GR-CPCI-GR740 Quick Start Guide
# Table of Contents

1. Introduction ...................................................................................................................... 3  
1.1. Overview ...................................................................................................................... 3  
1.2. References .................................................................................................................... 3  
2. Board Configuration ........................................................................................................ 4  
2.1. Overview ...................................................................................................................... 4  
2.2. Default configuration .................................................................................................... 4  
2.3. Clocking ......................................................................................................................... 4  
2.4. Bootstrap Signals ......................................................................................................... 4  
2.5. Pin multiplexing ............................................................................................................ 5  
2.5.1. PROM/I0, UART, CAN, MIL-STD-1553B, Spacewire, general purpose I/O .......... 6  
2.5.2. SDRAM, PCI, Ethernet port 1 .................................................................................. 8  
2.6. Interfaces ..................................................................................................................... 9  
2.6.1. JTAG FTDI ............................................................................................................... 9  
2.6.2. Ethernet .................................................................................................................. 10  
2.6.3. PROM Parallel flash ............................................................................................... 10  
2.6.4. UARTs .................................................................................................................... 10  
2.6.5. CAN ....................................................................................................................... 11  
2.6.6. MIL-STD-1553B .................................................................................................... 11  
2.6.7. Spacewire ............................................................................................................... 11  
2.6.8. PCI ......................................................................................................................... 11  
3. Comments on System-on-Chip Design ........................................................................ 12  
3.1. Overview ...................................................................................................................... 12  
3.2. Building Operating Systems for GR740 ..................................................................... 12  
3.3. Building Applications for GR740 .............................................................................. 12  
3.4. Running binaries linked to address 0x40000000 ......................................................... 12  
3.5. Considerations when enabling the Level-2 cache .................................................... 12  
4. GRMON3 hardware debugger .......................................................................................... 13  
4.1. Overview ...................................................................................................................... 13  
4.2. Debug-link alternatives ............................................................................................... 13  
4.2.1. Connecting via the FTDI USB/JTAG interface ....................................................... 13  
4.2.2. Connecting via the Ethernet debug interfaces ....................................................... 13  
4.2.3. Connecting via SpaceWire RMAP interface .......................................................... 14  
4.3. First steps .................................................................................................................... 14  
4.4. Connecting to the board .............................................................................................. 14  
5. Board Package .................................................................................................................. 20  
5.1. Overview ...................................................................................................................... 20  
5.2. MKPROM2 8kinit functions ......................................................................................... 20  
6. Frequently Asked Questions / Common Mistakes / Know Issues ................................ 23  
6.1. Clock gating ............................................................................................................... 23  
6.2. GRMON3 issues ......................................................................................................... 23  
6.3. Level-2 cache initialization .......................................................................................... 23  
6.4. Main memory problems ............................................................................................. 23  
6.5. Main memory interface EDAC .................................................................................... 23  
6.6. Flash programming ...................................................................................................... 24  
6.7. PROM EDAC .............................................................................................................. 24  
6.8. Ethernet ...................................................................................................................... 24  
6.9. UART ......................................................................................................................... 24  
6.10. Pin multiplexing (PROMIO/Peripherals) ................................................................... 24  
6.11. Can’t boot ................................................................................................................... 25  
6.12. FTDI/JTAG ............................................................................................................... 26  
6.13. SDRAM not working .................................................................................................. 26  
7. Support ............................................................................................................................. 27  
A. Default configuration ....................................................................................................... 28
1. Introduction

1.1. Overview

This document is a quick start guide for the GR-CPCI-GR740 CompactPCI 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-CPCI-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-CPCI-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-CPCI-GR740 data package and this document (including possibly newer revisions) are available from the GR-CPCI-GR740 product page at https://www.gaisler.com.

1.2. References

| 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-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-12 | VxWorks 7 SPARC architectural port and BSP [https://www.gaisler.com/index.php/products/operating-systems/vxworks-7] |

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-CPCI-GR740 Development Board User's Manual and the GR740 Data sheet and User's Manual. Before start using the GR-CPCI-GR740, clock sources have to be installed, boot-strap signals need to be set correctly and the desired interfaces have to be enabled. The GR740 shares some of the PROM/IO and SDRAM pins due to a limited number of pins. For that reason, the pin multiplexing has to be set up depending on the desired interfaces and memory configuration.

2.2. Default configuration

This guide provides a default configuration that uses flash PROM as boot memory and 96-bit SDRAM. The debug links are a) JTAG over FTDI and b) Ethernet. As general I/O, UART0 and UART1 are used. The complete default configuration can be found in Appendix A. If this is your first time using the GR-CPCI-GR740, please use this configuration as a starting point.

Default configuration

To achieve the default configuration please follow the instructions on each box note like this one.

2.3. Clocking

The default configuration of the board uses 50 MHz oscillators in the X2 (MEMCLK) and X4 (SYSCLK) sockets. If the SpaceWire interfaces are used then a 50 MHz oscillator must be present in the X1 (SPWCLK) socket. Also, an external clock source is required for the PCI interface, which can operate at 33 or 66 MHz, either from the PCI backplane or from an oscillator in socket X3 (PCICLK). Please refer to GR-CPCI-GR740 Development Board User's Manual for instructions on how to configure the board to function in a PCI system.

The internal frequencies for system, memory and spacewire depend on the oscillators installed on the board and on the PLL bypass[0:2] settings set on the front panel (FP-S3 switches 3, 4 and 5), as shown in Table 2.1 and Table 2.2 (Default configuration highlighted in grey). The default configuration with all clocks installed uses all PLLs enabled, as shown in the table. When changing settings, JP24 also needs to be set as shown in the table (see section FPGA for PCI Arbiter & Versaclock Controller in GR-CPCI-GR740 Development Board User's Manual for details). The clksel=0 options are intended to allow generating both system and memory clock with a single PLL but it multiplies the clock by 1 in default PLL configuration and is therefore not usable. It is recommended to keep clksel at the default (1) setting.

Table 2.1. System and Memory clock configuration

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>open</td>
<td>open</td>
<td>5xSYS=250</td>
<td>2xMEM=100</td>
<td>Sys, Mem</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>open</td>
<td>open</td>
<td>1xSYS=50</td>
<td>2xMEM=100</td>
<td>Mem</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>closed</td>
<td>closed</td>
<td>5xSYS=250</td>
<td>1xMEM=50</td>
<td>Sys</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>closed</td>
<td>closed</td>
<td>1xSYS=50</td>
<td>1xMEM=50</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>don't care</td>
<td>0</td>
<td>closed</td>
<td>closed</td>
<td>1xSYS=50</td>
<td>1xSYS=50</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2.2. Spacewire clock configuration

<table>
<thead>
<tr>
<th>Bypass[2]</th>
<th>SpW clock</th>
<th>PLLs enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8xSPW=400</td>
<td>SpW</td>
</tr>
<tr>
<td>1</td>
<td>1xSPW=50</td>
<td>-</td>
</tr>
</tbody>
</table>
Default configuration

The default configuration of the board uses 50 MHz oscillators installed on X1, X2 and X4 for SPWCLK, MEMCLK and SYSCLK respectively; and a 33 MHz oscillator in X3 for PCICLK.

Figure 2.1. GR-CPCI-GR740 default configuration as delivered

PLLs are enabled, i.e. bypass signals are disabled, which means that DIP switches FP-S3-3, FP-S3-4, FP-S3-5 are in CLOSED position to enable PLLs (see Figure 2.2). The result is System clock at 250 MHz, SDRAM at 100 MHz, Spacewire at 400 MHz and PCI at 33 MHz.

By default, grmon (version 3.2.2 or later) sets up the srdam timing parameters such that the memory works both at 50 MHz and 100 MHz SDRAM clock. However, to have the correct performance, user need to set startup option - sdfreq <mhz>. Please refer section Section 4.2 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-CPCI-GR740 via the DIP switches FP-S1, FP-S2, FP-S3 and S1. The bootstrap signals that are mapped to the general purpose I/O lines (DIP switch FP-S1 and FP-S2) 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 bootstrap signals controlled via DIP switch FP-S3 and S1 have a larger impact on the full system behaviour:

- The BREAK signal (S1) controls if the first processor will start execution after system reset. If it is on the right switch position, then it is not activated, meaning that the execution will start after system reset. On the left switch position, break is activated and the execution will not start.
- The DSU enable signal (FP-S3-1) controls if the design’s Debug Support Unit is enabled and also if the debug communication links are active. DIP switch S3-1 must be set to OPEN, i.e. DSU enabled, to connect to the board using the GRMON3 debug monitor.
- The MEM_CLKSEL signal (FP-S3-2) selects the clocks used for the SDRAM memory interface and the on-chip buses. If the DIP switch S3-2 is OPEN 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 (FP-S3-3, FP-S3-4, FP-S3-5) are explained in Section 2.3.
- The PLL_IGNLOCK signal (FP-S3-6) is explained in Section 2.3.
- The ETHCLK signal (FP-S3-7) is explained in Section 2.6.2.
- The WDEN signal (FP-S3-8) controls if a watchdog timeout in the GR740 will trigger a board reset. This switch should be set to OPEN in order to disconnect the watchdog from the reset circuit.

Some bootstrap signals are automatically set up by installing the configuration plug on J21, J22 or J23. See Section 2.5:

- The MEM_IFWIDTH signal selects the width of the primary memory interface.
- The PCI_MODE signal selects if the top-half of the SDRAM interface should be used for the PCI controller (HIGH) or Ethernet port 1 (LOW) when the SDRAM is not in full width.

The GPIO bootstrap signals are as follows:

- The GPIO[0:5] signals (FP-S1-1 to FP-S1-6) control EDCL IP address. All pulled up disable EDCL on both ETH0 and ETH1.
- The GPIO[6:7] signals (FP-S1-7 and FP-S1-8) control Spacewire router distributed interrupt configuration.
• The GPIO[8:9] signals (FP-S2-1 and FP-S2-2) control EDCL routing. Pulled up route EDCL traffic to the debug bus.

• The GPIO[10] signal (FP-S2-3) controls PROM width. Pulled down selects 8-bit, otherwise 16-bit.


• The GPIO[12:13] signals (FP-S2-5 and FP-S2-6) set Spacewire router's instance ID.

• The GPIO[14] signal (FP-S2-7) controls PROM EDAC. Pulled down disables PROM EDAC, otherwise enabled.

• The GPIO[15] signal (FP-S2-8) 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-CPCI-GR740 board and the configuration on the GR740 device.

First, to configure the GR-CPCI-GR740 board pin multiplexing, we need to set JP11 jumpers in the wanted configuration. Each JP11 jumper chooses each individual pin configuration as shown in Table 2.3 (Default configuration highlighted in grey). Set all JP11 jumpers on position AB if full PROM/IO mode (a) is used or position CD for peripheral mode (i.e. c and d). See section PROMIO / Interface configuration of the GR-CPCI-GR740 Development Board User's Manual.

Second, to configure the GR740 set GPIO[15] to LOW for full PROM/IO mode (a) or HIGH for peripheral mode (c). 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)). Table 2.3 shows the configuration for each bit. GPIO[15] LOW puts all bits to 1 and HIGH to 0. See section PROMIO / Interface configuration of the GR-CPCI-GR740 Development Board User's Manual and Register Bank For I/O and PLL configuration registers of the GR740 Data sheet and User's Manual.

It is also possible to create custom muxing configurations in between these three configurations, such as using the 8-bit PROM/IO along with the UART0 and UART1, which is the chosen default configuration. In such a configuration, the two most significant address pins and the two most significant data pins of the PROM/IO are used for the UARTs. To set up such custom muxing configurations, three steps are required: First, set the corresponding JP11 jumpers in the board (see Table 2.3). In our case, means that all JP11 jumpers are in position AB, or PROMIO mode, except jumpers 1, 2, 13 and 14 in position CD (UARTs pins). Note that the signals PROMIO_DATA[0:7] (JP11 13 and 14) are used only in the 16-bit PROM mode. The 8-bit PROM interface uses the signals PROMIO_DATA[8:15]. This means that the PROM/IO can only be used in 8-bit mode, up to 16 MiB instead of 64 MiB, when using the UART0 and UART1 along with the PROM, because the MSB address pins and the MSB data pins of the PROM/IO are used for the UARTs. Second, GPIO[15] needs to be LOW to set FTMEN and start in PROM mode. Third, software needs to change the configuration of the individual bits in FTMEN from a) PROM to b) peripheral mode. This means changing bits 8, 9, 20 and 21 to 0, that is writing 0xFF000 to FTMEN(0xFFA0B000). This can be done on the initialization boot code, as shown in Chapter 5 See section PROMIO / Interface configuration of the GR-CPCI-GR740 Development Board User's Manual and Register Bank For I/O and PLL configuration registers of the GR740 Data sheet and User's Manual.

Table 2.3. Jumper JP11 and FTMEN Register configurations

<table>
<thead>
<tr>
<th>JP11 jumper</th>
<th>FTMEN bit</th>
<th>JP11 jumper at AB / FTMEN bit at 1</th>
<th>JP11 jumper at CD / FTMEN bit at 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>21</td>
<td>PROMIO_ADDR27</td>
<td>UART_TXD0</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>PROMIO_ADDR26</td>
<td>UART_TXD1</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>PROMIO_ADDR25</td>
<td>1553TXA</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>PROMIO_ADDR24</td>
<td>1553TXNA</td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>PROMIO_ADDR23</td>
<td>1553RXENA</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>PROMIO_ADDR22</td>
<td>1553TXB</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>PROMIO_ADDR21</td>
<td>1553TXNB</td>
</tr>
<tr>
<td>8</td>
<td>14</td>
<td>PROMIO_ADDR20</td>
<td>1553RXENB</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>PROMIO_ADDR19</td>
<td>SPWDCL_DBG_TXD</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>PROMIO_ADDR18</td>
<td>SPWDCL_DBG_TXS</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>PROMIO_ADDR17</td>
<td>UART_RTSN0</td>
</tr>
<tr>
<td>12</td>
<td>10</td>
<td>PROMIO_ADDR16</td>
<td>UART_RTSN1</td>
</tr>
<tr>
<td>13</td>
<td>9</td>
<td>PROMIO_DATA7&lt;1</td>
<td>UART_RXD0</td>
</tr>
<tr>
<td>14</td>
<td>8</td>
<td>PROMIO_DATA6&lt;1</td>
<td>UART_RXD1</td>
</tr>
<tr>
<td>15</td>
<td>7</td>
<td>PROMIO_DATA5&lt;1</td>
<td>CAN_RX0</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>PROMIO_DATA4&lt;1</td>
<td>CAN_RX1</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>PROMIO_DATA3&lt;1</td>
<td>1553RXA</td>
</tr>
</tbody>
</table>
Default configuration

The default configuration of the board consist of a custom multiplexing configuration using PROM/IO and UARTs 0 and 1. To set the default configuration, set all JP11 jumpers on position AB except JP11-1, JP11-2, JP11-13 and JP11-14 in position CD (see Figure 2.4). FP-S2-8 (GPIO[15]) is Closed (see Figure 2.2) to start in PROM/IO mode and the software has to set up the register FTMEN(0xFFA0B000) to 0x000FFCFF to activate the UARTs pins. For instance, this can be done on the initialization boot code, as shown in Chapter 5 or in GRMON3 as shown in Chapter 4. See section Pin multiplexing control of the GR740 Data sheet and User's Manual.

![Figure 2.4. GR-CPCI-GR740 default configuration as delivered](image)

2.5.2. SDRAM, PCI, Ethernet port 1

The top half of the SDRAM interface shares pins with PCI and Ethernet port 1. There are three configurations, a) 96-bit memory interface; b) 48-bit memory interface and PCI; and c) 48-bit memory interface and Ethernet port 1. These configurations are selected by installing the configuration plug on the bottom side of the GR-CPCI-GR740 in J21 (b), J22 (c) or J23 (a) depending on the desired configuration and cannot be reprogrammed in software. Please note that for configuration a) 96-bit interface, 2 SODIMM modules are required. See section SDRAM Memory Interface configuration of the GR-CPCI-GR740 Development Board User's Manual.

<table>
<thead>
<tr>
<th>JP11 jumper</th>
<th>FTMEN bit</th>
<th>JP11 jumper at AB / FTMEN bit at 1</th>
<th>JP11 jumper at CD / FTMEN bit at 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>4</td>
<td>PROMIO_DATA2(^1)</td>
<td>1553RXNA</td>
</tr>
<tr>
<td>19</td>
<td>3</td>
<td>PROMIO_DATA1(^1)</td>
<td>1553RXB</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>PROMIO_DATA0(^1)</td>
<td>1553RXNB</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>PROMIO_CEN1</td>
<td>CAN_TX0</td>
</tr>
<tr>
<td>22</td>
<td>0</td>
<td>PROMIO_IOSN</td>
<td>CAN_TX1</td>
</tr>
</tbody>
</table>

\(^1\)PROMIO_DATA bits [0:7] are only used in 16-bit PROM mode. PROMIO_DATA[8:15] are used for 8-bit and 16-bit PROM mode.
Default configuration

The default configuration of the board uses the full 96-bit SDRAM width, in which the configuration plug is installed on the bottom side of the GR-CPCI-GR740 in J23.

Figure 2.5. GR-CPCI-GR740 default configuration as delivered

2.6. Interfaces

This section describes how to set up the main different interfaces of the GR-CPCI-GR740. If you are not interested in 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 4). Jumpers JP4 have to be installed to use JTAG to USB. Please see section *FTDI Serial to USB Interface* of the GR-CPCI-GR740 Development Board User's Manual.

Default configuration

The default configuration uses JTAG to USB conversion to connect to the board using GRMON3 by having all JP4 jumpers installed.

Figure 2.6. GR-CPCI-GR740 default configuration as delivered
2.6.2. Ethernet

There are two ethernet ports available in the GR-CPCI-GR740, port 0 and port 1. To use port 1, make sure that the pin multiplexing is configured to use ETH1, as explained in Section 2.5. FP-S3-7 chooses between gigabit mode and 100 Mbps. To use the Ethernet interfaces for the EDCL Debug link (see Chapter 4), it is necessary to appropriately set the GPIO signals at power-up/reset. GPIO[0..5] (FP-S1-1 to FP-S1-6) set the least significant address bits of the IP address of each port and also disable EDCL if all of them are Open. GPIO[8] and GPIO[9] (FP-S2-1 and FP-S2-2) set the routing of the ethernet traffic on the Debug AHB bus, required to use EDCL, for port 0 and 1 respectively. Please see section Ethernet Interface of the GR-CPCI-GR740 Development Board User’s Manual.

Default configuration

The default configuration uses only ethernet port 0, prepared to use EDCL by setting FP-S2-1 and FP-S2-2 Open (see Figure 2.2). But disabled by default to avoid conflicts when connecting the board by setting FP-S1-1 to FP-S1-6 to Open (see Figure 2.2). Please note that user should either change FP-S1-1 to FP-S1-6 to get a valid EDCL IP address at startup or set up any IP by using GRMON3 in order to use EDCL (see Chapter 4). Also FP-S3-7 should be adjusted according to the desired link speed.

2.6.3. PROM Parallel flash

Make sure that the pin multiplexing is configured to use PROM/IO, as explained in Section 2.5. The JS28F640J3 flash provides 8 MiB of non-volatile memory. This device can be configured for 8-bit or 16-bit mode. Please note that for 16-bit mode, the pin multiplexing has to be on full PROM/IO mode (a). GPIO[15] should also reflect this configuration. There is a hardware write-protection of the flash by installing jumper JP6a. The flash can be disconnected by removing jumper JP6b. Please see section PROMIO / Parallel flash of the GR-CPCI-GR740 Development Board User’s Manual.

Default configuration

The default configuration uses 8-bit PROM with EDAC disabled. Jumpers JP6 are all connected except JP6a to remove write-protection (see Figure 2.7). DIP switches FP-S2-3 and FP-S2-7 Closed to enable 8-bit PROM with EDAC disabled (see Figure 2.2).

Figure 2.7. GR-CPCI-GR740 default configuration as delivered

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 can either be connected to RS232 transceivers and the DSUB-9 connectors on the accessory PCB, or, they can be connected to the FTDI USB to serial converter. This configuration is chosen with jumpers JP2 and JP3 on the board. Please see section Serial Interface (RS232) of the GR-CPCI-GR740 Development Board User’s Manual.

Default configuration

The default configuration of the board uses the DSUB-9 connectors by having all JP2 and JP3 jumpers disconnected.

Figure 2.8. GR-CPCI-GR740 default configuration as delivered
2.6.5. CAN

Make sure that the pin multiplexing is configured to use CAN, as explained in Section 2.5. The CAN interfaces of the GR740 can be connected to the DSUB-9 connectors on the accessory PCB. Make sure that the correct bus termination is chosen with jumpers JP1 and JP2 on the accessory PCB. Please see section CAN 2.0 Interfaces of the GR-CPCI-GR740 Development Board User’s Manual.

2.6.6. 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 can be connected to the DSUB-9 connector on the accessory PCB. Make sure that the correct bus termination is chosen with jumpers JP3 on the accessory PCB. Please see section MIL-STD-1553 Interface of the GR-CPCI-GR740 Development Board User’s Manual.

2.6.7. Spacewire

There are nine Spacewire (SPW) interfaces provided by the GR740 that can be connected on the front panel with standard MDM9S connectors. One of them is a SPW Debug interface (DSU), which can be used by installing jumpers on positions 1-2 and 3-4 of JP7.

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 boots with the Spacewire router clock-gated off. Set GPIO[11] or FP-S2-4 to low to change this and enable the Spacewire router by default.


2.6.8. PCI

Make sure that the pin multiplexing is configured to use PCI, as explained in Section 2.5. The PCI interface of the GR740, can be connected directly to backplane CPCI connector or via a PCI-PCI Bridge (Texas Instruments, PCI2060). There are four different types of configuration: 1) Host with PCI bridge; 2) Peripheral with PCI bridge; 3) Host without PCI bridge; and 4) Peripheral without PCI bridge. All these configurations are automatically set up by connecting the different plug-on PCBs provided for each type of PCI configuration. The plug-on PCB's 24, 25, 26 and 27 have to be plugged on the J24, J25, J26 and J27 on the back of the GR-CPCI-GR740 respectively. Please see section PCI Interface of the GR-CPCI-GR740 Development Board User's Manual.

The complete default configuration can be found in Appendix A. 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 4).
4. GRMON3 hardware debugger

4.1. 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-CPCI-GR740 CompactPCI 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 DIP switch FP-S3-1 must be set to OPEN.

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].

4.2. Debug-link alternatives

4.2.1. 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>
```

4.2.2. 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 192.168.0.47. The GR-CPCI-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 where the first Ethernet debug link can be bootstrapped to an address in the range 192.168.0.16 - 192.168.0.31 and the second link to an address in the range 192.168.0.32 - 192.168.0.47 (see Section 2.6.2). The second link has to be enabled on the pin multiplexing (see Section 2.5.2).

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 DIP switch FP-S2-1 and FP-S2-2 are set to OPEN 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 192.168.0.16 the GRMON3 command to connect to the board is:

```
grmon -eth 192.168.0.16
```

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

```
grmon -eth 192.168.0.16 -sdfreq <mhz>
```

### 4.2.3. Connecting via SpaceWire RMAP interface

Please see Section 2.6.7 to configure SPW DSU interface.

GRMON3 has support for connecting to boards with SpaceWire interfaces as long as the SpaceWire has RMAP and automatic link start. An Ethernet to SpaceWire bridge (GRESB) is required to tunnel SpaceWire packets from the Ethernet network over to SpaceWire.

Please see the [RD-4] for information about connecting through a GRESB and optional parameters. Connect the GRESB SpW0 connector and the GR-CPCI-GR740’s SPW DSU connector, then issue the following command:

```
grmon -gresb
```

### 4.3. 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-CPCI-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].

### 4.4. 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
GRMON LEON debug monitor v3.0.12-86-gc455ea 64-bit internal version
Copyright (C) 2018 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 09/05/2019
Parsing -u
Parsing -ftdi
Commands missing help:
```
JTAG chain (1): GR740
Device ID: 0x740
GRLIB build version: 4153
Detected system: GR740 rev1
Detected frequency: 250 MHz

<table>
<thead>
<tr>
<th>Component</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>JTAG Debug Link</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRSPW2 SpaceWire Serial Link</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>EDCL master interface</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>EDM master interface</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 SPARC V8 Processor</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 SPARC V8 Processor</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 SPARC V8 Processor</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 SPARC V8 Processor</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>I/O Memory Management Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB-to-AHB Bridge</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>L2-Cache Controller</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB Memory Scrubber</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>IOMUX secondary master i/f</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB-to-AHB Bridge</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 Debug Support Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB/APB Bridge</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AMBA Trace Buffer</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB/APB Bridge</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Mixed PT DDR/SDRAM controller</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Memory controller with EDAC</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRPCI2 PCI/ANB bridge</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRSPW Router</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 Statistics Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRPCI2 Trace buffer</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Generic UART</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>General Purpose I/O port</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Multi-processor Interrupt Ctrl.</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Modular Timer Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Modular Timer Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Modular Timer Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Modular Timer Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Modular Timer Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRSPW Router DMA interface</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRSPW Router DMA interface</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GRSPW Router DMA interface</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>GR Ethernet MAC</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>CAN Controller with DMA</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>CAN Controller with DMA</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>SPI Controller</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Clock gating unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>MIL-STD-1553B Interface</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB Status Register</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>AHB Status Register</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>General Purpose I/O port</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>General Purpose Register</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>Temperature sensor</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>General Purpose Register Bank</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>CCSDS TDP / SpaceWire I/F</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>LEON4 Statistics Unit</td>
<td>Cobham Gaisler</td>
</tr>
<tr>
<td>64-bit PCI133 SDRAM Controller</td>
<td>Cobham Gaisler</td>
</tr>
</tbody>
</table>

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

grmon>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
adev2     Cobham Gaisler  EDCL master interface
          AHB Master 2
adev3     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: FF8400000 - FF8480000
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: 0x000000000 - 0x800000000
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: 000000000 - 800000000
AHB: 800000000 - C00000000
AHB: C00000000 - E00000000
AHB: F00000000 - F00000000

l2cache0  Cobham Gaisler  L2-Cache Controller
AHB Master 0
AHB: 000000000 - 800000000
AHB: F00000000 - F04000000
AHB: FFE000000 - FFE010000
IRQ: 28
L2C: 4-ways, cachessize: 2048 kbytes, mttr: 16, FT, AHB SPLIT support

memscrub0  Cobham Gaisler  AHB Memory Scrubber
AHB Master 1
AHB: FFE000000 - FFE010000
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: 800000000 - C00000000
AHB: C00000000 - D00000000
AHB: D00000000 - E00000000
AHB: F80000000 - F89000000

dsu0  Cobham Gaisler  LEON4 Debug Support Unit
AHB: E00000000 - E40000000
AHB trace: 256 lines, 128-bit bus
CPU0: win 8, mwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
, FT
stack pointer 0x0fffffff
IRQ: icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoop tags
CPU1: win 8, mwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
, FT
stack pointer 0x0fffffff
IRQ: icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoop tags
CPU2: win 8, mwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
, FT
stack pointer 0x0fffffff
IRQ: icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoop tags
CPU3: win 8, mwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
, FT
stack pointer 0x0fffffff
IRQ: icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoop tags

apbmst0  Cobham Gaisler  AHB/APB Bridge
AHB: E40000000 - E41000000

ahbtrace0  Cobham Gaisler  AMBA Trace Buffer
AHB: EFF000000 - EEF200000
Trace buffer size: 128 lines

apbmst1  Cobham Gaisler  AHB/APB Bridge
AHB: FF8000000 - FFA000000

apbmst2  Cobham Gaisler  AHB/APB Bridge
AHB: FFA000000 - FFB000000

ddrsdmux0  Cobham Gaisler  Mixed FT DDR/SDRAM controller
AHB: 000000000 - 800000000
AHB: FFE000000 - FFE010000
Backend: sdctl0

mcrl0  Cobham Gaisler  Memory controller with EDAC
AHB: C00000000 - D00000000
AHB: D00000000 - E00000000
AHB: F00000000 - F09310000
8-bit prom @ 0xc0000000

pci0  Cobham Gaisler  GRPCI2 PCI/AHB bridge
AHB: 800000000 - C00000000
AHB: F80000000 - F84000000
spwrtr0  Cobham Gaisler GRSPW Router
AHB: FF880000 - FF881000
IRQ: 31
Instance id: 19
SpW ports: 8  AMBA ports: 4  FIFO ports: 0
l4stat0  Cobham Gaisler LEXON4 Statistics Unit
APB: E4000200 - E4000400
Device is disabled
pcitrace0 Cobham Gaisler GRPCI2 Trace buffer
APB: E4040000 - E4080000
Device is disabled
uart0  Cobham Gaisler Generic UART
APB: FF900000 - FF900200
IRQ: 29
Baudrate 38390, FIFO debug mode
uart1  Cobham Gaisler Generic UART
APB: FF901000 - FF901100
IRQ: 30
Baudrate 38390, FIFO debug mode
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: FF909000 - FF90A000
IRQ: 1
16-bit scalar, 5 * 32-bit timers, divisor 250
gptimer1 Cobham Gaisler Modular Timer Unit
APB: FF90A000 - FF90B000
IRQ: 6
16-bit scalar, 4 * 32-bit timers, divisor 250
gptimer2 Cobham Gaisler Modular Timer Unit
APB: FF90B000 - FF90C000
IRQ: 7
16-bit scalar, 4 * 32-bit timers, divisor 250
gptimer3 Cobham Gaisler Modular Timer Unit
APB: FF90C000 - FF90D000
IRQ: 8
16-bit scalar, 4 * 32-bit timers, divisor 250
gptimer4 Cobham Gaisler Modular Timer Unit
APB: FF90D000 - FF90E000
IRQ: 9
16-bit scalar, 4 * 32-bit timers, divisor 250
grsuw1  Cobham Gaisler GRSPW Router DMA interface
APB: FF90E000 - FF90F000
IRQ: 20
Number of ports: 1
grsuw2  Cobham Gaisler GRSPW Router DMA interface
APB: FF90F000 - FF91000
IRQ: 21
Number of ports: 1
grsuw3  Cobham Gaisler GRSPW Router DMA interface
APB: FF91000 - FF91100
IRQ: 22
Number of ports: 1
grsuw4  Cobham Gaisler GRSPW Router DMA interface
APB: FF911000 - FF912000
IRQ: 23
Number of ports: 1
greth0  Cobham Gaisler GR Ethernet MAC
APB: FF940000 - FF941000
IRQ: 24
1000 Mbit capable
edcl ip 192.168.0.31, buffer 2 kbyte
greth1  Cobham Gaisler GR Ethernet MAC
APB: FF980000 - FF981000
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
groc0 Cobham Gaisler Clock gating unit
APB: FFA06000 - FFA06100
GRMON did NOT enable clocks during initialization
gr1553b0 Cobham Gaisler MIL-STD-1553B Interface
APB: FFA05000 - FFA05100
IRQ: 24
Device is disabled
ahbstat0 Cobham Gaisler AHB Status Register
APB: FFA07000 - FFA07100
IRQ: 27
ahbstat1 Cobham Gaisler AHB Status Register
APB: FFA08000 - FFA08100
IRQ: 27
gpio1 Cobham Gaisler General Purpose I/O port
APB: FFA09000 - FFA09100
IRQ: 16
gpreg0 Cobham Gaisler General Purpose Register
APB: FFA09800 - FFA09900
adev49 Cobham Gaisler Temperature sensor
APB: FFAA0000 - FFAA100
grpreg1 Cobham Gaisler General Purpose Register Bank
APB: FFA08000 - FFA08100
spwtdp0 Cobham Gaisler CCSDS TDP / SpaceWire I/F
APB: FFA0C000 - FFA0C200
IRQ: 31
ghstat1 Cobham Gaisler LEON4 Statistics Unit
APB: FFA00000 - FFA00200
AHB: 00000000 - 80000000
AHB: FFE00000 - FFE00100
64-bit sdram: 2 * 128 Mbyte @ 0x00000000,
col 9, cas 3, ref 7.8 us
sdctrl0 Cobham Gaisler 64-bit PC133 SDRAM Controller
AHB: E0000000 - E4000000
AHB trace: 256 lines, 128-bit bus
CPU0: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
CPU1: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
CPU2: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
CPU3: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
gmon> 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
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
CPU1: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
CPU2: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
CPU3: win 8, nwp 4, itrace 512, V8 mul/div, srmmu, lddel 1, GRFPU
stack pointer 0xffffffff0
icache 4 * 4 kB, 32 B/line, lru
dcache 4 * 4 kB, 32 B/line, lru, snoopy tags
gmon> l2cache invalidate
invalidate all cache lines
gmon> l2cache enable
gmon> load hello
00000000 .text 24.0kB / 24.0kB [=================>] 100%
00566000 .data 2.8kB / 2.8kB [=================>] 100%
Total size: 26.84kB (788.19kbit/s)
Entry point 0x00000000
Image /home/cg/hello loaded
gmon> run
Hello world!
CPU 0: Program exited normally.
CPU 1: Power down mode
CPU 2: Power down mode
CPU 3: Power down mode
gmon> hist
TIME ADDRESS INSTRUCTIONS/AHB SIGNALS RESULT/DATA
GRCLKGATE GR740 info:
Unlock register: 0x00000000
Clock enable register: 0x00000585
Reset register: 0x0000027a

GR740 decode of values:

<table>
<thead>
<tr>
<th>Gate</th>
<th>Core(s)</th>
<th>Description</th>
<th>Unlocked</th>
<th>Enabled</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>GRETH</td>
<td>Ethernet MAC 0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>GRETH</td>
<td>Ethernet MAC 1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>SPWRTR</td>
<td>SpaceWire router</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>PCI</td>
<td>PCI (GRPCI, PCIDMA)</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>GR1553B</td>
<td>MIL-STD-1553B</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>GRCAN</td>
<td>CAN core 0 &amp; 1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>LASTAT</td>
<td>L04M Statistics</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>APBUART</td>
<td>UART 0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>APBUART</td>
<td>UART 1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>SPICTRL</td>
<td>SPI Controller</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>MCTRL</td>
<td>PROM/IO</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

GRCLKGATE GR740 info:
Unlock register: 0x00000000
Clock enable register: 0x00000595
Reset register: 0x0000026a

GR740 decode of values:

<table>
<thead>
<tr>
<th>Gate</th>
<th>Core(s)</th>
<th>Description</th>
<th>Unlocked</th>
<th>Enabled</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>GRETH</td>
<td>Ethernet MAC 0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>GRETH</td>
<td>Ethernet MAC 1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>SPWRTR</td>
<td>SpaceWire router</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>PCI</td>
<td>PCI (GRPCI, PCIDMA)</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>GR1553B</td>
<td>MIL-STD-1553B</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>GRCAN</td>
<td>CAN core 0 &amp; 1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>LASTAT</td>
<td>L04M Statistics</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>APBUART</td>
<td>UART 0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>APBUART</td>
<td>UART 1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>SPICTRL</td>
<td>SPI Controller</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>MCTRL</td>
<td>PROM/IO</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

grmon3> grcg clkinfo
5. Board Package

5.1. Overview

The board package distributed together with this document contains GRMON3 scripts and MKPROM initialization functions that are specific to the GR740 or GR-CPCI-GR740. The package is named gr-cpci-gr740-<version>.

5.2. MKPROM2 bdinit functions

The subdirectory GRMON3 contains initialization code to be used with the MKPROM2 boot-PROM builder. To create boot-PROMs for GR740 MKPROM2 version 2.0.60 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-CPCI-GR740 is to compile the `bdinit.c` file. This is done with the command `sparc-elf-gcc -O2 -c -o bdinit.o gr-cpci-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-CPCI-GR740. The full MKPROM2 command for creating an image is:

```
/opt/mkprom2/mkprom2 -v \
 -stack <stack pointer> -ramsize <size of RAM in KiB> -sparcleon0 \ 
 -memcfg1 0x08030c0ff -memcfg3 0x08000000 \ 
 -dump -v -rstaddr 0x00000000 -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 256 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 -\texttt{Wl,-msparcleon0} flag.

```
/opt/mkprom2/mkprom2 -v  \
  -stack 0x0fffff00 -ramsize 262144 -sparcleon0  \
  -memcfg1 0x0803c0ff -memcfg3 0x08000000  \
  -dump -v -rstatdr 0xc0000000 -uart 0xFF900000 -freq 250 -baud 38400  \
  -bdinit hello -o hello.prom
```

The output of calling mkprom2 2.0.60 with the options above is:

```
bash-3.2$ /opt/mkprom2/mkprom2 -v  
  > -stack 0x0fffff00 -ramsize 262144 -sparcleon0  
  > -memcfg1 0x0803c0ff -memcfg3 0x08000000  
  > -dump -v -rstatdr 0xc0000000 -uart 0xFF900000 -freq 250 -baud 38400  
  > -bdinit hello -o hello.prom
```

The **hello.prom** image can now be programmed to the board's Flash. A transcript of this operation using GRMON3 is included below:

```
grmon3> flash erase all
    Erase in progress
    Block @ 0xc07e0000 : code = 0x80  OK
    Erase complete

grmon3> flash load hello.prom
    C0000000 .text                      31.1kB /  31.1kB   [===============>] 100%
    Total size: 31.12kB (404.72kbit/s)
    Entry point 0xc0000000
    Image /home/ag/hello.prom loaded

grmon3> verify hello.prom
    C0000000 .text                      31.1kB /  31.1kB   [===============>] 100%
    Total size: 31.12kB (6.07Mbit/s)
    Entry point 0xc0000000
    Image of /home/ag/hello.prom verified without errors

grmon3> flash
    Intel-style 8-bit flash on D[31:24]

    Manuf.  : Intel
    Device  : MT28F640J3
    Device ID  : 96eff440a00000403f
    User ID  : fffffffffffffff

    1 x 8 Mbytes = 8 Mbytes total @ 0xc0000000
```
CFI information
Flash family : 1
Flash size    : 64 Mbit
Erase regions : 1
Erase blocks  : 64
Write buffer  : 32 bytes
Lock-down     : Not supported
Region 0     : 64 blocks of 128 kbytes

grmon3>

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

MkProm2 boot loader v2.0
Copyright Gaisler Research - all rights reserved

system clock : 250.0 MHz
baud rate    : 38422 baud
prom         : 512 K, (2/2) ws (r/w)
sram         : 0 K, 1 bank(s), 0/0 ws (r/w)

decompressing .text to 0x00000000
decompressing .data to 0x0000A490

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_sdctr10.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.
  In the example above, the PROM width used was 8 bits. To use a 16-bit wide PROM interface, the field at bits 9:8 of MEMCFG1 should be changed from 0 to 1 (`-memcfg1 0x0803c1ff`). The PROM width setting must match with the PROM width selection made via the bootstrap signal GPIO[10] and the setting of jumper JP6f on the board. See section 4.5.4 of the GR-CPCI-GR740 Development Board User's Manual.

• 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 will still pressing BREAK.
6. Frequently Asked Questions / Common Mistakes / Known Issues

6.1. 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 \texttt{grc enable all} will have the same effect.

6.2. GRMON3 issues

When connected to the board, the message "stack pointer not set" will be shown by the command \texttt{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 persists, try stopping the kernel driver \texttt{ftdi_sio}.

6.3. 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 5. 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:

\begin{verbatim}
grmon3> l2cache invalidate
   invalidate all cache lines
grmon3> l2cache enable
\end{verbatim}

6.4. Main memory problems

If the memory is behaving erratically, please make sure that the configuration plug is properly adjusted on the connector J23.

6.5. 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:

\begin{verbatim}
grmon3> scrub clear 0 0xffffffff
0x00000000 256.0MB / 256.0MB   [=================>] 100%
grmon3> wmem 0xffffe00020 0x1
grmon3> l2cache invalidate
   invalidate all cache lines
grmon3> l2cache enable
\end{verbatim}

Note that the \texttt{scrub} above initialises 256 MiB of memory. If the system has 128 MiB of memory then the command is \texttt{scrub clear 0 0x07ffffffff}.

The scrubber monitors the Memory AHB bus for errors reported by the memory controller. The command \texttt{scrub} shows the status:

\begin{verbatim}
grmon3> scrub
AHB status register: Not triggered
Scrubber status:      Done init 00000000-0xffffffff
\end{verbatim}
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:

gromon3> 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

6.6. Flash programming

If Flash programming is too slow, use the EDCL debug link to program the flash, since it can offer up to 10x speed-ups.

6.7. PROM EDAC

To boot with PROM EDAC enabled, you need add the "-bch8 -romsize 8192" to your MKPROM2 flags. MKPROM2 will then create a PROM binary with the EDAC checkbits (-bch8) located at the end of the PROM memory area, given by the board's default configuration PROM size of 8 MiB (-romsize 8192, in KiB). The outfile file with .bch8 extension can be loaded to the PROM as usual. To enable PROM EDAC, set GPIO[14] to pull-up (PU).

When having custom pin multiplexing solutions, you need to make sure that the PROM address lines pin multiplexing is correct and matches the PROM size given to MKPROM2. For instance, to have a PROM size of 8 MiB flash, which is the default configuration, you need the PROM_ADDR[20:22] pins to access the EDAC checkbits. In the board pin multiplexing, these pins can be assigned to the PROM memory or to the 1553TXNB, 1553TXNB and 1553RXENB signals. If for instance, we need to use the 1553 interface signals, we cannot have a 8 MiB flash and therefore we need to adjust the MKPROM2 flags for a 1 MiB flash to be able to use the PROM_ADDR[20:22] pins for the 1553 interface. For that purpose, change the MKPROM flags from "-romsize 8192" to "-romsize 1024". Please note that this custom configuration leaves PROMIO_ADDR 21 and 20 (JP11 7 and 8) floating, which are connected to the flash memory (this is a board limitation).

6.8. Ethernet

If ethernet port 1 is not working properly, make sure that a) the pin multiplexing is correctly configured and b) the configuration plug is properly connected. See Section 2.5.2.

If EDCL is not working, please make sure that a) GPIO[8] and GPIO[9] are set to route traffic on the Debug AHB bus; b) a proper IP address has been selected either through GPIO[0-5] or via GRMON3; and c) FP-S3-7 selects the correct network speed. See Section 2.6.2.

6.9. 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.

6.10. Pin multiplexing (PROMIO/Peripherals)

The default full 8MiB PROM in 8-bit mode and the UART can be used at the same time. The 16-bit PROM mode cannot be used with UARTs at the same time. See Section 2.5.1.

In case the full PROM and all the interfaces are desired, what can be done is to connect the IOs from the GR740 to both the PROM and the interface transceivers, boot up in the full-PROM mode by pulling down gpio(15) during reset, and after completing the decompression of your PROM to RAM, switch over the IOs to their interface functions. This will have the side effect of toggling the interface IO lines according to the address driven to the PROM during the decompression phase which might cause unwanted interface communication. For 1553 this has been solved by making the TX-inhibit lines non-multiplexed which will prevent any 1553 data from going out.
For the other IOs you will have to make a system-level decision as to whether you can tolerate some IO interfaces toggling during boot-up, and if not, you could use one of the GPIOs to drive an enable signal to those IO interfaces transceivers to disable them during boot. Please note that this is not supported currently on the GR-CPCI-GR740 development board.

6.11. Can’t boot

First, check that the BREAK button is not asserted (should be in the rightmost position). Second, 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. Third, 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 mctrl10
Memory controller with EDAC
0xff903000 Memory config register 1 0x0003c0ff
  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  hexcn  0x0  Bus error enable
 23:20 iows  0x0  I/O wait states
 19  loen  0x0  I/O enable
 17:14 prombanksz  0xf  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 0x00000020
  31  sdramrf  0x0  SDRAM refresh enable
  30  sdramtrp  0x0  SDRAM TRP parameter
 29:27 sdramtrfc  0x0  SDRAM TRFC parameter
 26  sdramcas  0x0  SDRAM CAS 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  0x0  RAM bank size
  7  rbtdy  0x0  RAM bus read 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 0x00000000
  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  FROM EDAC enable
  7:0 tcb  0x0  Test checkbits
0xff903010 Memory config register 5 0x00000000
 29:23 iohw  0x0  I/O lead out
 13:7 romhw  0x0  ROM lead out
0xff903014 Memory config register 6 0x00000000
 13:7 ramhw  0x0  RAM lead out
0xff903018 Memory config register 7 0x00000000
 31:16 brdyncnt  0x0  Bus ready count
 15:0 brdynrlid  0x0  Bus ready reload value
```

```
grmon3> info reg -v sdctrl10
PCI33 SDRAM Controller
0xffe00000 SDRAM config register 0xfea087a4
  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 | 0x5 | 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 | 0x1 | SDRAM 64-bit data bus |
| 14:0 | reload | 0x7a4 | Refresh reload value |

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

### 6.12. FTDI/JTAG

If the JTAG connection is not working, check the following things:

- Make sure that the DSU is enabled (FP-S3-1) in open position (see Figure 2.2).
- Check that all JP4 jumpers are connected (see Figure 2.6).
- Check that the GPIO ribbon cable that goes from the front panel to the PCB J4 connector is properly connected (see Figure 6.1). Sometimes this connector moves a little bit out due to movement of the board and creates an unstable JTAG connection.
- 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.

![Figure 6.1. GR-CPCI-GR740 GPIO J4 connection.](image)

### 6.13. SDRAM not working

If the SDRAM is not working, check the following things:

- Make sure that the configuration plug is properly installed on the desired configuration (see Figure 2.5). This connector has created problems after significant physical manipulation and movement of the board, creating an unstable SDRAM connection. Try re-installing it.
7. Support

For support contact the Cobham Gaisler support team at support@gaisler.com.
## Appendix A. Default configuration

This appendix describes the jumper and switch positions for the default configuration of the board. See section Setting Up and Using the Board on the GR-CPCI-GR740 Development Board User's Manual.

### Table A.1. Jumper configuration

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Default configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>JP1</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP2</td>
<td>All disconnected.</td>
</tr>
<tr>
<td>JP3</td>
<td>All disconnected.</td>
</tr>
<tr>
<td>JP4</td>
<td>All connected.</td>
</tr>
<tr>
<td>JP5</td>
<td>1-2, 3-4 connected. 5-6 disconnected</td>
</tr>
<tr>
<td>JP6</td>
<td>All connected except 1-2.</td>
</tr>
<tr>
<td>JP7</td>
<td>All disconnected.</td>
</tr>
<tr>
<td>JP8</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP9</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP10</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP11</td>
<td>1, 2, 13 and 14 in position B-C. The rest in position A-B.</td>
</tr>
<tr>
<td>JP12</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP13</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP14</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP15</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP16</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP17</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP18</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP19</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP20</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP21</td>
<td>Disconnected.</td>
</tr>
<tr>
<td>JP22</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP23</td>
<td>Connected.</td>
</tr>
<tr>
<td>JP24</td>
<td>All disconnected.</td>
</tr>
</tbody>
</table>

### Table A.2. Switch configuration

<table>
<thead>
<tr>
<th>Switch</th>
<th>Default configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP-S1-1</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-2</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-3</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-4</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-5</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-6</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-7</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S1-8</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S2-1</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>Switch</td>
<td>Default configuration</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>FP-S2-2</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S2-3</td>
<td>Pull-down.</td>
</tr>
<tr>
<td>FP-S2-4</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S2-5</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S2-6</td>
<td>Pull-up.</td>
</tr>
<tr>
<td>FP-S2-7</td>
<td>Pull-down.</td>
</tr>
<tr>
<td>FP-S2-8</td>
<td>Pull-down.</td>
</tr>
<tr>
<td>FP-S3-1</td>
<td>Open.</td>
</tr>
<tr>
<td>FP-S3-2</td>
<td>Open.</td>
</tr>
<tr>
<td>FP-S3-3</td>
<td>Closed.</td>
</tr>
<tr>
<td>FP-S3-4</td>
<td>Closed.</td>
</tr>
<tr>
<td>FP-S3-5</td>
<td>Closed.</td>
</tr>
<tr>
<td>FP-S3-6</td>
<td>Open.</td>
</tr>
<tr>
<td>FP-S3-7</td>
<td>Closed.</td>
</tr>
<tr>
<td>FP-S3-8</td>
<td>Open.</td>
</tr>
<tr>
<td>S1</td>
<td>Right position.</td>
</tr>
</tbody>
</table>