Prevas Sierra - Real-Time Kernel in Hardware

- RTOS Acceleration

August, 2007 Product Specification

Prevas AB

Klockartorpsgatan 14
S – 723 44 Vasteras
Sweden

Phone: +46 (0)21 – 360 19 00
Fax: +46 (0)21 – 360 19 29
Email: info@prevas.se
URL: www.prevas.se

Features

- Scheduling mechanism in hardware working in parallel to the CPU
- Accelerates the scheduling, with **no** scheduling overhead
- 100 % deterministic
- Relieves pressure from the CPU
- 16 tasks, 8 external interrupts, 16 semaphores and 4 flags, timers for delay and periodic tasks
- Handles external interrupts as task with priority
- Small software API, only 2Kb in memory

Supported Devices

- Xilinx Virtex Family FPGAs
- Xilinx Spartan Family FPGAs
- Other programmable devices.

CORE Facts

Provided with Core	
Documentation	User's Reference Manual
Design File Formats	EDIF/ngc netlist; VHDL Source RTL (available at extra cost)
Constraints	UCF-file
Verification	Test bench and HW/SW system with PPC and MicroBlaze™ CPUs
Instantiation templates	VHDL
Reference designs & application notes	PPC and MicroBlaze Reference Platform Tooll Chain PPC and MicroBlaze Reference Platform application
Additional Items	A C-code API is delivered to utilize the functions of the Prevas Sierra RTOS.
Simulation Tool Used	
Model Technology ModelSim™ SE/EE	
Support	
Support provided by Prevas	

The Prevas Sierra Real-Time Kernel consists of:

- A priority driven scheduler
- A Resource Manager, which can be used for any kind of IPC, e.g. semaphores, flags
- A Time Manager, which contains functionality for handling delay and periodic start of tasks.
- An Intelligent Interrupt Handler

Family	Device tested	CLB Slices	Clock IOBs	IOBs	Performance (MHz)
Virtex-4	4VSX25ff668-12	551 (5 %)	1	47	141
Spartan-III	3S200ft256-5	553 (28 %)	1	47	77

Table 1: Example Implementation Statistics

Tmq	Time Manager
Rmq	Resource Manger
Irq	Interrupt Handler
GBI	Generic Bus Interface
TDBI	Technology Dependent Bus Interface

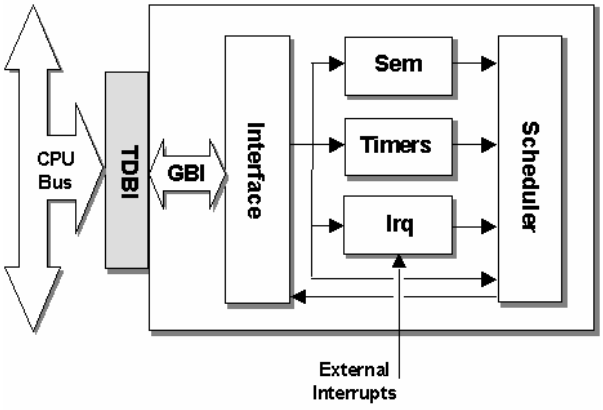

Figure 1 - Architecture of the Prevas Sierra

In general purpose OS´s time management control and all types of resource handling are time consuming due to large amount of queue handling. All types of queue handling, by that means searching in a specific queue, eat a lot of CPU cycles.

However, moving this to hardware much of the job can be performed in parallel to the CPU. The Sierra has an intelligent Interrupt Handler that makes interrupt handling much simpler. At the same time it improves performance and behavior.

Briefly, the external interrupts are routed to the Sierra and each interrupt service routine, ISR, is treated as a task waiting for an external event. When an interrupt occurs the ISR is scheduled as an ordinary task by the Sierra and will be started when it has the highest priority in the ready state.

Benefits

- When using the Sierra, there is almost no OS-footprint in memory. Merely a driver to communicate with the hardware kernel
- Less cache misses, in systems with a cache, since the Sierra driver has such a small memory footprint.
- Optimized system behavior as the CPU-load is minimized
- Complete system predictability since hardware is 100 % deterministic.
- Plug in architecture; add new components when needed.

Application

The Sierra solution is suitable for all kinds of small embedded systems, System-On-Chip (SoC) for example, and real-time systems in general to improve performance and predictability. Moreover, the Sierra will simplify your system design and shorten time to market.

Specifics

In this version, Sierra supports 16 tasks at 8 priority levels, 16 semaphores, 4 flags, and 8 external interrupts.

On request, the Sierra can be extended to other configurations, e.g. more tasks, resources etc. Contact Prevas for more information.

Functional description

The core is partitioned into modules as shown in and described in the text below.

Core Engine

The Sierra is partitioned into functional units

- Sierra Interface
- Scheduler
- Interrupt Handler
- Resource Manager; can be used for semaphores and flags
- Time Manager

Prevas Sierra interface

The interface to the Sierra is divided into a generic bus interface, GBI, and a technology dependent bus interface, TDBI. The GBI is bus independent while the TDBI is dependent on the specific bus in the system. This design of the Sierra makes it very easy to interface it towards different kinds of busses.

The Sierra can be delivered with TDBI for the CoreConnect - OPB or LMB bus for MicroBlaze and PowerPC CPU. The Sierra edif/ngc netlist has the GBI allowing the user to attach desired TDBI. The TDBI is delivered as source code. A customer can also develop other specific TDBIs or let Prevas do it for a reasonable fee.

All communication with the Sierra is carried out through a set of registers as service calls. The service calls are decoded in the Sierra interface and routed to the specific unit that will handle the service call.

Scheduler

The Scheduler controls all scheduling in the Sierra. The number of tasks that the Sierra can handle is 16 and there are 8 priority levels for the tasks.

Normally a number of tasks are created when the system is initialized. However, the Sierra allows tasks

to be created and deleted dynamically during runtime. When a task is created it is initialized to a specified state (blocked or ready). Tasks must have a priority and the priority must be initialized when the task is created. The scheduler guarantees that the task with the highest priority in the system that is ready. The scheduler uses a FIFO scheme for tasks on same priority level.

The Sierra supports the following task management primitives:

- Create/Delete task
- Block/Unblock task
- Enable/disable context switch
- Yield task
- Get task status

Interrupt Handler

The response time for interrupts generated by external devices must be kept short in all kinds of systems to obtain successful interaction with the external environment. Normally, external interrupts are routed to the CPU, which means that when an external interrupt occurs, the executing task will be interrupted. This has of course effects on the predictability of the system, as an ISR (Interrupt Service Routine) can preempt and run before a high priority scheduled task.

When using the intelligent Interrupt Handler, external interrupts are routed in to the Sierra. Each ISR is treated as a task waiting for an external event. When an interrupt occurs the ISR is scheduled by the Sierra and will start running when it has the highest priority in the ready queue.

The following primitives for interrupt handling exist:

- Initialize an interrupt task
- Wait for interrupt
- Remove interrupt
- Get ISR-task status

Resource Manager

The Sierra provides 16 semaphores and 4 flags. Each of these can be used for any form of interprocess communication, IPC, e.g. protecting of shared memory etc.

The following synchronization primitives are supported:

- Set flag/Clear flag/Wait for flag(s)
- Take/release semaphore
- Read semaphore value

Time Manager

There are two time handling functions implemented, delay and support for periodic tasks. In software implementation, these functions will increase the CPU load.

In ordinary software RTOS the principle for the delay functionality is as follows. Every clock tick the RTOS has to check the delay queue and decrease each tasks timer and examine if any timer has expired. When a task timer expires the task has to be scheduled by the system again. All this work has a cost in time and this time increases with number of tasks using the delay function.

When it comes to the Sierra, all handling with timers etc. is done in hardware and all cost in time have been removed from the system.

The Sierra supports following time management primitives:

- Set/get time base
- Initialize periodic time for a periodic task
- Wait for next period
- Start/stop period
- Delay/underlay

Pinout

The generic bus interface, GBI, signals are described in figure 2 and table 2 below.

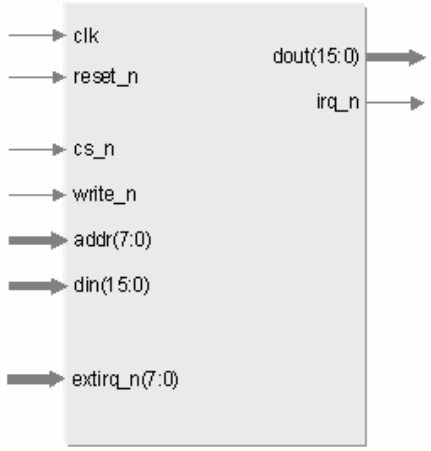

Figure 2: Sierra Generic Bus Interface (GBI)

Signal	Direction	Description
clk	input	System clock
reset_n	input	HW reset
cs_n	input	Chip select
write_n	input	Read / Write
addr(7:0)	input	Address bus
din(15:0)	input	Data bus in
extirq_n(7:0)	input	External interrupts
dout(15:0)	output	Data bus out
irq_n	output	Task switch interrupt

Table 2: Sierra Pinout

Verification Methods

Functional simulation of the Sierra RTOS carried out using Model Technology ModelSim™ SE/EE.

The core has also been verified and used in systems with Spartan and Virtex FPGAs.

Functional co-verification of the Sierra kernel hardware together with the software API has been executed in Xilinx Embedded Development Kit (EDK).

Recommended design experience

Users of this core should be familiar with HDL design and Xilinx design flows. Some knowledge in real time systems might help to understand how this core works. Experience in designing systems with CPUs and other devices are recommended. This core can easily be integrated in any kind of system that has a CPU.

Design Services

Prevas also offers core integration, core customization and other design services.

Ordering Information

This product is available from Prevas, under terms of the SignOnce IP License. See www.prevas.se for additional information about this product.

Pevas AB
Klckartorpsgatan 14
S – 723 44 Vasteras
Sweden

Phone: +46 (0)21 – 360 19 00
Fax: +46 (0)21 – 360 19 29
Email: info@prevas.se
URL: www.prevas.se

Related Information

Xilinx Programmable Logic

For information on Xilinx programmable logic or development system software, contact your local Xilinx sales office, or visit www.xilinx.com.