Reconfigurable testable bit-serial multiplier for DSP applications

Prof. M.A. Bayoumi
C.H. Yang

Indexing terms: Digital circuits, Reliability, Fault detection

Abstract: A new testable and reconfigurable bit-serial multiplier is proposed. Fault tolerance is established through built-in self-testing and dynamic reconfiguration. During the reconfiguration phase, the faulty modules of the multiplier are automatically isolated and the system reconfigures itself for the required function with no need for external interference. Quadruple-double modular redundancy (QDMR) techniques are employed to allow the isolation of the bad modules and the use of only good ones. The faulty cells are located and the diagnostic information is scanned out for evaluation. The followed design method supports a two-level testing strategy, in which the multiplier is tested by a small built-in test circuit and the test circuit itself is tested externally by a scan-path technique, providing high fault coverage. This design method adopts the functional building block concept and is especially suitable for linear arrays. The multiplier is based on the bit-serial approach, which has been proven to be an efficient implementation for several digital-signal-processing (DSP) structures; it accepts 2 s complement data and coefficients in a serial form. A prototype of an 8-bit multiplier has been implemented in single-layer metal 2 μm CMOS technology; it has an area of 2.64 mm × 1.91 mm (without I/O pads) and contains approximately 2500 devices.

1 Introduction

With recent advances in VLSI technology, very complex digital-signal-processing (DSP) algorithms can be cost-effectively implemented. But at the same time, the design complexity to achieve high-speed performance, efficient area and reliability becomes a major challenge. Reliability has recently gained wide importance, especially in several critical fields such as radar communications and real-time image processing for robotics applications. Non-traditional design methods and architectures are required to achieve this high performance within the VLSI constraints. In this paper, a novel reconfigurable testable multiplier architecture is presented. The multiplier is based on the bit-serial approach, to minimise the area. Pipelining is employed to enhance the speed and bring it closer to that of bit-parallel processing. Reliability is provided by including redundancy in hardware, testing mechanisms for all cells, fault detection and location at each cell, and dynamic reconfigurability.

In our design, reliability has been established through two phases: testing and fault tolerance. Testing has been a major challenge in integrated circuits. Traditionally, testing was considered only after the circuit had been fabricated, which requires providing probing testing pads, sophisticated test equipments, and heavy computation time for test generation and fault simulation. With the rapid increase in the complexity of VLSI systems, testing has become more difficult or even impossible [1]. This has been the motivation for adopting the 'design for testability' strategy, in which extra circuits are included on a chip to make testing manageable and economical. Scan-design, such as the level-sensitive scan design (LSSD) [2] and scan path [3], has been widely used to increase the circuit controllability and observability (the main testing criteria [4]) and to reduce the test generation for sequential circuit into combinational circuit. However, since the test data are generated outside the circuit under test (CUT) and the responses must be evaluated outside the circuit, the information must be shifted serially in and out of the circuit. This technique makes testing a time-consuming procedure. A more sophisticated approach, the built-in self-test (BIST), has been introduced to achieve a more effective and economical testing strategy [5, 6]. In this paper, a 'two-level testing' strategy is developed, which incorporates both BIST and scan-path approaches; BIST is used to reduce the test application and evaluation time while scan-path is used to guarantee the correctness of the BIST circuitry.

Fault-tolerance has also become highly desirable, since increasing VLSI silicon area means reducing yield. Although current technologies are much more reliable than the early ones, the resulting decrease in the failure rate has been offset by the increased complexity of today's VLSI circuits. In this paper, quadruple-double modular redundancy (QDMR) has been employed to provide fault diagnosis and fault tolerance. This technique is based on modulator duplication concept, and is applicable for one-dimensional unidirectional arrays. In the QDMR approach, the first cell contains four identical modules, and the rest contain two identical modules each. This would provide dynamic reconfiguration for fault-tolerance and would also provide fault diagnosis information. Designing a multiplier is the vehicle to demonstrate our proposed techniques.

The multiplier represents the main computational kernel and the bottleneck of most DSP algorithms. The performance of any DSP processor depends very much on the multiplier efficiency. The bit-serial approach [7, 8] has gained wide acceptance as a trade-off between the

Paper 6646E (C2), first received on the 8th July 1987 and in revised form on the 25th July 1988

The authors are with the Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, LA 70504 USA

design cost and the operation cost for real-time DSP applications. It offers maximum flexibility and extensibility, and high clock rate.

2 The multiplier design strategy

The main design philosophy is reliability, which is achieved in two phases:

(*a*) testing to detect faults

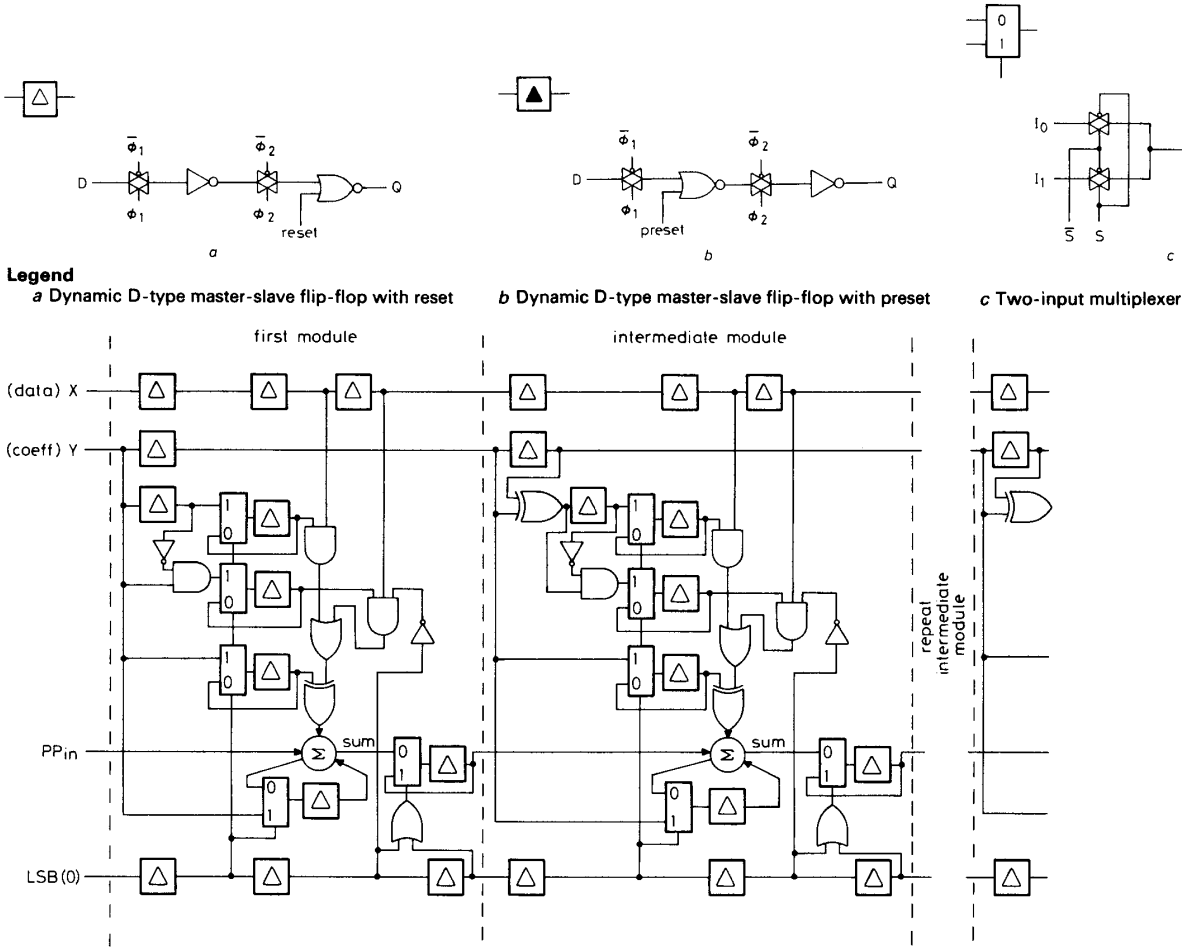
(*b*) locating faults and using dynamic reconfiguration to isolate fault modules and to utilise the good ones.

In the testing phase, the entire multiplier is tested by a small built-in test circuit and the test circuit itself is tested externally by a scan-path technique. This would significantly reduce the test turnaround time, while making sure that the testing is proceeded correctly. The second phase is based on utilising redundancy through dynamic reconfiguration. In this approach, the fault modules are isolated automatically and the system reconfigures itself to perform the required function. This technique is contrary to the common static reconfiguration, where an external process, such as a tailored discretionary interconnection, is needed to isolate the faulty modules and to connect the good ones. The dynamic reconfiguration techniques offer a good solution to the limitations experienced when using the static approach, such as: the need to probe each element prior to the final metallisation, the area consumed by the probing pads, and the cost of generating a distinct interconnection pattern and of exposing a tailored photoresist for the discretionary wiring.

Fig. 1 illustrates a non-fault-tolerant bit-serial multiplier. It is composed of three basic modules: the first module, the intermediate modules, and the last module. Its function will be discussed in detail in Section 3. The block diagram of the proposed multiplier is shown in Fig. 2. It has four different units: cell_*cu*, cell_1, cell_2 and cell_*N*. Cell_*cu* functions as a control unit for the entire multiplier. It includes multiplexers (MUX) to separate normal inputs with test inputs, a linear-feedback shift register (LFSR), a set of shift registers, a timing-signal generator, and a global comparator. Cell_1 has four first-multiplier modules (booth_1), two block comparators (CMP), and an arbiter circuit. Cell_2 has two intermediate multiplier modules (booth_2) and an arbiter, like the one in the cell_1. Cell_*N* has two last multiplier modules (booth_*N*) and an arbiter. The fault model used in this paper is similar to that used by previous researchers [9, 10]; it assumes single permanent faults, and the proposed multiplier can tolerate only one fault in each multiplier cell.

The proposed multiplier operates in three separate modes with two inputs, for 'reconfiguration_mode' and 'scan_mode'. These modes are summarised as follows:

(*a*) *The normal mode:* By keeping the scan_mode and reconfiguration_mode pins to logical low, our circuit will work as a normal serial multiplier (modified Booth



Legend

a Dynamic D-type master-slave flip-flop with reset

b Dynamic D-type master-slave flip-flop with preset

c Two-input multiplexer

Fig. 1 *Non-fault-tolerant serial modified Booth multiplier*

algorithm). The cell_*cu* has no influence on the multiplier circuit, but the LFSR in the cell_*cu* is activated all the time.

the AR sent from the preceding cell, and the arbiter will then let one module be responsible for the cell's operation. In this way cell_2 to cell_*N* have fault-tolerance

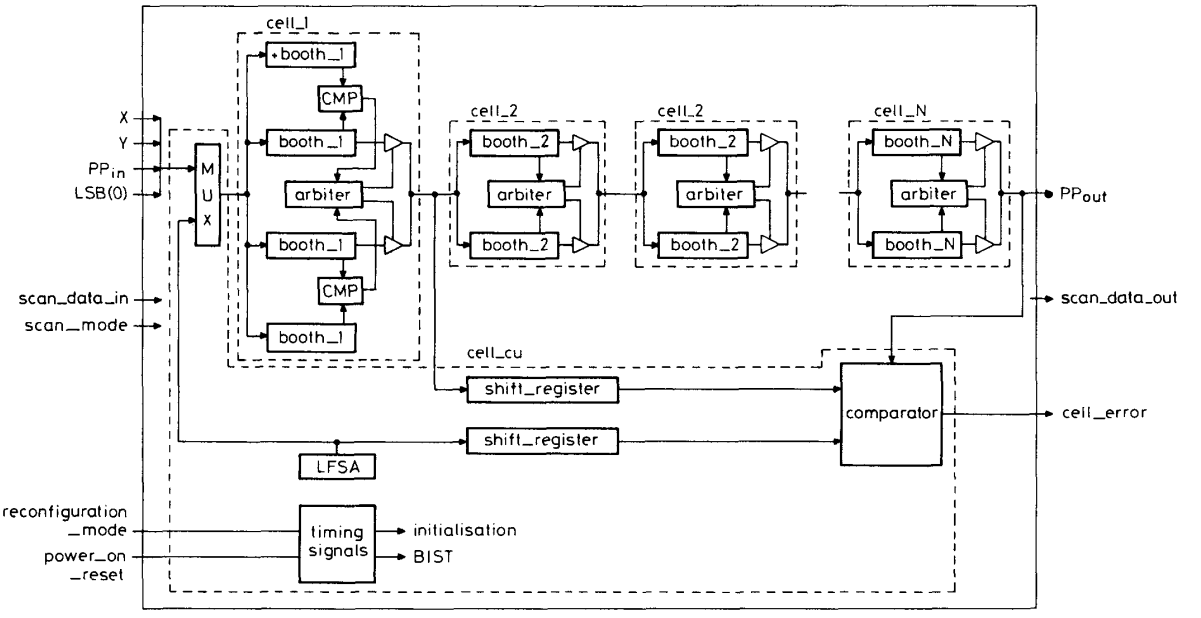


Fig. 2 *Fault-tolerant testable multiplier*

(*b*) *The reconfiguration mode:* The multiplier should set to this mode before the operation starts. By setting reconfiguration_mode pin to logical high, the multiplier will reconfigure itself. The control unit cell cell_cu immediately clears all internal flip-flops in cell_1 to cell_*N*, then fetches 4 bits from the LFSR as a pseudorandom test pattern and sends it to cell_1. The four identical modules (booth_1) in cell_1 are organised as two self-checking blocks, and each two modules in the same block simultaneously perform the same computation.

The 4-bit test pattern (X, Y, PP_{in}, and LSB(0)) will propagate through cell_1 at the same speed. Cell_1 also generates an extra 'anticipated response' (AR) bit and sends them to cell_2. The AR is actually the partial product output (PP_{out}) in the multiplier module during the normal mode, since now the test pattern is propagated through the cell and the original PP_{out} wire of the multiplier module is used for propagating PP_{in} signal during this reconfiguration mode. The AR of both modules in each block is being compared in each clock cycle. If the comparison results in a match, then both modules are assumed to be fault-free with respect to the current test pattern and the block is also assumed to be fault-free. The output of this block comparison (from the CMP) is sent to the arbiter; the arbiter will disable the block having a mismatch signal and allow only one module response for the cell's operation (Note: since the first and the last module in a modified Booth multiplier are slightly different from the intermediate modules, the booth_1 and booth_*N* will modify themselves to have the same function as booth_2). The second cell (cell_2) has two identical modules only; the basic idea is that each module will use its PP_{out} to compare with the AR sent from cell_1 and enable or disable its modular outputs. The test pattern and AR from the cell_1 will propagate through cell_2 to the next cell. Cell_*N* also has two modules and each modules will compare its PP_{out} with capability while keeping the redundant circuit to minimum (double modulator redundancy). Fig. 3*a* shows the multiplier operation when all modules are fault-free. Figs. 3*b* and *c* illustrate two independent reconfigurations due to faulty modules presented in the multiplier.

The AR produced by cell_1 is sent to a shift register in cell_*cu*, it also propagates through cell_2 to cell_*N* and then returns to cell_*cu*, where the global checking is done by comparing the AR from cell_1 with the AR from cell_*N*. The test patterns (generated by cell_*cu*) are sent to another shift register in cell_*cu* (for global checking), and at the same time are propagated through all multiplier cells and finally return from cell_*N* back to the control unit, where another global checking is done by comparing the original with the returned data. The global checking is done every clock cycle and the 'cell_error' output will go to logical high if any mismatch occurs. The cell_error is used to indicate that there are intolerable faults inside the multiplier.

(*c*) *The scan mode:* By setting the scan_mode pin to logical high, the multiplier operates in the scan mode. This mode allows the control unit cell_*cu* to be tested externally. A small number of deterministically generated patterns are sent serially from the scan_data_in pin, and then reset the scan_mode pin, wait for at least one clock cycle and then set scan_mode pin again to scan out the response. The fault diagnosis information, stored in the two error registers of the arbiter circuit in cell_1 to cell_*N*, will be available during the reconfiguration mode and this information is also shifted out for evaluation by using this scan mode.

3 Circuit design

A pipeline multiplier of K-bit data and N-bit coefficient requires $K + N$ bit times (clock cycles) per operation, but

can do one operation every K bit times, by starting one operation before finishing the previous operation, with

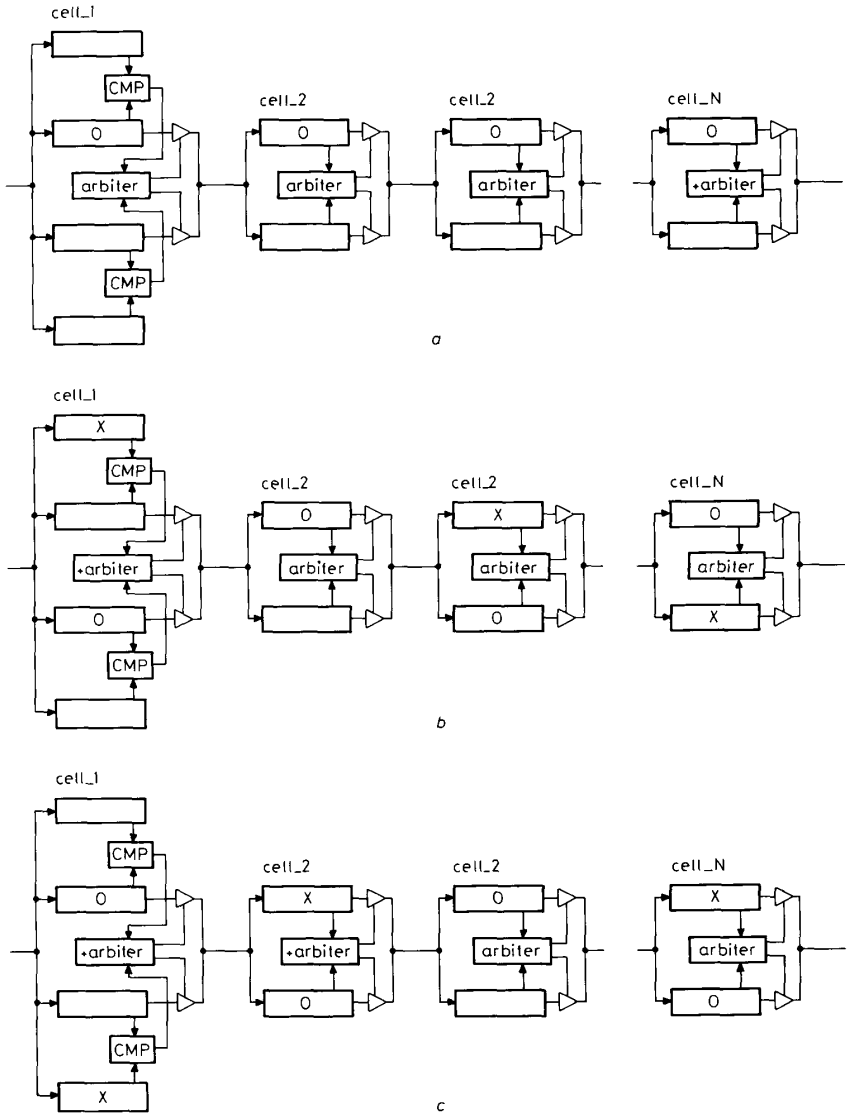


Fig. 3 *Reconfiguration scheme*
a All modules good
b Reconfiguration occurs owing to faulty modules
c Reconfiguration occurs owing to faulty modules
○: module responsible for the cell's operation
×: faulty modules

data and coefficient applied serially. The main constraint on bit-serial systems is the latency of the primitives. It is possible to reduce the latency of the multiplier by using the modified Booth algorithm [11, 12]. In this algorithm, the coefficients are recorded in such a manner, to halve the number of computational steps involved, and so halve the size of the processing array. The main achievement of this recoding scheme is that the products of these digits with the multiplicand (data) are implemented by optional shifts.

The circuit of the serial-modifed Booth multiplier [13], already shown in Fig. 1, consists of three basic modules: the first (leftmost) module, the intermediate modules, and the last (rightmost) module. The first module is different from the intermediate module in the recoding circuit, and the last module is different from the intermediate module in the sign extension circuit. For a $2N$-bit coefficient, the multiplier will have one first module, $(N-2)$ intermediate modules and one last module. The input data (X) and coefficient (Y), both in 2's complement form, are input serially with least significant bits (LSBs) first. The timing signal LSB(0) is true (high) to indicate the arrival of LSBs of new X and Y. The partial product input (PP_{in}) is used for truncation/rounding control. To avoid multiplication overflow, the top three bits of the data X must be identical. The LSB of the product output (PP_{out}) appears $3N+1$ clock cycles after the LSBs of the inputs X and Y.

The proposed multiplier takes the advantages of BIST and on-chip redundancy, while also providing fault-

diagnostic capability. The four basic cells of the multipliers, cell_1, cell_2, cell_N, and cell_cu, are discussed in more detail below:

(*a*) *cell_1:* This cell has quadruple-modular redundancy, four identical booth_1 multiplier modules. This 'module shadowing' concept was introduced by Intel [14]. Every two modules works as a self-checking block, and the outputs of the two blocks are controlled by an arbiter circuit. If one module in a block fails then the entire block is assumed to be faulty, and the output of the block comparator will be activated; this mismatch signal informs the arbiter to disable the faulty block. Fig. 4

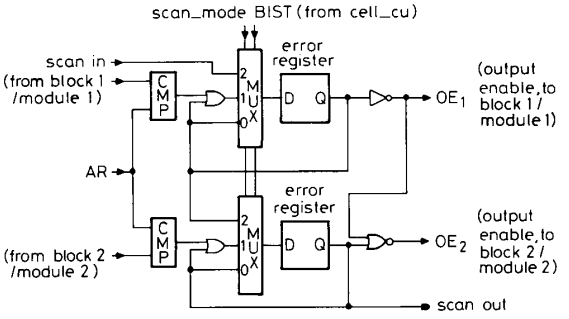


Fig. 4 *Arbiter circuit*

shows the arbiter circuit used in cell_1 to cell_N. The AR input in the arbiter of cell_1 is always logical low, since each block in cell_1 already have a comparator and cell_1 will generate the AR to the following cell. The error registers in the arbiter tells whether or not there is a mismatch in the blocks. This information can be serially scanned out during the scan mode. The circuit booth_1 is basically the multiplier first module with some extra circuitry to allow it to function as an intermediate multiplier module and propagate the test pattern through itself during the reconfiguration mode. At the reconfiguration mode, the test pattern (4 bits; X, Y, PP_{in}, and appear as inputs to cell_2. Although self-testing and fault-tolerance may also be achieved by using a triple redundancy with a voter unit in this cell, the quadruple redundancy with the arbiter just described (this arbiter is also used in cell_2 and cell_N) is considered to be easier to implement in VLSI owing to its regular structure.

(*b*) *cell_2:* This cell has double-modular redundancy (two booth_2 multiplier modules); the outputs are controlled by an arbiter circuit. The circuit of booth_2 is basically the multiplier intermediate module with some extra circuits to allow the test pattern and AR to propagate through it. During the reconfiguration mode, cell_2 will receive test patterns and AR from its preceding cell and the PP_{out} response of each booth_2 module is compared in the arbiter with the AR. The arbiter will use the comparison signals to decide which module should be responsible for the cell output, and enable the appropriate output buffer. The contents of the two error registers in the arbiter is scanned out for fault diagnosis during the scan mode. The test pattern and the AR (5 bits in total) will propagate through the cell to the following cell after 3 clock cycles.

(*c*) *cell_N:* This is similar to cell_2; it has double modular redundancy and two identical booth_N modules. The output is also controlled by an arbiter circuit. But, during the reconfiguration mode it functions as an intermediate module in cell_2 and generates the response which lets the arbiter decide which module should be responsible for the cell's outputs. The test pattern and AR will propagate through it without changing and, after 3 clock cycles, return to cell_cu.

(*d*) *cell_cu:* By activating the reconfiguration_mode input, cell_cu will generate a one-clock-cycle-width pulse to reset the internal flip-flops in cell_1 to cell_N and then start the self-testing and reconfiguration processes. Fig. 5

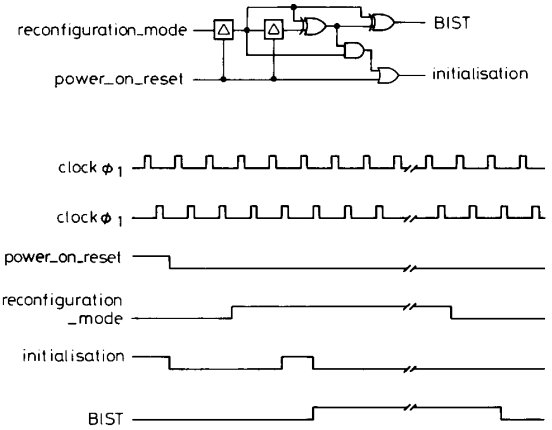


Fig. 5 *Control signal timing*

shows the circuit of a timing signal generator and the timing relationship between the signals; the 'power_on_reset' is only used to preset the LFSR. Cell_cu also contains a set of shift registers to store the test pattern generated by the free-running linear feedback shift register (LFSR) [15] and the AR generated by cell_1 for global checking. Global error checking of the entire multiplier is done by comparing the test pattern set to cell_1 with the test pattern returned from cell_N, and by comparing the AR generated by cell_1 with the AR returned from cell_N. Any mismatch in the comparison will activate the 'cell_error' output. The occurrence of cell_error signal should be interpreted as: there are too many faulty cells to be tolerated, or failure occurs in the interconnections between cells, or the failure occurs in the control unit.

4 Reliability analysis

The reliability function $R(t)$ is defined as the probability that a component will perform satisfactorily from time zero to time t, given that the operation is successful at time zero. The reliability function for a nonredundant component, using the general Poisson distribution, is $R(t) = e^{-\lambda t}$ where λ is the constant failure rate of the component. (We assume that the component is in the normal-life region, and therefore the failure rates λ is constant). For a general bit-serial multiplier with $2N$-bit coefficient, the nonredundant reliability R_0 is [16]:

$$R_0 = \prod_{i=1}^{i=N} R_i \approx R_m^N \tag{1}$$

where R_m is the reliability of the multiplier intermediate module. Since the circuit complexities of the multiplier modules (booth_1, booth_2, and booth_N) are approx-

imately the same, we consider them to have same reliability: i.e. $R_{\text{booth 1}} = R_{\text{booth 2}} = R_{\text{booth N}} = R_m$.

A simple reliability model for the redundant multiplier using our quadruple-double modulator redundancy (QDMR) technique is

$$R_{\text{QDMR}} = \bar{R}_1 \times \bar{R}_2^{(N-1)} \tag{2}$$

where $\bar{R}_1$ is the reliability of the entire cell_1, which includes four multiplier modules, and $\bar{R}_2$ that of cell_2. To compare R_{QDMR} with R_0, $\bar{R}_1$ and $\bar{R}_2$ should be correlated in terms of the module reliability R_m. Since cell_1 is composed of two parallel self-checking block outputs controlled by an arbiter, its reliability $\bar{R}_1$ is

$$\bar{R}_1 = R_{\text{hardcore 1}} \times (R_{\text{block}}^2 + 2R_{\text{block}}(1 - R_{\text{block}})) \tag{3}$$

where $R_{\text{hardcore 1}}$ is the reliability of hardcore circuitry in cell_1, which includes the arbiter, output switching, and matching circuitry. Since the failure rate λ tends to be proportional to the circuit complexity, we could estimate the relative failure rates from the relative transistor counts. By inspecting the circuit complexity of the hardcore circuitry to the basic multiplier module, we found that they have the relative circuit complexity of 0.28 : 1: therefore

$$R_{\text{hardcore 1}} = e^{-(0.28\lambda_m)t} = (e^{-\lambda_m t})^{0.28} = r_m^{0.28} \tag{4}$$

The self-checking block has two parallel modules, and its reliability R_{block} is therefore

$$R_{\text{block}} = R_{\text{CMP}} \times (R_m^2 + 2R_m(1 - R_m)) \tag{5}$$

where R_{CMP} is the reliability of the output comparator (a simple XOR gate) in the block of cell_1. The relative complexity of XOR gate to multiplier module is about 0.05 : 1, therefore $R_{\text{CMP}} = R_m^{0.05}$. In cell_2, two parallel module outputs are controlled by an arbiter. Its reliability R_2 is

$$\bar{R}_2 = R_{\text{hardcore 2}} \times (R_m^2 + 2R_m(1 - R_m)) \tag{6}$$

where $R_{\text{hardcore 2}}$ is the reliability of the hardcore circuitry in cell_2, and $R_{\text{hardcore 2}} = R_{\text{hardcore 1}} = R_m^{0.28}$.

From eqns. 2–6, and the aforementioned analysis, we get the reliability of the proposed multiplier R_{QDMR}, written as a function of the reliability R_m of the basic module as follows:

$$\begin{aligned} R_{\text{QDMR}} &= \bar{R}_1 \times \bar{R}_2^{(N-1)} \\ &= R_m^{1.33}(2 - R_m)(2 - R_m^{1.05}(2 - R_m)) \\ &\quad \times (R_m^{1.28}(2 - R_m))^{(N-1)} \end{aligned} \tag{7}$$

Values of R_{QDMR} and the reliability R_0 (eqn. 1) of a non-fault-tolerant multiplier are plotted in Fig. 6 as a function of module reliability R_m and N. It can be seen that the redundant multiplier maintains a higher reliability (the curves of a redundant multiplier are above those of a nonredundant multiplier). Fig. 7 shows a plot of the dependence of the reliability improvement (R_{QDMR}/R_0) on N and R_m. The plot illustrates a factor of 3 to 40 superiority in reliability predicted for the QDMR over the nonredundant model for 8 to 32 bit multipliers. Fig. 7 also shows that when the module reliability R_m is approximately zero, the reliability of redundant multiplier is lower than the nonredundant multiplier. This is because the redundant multiplier has spare components that tolerate failures, but when the probability of each component to fail is very high, there are merely more components to fail. Since this situation occurs in the wear-out period of the multiplier, we can ignore it.

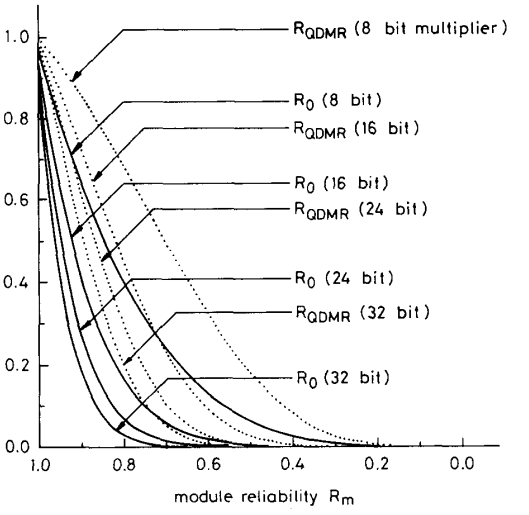


Fig. 6 *R_{QDMR} and R_0 as a function of R_m*

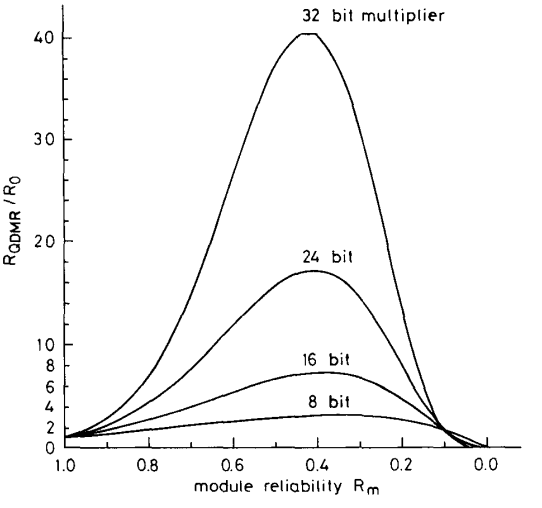


Fig. 7 *R_{QDMR}/R_0 for various coefficient lengths as a function of R_M*

5 Implementation

A prototype of an 8 bit coefficient of the proposed multiplier has been implemented in 2 μm CMOS single-metal technology using two approaches. In the first, a standard static CMOS logic is used, in which the multiplier (without I/O pads) occupies an area of 4.86 mm × 2.91 mm and contains approximately 3500 devices. In the second approach, a domino CMOS logic [17] is followed in which the implementation area measures 2.64 mm by 1.91 mm and has about 2500 devices. Approximately 35% of total silicon area is occupied by the BIST and scan-path circuitry for the 8 bit multiplier; however, this area overhead decreases as we increase the size of the multiplier. A simulation program indicates that the multiplier should operate at a maximum clock rate of about 25 MHz, i.e. with a cycle time of about 40 ns. The BIST and scan-path circuitry account for about 8 ns delay on the multiplier's critical path during the normal-mode operation. A layout has been generated, Fig. 8, and is ready to be implemented on a CMOS DSP chip. The design of a fault simulator and a dedi-

cated test pattern generator for the scan path is under way.

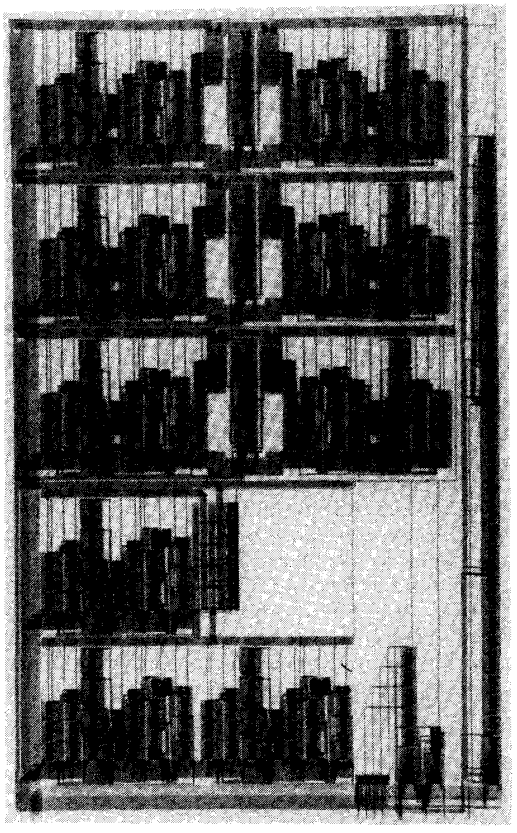


Fig. 8 *Layout of a domino CMOS prototype*

6 Conclusions

We have presented a novel, easily testable and fault-tolerant multiplier. By combining the built-in self-testing with self-reconfiguration, we eliminate the need for complicated test pattern generation, response analysis, and externally controlled reconfiguration. The employment of QDMR and two-level testing strategy provides complete fault tolerance, fault diagnosis and testability. The design is simple and has highly regular structure which produces an efficient VLSI implementation. The field reliability is significantly increased without seriously compromising the operation speed.

7 Acknowledgment

The authors would like to thank the referees for their helpful comments and suggestions to the original text.

8 References

1 IBARRA, O.H., and SAHNI, S.K.: 'Polynomially complete fault detection problems', *IEEE Trans.*, 1975, **C-24,** pp. 242–249
2 EICHELBERGER, E.B., and WILLIAMS, T.W.: 'A logic design structure for LSI testability', *Proc. 14th Design Automation Conf.*, 1977, pp. 462–468
3 FUNATSU, S., WAKATSUKI, N., and ARIMA, T.: 'Test generation systems in Japan', *Proc. 12th Design Automation Symp.*, 1975, pp. 114–122
4 WILLIAMS, T.W., and PARKER, K.P.: 'Design for testability — a survey', *Proc. IEEE*, 1983, pp. 98–112
5 KONEMANN, B., MUCHA, J., and ZWIEHOFF, G.: 'Built-in logic block observation techniques', *1979 Test Conf.*, pp. 37–41
6 LEBLANC, J.J.: 'LOCST: a built-in self-test technique', *IEEE Des. Test Comput.*, 1984, **1,** pp. 45–52
7 LYON, R.F.: 'A bit-serial VLSI architectural methodology for signal processing', *in* GRAY, J.P. (Ed.): VLSI 81 (Academic Press, 1981), pp. 131–140
8 MURRAY, A.F., and DENYER, P.B.: 'A CMOS design strategy for bit-serial signal processing', *IEEE J. Solid-State Circuits*, 1985, pp. 746–753
9 SHEN, J.P., and FERGUSON, F.J.: 'The design of easily testable VLSI array multipliers', *IEEE Trans. Comput.*, 1984, **C-33,** pp. 554–560
10 KANOPOULOS, N.: 'A bit-serial architecture for digital signal processing', *IEEE Trans. Circuits and Systems*, 1985, **CAS-32,** pp. 289–291
11 MACSORLEY, O.L.: 'High-speed arithmetic in binary computers', *Proc. IRE*, 1961, **49,** pp. 67–91
12 RUBINFIELD, L.P.: 'A proof of the modified booth's algorithm for multiplication', *IEEE Trans. Comput.*, 1975, **C-24,** pp. 1014–1015
13 LYON, R.F.: 'Two's complement pipeline multipliers', *IEEE Trans.*, 1976, **CON-24,** pp. 418–425
14 JOHNSON, D.: 'The Intel 432: a VLSI architecture for fault-tolerant computer systems', *Computer*, 1984, **17,** pp. 40–48
15 GOLUMB, S.W.: 'Shift register sequence' (Aegean Park Press, Laguna Hills, California, 1982)
16 INGLE, A.D., and SIEWIOREK, D.P.: 'A reliability model for various switch designs in hybrid redundancy', *IEEE Trans.*, 1976, **C-25,** pp. 115–133
17 KRAMBECK, R.H., LEE, C.M., and LAW, H.S.: 'High speed compact circuits with CMOS', *IEEE J. Solid-State Circuits*, 1982, **SC-17,** pp. 614–619