



**HUNT ENGINEERING**  
Chestnut Court, Burton Row,  
Brent Knoll, Somerset, TA9 4BP, UK  
Tel: (+44) (0)1278 760188,  
Fax: (+44) (0)1278 760199,  
Email: sales@hunteng.co.uk  
<http://www.hunteng.co.uk>  
<http://www.hunt-dsp.com>



# ***HUNT ENGINEERING***

## ***CPCI9-EM2*** ***Inter-board module***

### ***USER MANUAL***

***Hardware Rev B***  
***Document Rev D***  
***M.Siggins 24/7/03***

## **COPYRIGHT**

This documentation and the product it is supplied with are Copyright HUNT ENGINEERING 2002. All rights reserved. HUNT ENGINEERING maintains a policy of continual product development and hence reserves the right to change product specification without prior warning.

## **WARRANTIES LIABILITY and INDEMNITIES**

HUNT ENGINEERING warrants the hardware to be free from defects in the material and workmanship for 12 months from the date of purchase. Product returned under the terms of the warranty must be returned carriage paid to the main offices of HUNT ENGINEERING situated at BRENT KNOLL Somerset UK, the product will be repaired or replaced at the discretion of HUNT ENGINEERING.

Exclusions - If HUNT ENGINEERING decides that there is any evidence of electrical or mechanical abuse to the hardware, then the customer shall have no recourse to HUNT ENGINEERING or its agents. In such circumstances HUNT ENGINEERING may at its discretion offer to repair the hardware and charge for that repair.

Limitations of Liability - HUNT ENGINEERING makes no warranty as to the fitness of the product for any particular purpose. In no event shall HUNT ENGINEERING'S liability related to the product exceed the purchase fee actually paid by you for the product. Neither HUNT ENGINEERING nor its suppliers shall in any event be liable for any indirect, consequential or financial damages caused by the delivery, use or performance of this product.

Because some states do not allow the exclusion or limitation of incidental or consequential damages or limitation on how long an implied warranty lasts, the above limitations may not apply to you.

## **TECHNICAL SUPPORT**

Technical support for HUNT ENGINEERING products should first be obtained from the comprehensive Support section <http://www.hunteng.co.uk/support/index.htm> on the HUNT ENGINEERING web site. This includes FAQs, latest product, software and documentation updates etc. Or contact your local supplier - if you are unsure of details please refer to <http://www.hunteng.co.uk> for the list of current re-sellers.

HUNT ENGINEERING technical support can be contacted by emailing [support@hunteng.co.uk](mailto:support@hunteng.co.uk), calling the direct support telephone number +44 (0)1278 760775, or by calling the general number +44 (0)1278 760188 and choosing the technical support option.

# TABLE OF CONTENTS

<b>INTRODUCTION</b> .....	<b>5</b>
<b>LOCATION OF ITEMS ON THE CPC19-EM2</b> .....	<b>6</b>
<b>GETTING STARTED</b> .....	<b>7</b>
<b>MODULE FEATURES</b> .....	<b>8</b>
HEART CONNECTIONS .....	8
RESET.....	9
GIGABIT SERIAL CONNECTIONS.....	9
GIGABIT SERIAL CABLING .....	9
HERON SERIAL BUS.....	10
UDPRESET.....	10
MODULE ID.....	11
UMI CONNECTIONS.....	11
DONE LED .....	11
<b>HARDWARE DETAILS</b> .....	<b>12</b>
HERON MODULE TYPE.....	12
HARDWARE RESET .....	12
CONFIG.....	12
PHYSICAL DIMENSIONS OF THE MODULE .....	12
POWER REQUIREMENTS OF THE CPC19-EM2 .....	13
<i>Calculating Latency when using the EM2</i> .....	13
<i>Serial I/O characteristics</i> .....	14
<b>FITTING THE CPC19-EM2 TO YOUR COMPUTER</b> .....	<b>15</b>
<b>ACHIEVABLE SYSTEM THROUGHPUT</b> .....	<b>16</b>
<b>EXAMPLE CONFIGURATIONS</b> .....	<b>17</b>
<i>Example1 – two or more boards in the same host PC booted as one system -- method A</i> .....	17
<i>Example1a – two or more boards in the same host PC booted as one system -- method B</i> .....	20
<i>Example2 – two or more boards in the same host PC booted separately</i> .....	21
<i>Example3 – two or more boards in different host PCs booted separately</i> .....	23
<i>Example4 – two or more boards in different host PCs booted as one system</i> .....	23
<i>Example5 – more than two boards in a system booted separately</i> .....	24
<b>TROUBLESHOOTING</b> .....	<b>25</b>
HARDWARE .....	25
SOFTWARE .....	25
<b>CE MARKING</b> .....	<b>26</b>
<b>TECHNICAL SUPPORT</b> .....	<b>27</b>
<b>APPENDIX 1 – HERON SERIAL BUS COMMANDS</b> .....	<b>28</b>
MODULE ADDRESS .....	28
MODULE QUERY .....	28
INTERNAL REGISTER PROGRAM .....	28
<b>APPENDIX 2 – INTERNAL REGISTER DESCRIPTION</b> .....	<b>29</b>
RECEIVE CHANNEL TIMESLOT REGISTERS .....	29
TRANSMIT CHANNEL TIMESLOT REGISTERS .....	29
RECEIVE CHANNEL RESET-WITH-UMI REGISTERS .....	30
TRANSMIT CHANNEL RESET-WITH UMI REGISTERS .....	30
ZAP HEART REGISTER .....	30

RESET PROPAGATION REGISTER.....	30
HSB PROPAGATION REGISTERS.....	31
ZAP HSB PROPAGATION CONTROL.....	31
<b>APPENDIX 3 – GIGABIT SERIAL I/O CONNECTOR PINOUT .....</b>	<b>32</b>

The CPC19-EM2 is a module designed specifically for use with Hunt Engineering HECPC19 Heron Module Carriers, to provide point-to-point communication between HECPC19s, and HEPC19s.

The CPC19-EM2 plugs into a Rear-panel I/O slot of a CompactPCI host computer, such that it connects to the rear-panel I/O connections of an HECPC19 that is plugged into the corresponding Peripheral Slot . The J4 connector of the HECPC19 is designed for external modules to connect to the Heart architecture of the HECPC19, so that connections of the inter-node data may be made from one HECPC19 to another. The J4 connections are simply bypassed when there is no CPC19-EM2 fitted to it.

The interboard connectivity is achieved with up to six 1.25 Gigabit bi-directional links, over which the inter-node fifo data may be passed. The serial link connections between HECPC19s may be made according to the connectivity requirements of the end system. For example, it is possible that only one serial link may be required between a pair of HECPC19s. In other applications where multiple data paths exist between HECPC19s, multiple serial links may be used.

In addition to the internode data, the CPC19-EM2 also transmits the state of the HECPC19's reset signal, over any connected serial links, to other CPC19-EM2s. These other remote EM2's may be programmed, over the HSB, to propagate the state of the received reset onto the UDPRESET of the remote HECPC19, thus providing the function of remote resetting.

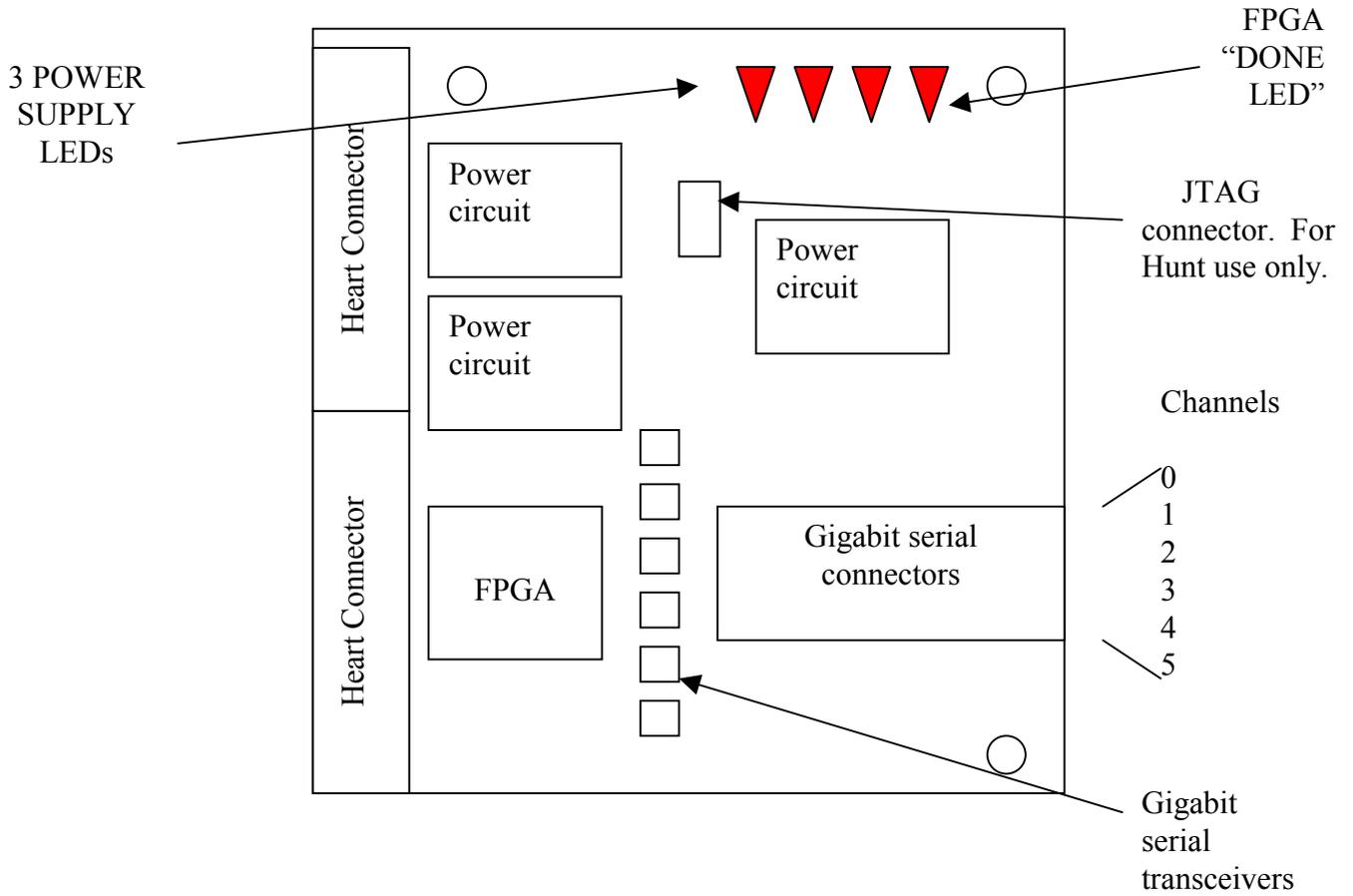
The CPC19-EM2 may also be programmed, again over the HSB, to propagate HSB messages over the serial links. It is therefore possible to perform all HSB functions on a remote HECPC19, such as heart configuration, module queries, and any other arbitrary HSB messaging.

The CPC19-EM2 module connects to the HECPC19 through several standard interfaces.

- The first is directly into the Heart architecture. The FIFO input and output interfaces, are embedded into the CPC19-EM2 itself. This is used for the main inter-node communications.
- The second is the HERON Serial Bus, used for configuration messages.

The last is the general control such as reset, power etc.

## Location of Items on the CPC19-EM2



The CPC19-EM2 is a module that plugs into a Rear-panel I/O slot of a CompactPCI host computer, for connection to the HECPCI9 HERON Module Carrier in the corresponding Peripheral slot.

The CPC19-EM2 should be fitted to the host computer along with any other carriers and their modules that your system has and their retaining nuts fitted (see a later section of this manual for details).

The connections provided by the CPC19-EM2 are point to point, and appear as FIFOs in the HEART system. Where you fit the cables is what determines the boards that are connected and the FIFO numbers that will be used. Each cable can make a connection in both directions.

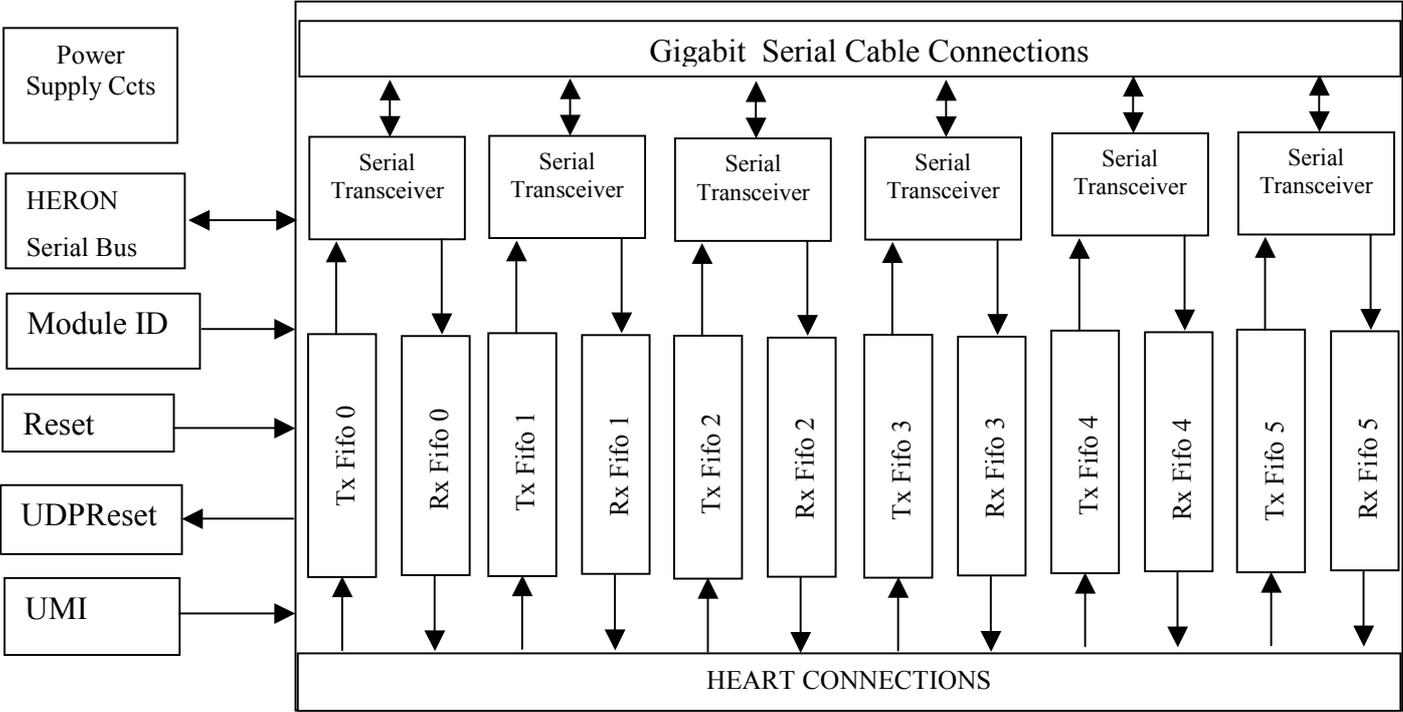
Because the connection can be programmed to propagate the hardware reset, it is possible to boot the “remote” or “slave” boards in the system over the CPC19-EM2 connection when using the Server/Loader program. Because the connections are also able to carry the HSB connections, it is also possible to configure FPGAs over this connection. In that case the Server/Loader will properly start the system.

If however you choose to boot the systems at the different ends of the CPC19-EM2 connection separately, the CPC19-EM2 should not propagate the reset and HSB between the systems. In that case the connection can be used to send and receive data, but careful consideration must be given to the synchronisation of those two systems. If for example the first system booted immediately starts to send data on a connection, it is probable that the other system will not yet be configured to receive. When that second system is booted, it will be reset, and its connections configured before it will be ready to send and or receive data. During that reset and configuration phase data will probably be lost from the system. It is important that your application software correctly handles this situation. There is a server/loader example on the HUNT ENGINEERING CD that demonstrates one possible way of handling this. Of course different applications will need to find different solutions.

The Gbit connection is designed to be totally reliable. However, if the cable is routed outside of your PC, it could be disturbed by noise or movement of the cable. In that case the connection should recover and work properly afterwards, but data could be lost/gained or corrupted during the disturbance. If your system is intended to run for long periods of time then you should build something into your application programs that can detect and recover from such a disturbance.

The connections to and from the CPC19-EM2 are programmed using the standard Heartconf tool at the same time as your other HEART connections. You can refer to the separate user manual for Heartconf and the Network File syntax to find the full details, but there are some examples later in this manual that show the typical uses.

This section describes the features of the CPC19-EM2 and why they are provided.



## Heart Connections

The External Ring Module slot on a HECPCI9 is bypassed when there is no CPC19-EM2 module fitted into its rear-panel I/O slot. All external modules ‘break into’ the heart architecture and become part of it when they are plugged in, and the module itself provides the Heart functionality for this slot. This feature is transparent to the user as the number of registers on the HEART ring is the same with and without the CPC19-EM2.

Programming of data paths between the Heart architecture and the CPC19-EM2 is exactly the same as for other module slots on the carrier board. No further detail is supplied here, as the heartconf software tool provided by HUNT ENGINEERING will be used. For reference only, the full description and addresses of the timeslot registers is detailed in the Appendix.

## **Reset**

The reset signal from the carrier board is driven into the CPC19-EM2, to initialise internal logic. Additionally, transitions of the reset signal are detected and codes for these transitions are sent to other CPC19-EM2s that may be connected via the gigabit serial connections. This allows a system wide reset to be selected if required, and this signal is then encoded on the same cable connection as the data.

## **Gigabit Serial Connections**

Each connection provides a differential PECL transmit pair, and a differential receive pair. These may connect directly to another CPC19-EM2, with cables provided by HUNT ENGINEERING. The serial data rate of each channel is 1.25Gbits per second. This equates to 125Mbytes per second.

These serial connections are mainly used to provide point-to-point connections between two or more HECPC19s. By appropriate programming of control registers of the carrier board and the CPC19-EM2s, data may be routed between HERON modules that are on different HECPC19s, in much the same way as if they were on the same HECPC19. It should be noted that the data bandwidth available for a serial channel, is 125Mbytes per second, while it is possible for inter-node communications on the same carrier to exceed this figure. Therefore, the system topology should be chosen so that any datapaths where the sustained data rate may exceed 125Mbytes/sec are constrained to be on the same carrier board.

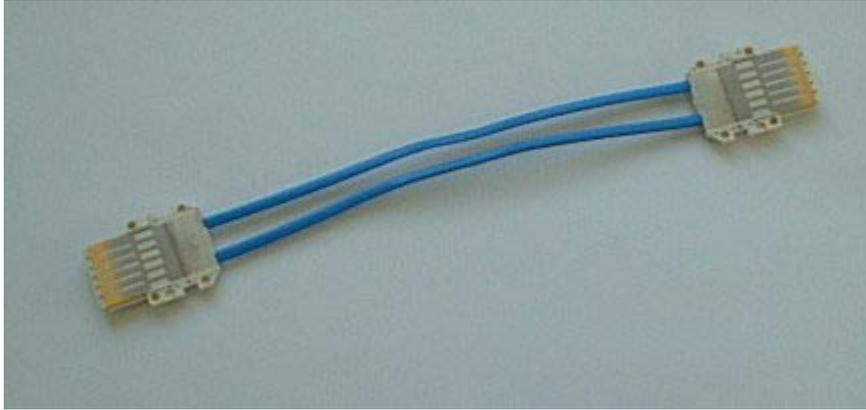
Data to be transmitted is routed from the heart architecture and split into 8-bit bytes. These bytes are then converted into 10-bit codes, which are then serialised and transmitted. At the receive end, the data stream is de-serialised into 10-bit codes, converted back into 8-bit data bytes, then 32-bit words, and then fed into the heart architecture. The 10-bit encoding is done for two reasons; firstly so that the serial data stream is 'edge-rich' to make clock recovery practical, and secondly to maintain the DC level of the serial signal as close to zero as possible. There is also the advantage that some of the spare 10-bit codes may be used for other functions, for example reset and HSB propagation. See Heron Serial Bus section, and UDPReset section for more detail on reset and HSB propagation.

The reset and HSB propagation takes priority over data transfer, but the loss of data bandwidth is insignificant and immeasurable.

## **Gigabit Serial Cabling**

The CPC19-EM2 is supplied with a single 15cm serial cable that has been specially manufactured for HUNT ENGINEERING. HUNT ENGINEERING also have 50cm cables made. These are suitable for making connections within the same PC. Additional cables can be purchased from HUNT ENGINEERING or your local HUNT ENGINEERING supplier.

These cables are polarised with a blanking plug to prevent incorrect insertion. The correct insertion is so that the metal side of the cable connector mates with the metal at the side of the PCB connector.



### **The standard cable supplied with CPCI9-EM2**

For connections outside of the PC more substantial cabling is necessary. The cabling used must be specifically optimised for Gbit serial transmission in both directions, line cables normally intended for Fibre-Channel applications, which despite its name is actually copper cable connections.

HUNT ENGINEERING can manufacture cables that take the EM2 connections to a 9 way D type connector that can be fitted to the backplate of your PC. Then standard Fibre-Channel cables (which can be purchased from HUNT ENGINEERING or your local HUNT ENGINEERING supplier) can be fitted directly to those 9 way D types.

### **Heron Serial Bus**

The Heron Serial Bus connects to the CPCI9-EM2, not only so that its internal registers may be programmed with it, but also so that HSB messages may be sent between HECPCI9s, via the gigabit serial connections.

This bridging of the HSB is achieved by use of some of the spare 10-bit codes, to transmit the various states of the local Heron Serial Bus, to the remote CPCI9-EM2, which in turn masters the remote HSB with identical bus behaviour. The propagation of HSB messages is programmable for each channel.

This programming is made using the Heartconf software utility.

The propagation of HSB across multiple boards is optional, but if selected this connection is then encoded on the same cable connection as the data.

The CPCI9-EM2 also responds to HSB Module Query commands.

### **UDPReset**

Changes in the state of the reset signal are transmitted by a CPCI9-EM2 using spare 10-bit codes. The UDPReset signal is output from the receiving CPCI9-EM2, onto its HECPCI9. The UDPReset is asserted low by the CPCI9-EM2 when a serial channel receives an active reset condition, AND that channel is programmed to propagate the reset locally. This feature may be used where one HECPCI9 may be remote and/or host-less, so that it may be reset by its UDPReset in this way.

## **Module ID**

The HERON specification assigns pins on the HERON module that give a HERON module access to the carrier ID of the carrier that it is plugged into, and a unique HERON slot identifier. The External Module slot of the HECPCI9 also has its unique module ID, so that an external module, such as the CPC19-EM2, is addressed on Heron Serial Bus (HSB) using this information.

## **UMI connections**

The CPC19-EM2 connects to the UMI (Uncommitted Module Interconnect) signals of the HECPCI9, so that they may be used to flush a FIFO when it is used in the non-blocking mode. This feature is sometimes useful in applications where it is not necessary to receive all data that is sent. For example, if live video data is being transmitted through the architecture, and a particular module's requirement is to occasionally receive the next frame of video. In this scenario, the CPC19-EM2 could be programmed to flush the relevant FIFOs when a particular UMI signal is asserted low, to get rid of old data. Each transmit and each receive FIFO may be individually programmed to be reset by any UMI signal. Multiple UMI signals may be used to reset a FIFO.

## **Done LED**

There is a Done LED, labelled "DONE". It is illuminated if the FPGA is not configured.

The "DONE" led should illuminate briefly at power on, and then go out showing that the FPGA is correctly configured.

### HERON Module Type

The CPC19-EM2 module is of type “Expansion module”.

The CPC19-EM2 does not have a processor so does not assert the “Module has processor” pin as defined in the HERON specification.

The CPC19-EM2 does not support JTAG so does not assert the “Module has JTAG” pin as defined in the HERON specification.

The CPC19-EM2 has a serial bus so asserts the “Module has serial bus” pin as defined in the HERON specification.

### Hardware Reset

Before the CPC19-EM2 can be used, it must be reset. This reset initialises the Heron Serial Bus circuitry into a state where it can be used. Depending on how this and any other CPC19-EM2s in the system have been programmed, this local reset may also reset other CPC19-EM2s and their module carriers, and similarly, a reset on a remote carrier may cause a reset to be propagated locally.

This signal is driven by the HECPC19.

### Config

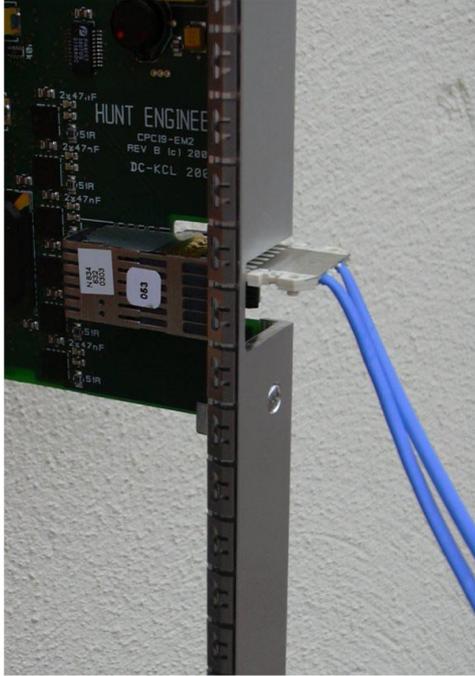
The CPC19-EM2 does not connect to the config signal.

### Physical Dimensions of the Module

The CPC19-EM2 module is 80mm by 233.35mm overall.

The 3.12mm limit on component height under the module is not violated by the CPC19-EM2.

The maximum height of the CPC19-EM2 above the module including mating connectors and cables is 14.5mm. This means that the 13.71mm limit on component height **IS** violated, and also that the assembly of a HECPC19 module carrier and the CPC19-EM2 is **MORE** than the 20mm single slot spacing of CompactPCI.



### **The arrangement of cabling with the CPC19-EM2**

The Gbit connectors of the CPC19-EM2 finish level with where the edge of the board would be, and are therefore inside the panel that may be fitted. The mating connectors however, will occupy the space outside of the panel.

### **Power Requirements of the CPC19-EM2**

The CPC19-EM2 only uses power from the 5V HECPCI9 supply. The 3.3V and 1.5V for the FPGA, and the 2.5V for the serial transceivers, are generated on board from this +5V.

The power demands of the CPC19-EM2 are governed by the amount of data flowing into and out of the serial channels, and also the amount of data flowing around the local HEART architecture. The maximum current for worst-case test patterns is approximately 1.5A on the 5V input.

### **Calculating Latency when using the EM2**

One feature of HEART is that the latency of a communication is controlled within limits that can be calculated. The way that the connections of HEART are pre-connected means that there are no arbitration delays. It is also not possible that a connection will fail to connect – it is already connected before the data is sent.

The use of FIFOs means that the latency varies depending on how you use them. For example if you use a block flag to determine when to write, that write may be delayed until there is space.

If we start with data to be transmitted from the EM2 being available at the HEART system,

1. Data that is valid on the ring is clocked into the transmit FIFO on the next 100MHz cycle.

2. Data will flow through the transmit FIFO in 6 cycles of the 100MHz.
3. Data is read from the transmit FIFO in bytes, converted to 10-bit codes, serialised and transmitted from the TX outputs, in 12 of the 125MHz clocks.
4. At the receiving end, the RX inputs are de-serialised, decoded, and written into the receive FIFO in 10 of the 125MHz clocks.
5. The data will be available to the HEART system in 13 cycles of the 100MHz.
6. The time slots travel around the HEART ring constantly, so the longest wait for a slot will be five 100MHz clocks, the data will be placed onto the ring in the sixth clock at the latest.
7. The data goes out onto the HEART system in the next 100MHz cycle.

I.e. Items 1, 2, 5, 6, and 7 add up to 21 + (up to 5) of the 100MHz cycles. Items 3 and 4 add up to 22 of the 125MHz cycles.

Therefore, the ADDITIONAL latency in using an EM2 in the data path will be 386nS to 436nS.  $((21 + M) \times 10\text{nS}) + (22 \times 8\text{nS})$ . Where  $M = 0$  to 5.

So, for modules on different carrier boards, minimum latency is achieved with fastest FIFO clocks, the sending module in the slot immediately upstream of the sending EM2, AND the receiving module in the slot immediately downstream of the receiving EM2 The times will be 646nS to 746nS.  $(6 \times 10\text{nS}) + ((41 + M) \times 10\text{nS}) + (22 \times 8\text{nS})$ , where  $M = 0$  to 10.

The maximum latency will occur with the slowest FIFO clocks, the sending module in the slot immediately DOWNstream of the sending EM2, AND the receiving module in the slot immediately UPstream of the receiving EM2 The times will be 1086nS to 1186nS.  $(6 \times 16.6\text{nS}) + ((41 + M) \times 10\text{nS}) + (4 \times 10 \times 10\text{nS}) + (22 \times 8\text{nS})$ , where  $M = 0$  to 10.

### **Serial I/O characteristics**

The serial I/O is compatible with the differential PECL standard, but is intended only for connection to another EM2 module.

## **Fitting the CPCI9-EM2 to your computer**

---

Fitting a CPCI9-EM2 module to your Compact PCI machine is very simple. Ensure that the computer does NOT have power applied when fitting modules or carrier boards, and normal anti-static precautions should be followed at all times.

The CPCI9-EM2 is supplied with a standard front panel and ejector handles. The use of these is optional.

Choose the rear-panel I/O slot that corresponds with the HECPCI9 peripheral slot that you wish to connect to. This slot should also have sufficient space around it for any cables that you have.

If you do not have a backplane fitted to your rack, you can use the supplied connectors to directly connect the CPCI9-EM2 to the HECPCI9 board.

Push the CPCI9-EM2 into the chosen slot, ensuring that it locates in the top and bottom card guides. Push the CPCI9-EM2 fully home.

DO NOT use excessive force on either the CPCI9-EM2 or the rack. Switch the power on and it should boot normally. If it doesn't then re-check that the CPCI9-EM2 is pushed fully home in the host computer.

## Achievable System Throughput

---

In a HERON system there are many factors that can affect the achievable system throughput. It must be remembered at all times that the part of the system that has the lowest limit on bandwidth will govern the throughput of the system.

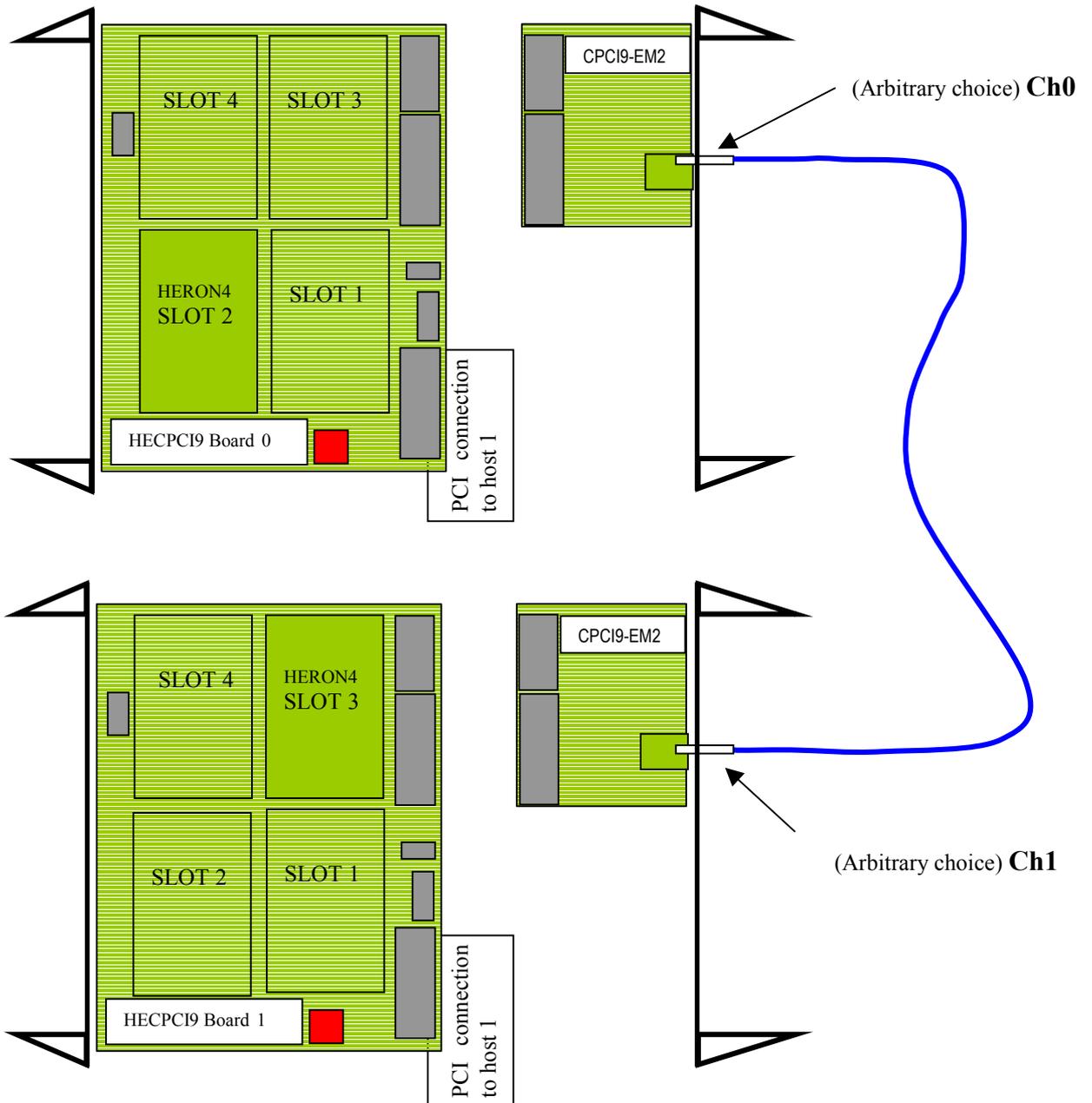
The CPCI9-EM2 can access its internal Heart FIFOs in 32-bit mode, while the serial channels can access the same FIFOs at 125Mbytes/sec.

For example, if a particular CPCI9-EM2 channel is programmed to read data from one timeslot on a heart system that is running at 100MHz, it will be taking data from the heart architecture at 66.6Mbytes/sec. The serial channel will be running at just over half-capacity. If however, a channel is programmed to use two timeslots, it is possible for data to be read from the heart system at 133.3Mbytes/sec. If this data rate were sustained, the FIFOs inside the CPCI9-EM2 would eventually fill. In this scenario, the EM2 may be programmed to either block the incoming data from being put onto the heart system, or to simply lose data when the internal FIFO is full.

## Example configurations

### Example1 – two or more boards in the same host PC booted as one system -- method A

This example shows the simplest use of the CPCI9-EM2, where there are two HECPCI9s in the same host machine. These two boards can be booted separately, routing their own connections to their EM2 modules. There is no need for reset or HSB propagation.



Programming of this functionality is straightforward.

The host resets each of the HECPCI9s and configures each HEART network.

The modules on each board can communicate with each other via the EM2 connections, and can communicate with the Host PC via their PCI interfaces.

In this case both boards can appear in the same network file, and Heartconf (or Server/Loader) can configure both of them at the same time.

Using the Server/Loader is the safest way to configure such a system as it will follow this sequence:-

- Reset Both boards
- Configure FPGA modules on both boards (if any defined in the network file)
- Reset both boards to ensure any FPGA modules are initialised
- Make HEART connections for booting DSPs
- Boot DSPs on both boards
- Re-configure both HEART networks to match the network file
- Use HSB to inform DSPs on both boards that they can start.

This sequence is important, as if one board were to be initialised and allowed to run before the other board was configured, then any data sent could be lost and the results after configuring the second board are undefined.

A network file for the above system could be :-

```
# BD API          Board_type      Board_Id      Device_Id
#-----
# Using API
BD API HEP9A 0 0
BD API HEP9A 1 0
#-----
# Nodes description
# ND  BD_nb  ND_NAME  ND_Type      HERON-ID  filename(s)
#-----
  c6  0      HERONA   ROOT          00000002  board0.out
  c6  1      HERONB   ROOT          0x13      board1.out
# Host interface:
pcif 0      hostA   normal        0x05
pcif 1      hostB   normal        0x15
# HEPC9 inter-board modules, currently EM1-C, EM1 or EM2:
em2  0      em2A    normal        0x06
em2  1      em2B    normal        0x16

#          from:slot  fifo  to:slot  fifo  timeslots
#-----
# Create a connection between host (fifo 0) and the 'C6x (fifo 0). It
uses
# 1 timeslot. The precise timeslot is chosen by Server/Loader or
HeartConf
heart      hostA  0      heronA  0      1
heart      hostB  0      heronB  0      1

# And create a connection back, from the 'C6x (fifo 0) to the host
(fifo 0)
heart      heronA  0      hostA  0      1
```

```

heart      heronB  0      hostB   0      1
# create the connections from the DSP modules to each other via the EM2
heart      heronA  1      em2A    0      1
heart      em2A    0      heronA  1      1
heart      heronB  1      em2B    1      1
heart      em2B    1      heronB  1      1

```

For full details of the syntax refer to the network file description document, but notice the correct connection to FIFO0 on the first EM2 module and FIFO1 on the second. This must reflect where the cable is plugged. Notice also that the cable carries data in both directions.

The system is now ready for use by the application. A Virtual FIFO exists in each direction between the two boards, and can be used in the same way that a virtual FIFO connection is used between modules on the same board.

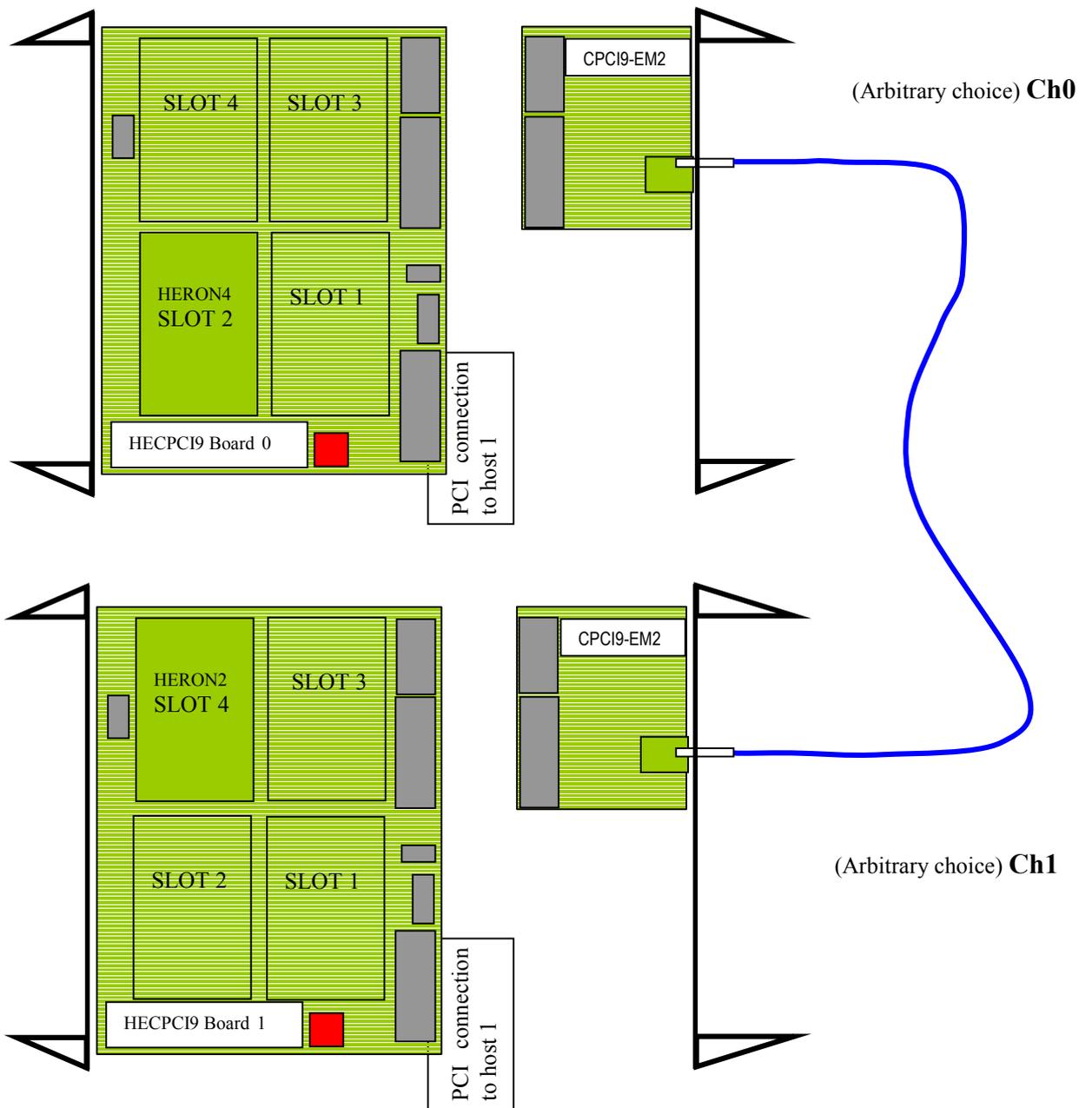
**Example1a – two or more boards in the same host PC booted as one system -- method B**

The hardware supports connecting the reset and HSB across multiple boards. Using this technique it is possible to boot and configure a multi-board system as if it were one board.

Currently (Jan 2003) the Server/Loader and Heartconf does not support this technique, but it will be possible as soon as work on those features are completed.

### Example2 – two or more boards in the same host PC booted separately

This example is a quite simple extension of the example above. The two HECPCI9s are in different host machines and are each booted and their HEART connections configured separately. There is no need for reset or HSB propagation.



Programming of this functionality is straightforward but some care must be taken over when each system is started.

Each host resets its HECPCI9 and configures its HEART network.

Once running the modules on each board can communicate with each other via the EM2 connections, and can communicate with the Host PC via their PCI interfaces.

In this case each boards will have its own network file and Heartconf (or Server/Loader) can configure each of them individually.

Now it is important to design your system correctly to avoid problems that can arise if data is sent to a system before it is booted. The act of booting the “second” half will mean that data is lost (because of the reset) and when that system is started you will not know where the data will start.

Network files for the above system could be :-

```
# BD API          Board_type      Board_Id          Device_Id
#-----
# Using API
BD API HEP9A 0 0
#-----
# Nodes description
# ND  BD_nb  ND_NAME  ND_Type          HERON-ID  filename(s)
#-----
#   c6  0      HERONA   ROOT              00000002  board0.out
# Host interface:
#   pcif 0      hostA   normal            0x05
# HEPC9 inter-board modules, currently EM1-C, EM1 or EM2:
#   em2  0      em2A   normal            0x06

#           from:slot  fifo  to:slot  fifo  timeslots
#-----
# Create a connection between host (fifo 0) and the 'C6x (fifo 0). It
# uses
# 1 timeslot. The precise timeslot is chosen by Server/Loader or
# HeartConf
#   heart      hostA  0      heronA  0      1

# And create a connection back, from the 'C6x (fifo 0) to the host
# (fifo 0)
#   heart      heronA  0      hostA  0      1

# create the connections from the DSP modules to each other via the EM2
#   heart      heronA  1      em2A   0      1
#   heart      em2A   0      heronA  1      1
```

And for the second board:-

```
# BD API          Board_type      Board_Id          Device_Id
#-----
# Using API
BD API HEP9A 1 0
#-----
# Nodes description
```

```

# ND  BD_nb  ND_NAME  ND_Type          HERON-ID  filename(s)
#-----
# c6  1      HERONB   ROOT          0x14      board1.out
# Host interface:
# pcif 1      hostB   normal        0x15
# HEPC9 inter-board modules, currently EM1-C, EM1 or EM2:
# em2  1      em2B    normal        0x16

#      from:slot  fifo  to:slot  fifo  timeslots
#-----
# Create a connection between host (fifo 0) and the 'C6x (fifo 0). It
uses
# 1 timeslot. The precise timeslot is chosen by Server/Loader or
HeartConf
heart      hostB  0      heronB  0      1

# And create a connection back, from the 'C6x (fifo 0) to the host
(fifo 0)
heart      heronB  0      hostB  0      1

# create the connections from the DSP modules to each other via the EM2
heart      heronB  1      em2B  1      1
heart      em2B  1      heronB  1      1

```

For full details of the syntax refer to the network file description document, but notice the correct connection to FIFO0 on the first EM2 module and FIFO1 on the second. This must reflect where the cable is plugged. Notice also that the cable carries data in both directions.

The system is now ready for use by the application. A Virtual FIFO exists in each direction between the two boards, and can be used in the same way that a virtual FIFO connection is used between modules on the same board.

For the synchronisation problem there are many solutions, all of which need to be made part of your application software.

One way could be that one end does not start until it receives a particular pattern, and that end is booted first. Then the other end when booted (last) sends that pattern to start the system.

### **Example3 – two or more boards in different host PCs booted separately**

This example is the same in every respect as example 2.

### **Example4 – two or more boards in different host PCs booted as one system**

The hardware supports connecting the reset and HSB across multiple boards. Using this technique it is possible to boot and configure a multi-board system as if it were one board.

Currently the Server/Loader and Heartconf do not support this technique, but it will be possible as soon as work on those features is completed.

### **Example5 – more than two boards in a system booted separately**

This example is simply an extension of examples 2 or 3. Now the synchronisation built into your application must arrange that the system boots correctly across all of the boards.

The following sections attempt to cover all likely problems. Please check through this section before contacting technical support.

### **Hardware**

If the Hardware has been installed according to the Instructions there is very little that can be wrong.

- Has the “DONE” LED gone out?
- Are the serial cables connected in the way that the network topology expects?
- Are the board numbers correctly selected on each module carrier?
- Are all board carriers powered up?

### **Software**

As long as the software has been installed using the installation program supplied on the HUNT ENGINEERING CD, there should be little problem with the software installation.

If you have problems then return to one of the example programs supplied with the system.

HUNT ENGINEERING has performed testing on its products to ensure that it is possible to comply with the European CE marking directives. The CPCI9-EM2 cannot be CE marked as it is a component in a system, but as long as the following recommendations are followed, a system containing the CPCI9-EM2 could be CE marked.

The immense flexibility of the HUNT ENGINEERING product range means that individual systems should be marked in accordance with the directives after assembly.

1. The host computer or housing in which the CPCI9-EM2 is installed is properly assembled with EMC and LVD in mind and ideally should itself carry the CE mark.

2. Any cabling between boards or peripherals is either entirely inside the case of the host computer, or has been assembled and tested in accordance with the directives.

The CPCI9-EM2 serial I/Os ARE protected against Static discharge, so if the cabling does exit the case, there is suitable protection already fitted.

HUNT ENGINEERING are able to perform system integration in accordance with these directives if you are unsure of how to achieve compliance yourself.

## Technical Support

---

Technical support for HUNT ENGINEERING products should first be obtained from the comprehensive Support section <http://www.hunteng.co.uk/support/index.htm> on the HUNT ENGINEERING web site. This includes FAQs, latest product, software and documentation updates etc. Or contact your local supplier - if you are unsure of details please refer to <http://www.hunteng.co.uk> for the list of current re-sellers.

HUNT ENGINEERING technical support can be contacted by emailing [support@hunteng.co.uk](mailto:support@hunteng.co.uk), calling the direct support telephone number +44 (0)1278 760775, or by calling the general number +44 (0)1278 760188 and choosing the technical support option.

## Appendix 1 – HERON Serial Bus Commands

---

### Module address

The CPC19-EM2 is configured to respond to Heron Serial Bus (HSB) commands addressed to it using the combination of the Board number and slot number that the module is fitted to. In this way all modules can be uniquely addressed in the same system. The HSB address is a 7-bit address that is formed by the bottom three bits of the slot number, with the 4 bits from the board number switch forming the top 4 bits of the seven.

e.g. on board number 1, slot 6 the address would be (board number<<3) || slot[2:0] which is 0x0E.

The CPC19-EM2 can respond to two different types of serial bus commands:

### Module Query

The CPC19-EM2 can receive a message requesting its module type:-

Master to EM2 module

module type query (01)-->address of requestor

It will then send a reply as follows:-

EM2 module to "original master"

module query response(02)-->module address (from)-->module type (05)

-->family number(02)-->option

### Internal Register Program

The Configuration transaction will be:-

Master to EM2 module

Register write (07)-->address of register--> data (--> address of register2--> data2)...

There is no reply to register program messages.

## Appendix 2 – Internal Register Description

---

The internal registers of the CPC19-EM2 are programmed via the Heron Serial Bus (HSB).

### Receive channel timeslot registers

Each of the six receive channels of the CPC19-EM2 may be individually programmed to output received data into any timeslot of the local Heart system. Only the least significant six bit are used, bit 0 is set to use timeslot 0, bit 1 for timeslot 1, up to bit 5 for timeslot 5. The use of multiple timeslots is allowed, although the use of more than two timeslots has no benefit as the serial transceivers cannot receive data fast enough to fully utilise the bandwidth of two timeslots. These bits are cleared at reset, and by an access to the 'ZAP Heart' register.

Address 0: Rx channel0 Timeslot Register [5..0].

Address 1: Rx channel1 Timeslot Register [5..0].

Address 2: Rx channel2 Timeslot Register [5..0].

Address 3: Rx channel3 Timeslot Register [5..0].

Address 4: Rx channel4 Timeslot Register [5..0].

Address 5: Rx channel5 Timeslot Register [5..0].

### Transmit channel timeslot registers

Each of the six transmit channels of the CPC19-EM2 may be individually programmed to read data from any timeslot of the local Heart system, for transmission. Only the least significant six bit are used, bit 0 is set to use timeslot 0, bit 1 for timeslot 1, up to bit 5 for timeslot 5. These bits are cleared at reset, and by an access to the 'ZAP Heart' register.

Additionally, bit 6 of each register is used to program the 'blocking' mode of the transmit channels. By default, this bit is set to "0", which means that data is blocked in the heart system on any timeslot that is sending to this channel, when the transmit FIFO becomes full. For certain applications, this bit may be programmed to "1", which allows data to continue flowing in those timeslots even when the transmit FIFO is full. In this mode, data may be lost, but soon as the FIFO can accept more data, the new data from the timeslots is more recent.

Address 6: Tx channel0 Timeslot Register [6..0].

Address 7: Tx channel1 Timeslot Register [6..0].

Address 8: Tx channel2 Timeslot Register [6..0].

Address 9: Tx channel3 Timeslot Register [6..0].

Address 10: Tx channel4 Timeslot Register [6..0].

Address 11: Tx channel5 Timeslot Register [6..0].

## **Receive channel reset-with-UMI registers**

Each of the six receive channels may have their data FIFOs reset, (flushed), by an active low assertion of any of the 4 UMI signals. (Uncommitted Module Interconnects). This feature may be useful when the non-blocking mode of the transmit channel is used. Only the least significant 4 bits are used. Bit 0 is set to make UMI0 reset the FIFO, bit 1 for UMI1, bit 2 for UMI2, and bit 3 for UMI3. Multiple bits may be set, so that more than one UMI signal may reset the FIFO. This register is cleared at reset.

Address 12: Channel 0 Receive FIFO reset-with-UMI [3..0]

Address 13: Channel 1 Receive FIFO reset-with-UMI [3..0]

Address 14: Channel 2 Receive FIFO reset-with-UMI [3..0]

Address 15: Channel 3 Receive FIFO reset-with-UMI [3..0]

Address 16: Channel 4 Receive FIFO reset-with-UMI [3..0]

Address 17: Channel 5 Receive FIFO reset-with-UMI [3..0]

## **Transmit channel reset-with UMI registers**

Each of the six transmit channels may have their data FIFOs reset (flushed), by an active low assertion of any of the 4 UMI signals. (Uncommitted Module Interconnects). This feature may be useful when the non-blocking mode of the transmit channel is used. Only the least significant 4 bits are used. Bit 0 is set to make UMI0 reset the FIFO, bit 1 for UMI1, bit 2 for UMI2, and bit 3 for UMI3. Multiple bits may be set, so that more than one UMI signal may reset the FIFO. This register is cleared at reset.

Address 18: Channel 0 Transmit FIFO reset-with-UMI [3..0]

Address 19: Channel 1 Transmit FIFO reset-with-UMI [3..0]

Address 20: Channel 2 Transmit FIFO reset-with-UMI [3..0]

Address 21: Channel 3 Transmit FIFO reset-with-UMI [3..0]

Address 22: Channel 4 Transmit FIFO reset-with-UMI [3..0]

Address 23: Channel 5 Transmit FIFO reset-with-UMI [3..0]

## **ZAP Heart Register**

This register is used as an easy means of clearing all timeslot bits that may be programmed in the Transmit channel timeslot registers, and Receive channel timeslot registers. No data bits are used, a write access alone clears all 12 registers.

Address 24: Zap Heart Control.

## **Reset propagation register.**

This six-bit register to determine which receive channels propagate a received reset condition from a remote carrier, onto the UDP Reset of the carrier that the CPC19-EM2 is

plugged into, which will reset the whole carrier board. This function will only normally be used on a slave or remote carrier that is NOT being reset by a host process, and this register will therefore normally be programmed from the 'master' carrier, over one of the serial links. Only the least significant 6 bits are used. Setting bit0 allows a reset to propagate from channel 0..... up to bit 5 to allow a reset to propagate from channel 5.

Naturally, these bits are NOT cleared at reset!!! They are just initialised to '0' at power-up.

Address 25: Reset\_propagation [5..0];

### **HSB propagation registers.**

These 1-bit registers are used to control whether or not a transmit channel will send HSB messages over its serial link to another CPCI9-EM2. Only bit 0 is used; setting bit 0 will allow HSB propagation. These bits are NOT cleared at reset!!! They are just initialised to '0' at power-up. Additionally, all six registers may be cleared using the Zap HSB Propagation Control.

Address 26: Transmit channel 0 HSB propagation.

Address 27: Transmit channel 1 HSB propagation.

Address 28: Transmit channel 2 HSB propagation.

Address 29: Transmit channel 3 HSB propagation.

Address 30: Transmit channel 4 HSB propagation.

Address 31: Transmit channel 5 HSB propagation.

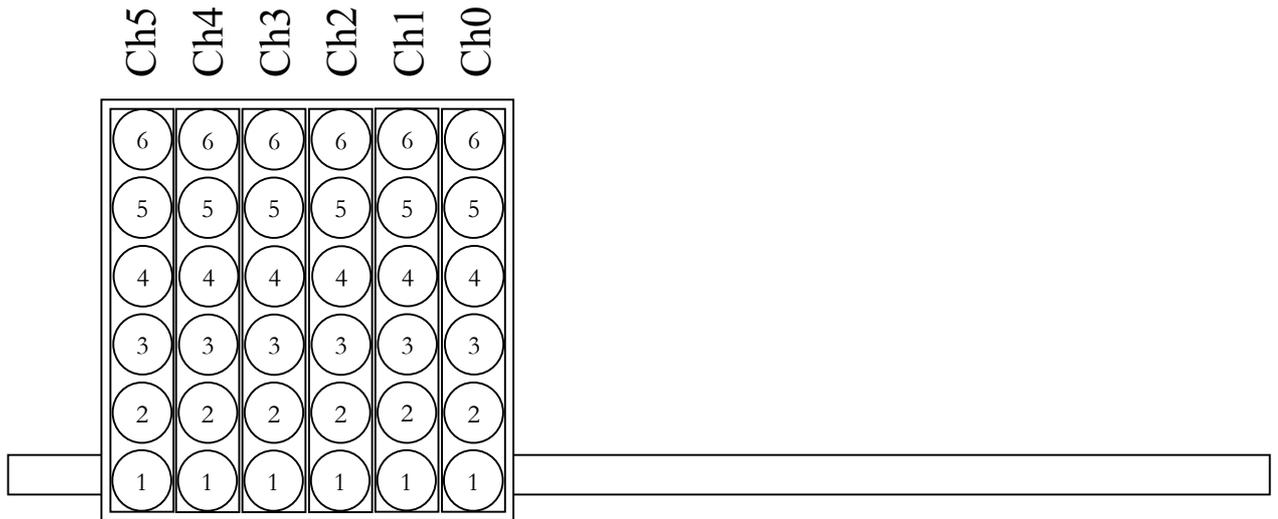
### **ZAP HSB propagation control.**

This register is used as an easy means of clearing all six HSB Propagation Registers. No data bits are used, a write access alone clears all 6 registers.

Address32: Zap HSB Propagation Control

## Appendix 3 – Gigabit serial I/O Connector pinout

The serial I/O connectors are pinned out so that the corresponding rx and tx pairs for one channel are housed in the same connector shell. The figure below shows the view looking onto the right-hand edge of the CPC19-EM2.



The pinout for each serial channel is as follows:

Pin number	Function.
1	Tx +
2	Tx -
3	Polarising pin
4	GND
5	Rx -
6	Rx +