Rad-Hard Microcontroller For Space Applications

Fredrik Johansson, Jan Andersson, Fredrik Sturesson,
Nils-Johan Wessman, Magnus Hjorth
Cobham Gaisler, Kungsgatan 12, SE-411 91, Göteborg, Sweden
Tel: +46 31 775 86 50
info@gaisler.com

Steven Redant, Wim Sijbers, Geert Thys
IMEC
Kapeldreef 75, 3001 Leuven, Belgium
Tel: +32 16 28 11 41
info@imec.be

Claudio Monteleone
European Space Agency,
Keplerlaan 1, PO box 299, NL-2220AG Noordwjik, The Netherlands
Tel: +31 71 56 56791
claudio.monteleone@esa.int

ABSTRACT

This paper describes a mixed-signal LEON3FT microcontroller ASIC (Application Specific Integrated Circuit) targeting embedded control applications with hard real-time requirements. The prototype device is currently in development at Cobham Gaisler, Sweden, and IMEC, Belgium, in the activity *Microcontroller for embedded space applications*, initiated and funded by the European Space Agency (ESA).

The presentation and paper will describe the architecture and functionality of the device. This abstract describes an on-going development where the devices are in the architectural design stage before detailed implementation phase. The presentation and final paper will contain further details on the device and will describe the latest progress made within the activity.

BACKGROUND

Software based data acquisition, dataprocessing and simple control applications are widely used in spacecraft subsystems. They allow implementation of software based control architectures that provide a higher flexibility and autonomous capabilities versus hardware implementations. For this type of applications, where limited processor performance are is required, general purpose microprocessors are usually considered incompatible due to high power consumption, high pin count packages, need of external memories and missing peripherals. Low-end microcontrollers are considered more attractive in many applications such as:

- Propulsion system control
- Sensor bus control
- Robotics applications control
- Simple motor control
- Power control
- Particle detector instrumentation
- Radiation environment monitoring
- Thermal control
- Antenna pointing control
- AOCS/GNC (Gyro, IMU, MTM)
- RTU control
- Simple instrument control
- Wireless networking

In these kind of applications the microcontroller device should have a relatively low price, a low power consumption, a limited number of pins and must integrate small amount of RAM and most of the I/O peripherals for control and data acquisition (serial I/Fs, GPIO's, PWM, ADC etc.). The requirements for memory and program length are usually minimal, with no or very simple operating system and low software complexity.

MICROCONTROLLER APPLICATIONS

Spacecraft subsystem control and monitoring of parameters such as power supply voltages, currents, pressures and temperatures are ideal applications for the LEON3FT microcontroller. Bridges between different communication standards or interface of an equipment towards a higher level controller or the central On Board Computer (OBC) are also ideal applications for the LEON3FT microcontroller.

The LEON3FT microcontroller can perform advanced data handling to offload any higher level controller or the central On Board Computer (OBC). By hiding the data handling details the transmitting data volume can be reduced and simplified functionalities and timing requirements are requested to the higher level controller.

The LEON3FT microcontroller integrates several on-chip data bus standards, such as SpaceWire, CAN, MIL-STD-1553, I2C, SPI, UART and can easily provide data packetization for serial communication using standard protocols. The microcontroller can also efficiently replace FPGAs in accomplishing the above functionalities. Generally the FPGA implementation is faster but much more complexity and flexibility can be captured in the software of a microcontroller even with limited processing capability. The correct use of FPGAs in space applications can be complex to achieve and also cost, package size and availability of integrated analog functions can favour the use of a microcontroller with respect to FPGA.

Below are listed a number of possible microcontroller use cases and specific applications.

- Nanosatellite controller
- Instrument Control Unit
- Remote Terminal control
- Mass Memory control
- Propulsion Unit control
- Electric Motor Control

MICROCONTROLLER ARCHITECTURE

Figure 1 shows an overview of the architecture. The system consists of three AMBA AHB buses, one main system bus, one debug bus and one bus for DMA traffic.

The main bus will include the LEON3FT core connected to a shared on-chip RAM and ROM. The main bus also connects all other peripheral cores in the design as well as the external memory controllers. Several peripherals are connected through two AMBA AHB/APB bridges where the bridges are integrated with the design's DMA controller.

The debug AMBA AHB bus connects a serial UART debug communications link and one JTAG debug communication link to the debug support unit and also to the rest of the system through an AMBA AHB bridge.

The third bus, a dedicated 32-bit Debug bus, connects a debug support unit (DSU), AHB trace buffers and several debug communication links. The Debug bus allows for non-intrusive debugging through the DSU and direct access to the complete system, as the Debug bus is not placed behind an AHB bridge with access restriction functionality.

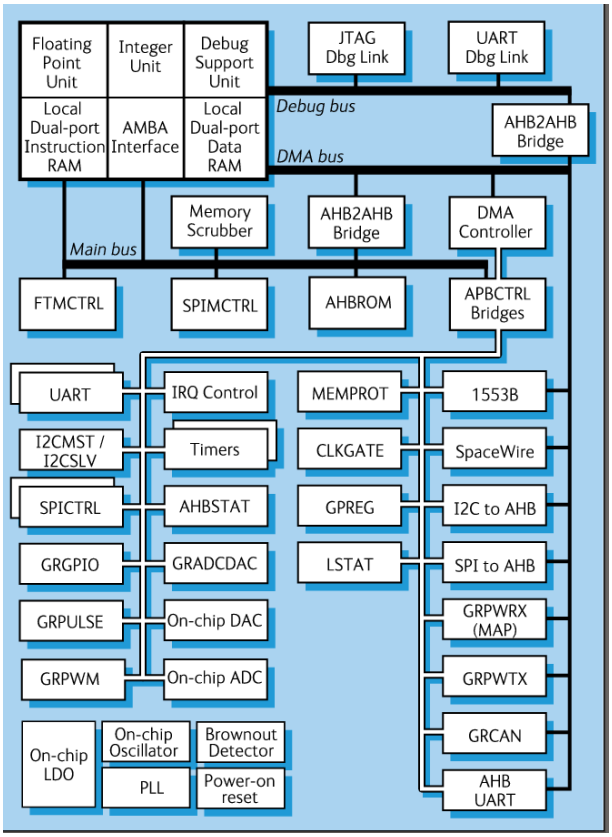

Fig. 1. Architecture overview

The list below summarizes the specification for the system:

- System Architecture
  - LEON3FT 32-bit SPARC V8 processor with LEON reduced instruction set
  - System AHB bus connecting processor, AHB/APB bridges
  - Separate debug AHB bus connecting debug communication links to DSU in order to allow non-intrusive debugging
  - On-chip ROM, SRAM.
  - Off-chip PROM, SRAM and memory mapped IO.
  - Pin sharing via switch matrix
  - Atomic operations
- Processor core
  - LEON3FT with tightly coupled data RAM and tightly coupled instruction RAM, 32-bit MUL
  - GRFPU floating-point unit with 4-word instruction FIFO
  - Reduced instruction set, without removing support for full instruction set.
  - SPARC V8e extensions (SVT, partial WRPSR, alternative window pointer)
  - Debug support unit with performance monitoring and AHB/instruction trace buffers

- Peripherals
  - Integrated on-chip 13-bit ADC/DAC, 100ksps
  - Power On Reset and Brown Out Detector
  - JTAG debug communication link
  - 8-bit UARTs
  - CCSDS CRC accelerator with DMA engine
  - MIL-STD-1553B interface
  - CAN controller
  - SpaceWire
  - General purpose I/O port
  - PWM and Pulse generator
  - PacketWire receiver and transmitter
  - I2C master and slave interface
  - SPI master/slave controller
  - Clock gate unit for power control
  - DMA controller
  - Hardware memory scrubber
  - Memory protection unit
  - Oscillator, PLL and pad control units
  - Temperature sensors
- External memory interfaces
  - Fault-tolerant 8 bit PROM/SRAM/IO controller with BCH ECC
  - SPI memory controller with support for Dual Memory Redundancy and BCH ECC
  - I2C memory controller with support for Dual memory Redundancy

Processor performance and determinism

In order to improve determinism, the LEON3FT microcontroller contains only a local instruction and data static RAM with fixed response times. All EDAC units in the system have the same latency and behaviour in the corrected as in the uncorrected case. This also applies to the CPU, so dynamic SEU handling schemes such as the LEON3FT pipeline restart on error options is not used.

Local instruction RAM tightly coupled to the LEON3FT CPU will be the main memory to execute the software. Due to its direct connection to the CPU, the execution of the software will be deterministic. For applications where full cycle-level determinism is not needed, it will also be possible to execute software from an external SRAM.

The local instruction memory will be implemented using dual-port RAM. The memory's second port will be connected to the main system AHB. This will allow modifying of the local instruction RAM without the intervention of the CPU. The contents of this memory will be protected against SEU errors with EDAC and scrubbing.

If the DMA peripherals and the processor are connected to a shared single-port memory, or to a memory via the same bus, and try to simultaneously access the shared resource then the DMA activity will have an effect on the execution time. On the other hand DMA activity will have no impact on SW execution time by using a dual-port on-chip data RAM and a separate bus for the DMA peripherals. This means that there is a separate access path for the CPU core to local instruction and data RAMs that is unaffected by concurrent DMA activity.

For applications demanding determinism on nested interrupts, a special interrupt handling scheme will be implemented in software where nested interrupts are allowed to occupy one additional register window. The number of levels of nested interrupts that can be handled without additional timing penalty depends on the complexity of the software implementation.

In the architecture, deterministic interrupt latency will be achieved by:

- Running software (including interrupt handlers) from local RAM and accessing any data needed during the interrupt handling through port separate from AMBA ports.
- Adapting the register window usage (using a flat model) structure to avoid unexpected window over/underflow traps. This is done in the compiler and application code, and most OS code does not need modification.
- The alternate window pointer feature from the SPARC V8E extension to allow window over/underflow handlers to run with traps enabled.
- Register file partitioning to allow partitioning of the register file (the windows) to different “contexts”. Contexts can for example be threads to speed up context switching and/or interrupt contexts to dedicate windows to ISRs.

SPARC Reduced instruction set

LEON-REX is an extension to the SPARCv8 instruction set. Similar extensions exist for other architectures such as THUMB/THUMB2 for ARM and MIPS16 for MIPS. The reduced SPARC V8 instruction set variant has been developed by Cobham Gaisler and is integrated into the device.

The main design goal has been to reduce code size, thereby reducing memory storage needed for the code, and to reduce memory bandwidth needed for the instruction code fetching.

Another design goal is to allow retrofitting the extension in existing LEON3/LEON3FT pipelines and into the existing software/compiler stack, and to provide backward compatibility. User can develop C code as usual (with bare-C or a small RTOS) and the existing

LEON environment (GRMON, compilers etc) can be used for development.

LEON REX is designed to allow gradual transition where existing SW environment can be used unmodified and converted piece by piece to use new instruction set.

The compressed instruction set is an optional extension of the SPARC V8 ISA, and existing code can be used without modification. Compressed and regular code can be mixed in the same application, thus the user can avoid changing critical code that has already been validated.

The first version of the instruction set extension has been specified and tested on prototype hardware and tests has shown that a compression ration of 30-50% compared to normal SPARC V8 code is achievable in a real world scenario.

LEON/REX and Runtime improvements

The new LEON/REX alternate window pointer feature (AWP) support and the improved interrupt single vector trap handler (SVT) have been tested and characterized in a series of measurements running on prototype hardware.

By delaying a timer interrupt N clocks into an overflow or underflow trap handling the interrupt latency and interrupt latency jitter as a result of SAVE/RESTORE can be quantified.

Five different software runtime configurations were benchmarked:

- Current BCC SVT
- Improved BCC SVT
- Improved BCC SVT with AWP
- Current BCC MVT
- Current BCC MVT with AWP

In order to understand where the latencies comes from the time from the interrupt is asserted to the time the ISR is reached is split up in three parts presented in the plots below:

- Interrupt assert to acknowledge (assert to first instruction of trap executed)
- Acknowledge to the Interrupt Service Routine (first instruction of trap to first instruction of ISR)
- Total latency (Assert to ISR first instruction)

The worst case interrupt latencies seen when an interrupt is asserted on top of a window underflow/overflow handler are presented in the table below. The highlighted rows are estimates results that can achieved in the LEON/REX environment.

Latency / Config	Assert to Acknowledge		Acknowledge to ISR		Total – Assert to ISR	
	Overflow	Underflow	Overflow	Underflow	Overflow	Underflow
CWP, SVT	134	143	539	281	673	424
CWP, new SVT	70	62	296	166	366	228
AWP, new SVT	**35**	**34**	**202**	**166**	**207**	**200**
CWP, MVT	60	52	262	152	322	204
AWP, MVT	**25**	**24**	**188**	**152**	**193**	**176**

Table 1: Worst case latencies measured

The new LEON/REX architecture also improves the “context” switching by allowing partitioning of the register file (the windows) to different “contexts”. By assigning windows to software threads or interrupts the software execution don't have to wait for the LEON3FT processor to store used windows on the stack.

Benchmark on prototype systems shows a large reduction of software execution time of switching “context”

Programmable DMA controller

Cobham Gaisler has developed a DMA controller able to preform concurrent programmable sequences of data transfers between any on-chip peripherals in the AMBA address space. The DMA controller is able to transfer data both between peripherals, between peripherals and memory and between memory areas. If the accessed memory is internal or external does not matter, as long as the memory is mapped into AMBA address space reachable from the AHB bus where the core is mapped.

The DMA controller has been specifically designed to offload the CPU and provide DMA capabilities to peripherals that do not have an internal DMA engine. The CPU is offloaded by the fact that the peripheral event is directly routed to the DMA controller. By routing events directly to the DMA controller or even directly between peripherals, these interrupts are in effect offloaded from the CPU. These reduce also the number of concurrent interrupts the CPU must handle and that may erode the system determinism.

Pin-multiplexing

The device shall be an attractive solution for a wide range of applications. Because of the small package and high number of interfaces, the functionality of the pins must be configurable and the pins must be shared between several peripherals. The number of configurable user pins has been chosen to be 64.

Clocking, reset and maximum frequency

The maximum operating frequency for the AMBA system is 50 MHz. The device can have separate clock signal inputs for system, SpaceWire, CAN and MIL-STD-1553B interfaces. The clocks signals can also be derived from single source via clock multipliers and dividers inside the device.

In order to avoid problems with reset sequencing, the device has one single reset input that is sequenced internally to provide reset signals to the different clock domains within the device.

SUPPORT FOR PROFILING AND DEBUGGING

The device provides debug interfaces via the JTAG and UART. The dedicated Debug bus allows non-intrusive debugging since the DSU, trace buffers and performance counters can be accessed without causing traffic on the Processor AHB bus.

The design also supports filtering for both the AHB and instruction trace buffers.

The LEON3 statistics unit provides performance counters, with support for filtering, for a large number of events, including:

- Data write buffer hold
- Branch prediction miss
- Total/Integer/Floating-point instruction count
- Total execution count
- AHB bus statistics for Processor AHB bus and Master I/O AHB bus

The interrupt controller in the design supports interrupt time stamping with time stamps interrupt line assertion and processor interrupt acknowledge.

SUPPORT FOR PROM-LESS APPLICATIONS

The device provides an easy access for systems that want to avoid having a boot-PROM connected to the device and prefer to upload software remotely.

The device can be accessed and remotely configured via SpaceWire, SPI, UART and I2C.

TARGET TECHNOLOGY AND PACKAGE

The technology used is UMC 180 nm, using the DARE library from IMEC, and the package is a 132 pin CQFP

SOFTWARE SUPPORT

The architecture is already supported by all operating systems and tool-chains provided by Cobham Gaisler.

CONCLUSION

The device in development is a SPARC V8 microcontroller that is based on the well known LEON architecture. The device is a prototype for a possible future device targeted at microcontroller applications and will have several new features that are not found in contemporary LEON devices. This includes architectural features to improve determinism, availability of the device in a low pin-count package, and support for the reduced instruction set.