Design and Evaluation of Fault-Tolerant Interleaved Memory Systems

Shyue-Kung Lu & Sy-Yen Kuo
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
R.O.C.

Cheng-Wen Wu[*]
Department of Electrical Engineering
National Tsin Hua University
Hsinchu, Taiwan
R.O.C.

Abstract

A highly reliable interleaved memory system for uniprocessor and multiprocessor computer architectures is presented. The memory system is divided into groups. Each group consists of several banks and furthermore, each bank has several memory units. Spare memory units as well as spare banks are incorporated in the system to enhance reliability. Reliability figures are derived to evaluate systems with various amounts of redundancy. The result shows that the system reliability can be significantly improved with little hardware overhead. User transparency in memory access is retained.

1 Introduction

A major approach to attain a memory system with a high bandwidth involves the use of several memory *banks* or *modules* connected in an interleaved fashion. Since a faulty memory bank will either degrade the system performance significantly (for a high-order interleaved memory) or even make the entire system malfunction (for a low-order interleaved memory), several approaches [1] have been proposed to exclude the faulty banks from normal operation. In this paper, we propose a novel *fault-tolerant interleaved memory (FTIM)* system based on the general ordered interleaved memory system and the results can be extended to other types of interleaved systems. This memory system is composed of *groups* of *banks* and the access to each bank uses the low-order interleaving technique. Each memory bank contains several *memory units (MUs)*. Each MU is usually a memory chip. Spare banks and spare MUs are added in each group and each bank, respectively. Faulty MUs are replaced by spare MUs within a bank first, and the replacement goes to the bank level if the faulty bank (a bank with faulty MUs) runs out of redundancy. If each bank (group) has at most two spare MUs (banks), then the *easily reconfigurable FTIM (ERFTIM)* is used. Alternatively, if there are more than two spare units in a bank, the *CAM-based FTIM (CBFTIM)* system and the *switch-based FTIM (SBFTIM)* system are proposed. The system performance is preserved until the memory system runs out of redundancy. Our approaches are much simpler in hardware complexity and more efficient in terms of time overhead than *SLAT* (second-level address translator) [2].

[*]This work was supported in part by the National Science Council, R.O.C., under Contract NSC83-0404-E007-019.

2 Review of Interleaved Memory System

2.1 Interleaved memory system

In general, in an N-way high-order or low-order interleaving, we have N memory banks where N is a power of 2, i.e., $N = 2^n$ and n is an integer. Assume that there is a total of $M = 2^m$ words in the memory system and therefore, a physical address has m bits. In such a system, n bits of the address suffice to select a bank and the remaining $m - n$ bits are used to select a word within a bank. If the n bits are the high-order bits of the address space, this scheme is a ***high-order interleaving*** scheme whereas a ***low-order interleaving*** scheme results if the lower order n bits are used to select a bank. The major drawback of low-order interleaving is that it is not modular. A failure in a single bank affects the entire address space and will almost certainly be catastrophic to the whole system.

2.2 Interleaved Memory System with Two-Level Redundancy

The memory structure of our FTIM system is shown in Fig. 1. The main memory with a total of $M = 2^m$ words is interleaved with the N-way low-order interleaving scheme, where $N = 2^n$. Moreover, these N independent memory banks are divided into $L(= 2^l)$ groups with each group consisting of K banks in the

FTIM system. By using an address latch in each group, a group of $K = 2^{n-l}$ banks ($l \leq n$) which are fully interleaved can be multiplexed on an internal memory bus. In this approach, K banks are accessed concurrently which increases the available bandwidth to K times the bandwidth of a single bank. If the main memory is further divided into pages with a page size of 2^p words (p is an integer and $p \leq m$), words within each page will be distributed evenly over all the memory banks.

The definition of address bits for our FTIM system is shown in Fig. 2(a). Compared with the original address format for the low-order interleaved memory as shown in Fig. 2(b), we can see the following differences: (1) the N memory banks are grouped into L groups with the higher order l bits of the n bits to select a group and the lower order $n - l$ bits to select a bank within a group; (2) depending on its size, a bank can have $Q = 2^q$ independent MUs.

Instead of having spare banks at the high level only, we can have a two-level redundancy scheme by introducing spare MUs in the low level. After the faulty MUs have been identified, spare MUs are used to replace them first, and then a faulty bank is replaced by a spare bank if the faulty bank runs out of spare MUs. An example of the hierarchical modular memory system is shown in Fig. 3. Each MU is designated as $MU(i, j)$ where i and j are the row number and the bank (column) number of the MU, respectively. An MU flagged by a R is a spare MU.

3 Design of FTIM Systems

We assume that there exists a mechanism to detect the presence of a faulty bank (MU) and to locate it. We only consider faults which will cause the loss of a complete MU. This fault model is more effective than the one presented in [3] where a fault will cause the

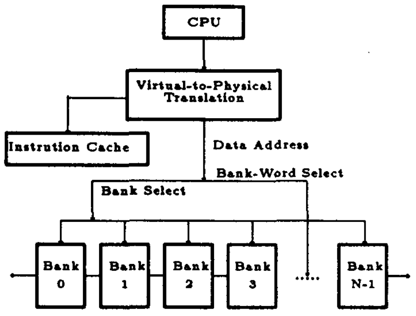

Figure 1: The memory system structure

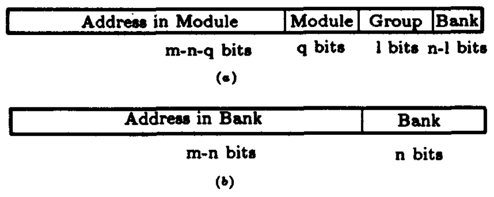

Figure 2: Addressing formats of interleaved memory systems

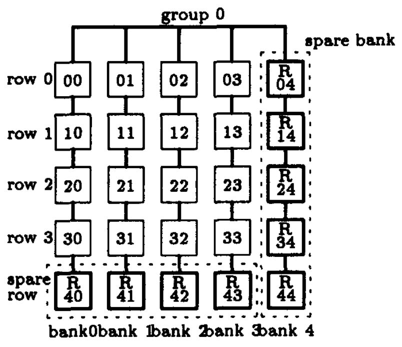

Figure 3: An interleaved memory system with two-level redundancy

loss of an entire memory bank.

When faults occur in the memory system, the program must be stopped, and correct information must be recovered from the backup store and stored in the designated MUs. The system management and address translation mechanism will be informed about the faulty modules and program execution can be restored.

The system structure of our FTIM system is shown in Fig. 4, where a *second-level address remapping controller (SLARC)* is included. When faults occurs in the memory system, the original MU/bank selection signals which enable the faulty MU/bank must be discarded. According to the original MU/bank address and the faulty status of the memory system, the SLARC will generate the remapped MU/bank selection signals to activate an appropriate spare MU/bank.

In Fig. 4, the CPU provides virtual address to the virtual-to-physical address translator, where $n + q$ bits of the virtual address are extracted and sent directly to the SLARC. If there are faulty MUs/banks in any of the bank/group, the *MUEI (module unit error indicator)* and *BEI (bank error indicator)* are both set to one. These error signals then activate CPU and SLARC to perform address remapping. When any of the error

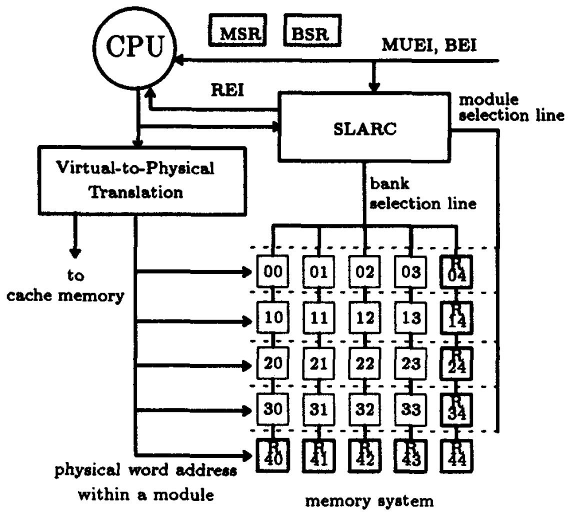

Figure 4: System structures of FTIM system.

indicator is set, the processor then read the contents of *MSR (module status register)* and *BSR (bank status register)* to get the address of the faulty MU and bank, respectively. Let the memory system have r spare MUs and c spare banks in each bank and group, respectively. Then the MSR register contains a vector of $2^q + r$ bits. Furthermore, the BSR register contains $2^{n-l} + c$ bits. Each bit in an MSR or BSR register indicates whether the corresponding MU or bank is faulty or not. According to the received faulty status, the SLARC then performs the appropriate remapping task. The output of the SLARC will set an appropriate bank/module selection line which always enables the fault-free module/bank. In addition to these selection signals, the SLARC also sends a *reference error indicator (REI)* signal to the *CPU*. Eventually, the system will run out of spares, the SLARC will then set the *REI* to indicate that an access is made to a faulty MU/bank which can not be replaced by a spare MU/bank. Based on the realization approach for the SLARC, we obtain the CBFTIM system or the SBFTIM system, as discussed in the following two subsections.

3.1 Design of CBFTIM System

In this subsection, two types of CBFTIM systems are described. The *type-I* CBFTIM system is shown in Fig. 5, where the SLARC is realized with an *MMT (module map table)*. This table is implemented by means of an associative memory of $2^q + r$ words with $q+1$ bits per word. Each word in the MMT contains a module number together with its corresponding *valid*

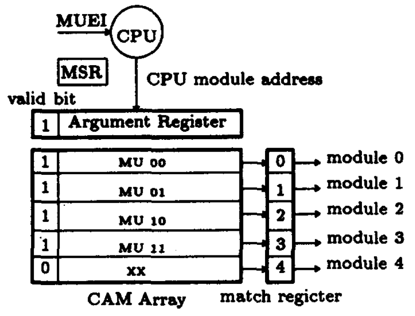

Figure 5: The MMT implementation of the SLARC, $r = 1$.

bit. The valid bit, if set, indicates that the corresponding module is good. Originally, if the memory system is fault free, the valid bits of the first 2^q words are set, and the other words are reset. It should be noted that the module address of $MU(a, X)$ is stored in word a in the MMT. The module field in each word is compared with the MU field in the virtual address if the corresponding valid bit is set. If a match occurs, the matched word will set its corresponding bit in the match register, which can be used to enable the corresponding module. That is, under a fault free situation, the spare MUs will not be selected.

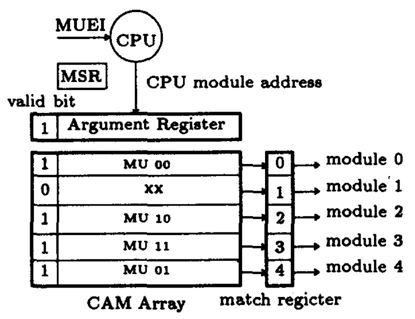

Figure 6: The replacement of the content of MMT when module 01 is faulty.

For example, if module 1 of a bank in Fig. 5 is faulty, then the valid bit of word 1 is reset. The module address content of word 1 is written into word 4, and the valid bit of word 4 is set. The reconfigured MMT table is shown in Fig. 6.

The type-I CBFTIM system is inefficient in memory space utilization and the allocation of spare MUs. For example, if there is only one faulty MU in a certain row, the entire row space is isolated after the second-level address remapping.

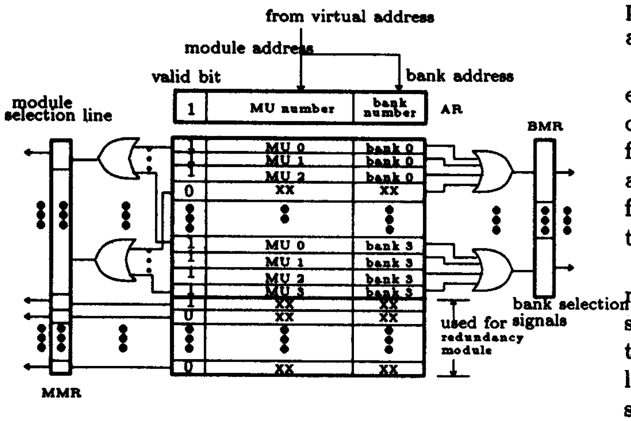

Figure 7: Type-II CBFTIM system, r = 2, AR = Argument register, BR = Bank Match Register, MMR = Module Match Register.

Instead of r spare MUs in each bank, a type-II CBFTIM system contains sm spare modules in each group that can be used to replace any faulty module in the group. That is, the sm spare modules do not belong to a same bank. The MMT table is reorganized as shown in Fig. 7, where $2^{n-l+q}+sm$ words of $n-l+q+1$ bits are used. Each word contains an $(n-l)$-bit bank number and a q-bit module number which are compared with the corresponding fields in the virtual address. The function of the valid bit and the matching approach is similar to the type-I CBFTIM system. These match signals then enable appropriate bank and module selection lines.

3.2 Design of SBFTIM System

In the SBFTIM system, an MU switching circuit (MUSC) as shown in Fig. 8 is used as the SLARC. The inputs to the MUSC consist of contents of the MSR (which contains the location of the faulty MUs in the bank) and the 2^q outputs from the decoder. The outputs of the MUSC are 2^q+r physical MU select signals. If the kth output line from MUSC is set, the kth physical MU is selected after the second-level address remapping.

The MUEI signal is a one-bit flag used to enable the SLARC when a fault occurs in the memory system. In a fault-free bank, the q-bit MU field in the virtual address are directly decoded to determine the MU to be selected. However, if faulty MUs exist in the memory bank, the actual MU selected will be remapped by the MUSC onto a physical MU in the following way. If there is one faulty MU in a bank, say physical MU k, then the output line $k+i$ $(0 \leq i)$ from the decoder is remapped to $k+i+1$. For multiple faults, the remapping mechanism is extended in a manner that the next available fault-free physical MU is selected.

Since the contents of the MSR is associated with each bank, each bit in the MSR represents the status of an entire row instead of a single MU in a bank. A faulty row is then bypassed and replaced by its nearest available fault-free row. This type of SBFTIM is referred to as type-I SBFTIM system as opposed to the type-II SBFTIM system discussed next.

Instead of using a single MSR register to store the memory unit status for all banks, the type-II SBFTIM system uses a pile of registers to maintain the information for all the banks. In an N-way low-order interleaved memory system, selection of a specific memory status word will then depend on the bank selection lines in the lowest n bits of the virtual address. The status words within the cache memory is updated every time a faulty MU is detected and located.

The major advantage of the type-I SBFTIM system is that the extra time delay incurred by the SLARC is approximately zero since no additional memory reference is introduced in this design. The type-II SBFTIM system has a more efficient memory space utilization, but it can introduce significant time delay since an additional cache memory reference cycle is required.

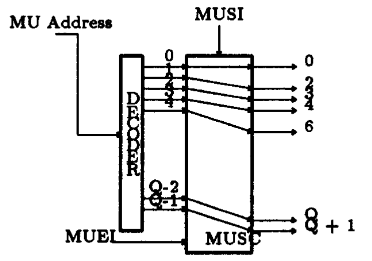

Figure 8: The changed content when MU_{03} is faulty.

4 Easily Reconfigurable Fault-Tolerant Interleaved Memory System

An *ERFTIM system* which can tolerate up to two faulty MUs in each bank and two faulty banks in a group is presented in this section.

In a ERFTIM system with two spare rows, the two spare MUs in each bank are placed at the bottom and the other at the top of the bank (Fig. 9). A switching element is associated with each MU (it can be attached to the "enable" line of each MU) to control the selection of the unit. The input and output connections of a switching element are shown in Fig. 10, which are controlled by two reconfiguration control registers (CRA

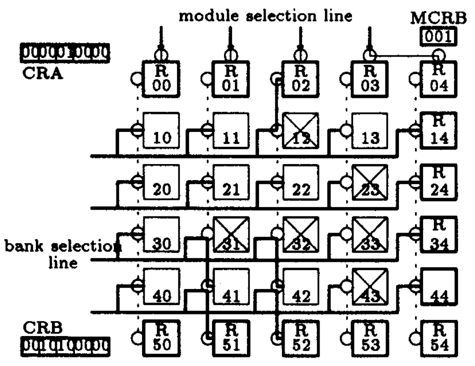

Figure 9: An example ERFTIM system.

and CRB). When the selection line $s = 0$, the corresponding switch is inactivated by the reconfiguration controller and thus the MU selected by the MU selection line is enabled. On the contrary, if $s = 1$, the switch is activated so that it skips the selected MU and replaces it with its adjacent top or bottom MU. The selection of top or bottom adjacent MU depends on the control register that is used to activate it. Each field in a reconfiguration control register controls a bank. The number of bits in each field is a function of the number of MUs in the corresponding bank. A bank with $2^q + r$ MUs (normal and spare) will have t bits in the corresponding field of both CRA and CRB where t is the smallest integer such that $2^t \geq 2^q + r - 2$. Memory units in a bank are numbered from 0 to $2^q + r - 1$. The bit-patterns in CRA and CRB determine whether a faulty MU will be replaced by its adjacent fault-free MU either at the top or the bottom. Fig. 11 illustrates the meaning of a field in CRA and CRB for the example shown in Fig. 9. For instance, if the field for bank i in CRA has the pattern 01, switches of *MU(0, i)* and *MU(1, i)* are activated and therefore the top spare MU in bank i will be used to replace a faulty MU in the bank.

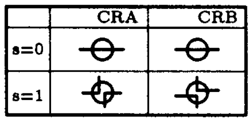

Figure 10: The switch mechanism.

In Fig. 9, there is a faulty MU (MU(3,1)) in bank 1. The corresponding switch setting to replace the faulty MU in bank 1 is completed by setting the contents in the second field of CRA and CRB to 00 and 10, respectively. Therefore, switches of MU(3,1), MU(4,1), and MU(5,1) in this bank are activated with CRB = 10, and switches of MU(0,1), MU(1,1) and MU(2,1) are not activated with CRA = 00. If there are two faulty MUs in a bank, e.g., MU(1,2) and MU(3,2) of bank 2 in Fig. 9, the bits in the third field of CRA and CRB are set to 01 and 10, respectively. This replaces faulty MU(1,2) by the spare MU at the top and faulty MU(3,2) by the spare MU at the bottom. The two spare MUs should not both be at the top or at the bottom of a bank, since from this example we can see that an MU can only be replaced by one of its adjacent neighbors due to the simple structure of the switch. As a result, this structure can tolerate up to two faulty MUs in each bank. However, it has the advantage of regular and easily reconfigurable structure as well as simple switches.

l=6 (r=2)		
	CRA	CRB
00	normal	normal
01	memory units 0-1	memory units 4-5
10	memory units 0-2	memory units 3-5
11	memory units 0-3	memory units 2-5

Figure 11: MUs activated by CRA and CRB.

The replacement of a faulty bank by a spare bank can be implemented with an approach similar to that for the MU level. Only a switching element is needed at each bank selection, and if it is activated the entire bank is replaced by its adjacent neighboring bank. A *bank control register (BCR)* is used to select the bank to be switched out if there is only one spare bank. Two control registers ($BCRA$ and $BCRB$) are needed if one spare bank is added before the first bank and one spare bank is added after the last bank. The bit-patterns of $BCRA$ ($BCRB$) are similar to those of CRA(CRB) except that $BCRA$ and the $BCRB$ have only one field.

5 Overhead Analysis

Let the memory system originally have $N = 2^n$ banks in each group and $Q = 2^q$ MUs in each bank. In the simple ERFTIM system, two spare rows and two spare columns are included. Each MU has a switch and a redundant link for fault tolerance. In addition, two reconfiguration control registers are used to control the functions of each switch. Since the switch function is very simple (Fig. 10), the hardware overhead introduced by the switches and reconfiguration control register is very small compared with the cost of the whole memory array. After the system has been reconstructed, extra time delay incurred due to the fault tolerance scheme is the time delay of a switch. This time delay is a two-gate delay since a switch can be implemented with a two-level logic.

The hardware overhead in the type-I SBFTIM system is caused by the MSR register and the SLARC. Although the cost of a SLARC is higher than the cost of a reconfiguration controller for the ERFTIM system, it is much less than that of the *second-level address translator (SLAT)* [3]. Since the SLARC can be implemented by the same technique as the SLU circuit in SLAT and the cost of the decoder and the reference error indicator is less than that of other circuits in SLAT (the SLAT has four more logical circuits and each of them is as complex as the SLU), the cost of SLARC is lower than that for SLAT. All the reconfiguration circuits can be implemented by combinational circuits so that the time overhead is not significant compared with the memory reference time. In addition, SLARC can be executed in parallel with the virtual-to-physical address translation which makes the time delay negligible.

An additional memory reference cycle is introduced in the type-II SBFTIM system in order to fetch the memory status of each bank from a pile of MSR registers. In a system with N banks and $Q+r$ MUs in each bank, this system will require N MSR registers. Each register has $Q+r$ bits to store the memory status information. Although the type-II SBFTIM system has a larger register file and a bigger time delay than the type-I SBFTIM system, it is more effective in improving the system reliability.

Among the proposed FTIM systems in this paper, the CBFTIM system has the highest efficiency and least performance penalty. The SLARC in this system is implemented with CAM which is suitable for VLSI implementation. Since the execution of the SLARC can be performed in parallel with the virtual-to-physical address translation, the time delay incurred by the SLARC is neglected if the page number is greater than the module number, which is always true in modern computer system. Let ca denote the cell area of the CAM array that implements the SLARC and ma denote that of the memory array. The word length of the memory system is w, and the size of each module is s. In type-I CBFTIM system, the CAM array contains $(2^q+r)\times(q+1)$ cells, and the memory system contains $(2^q+r)\times 2^{n-l}\times w$ memory cells. The hardware overhead is therefore

$$HO=\frac{(2^q+r)\times(q+1)\times ca}{(2^q+r)\times 2^{n-l}\times s\times w\times ma}.$$

For a system which has 16 banks with 16 MUs in each bank and $r=2$, the module size is 256 KBs, and the value of $HO\approx 0$ if ca is three times of ma.

In the type-II CBFTIM system, the CAM array contains $(2^{n-l+q}+sm)\times(n-l+q+1)$ CAM cells, and the memory system contains $(2^{n-l+q}+sm)\times s\times w$ memory cells. The hardware overhead is therefore

$$HO=\frac{((2^{n-l+q}+sm)\times(n-l+q+1))\times ca}{(2^{n-l+q}+sm)\times s\times w\times ma}.$$

For a system which has 16 banks with 16 MUs in each bank and $sm=10$, the module size is 256 KBs, and the value of $HO\approx 0.1\%$. In the analysis, we only consider the area of the CAM array. Other components (e.g., match register) in the SLARC are neglected as compared with the CAM array. From the analyzed results, we find that the cost of SLARC in CBFTIM system is almost negligible.

6 Conclusion

Fault-tolerant interleaved memory systems for highly reliable computer architectures have been presented. This system has two levels of redundancy. In the lower level, spare MUs are included in each bank to tolerate faulty MUs in the same bank. In the higher level, spare banks are used to replace faulty banks which do not have spare MUs left to replace faulty MUs. The structure of the reconfigurable memory system is designed in a way such that the replacement of faulty units (banks) by spare units (banks) will not disturb memory references. The system performance is preserved until the memory system runs out of redundancy. The system reliability of our FTIM system is much higher than that of a nonredundant system. Different types of address remapping controller are proposed which have less hardware overhead cost and time delays than other previous techniques.

REFERENCES

[1] M. S. Algudady, C. R. Das, and W. Lin, "Fault-tolerant task mapping algorithms for MIN-based multiprocessors," *Proc. International Conference on Parallel Processing*, 1990.

[2] M. Wang, M. Cutler, and S. Y. H. Su, "Refonfiguration of VLSI/WSI mesh array processors with two-level redundancy," *IEEE Trans. Comput.*, vol. c-38, pp. 547–554, Apr. 1989.

[3] K. C. Cheung, G. S. Sohi, K. K. Saluja, and D. K. Pradhan, "Design and analysis of a graceful degrading interleaved memory system," *IEEE Trans. Comput.*, vol. c-39, pp. 63–71, Jan. 1990.