TESTING THE BUS GUARDIAN UNIT OF THE FTMP

by

Manoj Kumar Jha

John P. Robinson

Electrial and Computer Engineering

University of Iowa

January 1983

Prepared for

Langley Research Center

NAG 1-195

TABLE OF CONTENTS

I. INTRODUCTION

The fault-tolerant multiprocessor (FTMP) is a digital computer architecture which has evolved over a decade in connection with several life-critical aerospace applications. It is designed to have a failure rate due to random causes of the order of 10^{-10} failures per hour, on ten-hour flights where no air-borne maintenance is available. The design is based on independent processor-cache memory modules and common memory modules which communicate via redundant serial buses. All information processing and transmission is conducted in triplicate so that local voters in each module can correct single errors. Modules can be retired and/or reassigned in any configuration. Job assignments are all made on a floating basis, so that any processor triad is eligible to execute any job step.

The FTMP comprises of ten identical line replaceable units (LRU's). Each LRU contains a processor region, a slave region, a clock generation region, two bus guardian units (BGU's), system bus interface circuits, and a power subsystem. The slave region contains a number of subsystems all of which are addressed, read, and written as locations in the system bus address space. These sybsystems are the system memory module, the I/O port, LRU control/status and communication registers, and a real-time clock/counter. The BGU's control the access that any element of an LRU has to the system bus. In order that the LRU be enabled to transmit on any line of the system bus, it is necessary that the enabling bits from both BGU's within the LRU be set. Either BGU can block the LRU's ability to transmit on a line.

The system bus interconnects all LRU's of the FTMP, and is the transfer medium for exchanging all data, clocking, control and status signals. The bus system is quintiply redundant, consisting of five identical bus sets. Each bus set is composed of four lines. These are a Poll (P) line, a processor tra smit (T) line, a processor receive (R) line, and a clock (C) line. Each LRU of the system is interfaced such that it always receives all bus lines. In addition it can be dynamically configured such that it may transmit on any bus line. The design of the interface circuitry of the LRU is such that any single point fault within an LRU can at most disable one of the bus sets. The BGU's within each LRU are responsible for providing enabling signals to the interface circuitry which allow that LRU to transmit on particular bus lines.

II. FTMP OPERATION

2.1 Basic Principles

The basic operating principles of the FTMP require that triads of processors and memories be formed and that each member of a triad be assigned to transmit on a different bus set. The triads operate in tight synchronism so that the transmissions from each element of a triad will be synchronized with one another. It is thus possible to listen to three separate bus lines on which a triad is transmitting, and to perform a majority vote to synthesize a correct transmission even if one element of the triad should fail. It is also possible to note and record any disagreements between elements of a triad which might become apparent during this voting process. The receiving party is responsible for performing the vote and error masking function.

Synchronous operation requires that all LRU's operate with a common time base. This common time base is provided by the clocking system. Clock generators within each LRU lock to a voted version of the redundant system clocks which apear on the system bus clock lines. These redundant clock signals are provided by gating the clock generator outputs of selected LRU's onto the C lines of the system bus. The selected LRU's effectively lock to each other, while all other LRU's lock to the selected LRU's.

2.2 System Bus Operation

The system bus interconnects all LRU's of the FTMP providing the timing, control, and data paths for all intermodule communication and synchronization. There are four types of buses, each having five redundant copies. The classification is based on their use, i.e., on the nature of information they carry. The construction of all twenty buses is identical.

Each bus operates as a wired 'OR'. If there are simultaneous transmissions from several LRU's onto one bus line, then the result is the logical 'OR' of the multiple transmission signals. The transmission line termination is such that the undriven state (when no transmissions are in progress) is a logical 'zero'.

2.2.1 The Clock Bus

The C buses are used to distribute redundant copies of a common system wide time base, called the system clock. At any one time either three or four of the C lines are active, the other line(s) being either failed or spare. All active C lines carry independent copies of the system clock. All of these copies are identical, except for some small time skews between them which result from circuit variations and propagation delay variations due to LRU location on the bus. An LRU receives all C lines, selects 3 of the 5 lines, performs a majority circuit reduction of those 3 signals to 1, and phase locks its own crystal oscillator to that reduced signal. Since all LRU's can be configured to lock to the active C lines, each LRU's crystal oscillator can in effect be locked to the same common time base.

The LRU's themselves are used as the source of the individual system clock signals. The system is either configured such that four LRU's have their internal clock signal gated out onto separate C lines and each of these LRU's selects and locks to the clock signals of the three other LRU's, or it is configured such that only three LRU's have their internal clock signal gated onto separate C lines and each of these LRU's selects and locks to the clock signals of the other two LRU's and itself The quad configuration allows all correctly functioning LRU's to remain synchronized with one another in the presence of any single fault in the clock system. LRU's which are not system clock sources can select and lock to any three or the four active C lines. The triplex configuration is nearly as good excepting for certain single point failures which could induce loss of synchronization. In this case since there are only three active C lines, every LRU selects and locks to the same three signals.

2.2.2 The Poll Bus

The poll lines are used to arbitrate among the processor triads seeking control of the system bus. At any one time three of the five P lines are active, the other two being either failed or spare. Each element of the processor triad is assigned to a different one of the three active P lines. A processor triad participates in a competitive poll when it seeks control of the system bus. During a poll simultaneous transmissions onto the P lines by multiple triads may occur. The signal value on any one line is the wired 'OR' of the multiple trasmissions onto that line. The resultant value read from the P lines is then the majority function of these wired 'OR' signals.

2.2.3 The Processor Transmit Bus

The T lines are used in transmitting processor triad read and write commands. Each element of a processor triad is assigned to transmit on a different T line. All elements of a processor triad transmit their read or write commands in tight synchronism with one another. The address and data bits are transmitted as a series of pulse width modulated pulses.

2.2.4 The Processor Receive Bus

The R lines are used in receiving slave region replies to processor triad system bus read commands. At any one time three of the five R lines are active, the other two being either failed or spare. Each slave region is assigned to transmitt on one of the active R lines. These assignments are made such that each element of a system memory triad is assigned to transmit on a different R line.

The read command may be replied by a single slave region, as is the case when reading I/O port registers of LRU status registers. In such cases the receiving processor triad must accept data from one of the lines of the active R lines. Voting among redundant copies of the same transmission is not done in this case, but instead this single copy is accepted as is. In this case synchronization among the incoming bits of multiple copies is not necessary. This is not the only form of a read transaction, however. Three slave regions may reply to a read comand, as is the case when reading system memory or the real-time clock/counter. In these cases the receiving processors must accept the redundant copies of the incoming data from the R bus, synchronize the multiple bit streams and vote so as to mask any single element errors. The data bits are transmitted as a series of pulse width modulated pulses, as in the case of T bus transmissions.

III. FAULT-MODELING IN THE BGU

3.1 The Role of BGU

The overall integrity of the system relies upon the ability to reliably control the access that any element of the LRU has to the system bus. It is the BGU's responsibility to protect the system bus from a failed or malfunctioning LRU or element within the LRU. The reliability of the BGU is, therefore, paramount in the overall performance of the system.

Each BGU provides 20 enabling lines, each of which runs to the driver of a bus interface circuit. The driver must have an enabling signal from both BGU's of the LRU before it will allow transmission by the LRU. The status of the enabling lines from a BGU is held by that BGU in enabling registers. There are four of them, each of 5 bits. Enabling register R1 contains the enabling bits for each of the five P bus lines. Similarly, R2 contains the enabling bits for each of the five T bus lines, R3 contains those for the five R bus lines, and R4 contains those for the five C bus lines. A one stored in a particular bit position provides an enabling signal, a zero provides a disabling signal. In addition, there is a fifth register R5, of four bits, which is used to hold the 3-out-of-5 select code to be used by the BGU's T bus input select circuitry.

3.2 Fault Behavior

If there is an error in the reception of an address word, a valid command may not be recognized or a command to another BGU may be executed. A command word failure may occur in either the register address field (bits 8 thru 10), and or the data field (bits 0 thru 4). If there is an error in just the data field, the proper register will be selected, but the data written into it will be corrupted. Because of an error in the address field, the addressed register within the BGU may not be written into, and/or some other register may be written into erroneously. In addition, faults could be present in the address comparison circuit itself, or on the lines that go to the registers.

The 3/5 input select, deskewers and voter hardware present a more complex testing problem. With all three slected T lines operating correctly, most single hardware faults in this area are masked. For example any one of the three deskewer units could fail completely and as long as the other two paths are receiving correct T bus data, the voter would yield correct information. To sensitize this area of masking hardware, tests with only 2 of 3 input T lines will be necessary.

3.3 Fault Diagnosis in Brief

A general procedure for the diagnosis of these faults would consist of the following steps:

1) Initialize registers to some known value.
2) Issue a write command to change the contents of one of the registers. Issue some of these commands in 2 of 3 form in order to test the masking hardware.
3) Verify that the data has been written correctly in the addressed register, and the contents of other registers have not been changed.

The BGU registers are not directly readable, and their contents can be observed only as their effects in the subsequent performance of the LRU. In order to make sure that the BGU under test is in control, the output lines from the other BGU will be assumed to be at logic one.

IV. TESTING THE BGU

4.1 Overview

The BGU can be broken down into two loop connected sections; from external inputs to voter output and from voter output to BGU registers. In order to test the input select, deskewers and voter section some input combinations corresponding to input errors are needed. For example if only 2 of the 3 selected T bus lines have good data the BGU should still recognize a valid command. The second section of the BGU (S/P converter, decoder and registers) does not incude error masking hardware and is conceptually more easily tested.

Any complete test of the BGU would include a sequence of data writes into the BGU registers followed by a test of their integrity, such that all bits in all the registers go through the complete cycle 0 1 0 (or 1 0 1). Since the initial power-up value is 0 for all bits in all registers, and also since 0 is a disable signal, it is more convenient to use the cycle $0 \rightarrow 1 \rightarrow 0$ rather than $1 \rightarrow 0 \rightarrow 1$. .

It is necessary to assume that the testing program is going to be part of the system program, and the system programmer has access to individual LRU's. In other words, the actual redundancy underlying the buses, system memory triads, procesor triads, etc., which is invisible to the applciations programmer, must be controllable by the test program. Otherwise there is no way in which the redundant hardware such as the deskewers and voter can be tested.

The system program itself can be run on either an external monitor or on one of the LRU's. However, using a single LRU as the testing unit may not be reliable enough. The reliability of the test can be improved by using a triad of procesors as the testing unit. It will be assumed that the test program is going to be run on a triad.

4.2 The Obervability Problem

Any testing algorithm includes the following three steps:

1) Initialize

2) Write

3) Verify

As mentioned in the section 3.3, verification for the BGU at the redundant bus level is possible only through indirect observation. This imposes some restrictions on the test data. Verification of a data write into one of the bus enable registers is done by making the LRUT (this abbreviation will be used for the LRU containing the BGUT, where BGUT itself stands for the BGU under test) transmit on the bus lines, and then verifying the presence of transmission on the lines that were enabled, and the absence of it on lines that were disabled. However, the presence or absence of a transmission cannot be observed directly, and it can be felt only as its effect on the LRU's receiving it. Since there are five buses of each type, and a LRU can select three of these, two LRU's are necessary and sufficient to observe the buses. One of these two LRU's can be configured to select buses 1, 2 and 4 of all types, and the other to select buses 3, 4 and 5 of all types. These LRU's will be denoted as L(1,2,4) and L(3,4,5) respecitvely. The select code for buses 1, 2 and 4 is 0000, the initial power-up value, which is one reason why this particular combination has been chosen.

Returning to the observability problem, note that an error bit is set whenever a selected input bus carries a bit different from what the other two selected. There are four error registers corresponding to the four bus types. Each register has five bits, corresponding to the five redundant bus lines. The error bit that is set indicates the bus that is in disagreement with the other two input buses. Thus in L(1,2,4) if the error bit corresponding to bus line C1 is set, it means that the data on C1 is different from that on C2 and C4. If transmission is present (or absent) or all three input buses, no error bit would be set, provided the same data stream has been transmitted on them. If transmission is present (absent) on two input buses, and absent (present) on the third bus, the error bit corresponding to the third bus would be set. Thus the observability is very limited — the most information that can be deduced from the error latches is a partitioning of the five redundant buses into two disjoint subsets, one of which is being used for transmission, the other not. There is no way to tell which one of the two subsets is active. This precludes the use of certain test patterns that would be more efficient, but verification can still be done in most cases, since only single errors are being considered.

4.3 Selection of Test Sequence

To verify all $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions in the bus enable registers, the fastest test cycle would be $00000 \rightarrow 11111 \rightarrow 00000$. However, because of the problem just mentioned in the preceding section, this cycle cannot be used. When all buses are disabled (enable register contains 00000), there is no transmission on any bus, and consequently, no error. When all buses are enabled (enable register contains 11111), there is transmission on every bus and again, no error. Thus, there is no way to distinguish the two situations, unless we go deeper into the system to observe directly the transmissions on the redundant bus structure (this problem occurs whenever all five bits are complemented). However, in order to avoid possible system problems, transmissions for testing purposes only would be chosen such that they do not affect anything beyond the error latches, i.e., they would be chosen to be 'harmless'. the only alternative left is to increase the length of the test cycle.

One such cycle is 00000 → 00111 (enable 1,2,3) → 11100 (enable 3,4,5) → 00000 (disable all). The leftmost bit corresponds to bus #5 and the rightmost bit to bus #1. However, there is a problem with this cycle for C buses. At least three clock lines are busy all the time (four in quad mode), leaving a maxium of two clock lines free at any time. Hence, if more than two clock uses are enabled (as in 00111 or 11100), the LRUT cannot be prevented from transmitting its own lock on a line that is already carrying the system clock signals. Multiple transmissions on a clock bus are detrimental to the system clock synchronization, and are to be avoided. But even if we assume that there is perfect synchronization in presence of multiple transmissions, it is not possible to do the verification, since only the presence/absence of a transmission on a bus can be detected (that too only in a realtive, not absolute sense), and there is no way to tell whether a transmission was single or multiple, or which LRU(s) it came from. As far as the error latches are concerned, it does not make a difference whether the LRUT transmitted or not, since a system clock is always present on that particular bus. Hence, a bus must be free before the LRUT is enabled to transmit on it. this is no problem for the P, R and T buses, but since only two C buses (at most) can be free at any time, no more than two C buses can be enabled for the LRUT at any time. With this restriction, and with symmetry in mind, the test cycle for clock buses can be selected as 00000 → 00011 → 00100 → 11000 → 00001. Note that the assignment of 3-out-of-5 clock buses for the system clock has to be dynamic for this part of the test. The quad clock can be used for testing the C enable register by making the test two steps longer, e.g. (00000 → 00001 → 00010 → 00100 → 0100 → 10000 → 00001).

4.4 The Input Select Register

For the T-input select register of the BGU, the observability problem is further aggravated by the fact that the BGU's do not have any circuitry for detection of errors on the T-input lines. There are no error latches that can be read to determine if some input line was in disagreement with the other two. This means that the so-called 'harmless' commands cannot give any indication as to which input lines are being selected. the only way to get some feedback is to make the command effective, and look for that effect. In case of the BGU, that means issuing a write command for one of the bus enable registers, and then verifying that these buses have indeed been enabled.

If the select code corresponds to T buses 1, 2 and 4, then verification of the select code can be done by issuing a command on two of these buses at a time, and verifying that those commands are carried out no matter which two of the three buses are chosen. This also tests the deskewers and voter hardware. The simplest command to use is to write something in the C enable register. Three of the clock buses are being used for verification purposes, but one of these will be needed later. Let C1 be chosen for input select verification. Initially the C bus enable register has all 0's in it. Issue a command to write 00001 in it on lines T1 and T2. Verify the write by reading C error latches in L(1,2,4). There should be no error since C2 and C4 were already carrying clock signals, and C1 has just been enabled. Then issue a command to write 00000 in the same register, this time on lines T2 and T4. Again, verify the write by reading C error latches in L(1,2,4). There should be an error on C1 since transmission on this line has been disabled. These two tests are theoretically sufficient to show that the lines T1, T2, and T4 are indeed being selected by the BGU. However, to improve the confidence level, one more test is recommended. Issue a command to write 00001 in the C enable register, this time on T1 and T4. Verify the write by reading C error latches in L(1,2,4). There should be no error.

The same testing procedure can be used for any other select code. At any time, the BGU is supposed to select three of the five T-input lines. Pick two of the three buses at a time, and transmit a command on them. Then verify that the command has been carried out by reading the C error latches in L(1,2,4). Since there is continuous transmission on C2 and C4, and transmission on C1 goes on/off, either there will be no error, or there will be an error on C1. However, it is apparent that the verification of a particular select code could be time-consuming, owing to the devious means of observation.

4.5 The Testing Routine

Now that everything has been considered, the testing routine can be written as a sequence of steps:

Step 1

Initialize the system. This means that for a given LRUT, select a triad to run the test program and provide the clock. Also, select two LRU's as L(1,2,4) and L(3,4,5). In order to let the BGUT have complete control, write all 1's in the enable registers of the other BGU in the LRUT. Let the system clock transmit on C buses 2, 4 and 5, unless otherwise specified, and reset all error latches. At this time, all registers in the BGUT should contain their power-up value, which is all 0's.

Step 2

Test the T bus enable registers for the BGUT using the test sequence 00000 → 00111 → 11100 → 00001. After every write, reset all error latches in L(1,2,4) and L(3,4,5), and make the LRUT transmit on all lines (T R, P and C) that have been enabled. Transmission on C lines is automatic if the corresponding enable bits are set. Transmission on R can be induced by issuing a system read command. Transmission on T and P must be explicitly asked for. Read all error latches in L(1,2,4) and L(3,4,5), and verify the data write by comparing the error syndrome with the expected syndrome. Also, verify the contents of input select register, as in Section 4.5.

The expected error syndrome is:

Data Written	Receiving LRU	Error on Lines T	P	R	C
00111	L(1,2,4)	4	None	None	1
	L(3,4,5)	3	None	None	3
11100	L(1,2,4)	4	None	None	1
	L(3,4,5)	None	None	None	3
00001	L(1,2,4)	1	None	None	1
	L(3,4,5)	None	None	None	3

Step 3

Test the P and R bus enable registers for the BGUT using the same test sequence, and the same set of operations as for the T enable registers in Step 2 above. The error syndrome is the same as above, except that column headings T and R are interchanged when the R enable register is being tested, and the headings T and P are interchanged when the P enable register is being tested.

Step 4

Test the C enable registers for the BGUT using the test sequence 00000 → 00011 → 00100 → 11000 → 00001. For this part of the test, the assignment of 3-our-of-5 clock buses for the system clock has to be dynamic. After every write, a in Step 2, reset all error latches in L(1,2,4) and L(3,4,5), and make the LRUT transmit on all lines (T,P,R and C) that have been enabled. Read all error latches in L(1,2,4) and L(3,4,5), and verify data write by comparing the error syndrome with the expected error syndrome. Also, verify the contents of the input select register as in Section 4.5.

The expected error syndrome is:

Data Written	C Buses Allocated To System Clock	Receiving LRU	Error In C Bus
00011	3, 4, 5	L(1,2,4)	None
		L(3,4,5)	None
00100	2, 4, 5	L(1,2,4)	1
		L(3,4,5)	None
11000	1, 2, 3	L(1,2,4)	None
		L(3,4,5)	None
00001	2, 4, 5	L(1,2,4)	None
		L(3,4,5)	3

No error is expected on buses T, P and R.

Step 5

Test the input select register for the BGUT using the test sequence 0000 → 1111 → 0000. Let the system clock transmit on C buses 2, 4, and 5. Now use the technique described in Section 4.5 with 2 of 3 selected buses to verify the selection of the T lines and to test the fault masking hardware.

Step 6

One BGU has now been completely tested. The next thing to do is to test the other BGU in the same LRU. Write 1's in all enable registers of the BGU that has just been tested, and perform exactly the same set of tests as before. When this is done, one of the LRU's will be done. Testing can be done for other LRU's in the same way by rotating the roles of LRU's whenever necessary.

V. CONCLUSION

We studied the problem of fault-testing in the BGU's of the FTMP, and proposed a testing algorithm. The algorithm is constrained by the limited observability of the BGU's. A small improvement in the observability will go a long way in reducing the test length. One possible improvement is to provide error latches for the T input select circuitry in the BGU's just as has been done for all other input select circuits.

The reliability of any test outcome depends on the reliability of the testing units. Five LRU's are used to test one LRU. Two of these five are being used as error detecting elements, and as such, they must be reliable. The other three form a triad to run the test program, and at least two of these must be fault-free for reliable testing. It is thus possible to come up with a configuration where the test outcome would be questionable. This possibility leads to the theory of system-level diagnosability, first developed by Preparata et al. [3]. They proposed a model which has come to be known as the Preparata-Metze-Chien (PMC) model. The model assumes symmetric invalidation, i.e., the test outcome is always reliable when the testing unit is fault-free, and always unreliable when the testing unit is faulty. Another model (the BGM model), proposed by Barsi, Grandoni and Maestrini [4], assumes asymmetric invalidation, i.e., a test performed on a faulty unit must always fail, regardless of whether the testing unit is faulty or fault-free. The rationale for this assumption is the conviction that when the unit being tested is faulty, there must be at least one disagreement between the actual test observations and the expected observations, even when the testing unit itself is faulty. However, because of limited observability, this conviction may not hold for the BGU's of the FTMP. For our purposes, the refore, the PMC model would be more acceptable.

While the BGU is being tested, it is assumed that the rest of the LRU, including the processor and the memory, is fault-free. The diagnostics for the processor and the memory are discussed in [2]. these diagnostics, combined with the system-level diagnosability ideas in [3-10], can be used to detect and identify a faulty LRU in the FTMP.

VI. REFERENCES

1. A.L. Hopkins, Jr., T.B. Smith, and J.H. Lala, "FTMP — A Highly Reliable Fault-Tolerant Multiprocessor for Aircraft," Proceedings of the IEEE, Oct. 1978, pp. 1221-1239.

2. Rockwell International, "Notes on FTMP".

3. F.P. Preparata, G. Metze, and R.T. Chien, "On the connection Assignment Problem of Diagnosable Systems," IEEE Trans. Electronic Computers, Dec. 197, pp. 848-854.

4. F. Barsi, F. Grandoni, and P. Maestrini, "A Theory or Diagnosability of Digital Systems," IEEE Trans. Computers, June 197, pp. 585-593.

5. J.D. Russell, and C.R. Kime, "System Fault Diagnosis: Closure and Diagnosability with Repair," IEEE Trans. Computers, Nove. 1975, pp. 1078-1088.

6. J.D. russell, and C.R. Kime, "System Fault Diagnosis: Masking Exposure, and diagnosability Without Repair," IEEE Trans. Computers, Dec. 1975, pp. 1155-1161.

7. S.L. Hakimi, and A.T. Amin, "Characterization of Connection Assignment of Diagnosable Systems," IEEE Trans. Computers, Jan. 1974, pp. 8-88.

8. G.G.L. Meyer, and G.M. Masson, "An Efficient fault diagnosis Algorithm for Symmetric Multiple Processor Architectures," IEEE Trans. Computers, Nov. 1978, pp. 1059-103.

9. F.J. Allan, T. Kameda, and S. Toida, "An Approach to the Diagnosability Analysis of a system," IEEE Trans. Computers, Oct. 1975, pp. 1040-1042.

10. S. Toida, "System Diagnosis and Redundant Tests," IEEE Trans. Computers, Nov. 197, pp. 1167-1170.