2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops

A Triple Core Lock-Step (TCLS) ARM® Cortex®-R5 Processor for Safety-Critical and Ultra-Reliable Applications

Xabier Iturbe, Balaji Venu, Emre Ozer, Shidhartha Das
ARM Research, Cambridge, UK
{xabier.iturbe, balaji.venu, emre.ozer, shidhartha.das}@arm.com

***Abstract*—This paper introduces the ARM Triple Core Lock-Step (TCLS) architecture, which builds up on the industry success of the ARM Cortex-R5 Dual-Core Lock-Step (DCLS) processor currently used in safety-critical real-time applications. The TCLS architecture adds a third redundant CPU unit to the DCLS Cortex-R5 system to achieve fail functional capabilities and hence increase the availability of the system. The TCLS architecture allows for transparent, quicker and more reliable resynchronization of the CPUs in the event of an error as the erroneous CPU can be identified by comparing its outputs, and the correct architectural state can be restored from one of the other two functionally correct CPUs. The quick resynchronization is also possible because there is no need to correct the state of the cache memories, which are shared and isolated from the CPUs. As the TCLS architecture provides reliability at the system level, individual CPUs do not need to be fault-tolerant, and can be implemented using commercial technology process that provides higher performance, better energy and cost efficiency than rad-hard process technology. The expectation is that the TCLS could increase reliability in the industrial applications where ARM processors are mainstream (e.g., automotive), as well as in new applications where there is currently no presence of ARM technology (e.g., space).**

Keywords—Reliability, Lock-Step, Safety, Automotive, Space, ARM Processor

I. INTRODUCTION

As electronic systems are becoming ubiquitous in our lives, higher reliability is required for them. There are so many ways in which our lives depend on the correct functioning of some safety-critical equipment controlled by a microprocessor (ranging from planes to pacemakers) that we need to guarantee these will not be altered even when operating under unexpected conditions, such as environmental changes or operator errors [1]. Functional safety standards, such as the IEC-61508, provide a common framework on which to design safety-critical electronic products and systems. In automotive, the ARM® lock-step Cortex®-R processor has become a widely used solution to meet the safety integrity requirements imposed by the ISO-26262 standard.

There is also a need for (ultra-)reliability in applications that might not pose any threat for human lives, but which could result in huge economic losses in case of failure (e.g., space missions). For space applications, the energy and cost efficiency as well as the high performance delivered by ARM CPUs has attracted some space agencies and satellite manufacturers [2][3]. They see ARM Cortex CPUs as a promising alternative to overcome the currently existing performance crisis in the space sector, as long as they develop an adequate degree of reliability to operate in harsh radiation space environments.

This paper presents one of the ongoing research efforts at ARM to design a cost and energy efficient fail-functional processor that could be used in safety-critical and ultra-reliable applications alike. In order to achieve this objective, three ARM Cortex-R5 CPUs are integrated in a Triple-Core Lock-Step (TCLS) architecture. Upon any divergence in the three CPUs, the TCLS brings them back to a safe (architectural) state within a very short time in the range of microseconds.

The remainder of this paper is organized as follows. Section II describes the related work in the field of CPU reliability. Section III describes the ARM Cortex-R5 CPU used in the TCLS architecture. Section IV describes the TCLS architecture and Section V presents the preliminary implementation and performance results. Section VI discusses two use-cases of the TCLS drawn from telecom satellites that demand ultra-reliability, and autonomous vehicles that demand safety-criticality. Finally, Section VII concludes the paper.

II. RELATED WORK

A classical solution to make a CPU radiation-tolerant is to use rad-hard process technology [4]. However, this technology also increases the chip manufacturing costs, reduces the performance and leads to overheads in area and energy.

Radiation-tolerance can also be improved by using redundant components to detect failures and, if possible, mask out individual failures. At Register Transfer Level (RTL), redundancy is found in Error Detection And Correction (EDAC) schemes, parity codes and redundant elements (e.g., registers) with majority voters. LEON-3FT [5] is a space-grade CPU that relies on redundancy at RTL level and has been used by ESA in several space missions. Very recently, a quad-core version of this processor has been released: the GR740 [6]. Gate-level redundancy was applied in the ARM Cortex-R4 processor in the past. The results reported in [7] suggest an important reduction in the achievable performance resulting from the modification of the original optimized processor design to include the redundant registers. Namely, the

978-1-5090-3688-2/16 $31.00 © 2016 IEEE
DOI 10.1109/DSN-W.2016.57

maximum clock frequency usable in the RTL-redundant version of the Cortex-R4 is only 30% of that in the original processor. In addition to this, the verification and certification of the modified CPU would lead to increased design costs.

At architecture level, redundant CPUs can be integrated in the same board [8], or in the same chip [9][10], to improve reliability. In order to detect any divergences/errors between the redundant CPUs, their outputs can be compared at every clock cycle (i.e., tightly-coupled hardware lock-step) [9][10], or periodically (i.e., loosely-coupled software lock-step) [11]. The benefit of using already verified and certified redundant CPUs (i.e., "Plug-and-Play"), is the reduced engineering costs. However, there are still area and power overheads to allocate the energy-consuming redundant CPUs on the chip. As proposed by Yogitech, the latter overheads can be reduced, by using a simplified version of the CPU as a fault supervisor, named fRCPU [12]. The fRCPU consists of a CPU sniffer unit that collects, computes and codes signals from the CPU boundary, a shadow processing unit that executes the same CPU flow, a register bank that stores a shadow value of the CPU register files, and a management unit that generates data and addresses. Another solution to reduce area overheads is to run several virtualized partitions using a Hypervisor on the same CPU and periodically check their outputs (i.e., virtual lock-step) [13]. However, this solution is not immune to CPU common-source failures, such as failures in the clock network.

III. THE ARM CORTEX-R5 CPU

The ARM Cortex-R class CPUs offer energy-efficiency, high performance, highly deterministic behaviour, and built in safety features. They are specifically designed for deeply embedded real-time applications such as Electronic Control Units (ECUs) used in safety-critical automotive and industrial applications.

Fig. 1 shows a block diagram of the Cortex-R5 dual-core processor, which implements the ARMv7-R instruction set and also supports Thumb-2 for high code density. It has an 8-stage pipelined in-order dual-issue micro-architecture with SIMD and DSP support. The CPU microarchitecture implements dynamic branch prediction and Floating-Point Unit (FPU) with single and double-precision capability. The Memory Protection Unit (MPU) allows for protecting the access to different memory regions. Tightly-Coupled Memories (TCMs) enable low-latency and predictable execution of critical routines, such as interrupt handling routines and real-time tasks. In addition, TCMs can be used to hold scratch pad data, data types whose locality properties are not well suited to caching, and critical data structures such as interrupt stacks. The high priority Low-Latency Peripheral Port (LLPP) allows fast peripheral reads and writes, and the Accelerator Coherency Port (ACP) ensures input/output coherency through the micro-Snoop Control Unit (µSCU) by providing external DMA engines and non-cached coherent masters with access to the same physical memory view as the processor cluster. The Cortex-R5 also provides AMBA3 AXI bus interconnect support for external connection.

The Cortex-R5 delivers 1.67 DMIPS/MHz performance and 62 DMIPS/mW efficiency when implemented in 28nm HPM commercial process technology [14].

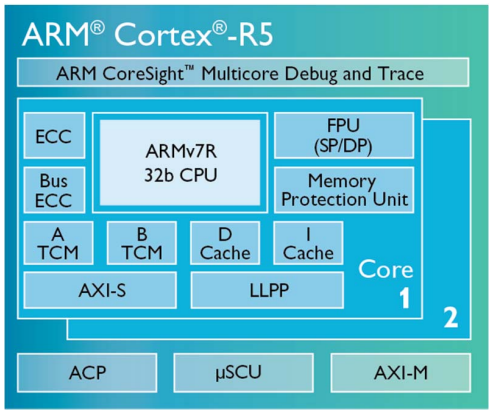

Figure 1. ARM Cortex-R5 CPU

As the Cortex-R5 is designed for safety-critical applications, it includes a number of hardware mechanisms to cope with soft and hard errors, such as parity checking and ECC (i.e., SECDED) in caches and TCMs. The AXI buses are also protected by ECC and parity bits (i.e., SECDED), thus extending soft error management out to the rest of the system. Besides, the Cortex-R5 can be used in Dual-Core Lock-Step (DCLS) configuration, where both cores share inputs and caches, and their outputs are compared at every clock cycle to detect errors.

IV. THE ARM TCLS CORTEX-R5

Fig. 2 shows the TCLS configuration for the Cortex-R5 CPU. This is a system-level solution to mitigate soft errors occurring inside the three redundant Cortex-R5 CPUs, which run in lock-step. Each Cortex-R5 CPU has its own clock tree, but share the instruction/data caches and TCMs, which are protected by SECDED ECC codes. A TCLS Assist Unit supports the lock-step functioning of the CPUs and drives the error recovery process. This unit consists of (1) Majority Voter, (2) Error Detection logic and (3) Resynchronization logic.

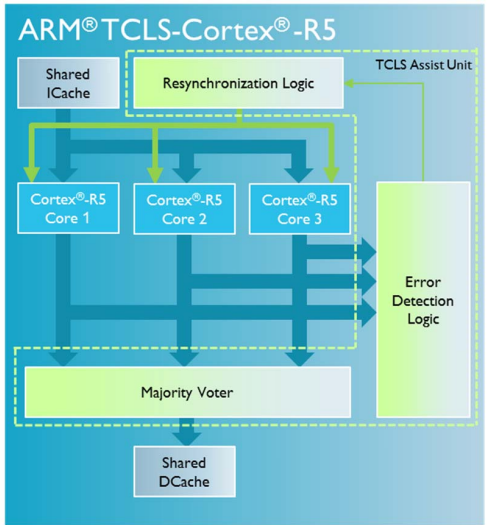

Figure 2. ARM TCLS-Cortex-R5

At every clock cycle, the instructions to execute are read from the shared instruction cache or TCM and distributed to the triplicated CPUs. The CPU outputs are majority-voted and forwarded to the shared data cache, TCM, and I/O ports. Simultaneously, the Error Detection logic checks if there is any mismatch in the outputs delivered by the three CPUs. If there is a mismatch, all CPUs are interrupted and the Error Detection logic identifies whether it is a correctable error (i.e., only one of the CPUs delivers a different set of outputs) or an uncorrectable one (i.e., all CPUs deliver different outputs). If the error is correctable, the TCLS passes the control to the Resynchronization logic to correct the architectural state of the erroneous CPU, that is, to resynchronize all the CPUs. Note here that the Majority Voter acts as an error propagation boundary, preventing correctable errors from propagating to memories and I/O ports. In the highly unlikely case that the error is uncorrectable, the TCLS transitions to a fail-safe operation state.

While the entirely combinational Majority Voter is on the critical path of the TCLS system, the Error Detection logic is out of this critical path and pipelined to increase performance.

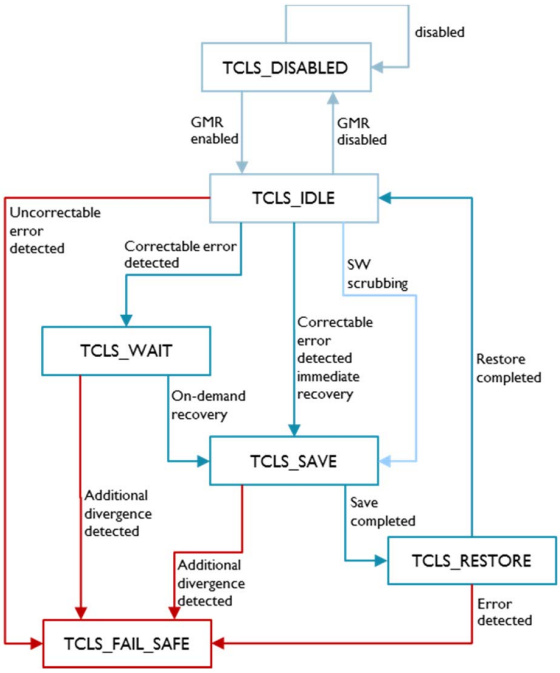

Figure 3. TCLS resynchronization FSM

Fig. 3 shows the flow-diagram of the Resynchronization logic's Finite State Machine (FSM). Upon a correctable error is detected, the TCLS Assist Unit can immediately trigger the CPU resynchronization process or wait for software to start it. This functioning permits to prevent the interruption of critical real-time tasks. Note that the TCLS system can still work safely with the two remaining CPUs, which are in a functionally correct state. In fact, the correct architectural state to be restored in the wrong CPU is recovered from these two CPUs. This is done by issuing an interrupt to the CPUs, which flushes their pipelines and pushes out their entire architectural state (i.e., register files, program counter and state registers). The CPU architectural states are passed through the Majority Voter and stored in a program stack mapped to the ECC-protected TCM. At the end of the interrupt, the CPUs enter the Wait for Event (WFE) low-power standby mode. When the Resynchronization logic observes the WFE signal, it issues a reset to the three CPUs to scrub away any soft error that might exist in their micro-architecture registers. This reset also wakes up the CPUs from the standby mode, initiating the restoration of the architectural state that was previously pushed onto the program stack. The last register to be restored is the program counter, thus resuming the normal operation of the CPUs at exactly the same point in the code where it was suspended to launch the resynchronization process.

Unlike in DCLS, where the error-recovery is application-dependent (e.g., checkpoint-rollback [15]), the recovery process in TCLS is automatic and transparent to the software.

V. Implementation and Performance Results

The area overhead of the TCLS Assist Unit is less than 5% of the three Cortex-R5 CPUs and 32 KB data and instruction caches. It takes up 18% of one Cortex-R5 CPU. Note that this percentage will decrease when using the TCLS architecture with newer and more complex ARM R-Class CPUs, such as Cortex-R7/8. The TCLS Assist Unit does not compromise the highest clock frequency of the Cortex-R5 processors, and can still operate at 1 GHz when implemented in 28nm HPM commercial process technology.

Table 1[*] shows the number of clock cycles needed to resynchronize the CPUs in TCLS when an error is detected. Note there is no equivalent behavior in DCLS, where error recovery is application-dependent. When using a 1 GHz clock signal, the CPU resynchronization is completed in less than 2 µs.

Table 1. TCLS resynchronization time (Clock cycles)

	Cache disabled	**Cache enabled**
Arch. state save	967	923
Arch. state restore	1,424	909
Total	**2,391**	**1,832**

For space applications, the TCLS Assist Unit is planned to be implemented using state-of-the-art rad-hard process technology. Note that this unit is critical as any malfunction could potentially result in undetected errors, memory corruption or wrong architectural state restoration. When targeting automotive applications, this unit will remain implemented using commercial process technology to preserve energy and cost efficiency, but perhaps using more robust cell libraries. With the objective of minimizing the probability that all CPUs are affected by a soft error at the same time, the three processors are to be spatially separated within the chip die, and oriented in different ways to minimize the amount of CPU surface that is exposed to radiation in the different directions.

[*] These numbers might be slightly different in the final TCLS implementation as we are now making minor improvements to it.

VI. USE-CASES FOR TCLS

A number of different software applications, even mixed-criticality applications, can be integrated onto the TCLS processor. This section briefly discusses two applications on which ARM is currently working to showcase the TCLS technology: telecom satellites and autonomous driving.

A. (Ultra-Reliable) Telecom Satellites

Three are the main demands for next-generation telecom satellites that are expected to be fulfilled by the TCLS processor: performance, reliability and availability. In effect, next-generation telecom satellites will need to harness more computing performance to deal with the increasing number of communication channels and to transition from channel switching to packet-switching, which demands more processing. Besides, new software architectures that also require more computing power, such as Time and Space Partitioning (TSP) (e.g., ARINC 653), are gaining momentum in the last ages. Telecom satellite availability is subject to very stringent requirements since customers require the service to be available at all times without any interruption. Finally, reliability is central in telecom satellites as the cost of designing and putting one satellite in operation exceeds hundreds of millions of dollars. Therefore, a satellite needs to be working (in the harsh space environment) for more than a decade in order to be profitable and any premature failure leads to huge economic losses.

B. (Safety-Critical) Autonomous Driving

ARM expects the processing demanded by autonomous cars compared to 2016 vehicles will increase 20x by 2018, 40-50x by 2020 and 100x by 2024 [16]. This increase will come from the necessity to execute complex navigation and guidance algorithms to transport the passengers from one location to another, real-time computer vision and sensor fusion algorithms to detect and track obstacles and potential threats for the navigation, as well as other control algorithms to drive the car's internal subsystems (e.g., engine) during the journey. In connection with this, a very low latency communication infrastructure will have to be deployed to direct the vehicles in a safely, coordinated and efficient way (e.g., warn about a blocked lane in a road), and to enable vehicle-to-vehicle (V2V) interactions. In this scenario, the communication infrastructure will have to deliver the appropriate instructions to each vehicle in a timely manner and the autonomous vehicles will have to follow the rules of the road strictly. Any single error in one of the processors involved in this functioning could potentially result in human live losses, and currently as the process technology node becomes smaller, soft errors are increasingly being detected in processors operating at ground level [17]. As a result, an unprecedented demand for real-time and safety-critical processors is expected in the next decades, and TCLS is thought to meet the performance and reliability requirements of some of the components involved.

VII. CONCLUSIONS

This paper has introduced the ARM Triple Core Lock-step (TCLS) architecture to build a cost and energy efficient ultra-reliable and safety-critical processor for telecom satellites and autonomous driving applications. The main advantage of TCLS compared to applying redundancy at gate-level is that it uses the original/optimized CPU design, thus resulting in reduced redesign costs and scaling well to further CPU generations. Besides, TCLS enables quick, automatic and transparent recovery from errors within microseconds, a feature that DCLS lacks. Future work will include testing the TCLS to evaluate its performance in a telecom satellite application, and injecting soft-errors to it to characterize its fault-tolerance capabilities.

ACKNOWLEDGEMENT

The research described in this paper has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 640354. Airbus DS, ARM, Atmel and Dolphin are part of this consortium and are collaborating to bring this innovative technology into the European space industry. Xabier Iturbe is funded by the European Union's FP7 Marie-Curie International outgoing fellowship program with agreement No. 627579.

REFERENCES

[1] M. Rausand, "Reliability of Safety-Critical Systems: Theory and Applications", Wiley, 2014.

[2] Statement of Work (SOW) For the Development of The High Performance Space Computing (HPSC) Processor: http://prod.nais.nasa.gov/eps/eps_data

[3] J. L. Poupat, B. Leroy and T. Helfers, "TCLS ARM for Space", Proc. of the Data Systems In Aerospace Conference, 2016.

[4] BAE Systems Plc., "RAD750 Radiation-Hardened PowerPC Microprocessor"', 2008.

[5] Aeroflex Gaisler, "UT699E 32-bit Fault-Tolerant SPARC V8 LEON 3FT Processor", 2013.

[6] Cobham, "GR740 Quad Core LEON4 SPARC V8 Processor", 2016.

[7] M. M. Ghahroodi, E. Ozer and D. Bull, "SEU and SET-Tolerant ARM Cortex-R4 CPU for Space and Avionics Applications", Proc. of the Workshop on Manufacturable and Dependable Multi-core Architectures at Nanoscale, 2013.

[8] Maxwell Technologies, "SCS750 Single Board Computer for Space". http://www.maxwell.com/images/documents/SCS750_rev8_r6.pdf

[9] ARM Ltd., "Cortex-R5 and Cortex-R5F. Technical Reference Manual", 2011.

[10] Infineon, "Infineon Tricore: Highly Integrated and Performance Optimized 32-bit Microcontrollers for Automotive and Industrial Applications", 2012.

[11] S. Resch, A. Steininger and C. Scherrer, "Software Composability and Mixed Criticality for Triple Modular Redundant Architectures", Proc. of the International Conference on Computer Safety, Reliability and Security, 2013.

[12] R. Mariani, T. Kuschel and H. Shigehara, "A Flexible Microcontroller Architecture for Fail-Safe and Fail-Operational Systems", Proc. of the HiPEAC Workshop on Design for Reliability, 2010.

[13] C. M. Jeffery and R. J. O. Figueiredo, "A Flexible Approach to Improving System Reliability with Virtual Lockstep", IEEE Transactions on Dependable and Secure Computing, Vol. 9, No. 1, pp. 2-15, 2012.

[14] http://www.arm.com/products/processors/cortex-r/cortex-r5.php

[15] D. Gizopoulos, M. Psarakis, S. Adve, P. Ramachandran, S. K. S. Hari, D. Sorin, A. Meixner, A. Biswas and X. Vera, "Architectures for Online Error Detection and Recovery in Multicore Processors", Proc. of the Design, Automation & Test in Europe Conference & Exhibition, 2011.

[16] http://www.arm.com/about/newsroom/arm-expects-vehicle-compute-performance-to-increase-100x-in-next-decade.php

[17] E. Normand, "Single Event Upset at Ground Level", IEEE Transactions on Nuclear Science, Vol. 43, No. 6, pp. 2742-2750, Dec. 1996.