

# TEMU

## ***GRSPW1 Model Manual***

Alberto Ferrazzi

Version 1.0, 2018-04-20

# Table of Contents

1. Introduction . . . . .

2. Configuration . . . . .

3. Attributes . . . . .

3.1. Properties . . . . .

3.2. Interfaces . . . . .

3.3. Ports . . . . .

4. Limitations . . . . .

5. Examples . . . . .

1

1

1

1

3

3

3

3

3

*Table 1. Record of Changes*

| Rev | Date       | Author | Note             |
|-----|------------|--------|------------------|
| 1.0 | 2018-04-20 | AF     | Initial version. |

## 1. Introduction

The GRSPW1 is part of the GRLIB IP library. It is available in libTEMUGrspw1.so.

## 2. Configuration

To work correctly, the device should be connected to an interrupt controller, the memory and another SpaceWire device.

There are several configuration parameters in the GrSpw1 device, summarized in the following table:

| Name                         | Description   |
|------------------------------|---|
| config.infiniteSpeed         | With this set, messages are sent immediately instead of being scheduled for the future based on the message length. This is the default option. |
| config.transmitter.frequency | Specify the SpaceWire transmitter frequency in Hz. Affects transfer speed when infinite speed is disabled.                                      |
| config.transmitter.dataRate  | SpaceWire port datarate: 1=single, 2=double, etc. Affects transfer speed when infinite speed is disabled.                                       |
| config.interrupt             | Influences the interrupt that is raised with the IRQ controller (setting this property also updates the APB PnP info).                          |
| config.realCrcCheck          | Set to use real crc check instead of packet crc flags. Real crc costs in terms of performance.  |

## 3. Attributes

### 3.1. Properties

| Name                 | Type    | Description                              |
|----------------------|---------|--|
| config.infiniteSpeed | uint8_t | Set to use infinite speed for transfers. |
| config.interrupt     | uint8_t | The interrupt index                      |

| Name                         | Type             | Description   |
|------------------------------|------------------|---|
| config.realCrcCheck          | uint8_t          | Set to use real crc check instead of packet crc flags |
| config.transmitter.dataRate  | uint8_t          | SpaceWire port datarate:<br>1=single, 2=double,...    |
| config.transmitter.frequency | uint32_t         | SpaceWire transmitter frequency in Hz                 |
| internal.linkState           | int32_t          | Link state  |
| internal.txDAddr             | uint32_t         | Data address for the scheduled dma engine transfer    |
| internal.txDLength           | uint32_t         | Data length for the scheduled dma engine transfer     |
| internal.txFlags             | uint32_t         | Flags for the scheduled dma engine transfer           |
| internal.txHAddr             | uint32_t         | Header address for the scheduled dma engine transfer  |
| internal.txType              | uint8_t          | Scheduled transmission type (dma engine/rmap)         |
| internal.uplinkNsPerByte     | uint32_t         | Transmitter speed                                     |
| irqCtrl                      | iref / <unknown> | Irq controller  |
| memory                       | iref / <unknown> | Memory used for DMA accesses                          |
| object.timeSource            | object           | Time source object (a cpu or machine object)          |
| pnp.bar                      | uint32_t         | Pnp BAR   |
| pnp.config                   | uint32_t         | Pnp configuration                                     |
| regs.clockDiv                | uint32_t         | Clock Divisor register                                |
| regs.control                 | uint32_t         | Control register                                      |
| regs.destKey                 | uint32_t         | Destination Key register                              |
| regs.dmaControl              | uint32_t         | Dma control registers                                 |
| regs.dmaRxDescTableAddr      | uint32_t         | Dma receive descriptor table address registers        |
| regs.dmaRxMaxLen             | uint32_t         | Dma rx maximum length registers                       |
| regs.dmaTxDescTableAddr      | uint32_t         | Dma transmit descriptor table address registers       |
| regs.nodeAddress             | uint32_t         | Node address register                                 |

| Name              | Type                   | Description                             |
|-------------------|------------------------|---|
| regs.statusIrqSrc | uint32_t               | Status / Interrupt-source register      |
| regs.time         | uint32_t               | Time register                           |
| spwUplink         | [2 x iref / <unknown>] | SpaceWire devices connected to the port |

## 3.2. Interfaces

| Name           | Type           | Description                |
|----------------|----------------|----------------------------|
| ApbIface       | ApbIface       | Apb interface              |
| DeviceIface    | DeviceIface    | Device interface           |
| MemAccessIface | MemAccessIface | Memory Access Interface    |
| ResetIface     | ResetIface     |                            |
| SpwPortIface   | SpwPortIface   | SpaceWire ports interfaces |

## 3.3. Ports

| Prop | Iface | Description |
|------|-------|-------------|
| -    | -     | -           |

## 4. Limitations

The following limitations/deviations from real hardware are known to exist with this model:

- No spill is currently not implemented
- Although the device already provides two spacewire ports, dual port is not yet implemented. This correspond to a device where the port VHDL parameter is set to 1. Therefore, in the control register, PO will be 0 and PS / NP bit are not available. Let us know if you need this feature implemented.
- The link interface currently effectively uses only ErrorReset, Ready, Connecting and Run states. Therefore, those are the only values that will be visible on the status register.
- RMAPEN signals not available

## 5. Examples

This example shows how to create two Grspw1 devices and connect them.

```
import BusModels
import TEMUGrspw1
object-create class=Grspw1 name=grspw0
object-create class=Grspw1 name=grspw1
spw-connect port1=grspw0:SpwPortIface[0] port2=grspw1:SpwPortIface[0]
```