DEVELOPMENT OF A DYNAMIC ROUTING SYSTEM FOR A FAULT TOLERANT SOLID STATE MASS MEMORY[1]

M. Ottavi, G.C. Cardarilli, P. Marinucci*, S. Pontarelli, M. Re, A. Salsano*
{g.cardarilli,marco.re}@ieee.org

Department of Electronic Engineering University of Rome "Tor Vergata", Italy
* ULISSE Consortium, Italy

ABSTRACT

This paper describes a fault tolerant Solid State Mass Memory (SSMM) for satellite applications. Definition of requirements plays an important role in the architectural solutions selected for the data storing system. The interconnection system proposed is based on a crossbar switch with half duplex links. All the connections have a complete flow control and the path of switched packets is dynamically reconfigurable. A controller tests functionality of each component and handles dynamic data allocation in the memory modules. Memory modules grant data integrity: an original tool developed by the authors derives data coding parameters.

1. INTRODUCTION

In the past, magnetic tape recorders have been used to store the large amount of data coming from instruments loaded on satellites. While a tape recorder can provide several gigabits of non-volatile storage, this electromechanical device has insufficient operational flexibility for advanced data systems. [5]

On the other hand, rapid growth in capacity of semiconductor memory devices, quadrupling every three years, now permits the development of solid-state mass memories which are competitive with tape recorders due to higher reliability, comparable density and better performances. Solid-state mass memories have no moving parts and their operational flexibility has made them suitable for many applications.

The first generation of SSMM was implemented with memory modules connected to the instruments via a bus. This solution is convenient if the number of apparatuses to connect is low, the distances are small and if there are not strong requests on bandwidth and latency. High performances buses do not overcome the reliability problems and, in any case, they are expensive since every module needs high speed logic in order to exploit the small time period allowed for each time division access. An alternative solution to buses is using crossbar matrix between the elements of the network. The more obvious advantage of such a topology, beyond the scalability and the possibility of reconfiguration, is the reliability. In fact, the failure of a connection does not compromise the entire connection of the network but only the access to a specific node.

This paper presents an application specific dynamic interconnection system and discusses the following points. Section 2 illustrates the choices made during the definition of interconnection system specifics. In Section 3, we describe the application of the above choices in implementing a SSMM system and the test bed we are using to achieve the final system simulation and validation.

2. SWITCH DESIGN REQUIREMENTS

The interconnection system has been developed taking into account the specs of a SSMM for space applications.

Specific conditions required for routing system in SSMM can be summarized in the following points:

1. Connections are required only between the apparatuses and the memory modules.
2. Apparatuses demand low latency connections.
3. At each access, memories typically are not read and written at the same time.
4. Most part of the apparatuses executes only memory writing.
5. The operations of reading from the memory mainly happen in prefixed periods. During the window of visibility, the satellite downloads the data to the earth station.

We will now show how these issues reflect in the project parameters of an interconnection system. In particular, we will consider the thoroughness of the traffic matrix, the management of file system using a virtual addressing modality and the kind of flow buffering required to reduce latency.

2.1 Traffic matrix

Consider a switch, with n bi-directional links. The thoroughness of the possible connections, and therefore the number of switches necessary, can be defined by introducing a $(n \times n)$ interconnection matrix. In this matrix the element (i,j) is equal to 1 if there is a connection between the input i and the output j, 0 if the connection is not present. In particular, for this matrix we can distinguish three typical cases:

- *Complete connection*: every input is connected to every output: $\forall i, j \mid i \in (1...n), j \in (1...n)\ \ mat(i,j)=1$
- *Without loopback*: no connection between i and j if $i=j$ $\forall i, j \mid i \in (1...n), j \in (1...n), i \neq j\ \ mat(i,j)=1$
- *With I/O subsets*: in this case consider two subsets of I/O interfaces: A and B of α and β elements respectively, being $\alpha+\beta=n$ $A \cup B=N$ and $A \cap B=0$, connections are present only between elements belonging to different subsets.

For each case, the number of necessary interconnections can be evaluated:

- *Complete connection*: requires $n*n$ connections
- *Without loopback*: requires $n*(n-1)$ connections

[1] Partially supported by ASI (Italian Space Agency) and ULISSE Consortium of Rome, Italy

- *With I/O subsets*: requires $2\alpha\beta$ connections

As we showed above, the third case is the typical case of the storing system in a SSMM. This means that the architectural complexity of the switching matrix is simpler compared to the full-connected general-purpose case.

2.2 Addressing modes

Data coming from the on board instruments are carried on serial links based on the *Spacewire* protocol [1]. According to the SpaceWire protocol, a header, a payload and an end of packet marker compose the packets sent from the apparatuses towards the switch matrix. Usually, the header indicates the address where the packet must be routed. Unfortunately, this approach implies that memory reconfiguration will affect the apparatus status. Therefore we have chosen a virtual addressing where the header indicates the file identity. This approach allows to manage dynamically the routing of packets inside the storing system. The reconfiguration of the routing is managed by an external microcontroller, while a local arbiter defines the physical connections.

2.3 Flow Buffering

The main problem related to the buffering of the data arriving to a switch system is the blocking of the head of the line (HOL). This problem occurs when a packet waiting to be routed towards a busy output blocks a queue of packets directed towards free outputs. If every input buffer is implemented with a single FIFO, HOL blocking can limit the throughput up to 58% of the maximum. This event occurs when all the inputs are 100% used and the traffic (packets) is uniformly distributed towards all the outputs. Many solutions to reach higher throughput have been proposed 5. These solutions imply higher hardware complexity with respect to the buffering with a single FIFO. In our application, the problem of the HOL can be limited using the virtual addressing of the packets operated by the microcontroller. In fact, microcontroller can change transparently the destination of a packet if its latency on an output is too high (reduction of the conflicts) or interrupt a packet before its end. Therefore, we opt for buffering with single FIFO.

According to the above specifications, we generated a routing system based on a crossbar switch matrix, an arbiter and a microcontroller. Such a system provides a connection between serial links and a set of independent memory modules. We will now describe the global architecture of the SSMM.

3. GENERAL SSMM ARCHITECTURE

In this section, we will describe the general architecture of SSMM [6]. In particular, we distinguish two main blocks:

- Memory array
- Routing system.

The memory array is implemented by a set of Independent Memory Array Modules (IMAM). The memory modules are independent, in fact, each module contains all the functions required for the detection and the correction of the errors as well as the functions required for the data transfer toward the I/O interfaces.

The routing system is composed of a microcontroller (system control unit) an arbiter, some I/O interfaces and an application specific interconnection system.

Systems for space applications must satisfy reliability requirements related to the hostile environment in which they operate. To achieve a fault tolerant system we have adopted two different strategies

1. Error Detection and Correction codes in the memory modules.
2. Redundancy and system reconfiguration in the routing system.

We will now describe, in some details, the memory array, the routing system and the test bed of the routing system and of the arbiter.

3.1 Memory Array.

Memory modules are the core of SSMM. The main characteristics required for such modules concern the capacity, the organization and the memory package. A memory module corresponds to a single board so the capacity of each module should be sufficiently large, even if the physical volume required by the memory chips limits it. The required memory capacity implies the use of a large number of memory chips. For this application the use of space qualified chips is unsuitable for two reasons. The first reason is related to the cost of space qualified chips that would make the SSMM very expensive. The second reason is the unavailability of space qualified memory chips with a sufficient level of integration. All the above reasons push toward the use of Commercial off-the-shelf (COTS) chips. Since these chips are not protected against the effects induced by spatial environment, the requirement of fault-tolerance implies the introduction of suitable design strategies.

The characteristics of the SSMM considered are the following

- Net capacity 16 Gbit/module beginning of life (BOL).
- Memory organization: array of r rows and c columns. The actual values of these two parameters must be chosen for obtaining a square board -in order to simplify the mechanical organization and the shielding of the SSMM.
- Suitable packaging of the memory chips. The chosen package must allow the control of the power supply for the single chip (we preferred a package containing 4 chips of 64 MBIT SDRAM).

The memory architecture uses n chips for the word group, k chips for data storing, n-k chips for storing the check symbols, and s chips for the cold spare. Each of the above chips belongs to a different package this choice avoids that a package fault could produce multiple symbol error. Each word group is arranged on 4 rows and each row is composed by $(n+s)/4$ packages (corresponding to $(n+s)/4$ columns). This board organization allows reconfiguring the memory module with different ECC (Error Correction Codes) structures, depending on the application considered. In fact, it is possible to reduce the codeword down to $1/4\ n$. Increasing the memory washing (i.e. periodical correction of data stored) frequency, it is also possible to reduce the code length maintaining the overall bit error rate (BER). Of course, this solution reduces the memory availability. Code parameters are defined using the optimization tools presented in [3][4].

3.2 Routing System

The main elements of the routing system (figure 1) are the I/O interfaces, the switching matrix, the arbiter and the system control unit. These components are responsible for the connection between the serial links -coming from the measurement instrumentation- and the memories. The main tasks that must be performed are

- *Flow control* for a connection between input and output interfaces. The flow control is mainly responsible for the generation and control of the handshaking signals.
- *Access arbitrating* for an output resource shared by several input links. In this case, the system must control the actual configuration of switches in the switching matrix. This level doesn't change the routing of the packets inside of the matrix..
- *File system management:* microcontroller dynamically define the route on the switching matrix, to connect the input interface to the output one, for a given file number.

While the microcontroller is an independent software programmable module, the following modules compose the core of the designed system.

3.2.1 Crossbar Switch Matrix

This component allows the physical interconnection among M input and N output interfaces. As explained above, the switch matrix shall implement only $M*N$ connections.

The use of half-duplex links allows simplifying the system implementation. In fact, a single physical link is required for each bi-directional connection. Each physical link is implemented with a channel for data and another channel for the flow control.

With this structure, the failure of a connection does not cause the failure of the whole memory system, in fact only a specific node cannot access the matrix. Moreover, the memory partitioning allows the concurrent access of several users, maintaining high access and transfer speeds. A connection system implemented using a switch matrix allows multiple parallel links between users and resources. This architectural redundancy increases the reliability of the system, in fact the failure of a link implies only partial loss of the system functionality.

3.2.2 Arbiter

The connection between the M serial interfaces and N memory modules is made in unidirectional way (half duplex communication). This implies that one of the two interfaces operates as master of the connection and the other as slave. The connections always start from a master that requests the arbiter to execute the switch towards the destination. The arbitration algorithm can be summarized as follows:

When the arbiter receives a request of connection between master i and slave j, it checks if i (as a slave) or j (as master or as slave) are already busy. If output i and j are free the connection i,j is established. On the other hand, if i or j are busy, the connection is established depending on an externally settable arbitrating policy and can be based on priority or timeout.

3.2.3 I/O Interfaces

Most of the I/O interfaces will be used as unidirectional ones. They correspond to the links carrying the measurement information. A small number of interfaces requires reading and writing the memories. The most important interface connects the memories to the telemetry circuitry. This circuit transmits collected informations from the satellite to the earth station.

All the interfaces access the switch matrix in half-duplex mode, and request arbitration via dedicated links. A shared bus interconnects all the I/O interfaces and the microcontroller to provide file system management and error detection. A generic I/O interface is composed of the main functional blocks (figure 2)

1. **LVDS I/F**. This block implements the electric interfacing between the differential signals LVDS (Low Voltage Differential Signaling) and Data and Strobe single ended signals.
2. **SPACEWIRE (1355 DS DE) I/F**. This interface interprets the serial signal, extracts the clock signal and translates it to a parallel word. It implements the flow and the parity control following the procedures of the *SpaceWire* protocol.
3. **FIFO**. The FIFO depth is chosen in a way to avoid data loss for the latency of the successive elements. Since serial link can reach 200Mbps, the FIFO speed is up to 20 Mtps (Mega tokens per second, with 10 bits per token).
4. **LINK I/F.** Represents the core of I/O interfaces. A master and a slave compose it. Master handles the virtual addressing via the message bus and the requests to the arbiter. All the data paths are fully handshaken.

The effectiveness of the above architecture is now under test. The validation will be performed by using a fast prototyping environment based on behavioral VHDL and FPGA implementation. In particular, this activity foreseen the following steps:

1. VHDL behavioral description and validation of SSMM modules. (fig 3, fig 4 show the VHDL test bench of arbiter and of 2*2 routing system between link I/F)
2. VHDL RTL design of single modules.
3. Back annotated simulation
4. H/W implementation of the test bed and its verification

Preliminary behavior simulation results show the correctness of the overall architecture. In particular, the behavior of routing and arbitrating systems agrees with the defined requirements.

4. CONCLUSIONS

In this paper we show an architectural description of a SSMM for satellite application. We focused our attention on the parameters involved in the design of the routing system. The objective to obtain a dynamically reconfigurable system is achieved considering the overlap of three complementary levels: flow control, access arbitrating and file system management. Routing system and memory modules were designed in order to obtain a system level fault tolerant architecture.

5. REFERENCES

[1] ECSS Secretariat ESA-ESTEC Requirements & Standards Division "SpaceWire draft e" ESA Publications Released on 13 September, 2000

[2] N. Ni, M. Pirvu and L. Bhuyan "Circular Buffered Switch Design with Wormhole Routing and Virtual Channels" Computer Design: VLSI in Computers and Processors, 1998. ICCD '98. Proceedings. International Conference on , 1998 , Page(s): 466 -473

[3] G.C. Cardarilli, P. Marinucci, A. Salsano, "Fault-tolerant Solid State Mass Memory for Satellite Applications", IMTC'98

[4] G.C. Cardarilli, P. Marinucci, S. Bertazzoni, M. Salmeri, A. Salsano, "Design of Fault-tolerant Solid State Mass Memory", DFT'99.

[5] D. Maeusli, F. Teston, P. Vuilleumier, R. Harboe-sorensen "ESA Developments In Solid Sate Mass Memories", in ESA publication

[6] G.C. Cardarilli, P. Marinucci, M. Ottavi, A. Salsano, "A Fault-tolerant 176 GBit Solid State Mass Memory Architecture", DFT'00

TABLE 1

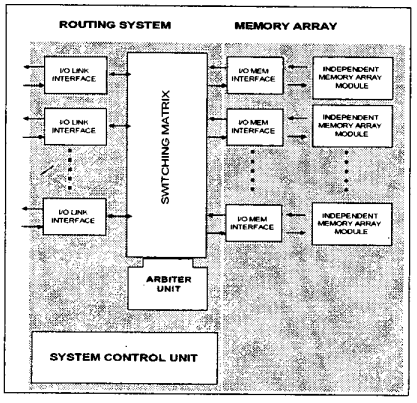

figure 1

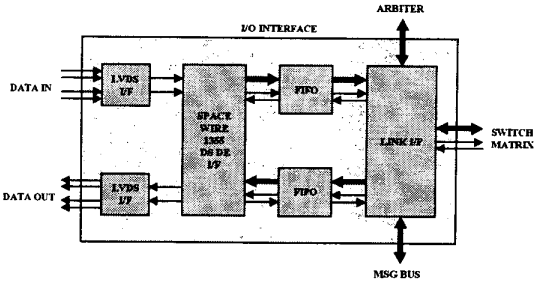

figure 2

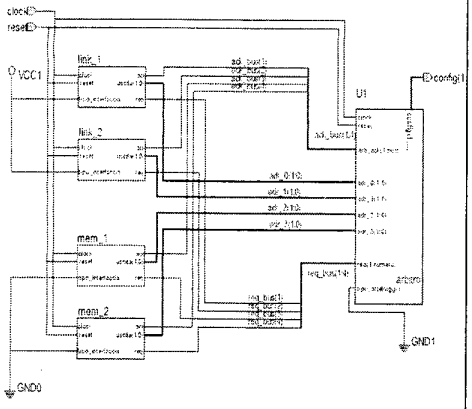

figure 3

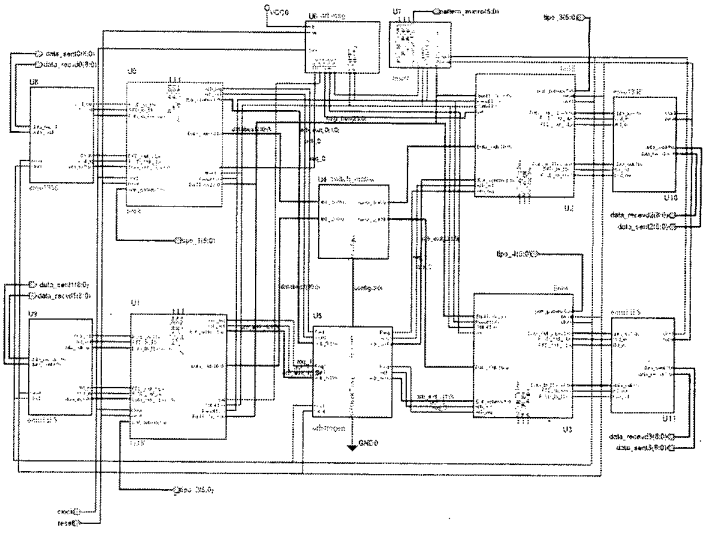

figure 4