T-EMU: MEC Device Model Manual

Prepared	Checked
Mattias Holm Technical Manager	Dan Søren Nielsen QA Manager
Approved	
Michela Alberti General Manager	

Record of Changes

Author	Description	Rev	Date
Mattias Holm	Initial Version	1.0	2015-03-01

Table of Contents

1. Introduction

The MEC (Memory Controller) device is used with the ERC32 processor. The device provides two UARTs, two timers and an interrupt interface. The interrupt interface allows for the raising and lowering of the 5 external interrupts provided by the ERC32 (IRQ 0 through (including) 4). The device model takes care of converting these to the relevant internal interrupts (i.e. SPARC IRQs 2,3,10,11 and 14). When raising (or lowering) a MEC interrupt you need to use numbers 0-4.

2. Configuration

2.1. Interrupt Delivery

The property irqControl should be connected to the device which the MEC raises interrupts on, this is normally a CPU object. The connection should be made to the CPU-object's interface of type IrqIface. Note that the CPU must support interrupts 1 through 15, this is in general case correct for SPARC based processors, but other CPUs may not be compatible.

2.2. UART Connections

Two serial interfaces exist, the UartAIface and the UartBIface, these can be connected to in order to receive data from remote serial port terminals (i.e. this is the RX direction). The uarta and uartb properties can be used to connect the TX direction of the UARTs.

2.3. Infinite UART Speed

Set config.infiniteUartSpeed to nonzero to enable infinite speed on the Tx channels. With infinite speed, a written byte is immediately forwarded to the destination device, with limited UART speed (the variable being zero) the timing due to UART scaler bits (upper 8 bits of the MecCtrlReg) will be simulated, leading to realistic byte rates over the serial port device. Note that individual bits are not transmitted only the bytes.

3. Limitations

The following deviations from real hardware are known to exist, if you need the correct behaviour (or simulation of it, contact us for more info):

- The UARTs do not support external (watchdog) clocks.
- The UARTs do not support parity, framing errors, break signals or stop bit configuration (although the transmission times are computed based on stop bit count and parity bit embedding).
- Write protection registers have no effect
- Timer values are lazily computed on reads, the content in the case a timer is disabled is estimated on disabling time. This is in principle correct. However, the prescaler counter write has no effect, only the reload value has an effect when written. This may cause an offset of 1024 cycles when re-enabling a timer.