

## LEON-PF-EX Quick Start Guide

# Table of Contents

1. Introduction .....	3
1.1. Overview .....	3
1.2. Availability .....	3
1.3. Prerequisites .....	3
1.4. References .....	3
2. Overview .....	4
2.1. Board .....	4
2.2. The design .....	4
2.3. Debug tools .....	4
3. Board Configuration .....	5
3.1. Debug connectors .....	5
3.2. LEDs .....	5
3.3. Push buttons .....	5
3.4. DIP switches .....	5
3.5. Memories .....	5
3.5.1. ....	5
3.6. Programming the FPGA .....	5
4. Software Development Environment .....	6
4.1. Overview .....	6
5. GRMON hardware debugger .....	7
5.1. Overview .....	7
5.2. Debug-link alternatives .....	7
5.2.1. Connecting via the USB JTAG connector .....	7
5.2.2. Connecting via the UART debug link .....	7
5.3. First steps .....	7
5.4. Connecting to the board .....	7
5.5. Get system information .....	8
5.6. Load a RAM application .....	8
6. RTEMS Real Time Operating System .....	9
6.1. Overview .....	9
6.2. Installing RCC .....	9
6.3. Building an RTEMS sample application .....	9
6.4. Running and debugging with GRMON .....	9
7. Support .....	11

# 1. Introduction

## 1.1. Overview

This document is a quick start guide for the LEON-PF-EX example designs.

The purpose of this document is to get users quickly started using the board.

This quick start guide does not contain as many technical details and is instead how-to oriented. However, to make the most of the guide the user should have glanced through the aforementioned documents and should ideally also be familiar with the GRMON debug monitor.

## 1.2. Availability

The FPGA bitstreams are available on the LEON-PF-EX web page: <https://www.gaisler.com/LEON-PF>.

Sample linux images to load and run are available at <https://www.gaisler.com/anonftp/linux/linux-2.6/images/>.

## 1.3. Prerequisites

To use the provided bitstream, the user needs:

- Microchip MPF300-SPLASH-KIT
- GRMON3, latest available version, available at <https://www.gaisler.com/index.php/downloads/debug-tools>.
- Microchip FlashPro Express (to program the FPGA). FPEXpress is available at <https://www.microsemi.com/product-directory/programming/4977-flashpro>.

## 1.4. References

Table 1.1. References

RD-1	The SPARC Architecture Manual, Version 8, Revision SAV080SI9308
RD-2	GRMON User's Manual [ <a href="https://www.gaisler.com/doc/grmon3.pdf">https://www.gaisler.com/doc/grmon3.pdf</a> ]
RD-3	RTEMS homepage [ <a href="https://www.rtems.org">https://www.rtems.org</a> ]
RD-4	RTEMS User Manual [ <a href="https://docs.rtems.org/branches/master/user/index.html">https://docs.rtems.org/branches/master/user/index.html</a> ]
RD-5	LEON/ERC32 RTEMS Cross Compilation System (RCC) [ <a href="https://www.gaisler.com/index.php/products/operating-systems/rtems">https://www.gaisler.com/index.php/products/operating-systems/rtems</a> ]
RD-6	RCC User's manual [ <a href="https://gaisler.com/anonftp/rcc/doc">https://gaisler.com/anonftp/rcc/doc</a> ]
RD-7	Cobham Gaisler RTEMS driver documentation [ <a href="https://gaisler.com/anonftp/rcc/doc">https://gaisler.com/anonftp/rcc/doc</a> ]
RD-8	Bare C Cross-Compilation System [ <a href="https://www.gaisler.com/index.php/products/operating-systems/bcc">https://www.gaisler.com/index.php/products/operating-systems/bcc</a> ]
RD-9	BCC User's Manual [ <a href="https://www.gaisler.com/doc/bcc2.pdf">https://www.gaisler.com/doc/bcc2.pdf</a> ]
RD-10	VxWorks 7 SPARC architectural port and BSP [ <a href="https://www.gaisler.com/index.php/products/operating-systems/vxworks-7">https://www.gaisler.com/index.php/products/operating-systems/vxworks-7</a> ]
RD-11	LEON-PF-EX User Manual [ <a href="https://www.gaisler.com/index.php/products/processors/leon-examples/leon-pf">https://www.gaisler.com/index.php/products/processors/leon-examples/leon-pf</a> ]
RD-12	MPF300-SPLASH-KIT User Guide [ <a href="https://www.microsemi.com/existing-parts/parts/150866#resources">https://www.microsemi.com/existing-parts/parts/150866#resources</a> ]

## **2. Overview**

### **2.1. Board**

The LEON-PF-EX example designs can be used on the following board:

- Microchip MPF300-SPLASH-KIT

### **2.2. The design**

The SoC system is described in the LEON-PF-EX User's manual LEON-PF-EX-UM, available at <https://www.gaisler.com/LEON-PF>. For details about the the interfaces' connections in the board see (Chapter 3).

### **2.3. Debug tools**

Non-intrusive debugging of the template design and application execution can be performed using the GRMON hardware debugger.

## 3. Board Configuration

This chapter describes boards items as used by the LEON-PF-EX design.

### 3.1. Debug connectors

- J1: USB JTAG/UART interface via FTDI with mini-USB connector. See (Chapter 5).

### 3.2. LEDs

- LED[ 1 . . 4 ]: Connected to GPIO outputs [0..3] (active LOW).
- LED[ 5 ]: When OFF the UART interface in J1 is configured as debug link. When ON the UART interface is configured as console UART.
- LED[ 6 ]: When OFF indicates that the CPU is in error mode.
- LED[ 7 ]: When ON indicates that the memory controller calibration is complete and the FPGA design has access to the on-board SDRAM.
- LED[ 8 ]: When OFF indicates that the reset is asserted.

### 3.3. Push buttons

- SW[ 3 ]: Connected to DSUBBREAK signal. Push to break software execution.
- SW[ 4 . . 6 ]: Connected to GPIO inputs [4..6] .

### 3.4. DIP switches

- DIP[ 1 ] acts as select signal for the UART interface. When "ON" it selects the UART debug link. When "OFF" it selects the console UART.
- DIP[ 2 . . 4 ]: Connected to GPIO inputs [7..9] .

### 3.5. Memories

The LEON-PF-EX has 1 GiB of SDRAM available on the on-chip bus.

### 3.6. Programming the FPGA

The bitstream folder contains several FPEXpress programming job files (.job) which represent different configurations of the processor (EX1,EX2,EX3 and EX4). Select one of the bitstreams (described in [RD-11]). and follow the instructions below to program the FPGA:

1. Connect the PC and the board using a standard micro-USB cable into the connector USB-JTAG J1.
2. Launch FlashPro Express.
3. Open a new job project and select the provided Programming Job File (.job).
4. Click on the "RUN" button and wait until the action is complete.
5. Once the FPGA has been programmed, it is possible to connect to the board through GRMON, using the command:

```
grmon -ftdi
```

Please see (Chapter 5) for further information regarding GRMON and the available debug links.

## 4. Software Development Environment

### 4.1. Overview

Cobham Gaisler provides a comprehensive set of software tools to run several different operating systems. The LEON5 platform supports the following:

BCC	the Bare C Cross-Compiler System is a toolchain to compile bare C or C++ applications directly on top of the processor without the services provided by an operating system
RTEMS	a hard Real Time Operating System. Cobham Gaisler provides, for LEON5, a preliminary toolchain and kernel to develop and compile RTEMS applications.
Linux	the open source operating system. Board Support Packages and tools to ease the compilation and deployment of the kernel are provided
VxWorks	an embedded real-time operating system developed by WindRiver. Cobham Gaisler provides a LEON architectural port (HAL) and a Board Support Package (BSP) in full source code

Cobham Gaisler also provides debug tools. The LEON5 platform is supported by the following:

GRMON	Used to run and debug applications on LEON-PF-EX hardware. See (Chapter 5).
-------	---

The recommended method to load software onto LEON-PF-EX is by connecting to a debug interface of the board through the GRMON hardware debugger (Chapter 5).

## 5. GRMON hardware debugger

### 5.1. Overview

GRMON is a debug monitor used to develop and debug GRLIB systems with NOEL and LEON processors. The target system, including the processor and peripherals, is accessed on the AHB bus through a debug-link connected to the host computer. GRMON has GDB support which makes C/C++ level debugging possible by connecting GDB to the GRMON's GDB socket. With GRMON one can for example:

- Inspect LEON5 and peripheral registers
- Upload applications to RAM with the **load** command.
- Program the FLASH with the **flash** command.
- Control execution flow by starting applications (**run**), continue execution (**cont**), single-stepping (**step**), inserting breakpoints/watchpoints (**bp**) etc.
- Inspect the current CPU state listing the back-trace, instruction trace and disassemble machine code.

The first step is to set up a debug link in order to connect to the board. The following section outlines which debug interfaces are available and how to use them on the LEON-PF-EX example designs. After that, a basic first inspection of the board is exemplified.

GRMON is described on the homepage [<https://www.gaisler.com/index.php/products/debug-tools>] and in detail in [RD-2].

### 5.2. Debug-link alternatives

#### 5.2.1. Connecting via the USB JTAG connector

Connect the PC and the board using a standard min-USB cable into the connector USB-JTAG J1 and issue the following command:

```
grmon -ftdi
```

If the the debug link is not established, please see the section "FTDI FT4232/FT2232" of [RD-2].

#### 5.2.2. Connecting via the UART debug link

Make sure that the switch DIP[ 1 ] select the UART debug link (ON position). Connect the PC and the board using a standard micro-USB cable into the connector USB-UART J4 and issue the following command:

```
grmon -uart /dev/ttyUSB0
```

### 5.3. First steps

The previous sections have described which debug-links are available and how to start using them with GRMON. The subsections below assume that GRMON, the host computer and the LEON-PF-EX board have been set up so that GRMON can connect to the board.

When connecting to the board for the first time it is recommended to get to know the system by inspecting the current configuration and hardware present using GRMON. With the **info sys** command more details about the system is printed and with **info reg** the register contents of the I/O registers can be inspected. Below is a list of items of particular interest:

- AMBA system frequency is printed out at connect, if the frequency is wrong then it might be due to noise in auto detection (small error). See `-freq` flag in [RD-2].
- Memory location and size configuration is found from the **info sys** output.

### 5.4. Connecting to the board

In the following example the JTAG debug-link is used to connect to the board. The auto-detected frequency, memory parameters and stack pointer are verified by looking at the GRMON terminal output below.

```
grmon -ftdi
```

GRMON debug monitor v3.2.8.3-101-gf451057 64-bit internal version

Copyright (C) 2020 Cobham Gaisler - All rights reserved.  
For latest updates, go to <http://www.gaisler.com/>  
Comments or bug-reports to [support@gaisler.com](mailto:support@gaisler.com)

This internal version will expire on 27/11/2021

Parsing -ftdi

Commands missing help:

JTAG chain (1): MPF300T  
GRLIB build version: 4260  
Detected frequency: 50.0 MHz

Component	Vendor
LEON5 SPARC V8 Processor	Cobham Gaisler
LEON5 Debug Support Unit	Cobham Gaisler
AHB Debug UART	Cobham Gaisler
JTAG Debug Link	Cobham Gaisler
L2-Cache Controller	Cobham Gaisler
AHB/APB Bridge	Cobham Gaisler
PolarFire FDDR4 Controller	Microsemi Corporation
General Purpose I/O port	Cobham Gaisler
AHB Status Register	Cobham Gaisler
Generic UART	Cobham Gaisler
Multi-processor Interrupt Ctrl.	Cobham Gaisler
Modular Timer Unit	Cobham Gaisler

Use command 'info sys' to print a detailed report of attached cores

## 5.5. Get system information

One can limit the output to certain cores by specifying the core(s) name(s) to the **info sys** and **info reg** commands. As seen below the memory parameters, first UART and first Timer core information is listed.

```
grmon3> info reg uart0
Generic UART
0x80000104  UART Status register          0x00000086
0x80000108  UART Control register           0x80000003
0x8000010c  UART Scaler register                   0x00000145
grmon3> info sys gptimer0
gptimer0  Cobham Gaisler  Modular Timer Unit
          APB: 80000300 - 80000400
          IRQ: 8
          16-bit scalar, 2 * 32-bit timers, divisor 100
```

## 5.6. Load a RAM application

An application linked to RAM can be loaded directly with the **load** and run with **run**.

```
grmon3> load hello.elf
40000000 .text          142.0kB / 142.0kB  [=====] 100%
400237D0 .rtmsroset      96B                [=====] 100%
40024840 .data           4.4kB / 4.4kB     [=====] 100%
Total size: 146.44kB (777.96kbit/s)
Entry point 0x40000000
Image hello.elf loaded

grmon3> forward enable uart0
I/O forwarding to uart0 enabled

grmon3> run
hello, world

CPU 0: Program exited normally.
```

The two lines starting with Hello World is the program output which is forwarded to the GRMON terminal.



## 6. RTEMS Real Time Operating System

### 6.1. Overview

RTEMS is a real time operating system maintained at [RD-3] that supports the LEON CPU family. Cobham Gaisler distributes a precompiled RTEMS toolchain for LEON called RCC [RD-7]. This section gives the reader a brief introduction on how to use RTEMS together with the LEON-PF-EX example designs. It will be demonstrated how to install RCC and build an existing sample RTEMS project from RCC and run it on the board using GRMON.

The RCC toolchain includes a prebuilt toolchain with GNU BINUTILS, GCC, NewlibC and GDB for Linux and Windows (mingw). It also contains prebuilt RTEMS kernels for the LEON2, LEON3/4/5 BSPs single-core and for multi-core development, see [RD-6] for more information. The LEON BSP specific drivers are documented in [RD-7].

Samples RTEMS projects are available within the toolchain package, installed into `rtems-x.y/src/samples`.

### 6.2. Installing RCC

The RCC toolchain is downloadable from the RCC homepage at [RD-7]. The full installation procedure is found in the RCC manual [RD-6]. Windows users are recommended to install the UNIX-like environment MSYS before proceeding.

The installation process of RCC is straight forward by first extracting the toolchain into `C:\opt` or `/opt` on Linux, then extracting the source distribution into the `/opt/rtems-x.y/src/` directory. In order for the compiler to be found one has to add the binary directory `/opt/rtems-x.y/bin` into the PATH variable as below:

```
$ cd /opt
$ tar -xf sparc-rtems-4.10-...-linux.tar.bz2
$ cd rtems-4.10/src
$ tar -xf rtems-4.10-...-src.tar.bz2
$ export PATH=$PATH:/opt/rtems-4.10/bin
```

### 6.3. Building an RTEMS sample application

Once the toolchain is set up, you can compile and link a sample RTEMS application by doing:

```
sparc-rtems-gcc -g -O2 rtems-hello.c -o rtems-hello
```

RCC's gcc creates executables for LEON3/4/5 by default. The default load address is at the start of the RAM, i.e. 0x40000000. All compilation options are described in [RD-6], but some useful options are reported below:

*Table 6.1. RCC's GCC compiler relevant options*

<code>-g</code>	generate debugging information - must be used for debugging with gdb
<code>-msoft-float</code>	emulate floating-point - must be used if no FPU exists in the system
<code>-mcpu=v8</code>	generate SPARC V8 mul/div instructions - needs hardware multiply and divide
<code>-O2</code> or <code>-O3</code>	optimize code maximum performance and minimal code size

### 6.4. Running and debugging with GRMON

Once your executable is compiled, connect to your LEON-PF-EX with GRMON. The following log shows how to load and run an executable. Note that the console output is redirected to GRMON by the use of the `-u` command line switch, so that printf output is shown directly in the GRMON console.

```
[andrea@localhost samples]$ grmon -ftdi -u

GRMON2 LEON debug monitor v2.0.42 internal version

Copyright (C) 2013 Aeroflex Gaisler - All rights reserved.
For latest updates, go to http://www.gaisler.com/
Comments or bug-reports to support@gaisler.com
```

```

Parsing -ftdi
Parsing -u

[...]

grmon2> load rtems-hello
40000000 .text                136.4kB / 136.4kB [=====] 100%
400221A0 .data                4.4kB / 4.4kB [=====] 100%
40023350 .jcr                  4B [=====] 100%
Total size: 140.83kB (780.05kbit/s)
Entry point 0x40000000
Image /home/andrea/Desktop/samples/rtems-hello loaded

grmon2> run
Hello World

CPU 0: Program exited normally.
CPU 1: Power down mode

```

To debug the compiled program you can insert break points, step and continue directly from the GRMON console. Compilation symbols are loaded automatically by GRMON once you load the executable. An example is provided below.

```

grmon2> load rtems-hello
40000000 .text                136.4kB / 136.4kB [=====] 100%
400221A0 .data                4.4kB / 4.4kB [=====] 100%
40023350 .jcr                  4B [=====] 100%
Total size: 140.83kB (781.11kbit/s)
Entry point 0x40000000
Image /home/andrea/Desktop/samples/rtems-hello loaded

grmon2> bp Init
Software breakpoint 1 at <Init>

grmon2> run

CPU 0: breakpoint 1 hit
0x400011f8: 1110007f sethi %hi(0x4001FC00), %o0 <Init+4>
CPU 1: Power down mode

grmon2> step
0x400011f8: 1110007f sethi %hi(0x4001FC00), %o0 <Init+4>

grmon2> step
0x400011fc: 4000003b call 0x400012E8 <Init+8>

grmon2> cont
Hello World

CPU 0: Program exited normally.
CPU 1: Power down mode

grmon2> Exiting GRMON

```

Alternatively you can run GRMON with the `-gdb` command line option and then attach a gdb session to it. For further information see Chapter 3 of [RD-6].

## 7. Support

For support contact the Cobham Gaisler support team at [support@gaisler.com](mailto:support@gaisler.com).

When contacting support, please identify yourself in full, including company affiliation and site name and address. Please identify exactly what product that is used, specifying if it is an IP core (with full name of the library distribution archive file), component, software version, compiler version, operating system version, debug tool version, simulator tool version, board version, etc.

There is also an open forum available at <https://glib.community>.

**Cobham Gaisler AB**  
Kungsgatan 12  
411 19 Gothenburg  
Sweden  
[www.cobhamaes.com/gaisler](http://www.cobhamaes.com/gaisler)  
[sales@gaisler.com](mailto:sales@gaisler.com)  
T: +46 31 7758650  
F: +46 31 421407

Cobham Gaisler AB, reserves the right to make changes to any products and services described herein at any time without notice. Consult Cobham or an authorized sales representative to verify that the information in this document is current before using this product. Cobham does not assume any responsibility or liability arising out of the application or use of any product or service described herein, except as expressly agreed to in writing by Cobham; nor does the purchase, lease, or use of a product or service from Cobham convey a license under any patent rights, copyrights, trademark rights, or any other of the intellectual rights of Cobham or of third parties. All information is provided as is. There is no warranty that it is correct or suitable for any purpose, neither implicit nor explicit.

Copyright © 2020 Cobham Gaisler AB