, "March LA: A test for linked memory faults," European Design & Test Conference 1997, p. 627.

[232] S. Hamdioui, "Testing multi-port memories: Theory and practice," Ph.D. Dissertation, Delft University, 2001.

[233] R. Dekker, F. Beenker, A. Thijssen, "Fault modeling and test algorithm development for static random access memories," International Test Conference 1988.

[234] R. Dekker, F. Beenker, LFSR. Thijssen, "A realistic self-test machine for static random access memories," International Test Conference 1988, pp. 353-61.

[235] J. Zhu, "An SRAM built-in self-test approach," EE-Evaluation Engineering, 8/1992, pp. 90-3.

[236] C. Hunter, et al., "The PowerPC 603 microprocessor: an array built-in self test mechanism," International Test Conference 1994, pp. 388-94.

[237] J.H. de Jonge, A.J. Smeulders, "Moving inversions test pattern is thorough, yet speed," Computer Design, May 1976, pp. 169-73.

[238] R.D. Adams, R. Mazzarese, "A high-speed pipelined memory built-in self-test state machine," Dartmouth – Thayer School of Engineering HDL-Based System Design Project, May 1996.

GLOSSARY / ACRONYMS

ABIST – Array built-in self-test. This term is synonymous with memory built-in self-test.
AF – Address decoder fault.
Aggressor – Cell which causes erroneous operation in a victim cell.
ATE – Automated test equipment.
ATPG – Automatic test pattern generation.

Beta ratio – Typically the ratio between the pull-down strength and transfer strength in an SRAM cell.
BIST – Built-in self-test
Bit oriented – A memory which is accessed one bit at a time.
Bridging defect – A short between to signal lines.

CAM – Content addressable memory
CF – Coupling fault.

DRAM – Dynamic random access memory.

EEPROM – Electrically erasable programmable read only memory.

FeRAM – Ferroelectric random access memory.
Flash – A type of EEPROM where large portions of memory can be erased simultaneously.

Galloping pattern – Pattern which ping-pong addresses through all possible address transitions.

LFSR – Linear feedback shift register. Used to generate pseudo-random patterns.

Marching pattern – Sequentially addresses memory, leaving new data type in its wake.
MISR – Multiple input signature register.
MRAM – Magnetoresistive random access memory.

NPSF – Neighborhood pattern sensitive fault.

PROM – Programmable read only memory.

ROM – Read only memory.

SOC – System on chip.
SOI – Silicon on insulator.
SRAM – Static random access memory.

Ternary CAM – CAM which stores "1", "0", or “don’t care” on a per bit basis.
TF – Transition fault.

Victim – Cell which is impacted by operation in aggressor cell.

Walking pattern – Sequentially addresses memory, leaving old data type in its wake.
Word oriented – A memory which is accessed one word at a time.

INDEX

ABOUT THE AUTHOR

R. Dean Adams has over 20 years of experience in the design and development of embedded and stand-alone memories, high-performance custom logic, and DFT/BIST circuits for IBM. He is currently a member of IBM's Design-For-Test (DFT) department, where he consults on numerous memory and advanced microprocessor design projects. Dean is a leading expert in high-performance memory BIST, as well as the modeling and testing of standard and custom logic for leading edge semiconductor technologies.

He holds Ph.D., M.S., and Master of Engineering Management degrees from Dartmouth, at the Thayer School of Engineering. He also has a B.S.E.E. degree from the University of Rhode Island. Dean is the inventor or co-inventor of over 20 patents, and has authored or co-authored numerous technical papers. Most of these papers and patents relate to memory testing, memory self-test, and memory design.