Implementation of Middleware Switch ASIC Processor

Vladimir Petrovic, Marko Ilic, Gunter Schoof, and Sergio Montenegro

***Abstract* — The new spacecraft area network (SCAN) system for internal satellite communication provides data transfer between different satellite components. The satellite components have their own communication protocols, which are translated by middleware (MW) switch processor into a universal middleware protocol, understandable for the satellite operating system. The MW switch processor is the main part of a *new* proposed approach. Instead of current board computer based systems, the new data transfer approach based on network provides a more reliable solution. The processor is fabricated in the 250 nm IHP technology. This paper introduces a description of MW switch architecture, a comparison between possible architecture approaches and the characteristics of the implemented MW Switch processor.**

***Keywords* — Architecture, ASIC, communication switching, middleware, reliability, satellite communication.**

I. Introduction

In the development process of a new satellite system, designers are facing complex problems. The first problem is the long development time of an avionics system and the huge costs because of the long time required for defining a new interface specification. The second important problem is the development of very expensive board computers. All devices in a satellite are connected to the board computer and for every new system it is required to define a new device configuration and redesign the board computer. The way of solving these problems was the implementation of a SCAN network [1], [3]. The central part of the whole SCAN system is the MW switch processor presented in this paper.

Spacecraft Area Network (SCAN) approach is based on a message distribution protocol defined by an interconnection link. The message distribution system is based on a publisher/subscriber model where each message has its own message topic identifier (TID). The switch controller is routing the received message to all subscribers which are related to the received TID. In the represented ASIC implementation of MW switch processor there are two types of serial interfaces [1] – S3P (Simple Synchronous Serial Protocol) and UART (Universal Asynchronous Receiver Transmitter).

Publishers (sensors, computing nodes) provide messages, which are public under a given topic. Subscribers (memories, actuators, computing nodes), configured to receive the messages from the defined topic, can receive all published messages (Fig. 1).

This system, based on the topic share between providers and consumers, provides an interconnect service. Services will be published as topics, regardless of whether they are produced by software tasks or by hardware devices. Therefore, the SCAN network can attach and use COTS [7] (Commercial off the Shelf) devices (with their own protocols) although they are not space verified. The network provides required interfaces and protocol converters. The COTS devices receive and send their own messages, and then the protocol converters translate them into our internal “universal language”. The network performs all required transformations in order to make the message transfer transparent. In this respect, the SCAN based system can integrate COTS components and provide a reliable architecture.

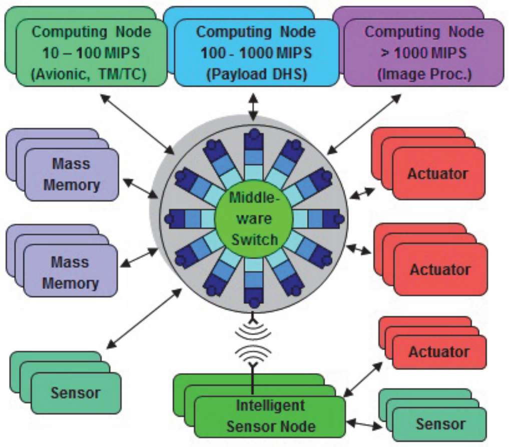

Fig. 1. MW switch processor integrated in the satellite system.

In Fig. 1 is presented the SCAN network. As we can see, the heart of the SCAN is the MW Switch processor, which performs all reliable actions.

The MW switch processor is implemented in a standard IHP 250 nm process [6] where the processor production funding is initialized from DLR [11]. Further development will be based on the radiation hard IHP 250 nm process using a fault tolerant technique, also provided from IHP [2], [5].

Vladimir Petrovic is with IHP, Im Technologiepark 25, 15236 Frankfurt Oder, Germany (phone: 49-335-5625357, e-mail: petrovic@ihp-microelectronics.com).

Marko Ilic is with Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia (phone: 381-63-1026759, e-mail: markoilic@elfak.rs).

Gunter Schoof is with IHP, Im Technologiepark 25, 15236 Frankfurt Oder, Germany (phone: 49-335-5625357, e-mail: schoof@ihp-microelectronics.com).

Sergio Montenegro is with Würzburg University, Am Hubland, 97074 Würzburg, Germany (e-mail: montenegro@informatik.uni-wuerzburg.de).

This paper provides the architecture description of the ASIC MW switch, different architectural approaches and the results of the MW switch processor implementation as well as measurements.

II. Middleware Switch Architecture

As the MW switch processor is the main component of the SCAN system, it needs to fulfill three important architectural requirements. The first requirement is high reliability - if the MW processor is not functioning, the complete SCAN system is not usable. In order to provide reliable usage of a SCAN system in the space environment, it is necessary to apply a new design concept. It is independent of the system architecture and used together with radiation hardened technologies. The ASIC design concept, presented in [5], provides a tradeoff between production costs and high system reliability.

The second important architectural requirement is scalability – fast and uncomplicated changes of active transferring data port number. Regarding the current data transfer standards in the space industry [8], the MW switch processor can achieve the required data throughput only in target topologies (parallel MW switch architecture or a combination of the pipelined-parallel architecture).

The third architectural requirement is flexibility – highly configurable ports for the communication of different device types connected to the SCAN.

It is important to note that more MW switch processors connected to the network also provide higher reliability and scalability of the complete satellite system.

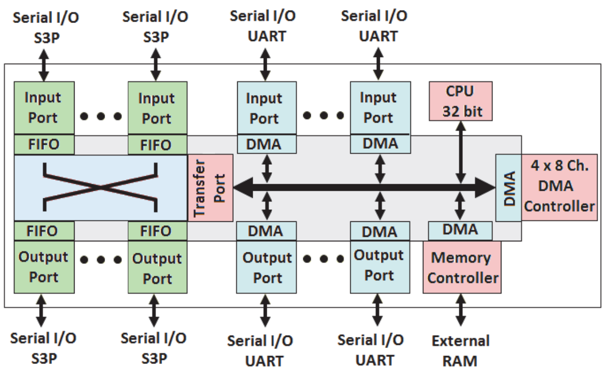

Fig. 2. Block diagram of an ASIC middleware switch.

The hierarchical view of the MW switch processor is divided into three main modular groups: communication ports, communication links and a control unit (CPU – Central Processor Unit). The basic block diagram of the implemented MW switch processor is represented in Fig. 1. Communication ports are connected via a switch matrix or via DMA (direct memory access) channels [10]. The switch matrix can be realized as a serial (one bit to one bit communication) or parallel (32 bit package to 32 bit package communication) one. The DMA channels are configured to transfer data between ports and main memory through a CPU memory controller. The switch matrix is controlled by the CPU (Leon2) in real-time and may be used to connect high speed ports directly to each other. Furthermore, one link of the switch matrix is connected to a DMA channel to allow data traffic to and from the memory. The CPU may communicate with each port, switch matrix and DMA controller to monitor and control their configuration, data traffic and flow control. The processor reads the message topic identifier (TID), directly from a message and, after processing in software, gives a corresponding instruction to the switch matrix and provides correct data traffic [1].

Simple synchronous serial ports (SSSP or S3P) are using a switch matrix for data transfer. Data are buffered in the FIFO buffers, where each port has one FIFO buffer for received data and another FIFO buffer for the data which will be sent. This concept provides different network protocol implementations (point-to-point, multicast, broadcast). Because of the required message length, implemented FIFO buffers are 2x4k bytes per S3P port, what requires more area.

The UART ports are using DMA channels and the AHB processor bus to transfer data between ports and external memory (in Fig. 2 an external memory link is shown). Before the transmission of messages from external memory to any destination device starts, all messages and destination configuration may be manipulated by the CPU.

The MW switch processor has the integrated hardware flow control. As there are different communication protocols, software flow control needs to be recognized and handled by port logic. Therefore, devices are sending or receiving flow control commands through the MW switch to avoid retransmissions or data losses. This is helpful at different transmission speeds to avoid FIFO or DMA buffer overflows.

The implemented MW switch uses a uniform internal protocol in order to provide simple switchable connections between different devices. The internal communication protocol is based on the S3P protocol and therefore S3P ports have very simple protocol converters. To allow different device types to be connected on the same UART port pin, the UART protocol convertor must be configurable, what provides required flexibility.

The CPU control of a S3P port is done through configuration registers connected on the APB processor bus. The CPU can configure the switch matrix, to initiate data transfer through the switch matrix, to send flow control commands to the external device and to start or stop data transfer from port to the external device. From another side, the S3P port can initiate the interrupts in different cases – FIFO is full, detected flow control sequence on input etc. The UART port is controlled by the CPU through APB processor bus, where the CPU can define in which mode the protocol convertor will be. The DMA controllers are set by the CPU in order to control data traffic.

Further implemented functions are a debug unit, programmable timers, pulse generators, and counters as well as scan chains and built-in-self-test (BIST) features. In the MW switch is also implemented the EDAC (error detection and correction) block between memory controller and external memory device. The EDAC is based on Hsiao [9] coding scheme and it is possible to detect and correct one bit error and to detect two bit errors without correction.

The system complexity of the middleware switch processor requires the analysis and classification of different architecture approaches which are used in the development process. The comparison of architecture approaches is based on four parameters. Classification of the analyzed architectures is done by answering the question - which effects can each part of the architecture have on the following parameters:

1. Implementation limits of data port transmitter/ receiver;
2. Implementation limits of internal MW switch processor data transfer system;
3. Number of data port types;
4. CPU resources and data traffic dependability;
5. CPU direct protocol conversion option.

Implementation limit of data port transmitter / receiver (ILTR) is the parameter that shows the possibility of the data transmitter or receiver architecture to be implemented on an ASIC chip. Example: Hardware realization of protocol convertors on receiving or transmitting side may require more hardware resources. Implementation limit is presented in percents and can be mathematically described as a relation between area required just for the standard cells (P_{std}) and area required for the standard cells after routing (P_{route}), related to equation (1). If the ILTR parameter is above 50%, it is recognized as a violation.

$$ILTR = \frac{P_{std}}{P_{route}} \times 100\,\%. \tag{1}$$

In order to explain the above mentioned parameter, we can discuss this effect, presented in Fig. 3. The area before routing is presented in Fig. 3(a). The cells are forming the minimum required area. After routing, the occupied area increases because of the routing resources. The routing resources are technology dependable (more routing layers - more saved silicon area).

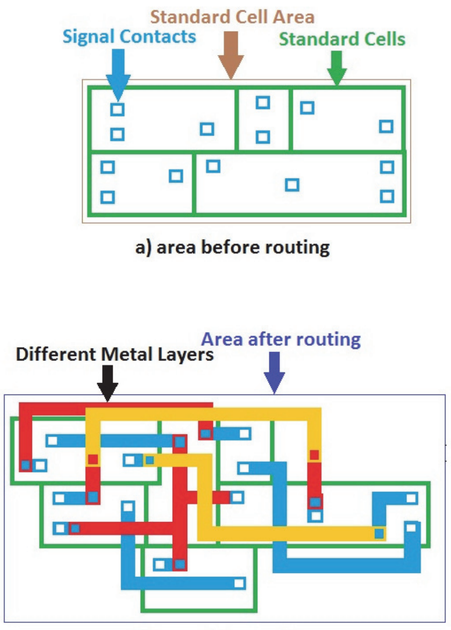

Fig. 3. Implementation limit of an ASIC digital system: (a) occupied silicon area before routing, (b) occupied silicon area after routing.

An implementation limit of internal MW switch processor data transfer system (ILTS) is the parameter that shows the ability to implement an internal data communication digital system between different data ports. The multiplexer is a building component of the MW switch internal data transfer system. Example: Implementation of switch matrix can be done in two different ways – serial and parallel internal data transfer. Implementation limit of internal data transfer system is calculated in the same way as ILTR parameter and presents the relation between area required for multiplexers (P_{mux}) and area required for multiplexers after routing (P_{route}), as it is described in equation (2). If the ILTS parameter is above 50%, it is recognized as a violation.

$$ILTS = \frac{P_{mux}}{P_{route}} \times 100\,\%. \tag{2}$$

Number of data port types (PT) is the parameter that shows how many different devices with different protocols can be connected to the network. Example: The MW switch implementation with only S3P protocol has one data port type and cannot transfer data from the commercial-off-the-shelf (COTS) devices with UART interface [7].

CPU resources and data traffic dependability (DTD) shows how data traffic is organized. Example: Data transfer process can be realized through special defined lines (switch matrix) or by using the CPU resources as APB or AHB bus.

CPU direct protocol conversion option (PCO) is the parameter that shows how many protocol versions are supported per data port type. Example: Usage of different COTS devices connected on the SCAN system e.g. data transfer on sensor – actuator net. The UART port can provide up to five different serial protocol types: without framing, begin and end of the message, terminated end of message, timing gap framing and header message recognition.

III. Architectural Approaches

The development process of completely new hardware from scratch requires a more detailed analysis. This is done in order to provide required reliability for satellites systems and better recognition of possible designing errors which increases the chip production price.

In order to test the capabilities and the functionality of the SCAN system, it was required to have enough communication ports. Before a detailed analysis, it was considered to use 32 communication ports. As there are two data port types (S3P and UART) and it is possible to connect them on the three different data transfer types (serial switch matrix, parallel switch matrix and DMA), we need to analyze throughputs of potential combinations.

Table 1 represents estimated data throughputs for different data port types (S3P, UART) connected on different data transfer types (serial switch matrix, parallel switch matrix and DMA) for only one message (1 kByte) and empty FIFOs. The S3P ports are designed as a high speed link up to 50 Mbits/s and in this estimation for the UART port is used baud rate of 115.2 kbit/s. From the data throughput table -Table 1, it is possible to see how a slow UART port connected on a DMA controller has better results than a fast S3P port connected on a DMA controller. Therefore, S3P port can be connected just on the switch matrix data transfer type in order to save processor resources. We can also see from the table that the parallel switch data transfer type can provide double faster data transfer, compared to the serial switch data transfer type. The UART port can be connected on all three data transfer types. The time (percent) of AHB usage per one message transmission (from/to one UART port to/from external memory) is estimated at 100MHz.

TABLE 1: DATA THROUGHPUT FOR PORT AND DATA TRANSFER TYPES.

	Max port speed [Mbit/s]	*Message delay time [us]*	*AHB usage per msg. [us]*	*AHB usage per msg. [%]*
S3P + serial switch	100	248.6	0	0
S3P + parallel switch	100	166.7	0	0
S3P + DMA	100	166.8	20.48	12.5
UART + serial switch	0.115	89.320	0	0
UART + parallel switch	0.115	89.238	0	0
UART + DMA	0.115	89.238	20.48	0.02

This section of the paper gives a more detailed and system oriented view of the central component of a SCAN satellite system, what the MW switch presents. The discussion is presented using four different architectures based on the combinations of transfer port types and data port types:

a) SM32 – architecture realized with 32 serial S3P ports, Leon2 CPU and a serial switch matrix;
b) SM16UM16 – architecture realized with 16 S3P ports and 16 UART ports all connected through a serial switch matrix and Leon2 CPU;
c) SM16UDMA16 – architecture realized with 16 UART, 16 S3P and Leon2 CPU where S3P ports are communicating through a serial switch matrix and UART ports are communicating through DMA controllers;
d) SM6UDMA8 – architecture with a reduced number of ports and the same strategy as the architecture c).

A. SM32 Architecture

The architecture is based on the S3P type of serial protocol and it is represented in Fig. 4. System consists of the CPU (Leon2) as a control unit, the serial switch matrix for controlled direction of data traffic and 32 S3P ports.

If we go through defined parameters for architectural analysis, the results we get are the following:

1. Implementation limits of data port transmitter/ receiver are not in risk for this type of architecture because of the simple protocol convertors;
2. Implementation limits of internal MW switch processor data transfer system in this case are critical because of the serial switch matrix which consists of the multiplexers (MUX) and big FIFO buffers. For the 32 S3P ports, we need 64 FIFO buffers;
3. The number of data port types is one, because we can connect just one type of devices based on S3P protocol;
4. CPU resources are not critically affected in this case, because the CPU controls just the serial switch matrix. All other BUS connections are free for software usage.

The processor cannot change the data which are in traffic. The CPU can control just the data destination.

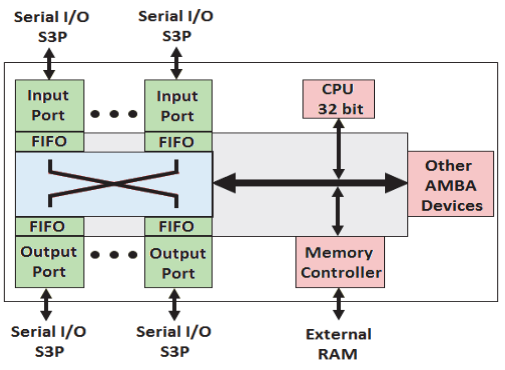

Fig. 4. Block diagram of SM32.

B. SM16UM16 Architecture

The architecture, described in this section, consists of 16 S3P ports, 16 UARTS, CPU (Leon2), two reduced serial switch matrices – one for the S3P ports and the second for the UART ports. Instead of S3P port concept, the UART port has five different programmable protocol converters (1. No framing, 2. Begin of message – end of message protocol, 3. Character terminated string, 4. Time-gaps, 5. Header/message length) to support the COTS devices connected to the network, where the CPU can set and control all protocol parameters. The architecture is represented in Fig. 5.

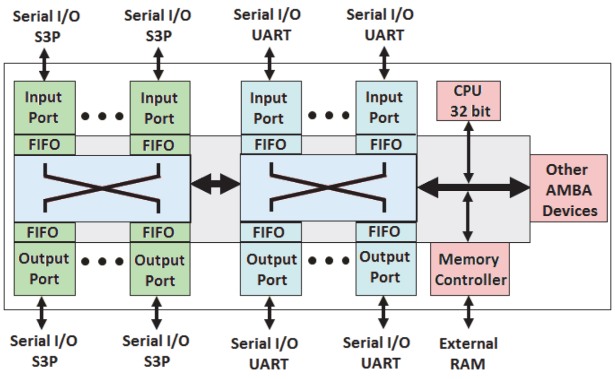

Fig. 5. Block diagram of SM16UM16.

The results after defined parameter analysis:

1. Implementation limit of data port transmitter/ receiver is 50% worse for this type of architecture because of the complex protocol convertors of UART data ports comprising a half of data ports of the whole system;
2. Implementation limit of internal MW switch processor data transfer system is still critical – instead of one big serial switch matrix there are two of them. The number of FIFO buffers is the same (64). Compared to the SM32 architecture, the routing resources are relaxed - the occupied silicon area is the same;
3. The number of data port types is increased from one to two;

4. The CPU resources and the data traffic dependability are not affected. All data transfers are done through the serial switch matrix without using any of the CPU data transfer resources;
5. For this architectural approach, it is possible to control directly the protocol conversion on the UART ports which is enough for the first SCAN tests.

C. *SM16UDMA16 Architecture*

The SM16UDMA16 architecture is represented in Fig. 6. It consists of 16 S3P ports, 16 UART ports, and a serial switch matrix for the S3P data transfer, 4 DMA controllers and a transfer port for data transfer from one data port type to another.

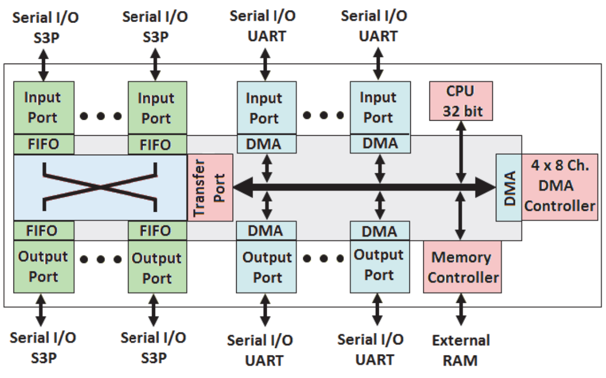

Fig. 6. Block diagram of SM16UDMA16.

The analysis of the SM16UDMA16 architecture based on defined parameters is as follows:

1. Implementation limit of data port transmitter/ receiver is the same as in previously discussed architecture;
2. Implementation limit of internal MW switch processor data transfer system is not critical any more – there is one serial switch matrix with 32 FIFOs for the S3P ports;
3. The number of data port types is the same as in the previous version (the SM16UM16 architecture);
4. CPU resources are affected but not critically. The DMA controller uses AHB BUS for the data traffic between memory controller and UART ports;
5. Control protocol conversion is performed by the CPU directly on the UART ports.

D. *SM6UDMA8 Architecture*

The SM6UDMA8 architecture is a reduced version of the SM16UDMA16 architecture due to reducing the area and costs of the chip fabrication.

IV. Architecture Analysis

After having analyzed the first architecture, the SM32, new ideas were opened for a better hardware realization. The first idea was the modification of the switch matrix approach to provide a better implementation process. The second point was based on reducing the area, providing more data port types (not just S3P) and the option to have different and programmable protocol converters.

Two switch matrix types were proposed – a serial and parallel type. The post-synthesis results of both switch matrix types have shown critical differences in power and area estimation, as it is presented in Table 2. Therefore, in the design we have used only a serial switch type regarding better implementation results.

Table 2: Power and area estimation for serial and parallel switch type.

	Serial Switch	***Parallel Switch***
Power [mW]	1.286	7.911
Area [um^2]	2009.904	33974.640

As one of the most important parameters – implementation limit of internal MW switch processor data transfer system was violated in both architectures (SM32 and SM16UM16), it was clear that a new solution was needed to be implemented. The first one was based on larger CPU resources usage (APB BUS or AHB BUS) for the data transfer in order to relax the serial switch matrix implementation and the number of FIFOs. The second solution was the option to transfer data from one data port type to another.

In the architecture SM16UDMA16, the serial switch matrix implementation problem was resolved by introducing DMA (Direct Memory Access) controllers [10]. The DMA controller provides data transfer from a port to the memory in a receiving mode and from the memory to a port in a transmitting mode through AHB BUS and a memory controller. The analysis of CPU resources, made by our tests, has shown that the usage of the AHB bus is about 50% without DMA controllers in a normal processor operation mode. Because of the slow UART ports, the data transfer through DMA controllers and the AHB bus doesn't have a critical impact on the CPU resources as it was shown in Table 1.

Table 3: Summarized dependences of different approaches.

Architecture / ***Parameter***	**SM32**	**SM16UM16**	**SM16U DMA16**	**SM6U DMA8**
ILTR	0%	0%	50%	50%
ILTS	Viol.	Viol.	OK	OK
PT	1	2	2	2
DTD	NO	NO	YES (AHB)	YES (AHB)
PCO	NO	YES	YES	YES

After parameter results analysis of all architectures, we can summarize results, which are shown in Table 3. From the information provided in Table 3, it is possible to note the dependence between architectural approaches and the required parameters of design. It is also possible to compare parameters, as power and area, after synthesis process of different architectural approaches what is represented in Table 4. Results are without calculations of power consumptions of FIFO buffers, processor caches and IO pads. This is done in order to compare only the results of the implemented logic.

If the design is based on the S3P port, the area will be bigger because of the FIFO buffers which are implemented. Instead of an S3P port, the UART port based architecture requires a smaller area but higher power consumption.

TABLE 4: POWER AND AREA CONSUMPTIONS OF ALL ARCHITECTURES.

Architecture	SM32	SM16UM 16	SM16U DMA16	SM6U DMA8
Parameter				
Area [mm^2]	65	59.73	54.14	34.6
Power Consumption [mW]	745.48	905.48	1195.56	755.04

As the SM6UDMA8 architecture is a reduced version of the SM16UDMA16 and fulfills all requirements for SCAN system tests, it is chosen for implementation.

V. IMPLEMENTATION

Technology used for MW switch implementation is SGB25IHP [6] technology because of the future planed implementation in radiation hardened IHP SGB25RH technology.

The implemented functionality of the S3P ports is based on using a serial switch matrix for data transfer. Data is buffered in FIFOs, where each port has one FIFO for data receiving and another FIFO for data sending. This concept provides different network protocol implementations (point-to-point, multicast, broadcast). Because of the required message length, implemented FIFOs are 2x4k bytes per S3P port. From another side, the UART ports are using DMA channels and the AHB processor bus to transfer data between ports and external memory.

The measurement results of the MW switch processor in the nominal temperature, voltage and process conditions are provided in Table 5. As we can see, the estimated power consumption presented in Table 4, is about 25% different. This difference is related to the worst case parasitic effects used for estimation during the chip development and production.

TABLE 5: MEASUREMENT RESULTS OF MW SWITCH PROCESSOR.

Implemented Architecture	SM6UDMA8
Parameter	
Maximal Frequency [MHz]	105.4
Power Consumption [mW]	537
Maximal Operating Temperature[ºC]	49.07
Power Supply [V]	2.5

During measurements we have provided the temperature analysis of the processor in order to find the potential "hot points". The results of thermal analysis are presented in Fig. 7. We can see that the MW switch processor has an almost uniform temperature distribution. Maximal temperature differences are less than ± 2 ºC.

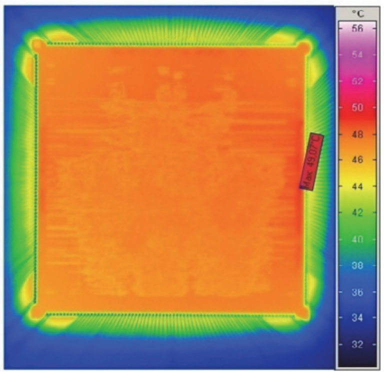

Fig. 7. Thermal analysis of MW switch during operation in nominal conditions.

The implemented MW Switch processor architecture (SM6UDMA8), after production, packaging and bonding is presented in Fig. 8.

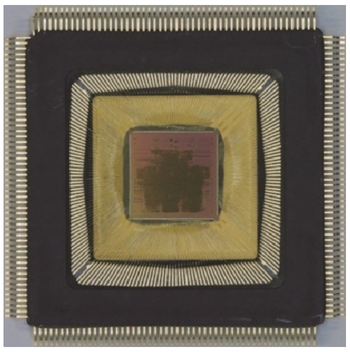

Fig. 8. The MW Switch processor after bonding and packaging.

VI. CONCLUSION

As the MW switch is the main part of a SCAN system, the analysis of different architectures and their classification in order to provide the best solution have been described in this paper. The chip is fabricated at IHP where all the required functionality tests have been successfully performed. Further system tests after integration of the complete SCAN system will be done in DLR. Future work will be based on the radiation hardened implementation of the MW switch processor with fault tolerant techniques in order to provide sufficient reliability. It is planned to use the IHPSGB13 (130nm) process to implement more communication ports and to increase data throughput.

REFERENCES

[1] S. Montenegro, V. Petrovic, and G. Schoof, "Network Centric Systems for Space Application," *IEEE conference SPACOM*, 2010.

[2] G. Schoof, R. Kraemer, U. Jagdhold and C. Wolf, "Fault Tolerant design for applications exposed to radiation," *Proceedings of the conference DASIA 2007 – Data Systems in Aerospace*, 2007.

[3] S. Montenegro and E. Hariran, "A Fault-Tolerant Middleware Switch for Space Applications," *Third IEEE International Conference on Space Mission Challenges for Information Technology*, pp. 333-340, 2009.

[4] S. Montenegro, F. Dannemann, B. Vogel, J. Gacnik, and M. Hannibal "(Spacecraft Bus Controller + automotive ECU)/2 = Unimate Controller" *Workshop ENVISION2020*, Paderborn, February 22-26, 2010.

[5] V. Petrovic, M. Ilic, G. Schoof, and Z. Stamenkovic, "Design Methodology for Fault Tolerant ASICs," *The 15th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems Symposium (DDECS 2012)*, Tallinn, Estonia, April 18 - 20, 2012.

[6] IHP SGB25V/SGB25RH technology and MPW shuttle service, Avaliable: http://www.ihp-microelectronics.com/12.0.html#36

[7] J. R. Marshall, "Building standards based COTS multiprocessor computer systems for space around a high speed serial bus network," *Digital Avionics Systems Conference*, Proceedings: 17th DASC. The AIAA/IEEE/SAE, vol. 1, 1998.

[8] S. Parkes, "SpaceWire for Adaptive Systems," Adaptive Hardware and Systems, 2008. AHS '08. NASA/ESA Conference on Digital Object Identifier: 10.1109/AHS.2008.30, pp. 77-82, 2008.

[9] M. Y. Hsiao, "A class of optimal minimum oddweight-column SECDED Codes," *IBM J. Res. Dev.*, 1970, 14, pp. 395-401.

[10] Synopsys DesignWare IP Library "DMAc", Avaliable: http://www.synopsys.com/IP/SoCInfrastructureIP/DesignWare/Pages/default.aspx

[11] DLR, Deutsches Zentrum für Luft- und Raumfahrt, Avaliable: www.dlr.de