GR-VPX-GR740 Quick Start Guide

Table of Contents

1. Introduction

1.1. Overview

This document is a quick start guide for the GR-VPX-GR740 VPX Development Board.

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

For a complete description of the board please refer to the GR-VPX-GR740 Development Board User's Manual.

The GR740 system-on-chip is described in the GR740 Data sheet and User's Manual.

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.

Information in this document applies to GR-VPX-GR740 Revision 1.2 or later. Please contact support@gaisler.com for technical questions and for document versions applicable to earlier board revisions. The GR-VPX-GR740 data package and this document (including possibly newer revisions) are available from the GR-VPX-GR740 product page at https://www.gaisler.com.

1.2. References

Table 1.1. References

RD-1	GR-VPX-GR740 Development Board User's Manual
RD-2	GR740 Data sheet and User's Manual [https://www.gaisler.com/doc/gr740/GR740-UM-DS-2-3.pdf]
RD-4	GRMON User's Manual [https://www.gaisler.com/doc/grmon3.pdf]
RD-6	RTEMS homepage [https://www.rtems.org]
RD-7	LEON/ERC32 RTEMS Cross Compilation System (RCC) [https://www.gaisler.com/index.php/products/operating-systems/rtems]
RD-8	RCC User's manual [https://gaisler.com/anonftp/rcc/doc]
RD-9	Cobham Gaisler RTEMS driver documentation [https://gaisler.com/anonftp/rcc/doc]
RD-10	Bare C Cross-Compilation System [https://www.gaisler.com/index.php/products/operating-systems/bcc]
RD-11	BCC User's Manual [https://www.gaisler.com/doc/bcc2.pdf]
RD-12	VxWorks 7 SPARC architectural port and BSP [https://www.gaisler.com/index.php/products/operating-systems/vxworks-7]
RD-13	MKPROM2 User Manual [https://gaisler.com/doc/mkprom.pdf]

The referenced documents can be downloaded from https://www.gaisler.com.

2. Board Configuration

2.1. Overview

The primary sources of information are the GR-VPX-GR740 Development Board User's Manual and the GR740 Data sheet and User's Manual. Before start using the GR-VPX-GR740, bootstrap signals need to be set correctly and the desired interfaces have to be enabled.

2.2. Default configuration

The complete default configuration can be found in GR-VPX-GR740 Development Board User's Manual section 5.1.1 and 5.2. If this is your first time using the GR-VPX-GR740, please use this configuration as a starting point.

2.3. Clocking

The board uses a common 50 MHz oscillators and a clock splitter to generate the memory (MEMCLK), SpaceWire (SPWCLK) and System (SYSCLK) clocks.

The internal frequencies for system, memory and spacewire depend on the PLL bypass[0:2] settings set on the onboard switches (S9, S11, S13), as shown in Table 2.1 and Table 2.2 (Default configuration highlighted in light blue). The default configuration with all clocks installed uses all PLLs enabled, as shown in the table.

Table 2.1. System and Memory clock configuration

Bypass[0]	Bypass[1]	clksel	JP13 1-2	JP13 3-4	System clock	SDRAM clock	PLLs enabled
0	0	1	open	open	5xSYS=250	2xMEM=100	Sys, Mem
1	0	1	open	open	1xSYS=50	2xMEM=100	Mem

Table 2.2. Spacewire clock configuration

Bypass[2]	SpW clock	PLLs enabled
0	8xSPW=400	SpW
1	1xSPW=50	-

Default configuration

The default configuration of the board uses 50 MHz oscillators and a clock splitter to generate the memory (MEMCLK), SpaceWire (SPWCLK) and System (SYSCLK) clocks.

PLLs are enabled, i.e. bypass signals are disabled. The result is System clock at 250 MHz, SDRAM at 100 MHz and Spacewire at 400 MHz.

By default, grmon(version 3.2.2 or later) sets up the sdram timing parameters such that the memory works both at 50MHz and 100MHz SDRAM clock. However, to have the correct performance, user need to set startup option -sdfreq <*mhz*>. Please refer section Section 5.2 for grmon command formats for different debug link connections.

2.4. Bootstrap Signals

Bootstrap signals configure the chip on reset and are listed in section *Bootstrap signals* of GR740 Data sheet and User's Manual. Some of these signals can be controlled on the GR-VPX-GR740 via the switches S1 to S22. The bootstrap signals that are mapped to the general purpose I/O lines control settings such as the reset address for the Ethernet debug communications link (EDCL), routing of EDCL traffic, boot-PROM width and PROM EDAC enable.

- The DSU enable signal S5 controls if the design's Debug Support Unit is enabled and also if the debug communication links are active. Switch S5 must be set to HIGH, i.e. DSU enabled, to connect to the board using the GRMON3 debug monitor.

- The MEM_CLKSEL signal S7 selects the clocks used for the SDRAM memory interface and the on-chip buses. If the S7 is HIGH the source for the memory clock is the MEM_EXTCLOCK clock input, otherwise the memory clock and the system clock have the same source.
- The BYPASS_PLL signals (S9, S11, S13) are explained in Section 2.3.
- The PLL_IGNLOCK signal (S15) is explained in GR740 Data sheet and User's Manual.
- The WDEN signal, JP7 jumper controls if a watchdog timeout in the GR740 will trigger a board reset. This jumper should be OPEN in order to disconnect the watchdog from the reset circuit.

The GPIO bootstrap signals are as follows:

- The GPIO[0:3] signals (S1, S2, S3 and S4) control EDCL IP address. All pulled up disable EDCL on ETH0.
- The GPIO[6:7] signals (S10 and S12) control Spacewire router distributed interrupt configuration.
- The GPIO[8] signals (S14) control EDCL routing. Pulled up route EDCL traffic to the debug bus.
- The GPIO[10] signal (S17) controls PROM width. This must be set LOW since the board only support 8-bit interface. Pulled down selects 8-bit, otherwise 16-bit.
- The GPIO[11] signal (S18) controls Spacewire router clock gating settings. Pulled up disables the Spacewire router on the clock gating unit.
- The GPIO[12:13] signals (S19 and S20) set Spacewire router's instance ID.
- The GPIO[14] signal (S21) controls PROM EDAC. Pulled down disables PROM EDAC, otherwise enabled.
- The GPIO[15] signal (S22) controls PROM/IO pin multiplexing (after reset). Pulled down enables PROM interface, otherwise alternative functions are enabled. Note that this can be changed later by software (see Section 2.5).

2.5. Pin multiplexing

2.5.1. PROM/IO, UART, CAN, MIL-STD-1553B, Spacewire, general purpose I/O

The GR740 shares some of the PROM/IO pins due to a limited number of pins. There are three configurations available: a) PROM/IO; b) UART, CAN, MIL-STD-1553B and Spacewire debug interfaces; and c) general-purpose I/O. See section *Pin multiplexing* of the GR740 Data sheet and User's Manual.

There are two things to take into consideration when configuring the pin multiplexing: The configuration on the GR-VPX-GR740 board and the configuration on the GR740 device. First, to configure the board set GPIO[15] to LOW for full PROM/IO mode (a) or HIGH for peripheral mode (b). GPIO[15] controls the reset value of the FTMEN register in the GR740, which belongs to the GRGPRBANK core (mapped in memory at 0xFFA0B000). FTMEN configures each pin between configurations (a) and (b) (another register, called ALTEN is used for configuration (c)). GPIO[15] LOW puts all bits to 1 and HIGH to 0. Second, is to configure the GR740 device registers, FTMEN (mapped in memory at 0xFFA0B000) must be configured to 0x00000400 and ALTEN (mapped in memory at 0xFFA0B004) must be configured to 0x003FC73C.

Note the board make use of PROMIO_ADDR[16] (128 KiB MRAM boot memory), UART0 and UART1 without flowcontrol, MIL-STD-1553B, GPIO2[0], GPIO2[1], GPIO2[6], GPIO2[7], GPIO2[11], GPIO2[12] and GPIO2[13] from the GR740 device. The GR740 Data sheet and User's Manual section *PROM/IO interface multiplexing* and section *Register Bank For I/O and PLL configuration registers* provides additional information.

In this board the GPIO[15] is set to HIGH (default during delivery of the board) which enables the use of UART and 1553 available on board. This configuration is usefull only for a system which does not make use of the MRAM boot memory. To properly boot the system using the MRAM boot memory set the GPIO[15] to be LOW, and program the FTMEN and ALTEN to the required values.

2.5.2. SDRAM, PCI, Ethernet port 1

In the GR740 SOC the top half of the SDRAM interface shares pins with PCI and Ethernet port 1. In this board only the PCI interface is used and mapped to the mezzanine interface connector.

2.6. Interfaces

This section describes how to set up the main different interfaces of the GR-VPX-GR740. If you are not interested on a specific interface not used on the default configuration, skip that part.

2.6.1. JTAG FTDI

A FTDI FT423HL chip provides a JTAG to USB conversion supported by GRMON3 (see Chapter 5). Jumpers JP8 have to be installed to use JTAG to USB. The JTAG interface is connected to the ADBUS of the FTDI FT423HL chip. Please see section *FTDI (USB Serial)* of the GR-VPX-GR740 Development Board User's Manual.

2.6.2. Ethernet

There is one ethernet ports available in the GR-VPX-GR740, port 0 from the GR740 SOC. To use the Ethernet interfaces for the EDCL Debug link (see Chapter 5), it is necessary to appropriately set the GPIO signals at power-up/reset. GPIO[0..5] (Switch S1 to S4) set the least significant address bits of the IP address of each port and also disable EDCL if all of them are High. GPIO[8] (Switch S14)set the routing of the ethernet traffic on the Debug AHB bus, required to use EDCL, for port 0. Please see section *Ethernet* of the GR-VPX-GR740 Development Board User's Manual.

2.6.3. Boot memory MRAM

Make sure that the pin multiplexing is configured to use PROM/IO, as explained in Section 2.5. The Everspin MR0A08B, which is a 128k x 8bit, 3.3V device. There is a hardware write-protection of the memory by removing jumper JP1. Please see section *PROMIO / Parallel flash* of the GR-VPX-GR740 Development Board User's Manual.

2.6.4. UARTs

Make sure that the pin multiplexing is configured to use UARTs, as explained in Section 2.5. The UARTs of the GR740 is connected to the FTDI USB to serial converter. This configuration is chosen with jumpers JP9 on the board. The UART0 interface is connected to the CDBUS of the FTDI FT423HL chip. The UART1 interface is connected to the DDBUS of the FTDI FT423HL chip. Please see section *FTDI (USB Serial)* of the GR-VPX-GR740 Development Board User's Manual.

2.6.5. MIL-STD-1553B

Make sure that the pin multiplexing is configured to use MIL-STD-1553B, as explained in Section 2.5. The MIL-STD-1553B interfaces of the GR740 is connected to the DSUB-9 connector on the Front panel. Please see section *MIL-STD-1553 Interface* of the GR-VPX-GR740 Development Board User's Manual.

2.6.6. SPI

The SPI interface of the GR740 is connected to a SPI FLASH device on the GR-VPX-GR740. The GR-VPX-GR740 provides the S79FL512S device, selected with the SPI_CS0 output of the ASIC. It consists of two SPI devices internally. Clock and chip select is common for the two internal chips bus the MOSI/MOSI buses are independent. Only the first data bus is connected with the GR740 and the GR740 has only one SPI bus. The consequence is that only half of the device capacity can be used. Hence S79FL512S provides 32 MiB memory capacity. Please see section *SPI Interface* of the GR-VPX-GR740 Development Board User's Manual.

2.6.7. Spacewire

The board incorporates a large number of SpaceWire Links distributed between the VPX backplane, GR740 processor, mezzanine connector, external front panel connectors and on board header/connector. Please see section *SPW Interfaces* of the GR-VPX-GR740 Development Board User's Manual.

Make sure that the Spacewire router is not clock-gated off by the clock gating unit in order to use the Spacewire ports. The default configuration of the board do not clock gate the Spacewire router. Set Switch S18 to HIGH to change this and enable the Spacewire router clock gating by default.

The complete default configuration can be found in section 5.1.1 and 5.2.1 of the GR-VPX-GR740 Development Board User's Manual. The referenced documents can be downloaded from https://www.gaisler.com/gr740.

3. Comments on System-on-Chip Design

3.1. Overview

The goal of this section is to summarize differences with the GR740 device compared to other contemporary LEON systems and what these differences mean to users. The GR740 Data sheet and User's Manual has a section named Technical notes that contain additional information on this topic specific for the GR740.

3.2. Building Operating Systems for GR740

Operating systems must take into account that the RAM starts at 0x0 in the GR740 design. This is typically done by specifying special build options. Operating systems distributed by Cobham Gaisler have been extended to support linking to address 0x0 instead of the 0x40000000 address that is traditionally used in LEON systems. Please refer to the operating system documentation for additional information.

Software must also take into account that in the GR740 the peripheral units are connected through an AHB-to-AHB bridge. This has no impact for normal memory accesses but existing software may not support AMBA plug and play scanning over the AHB-to-AHB bridges. All recent versions of operating systems distributed by Cobham Gaisler will correctly detect the peripheral devices in GR740. Support for recursive plug and play scanning over bridges is present in BCC version 1.0.41 (software compiled by earlier versions of BCC need to have been built using the `-qambapp` flag), RTEMS version 4.10, VxWorks 6.3/6.5/6.7 release 1.0.3, MKPROM2 version 2.0.56, and later versions of these software releases available from Cobham Gaisler.

3.3. Building Applications for GR740

No special consideration needs to be taken for applications that run on top of an operating system.

Special flags must be specified when using the Bare-C Compiler (BCC) toolchain available from Cobham Gaisler. In order to link the application to address 0 the flags `-Wl,-msparcleon0` must be specified. In order to enable plug and play scanning over AHB-to-AHB bridges, the `-qambapp` must be specified. The functionality enabled by `-qambapp` is enabled by default starting with BCC version 1.0.41. With version 1.0.41 or higher of BCC the following call would compile a hello world application: **sparc-elf-gcc -Wall --Wl,-msparcleon0 hello.c -o hello**

3.4. Running binaries linked to address 0x40000000

Note that legacy software linked to address 0x40000000 may still be used on the GR740 under the right conditions. Either by having 2 GiB of memory installed or by ensuring that the memory will wrap at address 0x40000000.

The memory controllers will wrap at the end of RAM so by installing a smaller amount of memory (less than 2 GiB) it is possible to run applications from 0x40000000 as the memory area will wrap and it will access the same external memory positions as an access to address 0x0. For this workaround to properly work the following conditions must be met:

- The installed memory detected by GRMON3 must be of a size so that the memory will actually wrap at address 0x40000000. This will happen for memories that have bank sizes less than, or equal to, 500 MiB. For example, a 1 GiB single rank memory will not work as the first 1 GiB will be mapped in the range 0x00000000 - 0x3FFFFFFF, the area 0x40000000 - 0x7FFFFFFF will then cause the second chip select to be asserted.
- The stack pointer must be set so that it takes the 0x40000000 offset into account.
- The software may not assume that it can access the range 0x00000000 - 0x3FFFFFFF (this area will contain the PROM and memory mapped I/O areas on legacy LEON systems). Accesses to this range will modify the RAM.
- The software must support plug and play scanning over bridges, or not depend on finding peripherals through plug and play scanning.

3.5. Considerations when enabling the Level-2 cache

The Level-2 cache is disabled after system reset and should be enabled in order to improve system performance. It is important that the Level-2 cache contents is invalidated before the cache is enabled. Otherwise the power-on contents of the cache RAMs may be interpreted as valid data by the cache. Please note that L2 cache is automatically enabled by GRMON2/GRMON3 (see Chapter 5).

4. Software Development Environment

4.1. Overview

Cobham Gaisler provides a comprehensive set of software tools to run several different operating systems. The GR740 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 RCC, a toolchain to develop and compile RTEMS applications specifically for the LEON
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 a set of debug tools. The GR740 platform is supported by the following:

GRMON	Used to run and debug applications on GR-VPX-GR740 hardware. See (Chapter 5).

Developer tools are generally provided for both Linux and Windows host operating systems. Cobham Gaisler also provides an integrated, easy-to-use solution to help programmers with the task of developing for the LEON. The LEON Integrated Development Environment for Eclipse (LIDE) is an Eclipse plug-in integrating compilers, software and hardware debuggers in a graphical user interface. The plugin makes it possible to cross-compile C and C++ application for LEON, and to debug them on either simulator and target hardware (TSIM or GRMON3).

The recommended method to load software onto a LEON board is by connecting to a debug interface of the board through the GRMON3 hardware debugger (Chapter 5). Execution of programs by a PROM-loaded boot loader is also possible.

4.2. Boot Loaders

Cobham Gaisler provides three boot loaders for the ERC32, LEON2, LEON3 and LEON4 processors listed below for more information. The boot loaders covers different use cases and requirements on software quality level. The boot loaders are all capable of booting all the supported Operating Systems provided by Cobham Gaisler.

MKPROM2	MKPROM2 is a free open-source boot loader supporting a minimal system initialization, extraction of a single ROM application image into main memory and booting it. No system self-tests are performed by MKPROM2.
GR712RC Boot SW	The GR712RC Boot SW was specifically developed for the ESA JUICE satellite and will be used in several of its GR712RC based payloads on board following the requirements of ESA's *Flight Computer Initialisation Sequence* requirement document. It supports initialization, self-tests, a SpaceWire remote terminal as Standby Mode and CRC checking application loader handing over a boot report. The software was developed according to ESA's software engineering standards ECSS-E-ST-40C and ECSS-Q-ST-80C, software criticality category B, reviewed successfully by ESA and third party (ISV&V).
GRBOOT	The GRBOOT boot loader software is based on the GR712RC Boot SW using the same ECSS software engineering standards previously used to guarantee a high reliability for flight. By isolating mission and device specific parts into BSPs and generalizing the implementation, GRBOOT provides similar a reusable feature set for systems based on LEON3/4FT processor devices acting as either payload or OBC. One or more application images can be located in parallel flash or SPI flash. Multiprocessor application booting is supported.

	GRBOOT is available for GR712RC and GR740 based systems together with the appropriate quality proofs, documentation and test suites. A version without references to the ESA requirements documents is also available.
u-boot	Currently u-boot for the GR-VPX-GR740 VPX Development Board is not provided by Cobham Gaisler.

Table 4.1. Boot Loader feature table

Feature	**MKPROM2**	**GR712RC Boot SW**	**GRBOOT**
Supported processors	• Most LEON	• GR712RC	• GR712RC • GR740 Additional system support in progress.
Processor self-tests	No	Yes	Yes
Memory self-tests	No	Yes	Yes
Application storage memory	• PROM • FLASH • MRAM	MRAM	• PROM • FLASH • MRAM • SPI flash
Number of application images	1	2	Unlimited
Standby Mode	No	Yes PUS over SpaceWire	Prepared for user extensions. PUS over SpaceWire in development.
Validation and unit test suite	No	Yes	Yes
Documentation covering SW requirements, design and quality	No	Yes	Yes
Compatible standards	None	`TEC-SWS/10-373`, `ECSS-E-70-41A`	`SAVOIR-GS-002`

4.3. Software Drivers

The operating system environments include software drivers for most I/O units of the GR740.

5. GRMON3 hardware debugger

5.1. Overview

GRMON3 is a debug monitor used to develop and debug GRLIB/LEON systems. The target system, including the processor and peripherals, is accessed on the AHB bus through a debug-link connected to the host computer. GRMON3 has GDB support which makes C/C++ level debugging possible by connecting GDB to the GRMON3's GDB socket. With GRMON3 one can for example:

- Inspect LEON 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 GR-VPX-GR740 VPX Development Board. After that, a basic first inspection of the board is exemplified.

Note the Debug Support Unit and the Debug AHB bus must be enabled if GRMON3 is to be used to connect to the board. The Switch S5 must be set to drive HIGH.

Note that the GR740 requires that GRMON3 version 2.0.71 is used. Earlier versions will not use the correct JTAG version and will not recognize all the clock-gated cores in the clock-gating unit.

Several of the SoC's peripherals may be clock gated off. GRMON3 will enable all clocks if started with the flag **-cginit**. Within GRMON3, the command **grcg enable all** will have the same effect.

GRMON3 is described on the homepage [https://www.gaisler.com/index.php/products/debug-tools] and in detail in [RD-4].

5.2. Debug-link alternatives

5.2.1. Connecting via the FTDI USB/JTAG interface

Please see Section 2.6.1 to configure FTDI interface.

Please see GRMON User's Manual for how to set up the required FTDI driver software. Then connect the PC and the board using a standard USB cable into the FTDI USB connector and issue the following command:

```
grmon -ftdi
```

For grmon version 3.2.2 and later, please use the following command:

```
grmon -ftdi -sdfreq <mhz>
```

5.2.2. Connecting via the Ethernet debug interfaces

The design has two Ethernet debug communication links (EDCL). These links have default addresses in the range 192.168.0.16 to 192.168.0.31. The GR-VPX-GR740 should not be connected to an existing network where these addresses may be already occupied. The selection of address can be controlled via bootstrap signals, the Ethernet debug link can be bootstrapped to an address in the range 192.168.0.16 - 192.168.0.31 (see Section 2.6.2).

If another address is wanted for the Ethernet debug link then one of the other debug links must be used to connect GRMON3 to the board. The EDCL IP address can then be changed using GRMON3's **edcl** command. This new address will persist until next system reset.

Note that the Ethernet debug link traffic can be routed either to the Master I/O AHB bus or to the Debug AHB bus. In order to control the LEON processors the debug link must be routed to the Debug AHB bus, otherwise GRMON3 will not be able to use the debug link to access the Debug Support Unit. For all uses except testing of IOMMU functionality it is recommended that switch S14 is set to HIGH to route debug Ethernet traffic via the Debug AHB bus.

After reset the first Ethernet debug communication link will attempt to configure the Ethernet PHY. In order for this to succeed, the Ethernet 0 port must be connected to a switch or other networking equipment. Once the PHY for Ethernet 0 has been configured then control over the shared MDIO bus will be given to Ethernet 1. This means that in order to use the Ethernet 1 debug communication link, Ethernet 0 must also be connected to a network.

With the Ethernet Debug Communication Link 0 address set to 192.168.0.23 the GRMON3 command to connect to the board is:

```
grmon -eth 192.168.0.23
```

For grmon version 3.2.2 and later, please use the following command:

```
grmon -eth 192.168.0.23 -sdfreq <mhz>
```

5.3. First steps

The previous sections have described which debug-links are available and how to start using them with GRMON3. The subsections below assume that GRMON3, the host computer and the GR-VPX-GR740 board have been set up so that GRMON3 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 GRMON3. 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 the GRMON User's Manual [RD-4].
- Memory location and size configuration is found from the **info sys** output.
- The GR740 has a clock-gating unit which is able to disable/enable clocking and control reset signals. Clocks must be enabled for all cores that LEON software or GRMON3 will be using. The **grcg** command is described in the GRMON User's Manual [RD-4].
- If the Ethernet debug link is present, one can view and change the EDCL IP using the **edcl** command as described in the GRMON User's Manual [RD-4].

5.4. Connecting to the board

The transcript below shows a example session with GRMON3. GRMON3 is started with the `-u` flag in order to redirect UART output to the GRMON3 terminal.

```
cg@hwlin0:~$ grmon -ftdi -u -sdfreq 100

  GRMON debug monitor v3.2.7-37-gfe410a3 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

  This internal version will expire on 12/10/2021

Parsing -ftdi
Parsing -u
Parsing -sdfreq 100

Commands missing help:

JTAG chain (1): GR740
  Device ID:           0x740
  GRLIB build version: 4153
  Detected system:     GR740 rev1
  Detected frequency:  249.0 MHz

  Component                            Vendor
  JTAG Debug Link                      Cobham Gaisler
  GRSPW2 SpaceWire Serial Link         Cobham Gaisler
  EDCL master interface                Cobham Gaisler
  EDCL master interface                Cobham Gaisler
  LEON4 SPARC V8 Processor             Cobham Gaisler
  LEON4 SPARC V8 Processor             Cobham Gaisler
  LEON4 SPARC V8 Processor             Cobham Gaisler
  LEON4 SPARC V8 Processor             Cobham Gaisler
  IO Memory Management Unit            Cobham Gaisler
  AHB-to-AHB Bridge                    Cobham Gaisler
  L2-Cache Controller                  Cobham Gaisler
```

```
  AHB Memory Scrubber                 Cobham Gaisler
  IOMMU secondary master i/f          Cobham Gaisler
  AHB-to-AHB Bridge                   Cobham Gaisler
  LEON4 Debug Support Unit            Cobham Gaisler
  AHB/APB Bridge                      Cobham Gaisler
  AMBA Trace Buffer                   Cobham Gaisler
  AHB/APB Bridge                      Cobham Gaisler
  AHB/APB Bridge                      Cobham Gaisler
  Muxed FT DDR/SDRAM controller       Cobham Gaisler
  Memory controller with EDAC         Cobham Gaisler
  GRPCI2 PCI/AHB bridge               Cobham Gaisler
  GRSPW Router                        Cobham Gaisler
  LEON4 Statistics Unit               Cobham Gaisler
  GRPCI2 Trace buffer                 Cobham Gaisler
  Generic UART                        Cobham Gaisler
  Generic UART                        Cobham Gaisler
  General Purpose I/O port            Cobham Gaisler
  Multi-processor Interrupt Ctrl.     Cobham Gaisler
  Modular Timer Unit                  Cobham Gaisler
  Modular Timer Unit                  Cobham Gaisler
  Modular Timer Unit                  Cobham Gaisler
  Modular Timer Unit                  Cobham Gaisler
  Modular Timer Unit                  Cobham Gaisler
  GRSPW Router DMA interface          Cobham Gaisler
  GRSPW Router DMA interface          Cobham Gaisler
  GRSPW Router DMA interface          Cobham Gaisler
  GRSPW Router DMA interface          Cobham Gaisler
  GR Ethernet MAC                     Cobham Gaisler
  GR Ethernet MAC                     Cobham Gaisler
  CAN Controller with DMA             Cobham Gaisler
  CAN Controller with DMA             Cobham Gaisler
  SPI Controller                      Cobham Gaisler
  Clock gating unit                   Cobham Gaisler
  MIL-STD-1553B Interface             Cobham Gaisler
  AHB Status Register                 Cobham Gaisler
  AHB Status Register                 Cobham Gaisler
  General Purpose I/O port            Cobham Gaisler
  General Purpose Register            Cobham Gaisler
  Temperature sensor                  Cobham Gaisler
  General Purpose Register Bank       Cobham Gaisler
  CCSDS TDP / SpaceWire I/F           Cobham Gaisler
  LEON4 Statistics Unit               Cobham Gaisler
  64-bit PC133 SDRAM Controller       Cobham Gaisler

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

grmon3> info sys
  ahbjtag0  Cobham Gaisler  JTAG Debug Link
            AHB Master 0
  grspw0    Cobham Gaisler  GRSPW2 SpaceWire Serial Link
            AHB Master 1
            APB: e4000000 - e4000100
            Number of ports: 1
  edcl0     Cobham Gaisler  EDCL master interface
            AHB Master 2
  edcl1     Cobham Gaisler  EDCL master interface
            AHB Master 3
  cpu0      Cobham Gaisler  LEON4 SPARC V8 Processor
            AHB Master 0
  cpu1      Cobham Gaisler  LEON4 SPARC V8 Processor
            AHB Master 1
  cpu2      Cobham Gaisler  LEON4 SPARC V8 Processor
            AHB Master 2
  cpu3      Cobham Gaisler  LEON4 SPARC V8 Processor
            AHB Master 3
  iommu0    Cobham Gaisler  IO Memory Management Unit
            AHB Master 4
            AHB: ff840000 - ff848000
            IRQ: 31
            Device index: 0
            Protection modes: APV and IOMMU
            msts: 11, grps: 8, accsz: 128 bits
            APV cache lines: 32, line size: 16 bytes
            cached area: 0x00000000 - 0x80000000
            IOMMU TLB entries: 32, entry size: 16 bytes
            translation mask: 0xff000000
            Core has multi-bus support
            Core has 4 ASMP register blocks
  ahb2ahb0  Cobham Gaisler  AHB-to-AHB Bridge
            AHB Master 5
            AHB: 00000000 - 80000000
            AHB: 80000000 - c0000000
            AHB: c0000000 - e0000000
```

```
          AHB: f0000000 - 00000000
          USR: 00000113
          USR: fff00000
l2cache0  Cobham Gaisler  L2-Cache Controller
          AHB Master 0
          AHB: 00000000 - 80000000
          AHB: f0000000 - f0400000
          AHB: ffe00000 - fff00000
          USR: 00000114
          USR: ffe00000
          IRQ: 28
          L2C: 4-ways, cachesize: 2048 kbytes, mtrr: 16, FT, AHB SPLIT support
memscrub0 Cobham Gaisler  AHB Memory Scrubber
          AHB Master 1
          AHB: ffe01000 - ffe01100
          IRQ: 28
          burst length: 32 bytes
adev12    Cobham Gaisler  IOMMU secondary master i/f
          AHB Master 2
ahb2ahb1  Cobham Gaisler  AHB-to-AHB Bridge
          AHB Master 0
          AHB: 80000000 - c0000000
          AHB: c0000000 - d0000000
          AHB: d0000000 - e0000000
          AHB: ff800000 - ff900000
          USR: 00000118
          USR: ff800000
dsu0      Cobham Gaisler  LEON4 Debug Support Unit
          AHB: e0000000 - e4000000
          AHB trace: 256 lines, 128-bit bus
          CPU0:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                 , FT
                 stack pointer 0x07ffff0
                 icache 4 * 4 kB, 32 B/line, rnd
                 dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
          CPU1:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                 , FT
                 stack pointer 0x07ffff0
                 icache 4 * 4 kB, 32 B/line, rnd
                 dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
          CPU2:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                 , FT
                 stack pointer 0x07ffff0
                 icache 4 * 4 kB, 32 B/line, rnd
                 dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
          CPU3:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                 , FT
                 stack pointer 0x07ffff0
                 icache 4 * 4 kB, 32 B/line, rnd
                 dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
apbmst0   Cobham Gaisler  AHB/APB Bridge
          AHB: e4000000 - e4100000
ahbtrace0 Cobham Gaisler  AMBA Trace Buffer
          AHB: eff00000 - eff20000
          Trace buffer size: 128 lines
apbmst1   Cobham Gaisler  AHB/APB Bridge
          AHB: ff900000 - ffa00000
apbmst2   Cobham Gaisler  AHB/APB Bridge
          AHB: ffa00000 - ffb00000
ddrsdmux0 Cobham Gaisler  Muxed FT DDR/SDRAM controller
          AHB: 00000000 - 80000000
          AHB: ffe00000 - ffe00100
          Backend: sdctrl0
mctrl0    Cobham Gaisler  Memory controller with EDAC
          AHB: c0000000 - d0000000
          AHB: d0000000 - e0000000
          USR: 00000003
          APB: ff903000 - ff903100
          8-bit prom @ 0xc0000000
pci0      Cobham Gaisler  GRPCI2 PCI/AHB bridge
          AHB: 80000000 - c0000000
          AHB: ff800000 - ff840000
          APB: ffa00000 - ffa00100
          IRQ: 11
          Trace buffer size: 256 lines
spwrtr0   Cobham Gaisler  GRSPW Router
          AHB: ff880000 - ff882000
          IRQ: 31
          Instance id: 16
          SpW ports:  8  AMBA ports:  4  FIFO ports:  0
l4stat0   Cobham Gaisler  LEON4 Statistics Unit
          APB: e4000200 - e4000400
          Device is disabled
```

```
pcitrace0 Cobham Gaisler  GRPCI2 Trace buffer
          APB: e4040000 - e4080000
          Trace buffer size: 256 lines
uart0     Cobham Gaisler  Generic UART
          APB: ff900000 - ff900100
          IRQ: 29
          Baudrate 38378, FIFO debug mode available
uart1     Cobham Gaisler  Generic UART
          APB: ff901000 - ff901100
          IRQ: 30
          Baudrate 38378, FIFO debug mode available
gpio0     Cobham Gaisler  General Purpose I/O port
          APB: ff902000 - ff902100
          IRQ: 16
irqmp0    Cobham Gaisler  Multi-processor Interrupt Ctrl.
          APB: ff904000 - ff908000
          EIRQ: 10
gptimer0  Cobham Gaisler  Modular Timer Unit
          APB: ff908000 - ff908100
          IRQ: 1
          16-bit scalar, 5 * 32-bit timers, divisor 249
gptimer1  Cobham Gaisler  Modular Timer Unit
          APB: ff909000 - ff909100
          IRQ: 6
          16-bit scalar, 4 * 32-bit timers, divisor 249
gptimer2  Cobham Gaisler  Modular Timer Unit
          APB: ff90a000 - ff90a100
          IRQ: 7
          16-bit scalar, 4 * 32-bit timers, divisor 249
gptimer3  Cobham Gaisler  Modular Timer Unit
          APB: ff90b000 - ff90b100
          IRQ: 8
          16-bit scalar, 4 * 32-bit timers, divisor 249
gptimer4  Cobham Gaisler  Modular Timer Unit
          APB: ff90c000 - ff90c100
          IRQ: 9
          16-bit scalar, 4 * 32-bit timers, divisor 249
grspw1    Cobham Gaisler  GRSPW Router DMA interface
          APB: ff90d000 - ff90e000
          IRQ: 20
          Number of ports: 1
grspw2    Cobham Gaisler  GRSPW Router DMA interface
          APB: ff90e000 - ff90f000
          IRQ: 21
          Number of ports: 1
grspw3    Cobham Gaisler  GRSPW Router DMA interface
          APB: ff90f000 - ff910000
          IRQ: 22
          Number of ports: 1
grspw4    Cobham Gaisler  GRSPW Router DMA interface
          APB: ff910000 - ff911000
          IRQ: 23
          Number of ports: 1
greth0    Cobham Gaisler  GR Ethernet MAC
          APB: ff940000 - ff940100
          IRQ: 24
          1000 Mbit capable
          edcl ip 192.168.0.23, buffer 2 kbyte
greth1    Cobham Gaisler  GR Ethernet MAC
          APB: ff980000 - ff980100
          IRQ: 25
          Device is disabled
grcan0    Cobham Gaisler  CAN Controller with DMA
          APB: ffa01000 - ffa01400
          IRQ: 16
          Device is disabled
grcan1    Cobham Gaisler  CAN Controller with DMA
          APB: ffa02000 - ffa02400
          IRQ: 16
          Device is disabled
spi0      Cobham Gaisler  SPI Controller
          APB: ffa03000 - ffa03100
          IRQ: 19
          Device is disabled
grcg0     Cobham Gaisler  Clock gating unit
          APB: ffa04000 - ffa04100
          GRMON did NOT enable clocks during initialization
gr1553b0  Cobham Gaisler  MIL-STD-1553B Interface
          APB: ffa05000 - ffa05100
          IRQ: 26
          Device is disabled
ahbstat0  Cobham Gaisler  AHB Status Register
          APB: ffa06000 - ffa06100
```

```
            IRQ: 27
  ahbstat1  Cobham Gaisler  AHB Status Register
            APB: ffa07000 - ffa07100
            IRQ: 27
  gpio1     Cobham Gaisler  General Purpose I/O port
            APB: ffa08000 - ffa08100
            IRQ: 16
  gpreg0    Cobham Gaisler  General Purpose Register
            APB: ffa09000 - ffa09100
  adev49    Cobham Gaisler  Temperature sensor
            APB: ffa0a000 - ffa0a100
  gpreg1    Cobham Gaisler  General Purpose Register Bank
            APB: ffa0b000 - ffa0b100
  spwtdp0   Cobham Gaisler  CCSDS TDP / SpaceWire I/F
            APB: ffa0c000 - ffa0c200
            IRQ: 31
  l4stat1   Cobham Gaisler  LEON4 Statistics Unit
            APB: ffa0d000 - ffa0d200
            cpus: 4, counters: 16, i/f index: 1
  sdctrl0   Cobham Gaisler  64-bit PC133 SDRAM Controller
            AHB: 00000000 - 80000000
            AHB: ffe00000 - ffe00100
            32-bit sdram: 2 * 64 Mbyte @ 0x00000000,
            col 9, cas 2, ref 3.1 us

grmon3> info sys dsu0
  dsu0      Cobham Gaisler  LEON4 Debug Support Unit
            AHB: e0000000 - e4000000
            AHB trace: 256 lines, 128-bit bus
            CPU0:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                   , FT
                   stack pointer 0x07fffff0
                   icache 4 * 4 kB, 32 B/line, rnd
                   dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
            CPU1:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                   , FT
                   stack pointer 0x07fffff0
                   icache 4 * 4 kB, 32 B/line, rnd
                   dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
            CPU2:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                   , FT
                   stack pointer 0x07fffff0
                   icache 4 * 4 kB, 32 B/line, rnd
                   dcache 4 * 4 kB, 32 B/line, rnd, snoop tags
            CPU3:  win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
                   , FT
                   stack pointer 0x07fffff0
                   icache 4 * 4 kB, 32 B/line, rnd
                   dcache 4 * 4 kB, 32 B/line, rnd, snoop tags

grmon3> l2cache invalidate
  invalidate all cache lines

grmon3> l2cache enable

grmon3> load hello
                0 .text             24.0kB /  24.0kB   [===============>] 100%
             6000 .data              2.8kB /   2.8kB   [===============>] 100%
  Total size: 26.84kB (785.37kbit/s)
  Entry point 0x00000000
  Image /home/anandhavel/hello loaded

grmon3> run
Hello world!

  CPU 0:  Program exited normally
  CPU 1:  Power down mode
  CPU 2:  Power down mode
  CPU 3:  Power down mode

grmon3> hist
       TIME     ADDRESS   INSTRUCTIONS/AHB SIGNALS      RESULT/DATA
       9080084  00001940  restore                       [00000000]
       9080085  00001944  retl                          [00001944]
       9080086  00001948  nop                           [00000000]
       9080089  0000106c  call  0x00004c70             [0000106c]
       9080090  00001070  nop                           [00000000]
       9080091  00004c70  mov  0x1, %g1                 [00000001]
       9080092  00004c74  ta  0                         [  TRAP  ]
       9080149  00000800  AHB read   mst=0  size=4      [91d02000 01000000 01000000 01000000]
       9080150  00000810  AHB read   mst=0  size=4      [91d02000 01000000 01000000 01000000]
       9080155  00000800  ta  0                         [  TRAP  ]
```

```
grmon3> inst
      TIME     ADDRESS   INSTRUCTION                 RESULT      SYMBOL
      9080080  0000453c  jmp  %l1                    [0000453c]  -
      9080081  00004540  rett  %l2                   [00001944]  -
      9080084  00001940  restore                     [00000000]  main+0x1c
      9080085  00001944  retl                        [00001944]  main+0x20
      9080086  00001948  nop                         [00000000]  main+0x24
      9080089  0000106c  call  0x00004c70           [0000106c]  -
      9080090  00001070  nop                         [00000000]  -
      9080091  00004c70  mov  0x1, %g1               [00000001]  _exit+0x0
      9080092  00004c74  ta  0                       [  TRAP  ]  _exit+0x4
      9080155  00000800  ta  0                       [  TRAP  ]  -

grmon3> grcg clkinfo
  GRCLKGATE GR740 info:
  Unlock register:       0x00000000
  Clock enable register: 0x0000058d
  Reset register:        0x00000272

  GR740 decode of values:
  +------+----------+---------------------------+----------+---------+-------+
  | Gate | Core(s)  | Description               | Unlocked | Enabled | Reset |
  +------+----------+---------------------------+----------+---------+-------+
  |   0  | GRETH    | Ethernet MAC 0            |    0     |    1    |   0   |
  |   1  | GRETH    | Ethernet MAC 1            |    0     |    0    |   1   |
  |   2  | SPWRTR   | SpaceWire router          |    0     |    1    |   0   |
  |   3  | PCI      | PCI (GRPCI, PCIDMA)       |    0     |    1    |   0   |
  |   4  | GR1553B  | MIL-STD-1553B             |    0     |    0    |   1   |
  |   5  | GRCAN    | CAN core 0 & 1            |    0     |    0    |   1   |
  |   6  | L4STAT   | LEON4 Statistics          |    0     |    0    |   1   |
  |   7  | APBUART  | UART 0                    |    0     |    1    |   0   |
  |   8  | APBUART  | UART 1                    |    0     |    1    |   0   |
  |   9  | SPICTRL  | SPI Controller            |    0     |    0    |   1   |
  |  10  | MCTRL    | PROM/IO                   |    0     |    1    |   0   |
  +------+----------+---------------------------+----------+---------+-------+

grmon3> grcg enable 4

grmon3> grcg clkinfo
  GRCLKGATE GR740 info:
  Unlock register:       0x00000000
  Clock enable register: 0x0000059d
  Reset register:        0x00000262

  GR740 decode of values:
  +------+----------+---------------------------+----------+---------+-------+
  | Gate | Core(s)  | Description               | Unlocked | Enabled | Reset |
  +------+----------+---------------------------+----------+---------+-------+
  |   0  | GRETH    | Ethernet MAC 0            |    0     |    1    |   0   |
  |   1  | GRETH    | Ethernet MAC 1            |    0     |    0    |   1   |
  |   2  | SPWRTR   | SpaceWire router          |    0     |    1    |   0   |
  |   3  | PCI      | PCI (GRPCI, PCIDMA)       |    0     |    1    |   0   |
  |   4  | GR1553B  | MIL-STD-1553B             |    0     |    1    |   0   |
  |   5  | GRCAN    | CAN core 0 & 1            |    0     |    0    |   1   |
  |   6  | L4STAT   | LEON4 Statistics          |    0     |    0    |   1   |
  |   7  | APBUART  | UART 0                    |    0     |    1    |   0   |
  |   8  | APBUART  | UART 1                    |    0     |    1    |   0   |
  |   9  | SPICTRL  | SPI Controller            |    0     |    0    |   1   |
  |  10  | MCTRL    | PROM/IO                   |    0     |    1    |   0   |
  +------+----------+---------------------------+----------+---------+-------+

grmon3>
```

6. Board Package

6.1. Overview

The board package distributed together with this document contains MKPROM initialization functions that are specific to the GR740 or GR-VPX-GR740. The package is named gr-vpx-gr740-<*version*>.

6.2. MKPROM2 bdinit functions

To creare boot-PROMs for GR740 MKPROM2 version 2.0.62 or later must be used.

The board package's MKPROM2 directory contains the following files:

- *bdinit.c* - bdinit functions that will be called by MKPROM2.
- *bdinit_gr740_sdctrl0.ci* - File included by *bdinit.c*. Contains initialization code for the GR740 SDRAM memory controller.
- *bdinit_gr740_mctrl0.ci* - File included by *bdinit.c*. Contains initialization code for the GR740 PROM memory controller.
- *bdinit_gr740_l2cache.ci* - File included by *bdinit.c*. Contains initialization code for Level-2 cache.
- *bdinit_gr740_uart.ci* - File included by *bdinit.c*. Contains initialization code for FTMEM and ALTEN register that set up the pin multiplexing so that uart0 and uart1 can be used along with the PROM.
- *bdinit_gr740_grcg.ci* - File included by *bdinit.c*. Contains clock-gating unit enabling and disabling routines.

The bdinit sequence (defined in bdinit.c file) contains the following steps:

- *bdinit0*: Called before the LEON registers have been initialized but before the memory has been cleared.

  1. Setup memory controller (SDCTRL). Please note that the board package assumes the default board configuration and provided memory modules with the board. If you change the memory modules, you must check if this setup still applies.
  2. Enable 2T signaling on memory controller (SDCTRL). This improves the SDRAM signal timing, required for some memory modules.
  3. Initialize memory controller (SDCTRL), which is required after enabling 2T signaling.
  4. Setup PROM write lead out cycles (MCTRL), which is required to be able to write to the PROM.
- *bdinit1*: Called after the LEON registers have been initialized but before the memory has been cleared.

  1. Invalidate L2 cache contents.
  2. Enable L2 cache after invalidate has finished.
- *bdinit2*: Called after the memory has been initialized but before the application is loaded.

  1. Setup pin multiplexing to enable UART0 and UART1. This assumes that the board is configured in the default board configuration.
  2. (Optional) Enable/disable cores on clock-gating unit. Please note that UART0 and MCTRL cores are enabled by default and should not be enabled/disabled here, since that will cause a reset on the cores that are being used by the boot code.

The first step in creating a boot-PROM image for GR-VPX-GR740 is to compile the bdinit.c file. This is done with the command **sparc-elf-gcc -O2 -c -o bdinit.o gr-vpx-gr740-bp/MKPROM2/bdinit.c**. Note that this requires the Bare-C Compiler (BCC) available from https://www.gaisler.com. Also note that the **-O2** is important, since MKPROM requires bdinit0 and bdinit1 functions to be leaf and not allocate any stack space, i.e. no local variables, see [RD-13] for more information.

The next step is to run **mkprom2** specifying flags that are specific for the GR-VPX-GR740. The full MKPROM2 command for creating an image is:

```
/opt/mkprom2/mkprom2 -v  \
     -stack <stack pointer> -ramsize <size of RAM in KiB>-romsize 128 -sparcleon0 \
     -memcfg1 0x080100ff -memcfg3 0x08000000   \
     -dump -v -rstaddr 0xc0000000 -uart 0xFF900000 -freq <frequency> -baud 38400 \
     -bdinit <image> -o <output name>
```

Making a boot-PROM image for a hello world application, named *hello* for a 250 MHz system frequency with 128 MiB of RAM gives requires the MKPROM2 command line below. Note that the application should be linked for RAM starting at address 0. For BCC, this is accomplished with the `-Wl,-msparcleon0` flag.

```
/opt/mkprom2/mkprom2 -v  \
     -stack 0x07ffff00 -ramsize 131072 -romsize 128 -sparcleon0 \
     -memcfg1 0x080100ff -memcfg3 0x08000000   \
     -dump -v -rstaddr 0xc0000000 -uart 0xFF900000 -freq 250 -baud 38400 \
     -bdinit hello -o hello.prom
```

The output of calling mkprom2 2.0.62 with the options above is:

```
mkprom2 -v -stack 0x07ffff00 -ramsize 131072 -romsize 128 -sparcleon0 -memcfg1 0x080100ff -memcfg3 0x08000000
-dump -v -rstaddr 0xc0000000 -uart 0xFF900000 -freq 250 -baud 38400 -bdinit ./hello -o ./hello.prom

LEON2/3/ERC32 MKPROM prom builder for BCC, ECOS, RTEMS and ThreadX v2.0.62
Copyright Cobham Gaisler AB 2004-2007, all rights reserved.

phead0: type: 1, off: 65536, vaddr: 0, paddr: 0, fsize: 27488, msize: 29544
phead1: type: 1, off: 95080, vaddr: 7368, paddr: 7368, fsize: 0, msize: 4
section: .text at 0x0, size 24576 bytes
Uncoded stream length: 24576 bytes
Coded stream length: 13466 bytes
Compression Ratio: 1.825
section: .data at 0x6000, size 2912 bytes
Uncoded stream length: 2912 bytes
Coded stream length: 827 bytes
Compression Ratio: 3.521

creating LEON3 boot prom: ./hello.prom
Searching for compiler to use (sparc-elf, sparc-rtems or sparc-linux):
sparc-elf-gcc (BCC 4.4.2 release 1.0.45) 4.4.2
Copyright (C) 2009 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.


sparc-elf-gcc -O2 -g -N -T/opt/mkprom2/linkprom -Ttext=0xc0000000  /opt/mkprom2/promcore.o /opt/mkprom2/prominit.o
/opt/mkprom2/prominit_leon3.o /opt/mkprom2/promcrt0.o /opt/mkprom2/promload.o /opt/mkprom2/promdecomp.o
-nostdlib /opt/mkprom2/prombdinit.o  dump.s bdinit.o  -o ./hello.prom
multidir:
```

The *hello.prom* image can now be programmed to the board's MRAM. Set the GPIO[15] Switch S22 to LOW before programming and boot the system from MRAM. A transcript of this operation using GRMON3 is included below

```
grmon3> set mctrl0::mcfg1::pwen 1

grmon3> set mctrl0::mcfg1::prombanksz 4

grmon3> set gpreg1::ftmfunc 0x400

grmon3> wash 0xc0000000 0xc0020000
  c0000000                          128.0kB / 128.0kB   [===============>] 100%
  Finished washing!

grmon3> load hello.prom
          C0000000 .text             20.7kB /  20.7kB   [===============>] 100%
  Total size: 20.72kB (789.43kbit/s)
  Entry point 0xc0000000
  Image /home/anandhavel/validation-gr-vpx-gr740-oct2-2020/helloworld-play/hello.prom loaded

grmon3> verify hello.prom
          C0000000 .text             20.7kB /  20.7kB   [===============>] 100%
  Total size: 20.72kB (372.21kbit/s)
  Entry point 0xc0000000
  Image of /home/anandhavel/validation-gr-vpx-gr740-oct2-2020/helloworld-play/hello.prom verified without errors

grmon3>
```

When the board is power-cycled, the following will appear at UART 0 (38400, 8N1):

```
 MKPROM2 boot loader v2.0.62
  Copyright Cobham Gaisler AB - all rights reserved

  system clock    : 250.0 MHz
  baud rate       : 38390 baud
  prom            : 128 K, (2/2) ws (r/w)
  sram            : 131072 K, 1 bank(s), 0/0 ws (r/w)

  decompressing .text to 0x00000000
  decompressing .data to 0x00006000

  starting ./hello

Hello world!
```

Connecting to the GR740 after the boot will show the Level-2 cache as enabled, since it is enabled by the bdinit functions used for the PROM image.

Additional notes on creating boot-PROMs:

- The value written on bdinit_gr740_sdctrl0.ci are the parameters written into the SDRAM controller registers. These parameters depend on the type of SDRAM SODIMM used. If the SDRAM SODIMM is replaced then the simplest way to obtain new parameters is to connect to the design with GRMON3 and issue **info reg** and copy the values that GRMON3 has initialized the SDRAM memory controller with.
- The *-stack* and *-ramsize* parameters should be set according to the amount of available memory. In the example above the stack can be set at top of the 256 MiB area and still work when switching memory interface to the 128 MiB SDRAM as the memory area will wrap.
- The *-memcfg* parameters specify values written to the PROM/IO memory controller configuration registers. The PROM width must be 8 bits. The PROM width setting must match with the PROM width selection made via the bootstrap signal GPIO[10]. GPIO[10] must be set LOW since this board only support 8-bit interface.
- In case the PROM has been programmed with a destructive application it is possible to prevent the processor's from starting up by holding the RESET and BREAK buttons on the front-panel, and then releasing RESET with still pressing BREAK.

7. Frequently Asked Questions / Common Mistakes / Know Issues

7.1. Clock gating

Several of the design's peripherals may be clock gated off. GRMON3 will enable all clocks if started with the flag `-cginit`. Within GRMON3, the command **grcg enable all** will have the same effect.

7.2. GRMON3 issues

When connected to the board, the message "stack pointer not set" will be shown by the command **info sys** in case GRMON3 doesn't find any memory.

When connecting to the board with FTDI in a Linux based system, the message "unable to claim usb device. Make sure the default FTDI driver is not in use" is printed. Make sure that you have set the FTDI udev rules. Please refer to [RD-4]. If the problem persist, try stopping the kernel driver *ftdi_sio*.

7.3. Level-2 cache initialization

The Level-2 cache (L2C) should always be enabled in order to obtain adequate performance. When connection to the board is established using GRMON3/GRMON2, l2cache is automatically enabled by GRMON debugger. Otherwise, the L2C has to be enabled by software, as shown in Chapter 6. When enabling L2C it is important that all entries in the cache are invalidated. Otherwise power-on values in the cache's internal memories may be interpreted as valid cache data.

The Level-2 cache contents can be invalidated, and the cache then enabled with the following GRMON3 sequence:

```
grmon3> l2cache invalidate
  invalidate all cache lines

grmon3> l2cache enable
```

7.4. Main memory interface EDAC

EDAC on the main memory interface can be enabled via GRMON3. First all memory that will be used needs to be initialized. This can be done with help of the hardware memory scrubber. Next, the EDAC is enabled by EDAC enable bit must be set in the memory controller's FT Configuration Register. The full initialization sequence, with also Level-2 cache enabled becomes:

```
grmon3> scrub clear 0 0x07ffffff
  0x00000000                        128.0MB / 128.0MB   [===============>] 100%

grmon3> wmem 0xffe00020 0x1

grmon3> l2cache invalidate
  invalidate all cache lines

grmon3> l2cache enable
```

Note that the **scrub** above initialises 128 MiB of memory. If the system has 256 MiB of memory then the command is **scrub clear 0 0x0fffffff**.

The scrubber monitors the Memory AHB bus for errors reported by the memory controller. The command **scrub** shows the status:

```
grmon3> scrub
  AHB status register: Not triggered
  Scrubber status:     Done init 00000000-0fffffff

  Error count/limits:  UE:0/0, CE:0/0, SEC:0/off, SBC:0/off
  Scrubber config:     Loop:0, Extstart:0, Extclear:0, Delay: 0
```

In case errors have been detected, the **scrub** will show the affected address:

```
grmon3> scrub
  AHB status register: ERROR at addr 0f4027c0 (mst 0 hsize 4 hwrite 0)
  Scrubber status:     Idle

  Error count/limits:  UE:1/0, CE:0/0, SEC:0/off, SBC:0/off
  Scrubber config:     Loop:0, Extstart:0, Extclear:0, Delay: 0
```

7.5. MRAM with EDAC

To boot with MRAM EDAC enabled you need add the "-bch8 -romsize 128" to your MKPROM2 flags. MKPROM2 will then create a PROM binary with the EDAC checkbits (-bch8) located at the end of the MRAM memory area, given by the board's MRAM size of 128 KiB (-romsize 128, in KiB). The outfile file with .bch8 extension can be loaded to the MRAM as usual. To enable MRAM/PROM EDAC, set GPIO[14] to pull-up (PU).

7.6. Ethernet

If EDCL is not working, please make sure that a) GPIO[8] is set to route traffic on the Debug AHB bus; b) a proper IP address has been selected either through GPIO[0-3] or via GRMON3. See Section 2.6.2.

7.7. UART

If the UART is either displaying rubbish characters or not displaying anything, make sure that a) the pin multiplexing is properly configured and b) the FTMEN and ALTEN registers allow UART operation. See Section 2.5.1.

7.8. Can't boot

Check that your boot image is properly loaded into the flash memory (starting address 0xc0000000). You can use the **verify** command of GRMON3 that will do it for you. Make sure that both memory controllers (mctrl0 and sdctrl0) are properly initialized. To check if that is happening, you can connect with GRMON3 with no initialization flag (-ni) once your system has been powered up. For instance, using JTAG/FTDI debug link, the command is **grmon -ftdi -ni**. Please note, that if you use GRMON3 without the **-ni** option, GRMON3 initializes both memory controllers and thus, the state left by your boot code cannot be analysed. Once in grmon, check the value of the memory controller configuration registers, as shown below:

```
grmon3> info reg -v mctrl0
  Memory controller with EDAC
      0xff903000  Memory config register 1               0x080100ff
      30     pbrdy             0x0         PROM area bus ready enable
      29     abrdy             0x0         Asynchronous bus ready enable
      28:27  iobusw            0x1         I/O bus width
      26     ibrdy             0x0         I/O bus ready enable
      25     bexcn             0x0         Bus error enable
      23:20  iows              0x0         I/O wait states
      19     ioen              0x0         I/O enable
      17:14  prombanksz        0x4         PROM bank size
      11     pwen              0x0         PROM write enable
       9:8   promwidth         0x0         PROM width
       7:4   promwws           0xf         PROM write wait states
       3:0   promrws           0xf         PROM read wait states

      0xff903004  Memory config register 2               0x00001c20
      31     sdramrf           0x0         SDRAM refresh enable
      30     sdramtrp          0x0         SDRAM TRP parameter
      29:27  sdramtrfc         0x0         SDRAM TRFC parameter
      26     sdramtcas         0x0         SDRAM TCAS parameter
      25:23  sdrambanksz       0x0         SDRAM bank size
      22:21  sdramcolsz        0x0         SDRAM column size
      20:19  sdramcmd          0x0         SDRAM command
      18     d64               0x0         SDRAM 64-bit data bus
      17     sdpb              0x0         SDRAM page burst
      14     se                0x0         SDRAM enable
      13     si                0x0         SRAM disable
      12:9   rambanksz         0xe         RAM bank size
       7     rbrdy             0x0         RAM bus ready enable
       6     rmw               0x0         Read-modify-write enable
       5:4   ramwidth          0x2         RAM width
       3:2   ramwws            0x0         RAM write wait states
       1:0   ramrws            0x0         RAM read wait states
```

```
      0xff903008  Memory config register 3               0x08000000
      28     rse               0x0         Reed-Solomon EDAC enable
      27     me                0x1         Memory EDAC available
      26:12  sdramreload       0x0         SDRAM refresh counter reload value
      11     wb                0x0         EDAC diagnostic write bypass enable
      10     rb                0x0         EDAC diagnostic read bypass enable
       9     re                0x0         RAM EDAC enable
       8     pe                0x0         PROM EDAC enable
       7:0   tcb               0x0         Test checkbits

      0xff903010  Memory config register 5               0x00001c00
      29:23  iohws             0x0         IO lead out
      13:7   romhws            0x38        ROM lead out

      0xff903014  Memory config register 6               0x00000000
      13:7   ramhws            0x0         RAM lead out

      0xff903018  Memory config register 7               0x00000000
      31:16  brdyncnt          0x0         Bus ready count
      15:0   brdynrld          0x0         Bus ready reload value

grmon3> info reg -v sdctrl0
  64-bit PC133 SDRAM Controller
      0xffe00000  SDRAM config register                  0xfe20079c
      31     refresh           0x1         SDRAM refresh enable
      30     trp               0x1         SDRAM TRP parameter
      29:27  trfc              0x7         SDRAM TRFC parameter
      26     tcas              0x1         SDRAM TCAS parameter
      25:23  banksz            0x4         SDRAM bank size
      22:21  colsz             0x1         SDRAM column size
      20:18  cmd               0x0         SDRAM command
      17     pb                0x0         Pageburst
      16     ms                0x0         Mobile SDR support
      15     d64               0x0         SDRAM 64-bit data bus
      14:0   reload            0x79c       Refresh reload value

      0xffe00004  SDRAM configuration register 2         0x40f08000
      30     ce                0x1         Clock enable
      15     en2t              0x1         Enable 2T signaling
      14     dcs               0x0         Double chip select mode
      13     bpark             0x0         Bus parking enable
```

These registers have to be set up by the boot code in order to get a working memory.

7.9. FTDI/JTAG

If the JTAG connection is not working. Check the following things:

- Make sure that the DSU is enabled (Switch S5)
- Check that all JP8 jumpers are connected
- Some USB cables do not fit properly on the board connector due to the plastic protector of the cable that prevents the connector to go deeper into the board connector. Try removing a little bit of the plastic protection and see if that works.

7.10. SDRAM not working

If the SDRAM is not working. Check the following things:

- Make sure the SDRAM device is mounted firmly on the SODIMM.

8. Support

For support contact the Cobham Gaisler support team at 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.

The support service is only for paying customers with a support contract.

Cobham Gaisler AB
Kungsgatan 12
411 19 Gothenburg
Sweden
www.cobhamaes.com/gaisler
sales@gaisler.com
T: +46 31 7758650
F: +46 31 421407