

T-EMU: MEC Device Model Manual



Prepared

Mattias Holm
Technical Manager

Approved

Michela Alberti
General Manager

Checked

Dan Søren Nielsen
QA Manager



Record of Changes

Author	Description	Rev	Date
Mattias Holm	Add note about IRQ behaviour.	1.2	2018-08-29
Mattias Holm	Auto gen tables	1.1	2016-05-12
Mattias Holm	Initial Version	1.0	2015-03-01

Table of Contents

1. Introduction	2
2. Configuration	2
2.1. Interrupt Delivery	2
2.2. UART Connections	2
2.3. Infinite UART Speed	3
3. Attributes	3
3.1. Properties	3
3.2. Interfaces	4
3.3. Ports	4
4. Notes	5
5. Limitations	5

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. Attributes

3.1. Properties

Name	Type	Description
<code>accessProtSegment1Base</code>	<code>uint32_t</code>	
<code>accessProtSegment1End</code>	<code>uint32_t</code>	
<code>accessProtSegment2Base</code>	<code>uint32_t</code>	
<code>accessProtSegment2End</code>	<code>uint32_t</code>	
<code>config.infiniteUartSpeed</code>	<code>uint32_t</code>	
<code>cpu</code>	<code>iref / <unknown></code>	
<code>errorAndResetStatus</code>	<code>uint32_t</code>	
<code>failingAddr</code>	<code>uint32_t</code>	
<code>gpiConfig</code>	<code>uint32_t</code>	
<code>gpiData</code>	<code>uint32_t</code>	
<code>gptCounter</code>	<code>uint32_t</code>	
<code>gptCounterProgramReg</code>	<code>uint32_t</code>	
<code>gptScaler</code>	<code>uint32_t</code>	
<code>gptScalerProgramReg</code>	<code>uint32_t</code>	
<code>ioConfig</code>	<code>uint32_t</code>	
<code>irqClear</code>	<code>uint32_t</code>	
<code>irqControl</code>	<code>iref / <unknown></code>	
<code>irqForce</code>	<code>uint32_t</code>	
<code>irqMask</code>	<code>uint32_t</code>	
<code>irqPending</code>	<code>uint32_t</code>	
<code>irqShape</code>	<code>uint32_t</code>	
<code>mecCtrl</code>	<code>uint32_t</code>	
<code>memoryConfig</code>	<code>uint32_t</code>	
<code>object.timeSource</code>	<code>object</code>	Time source object (a cpu or machine object)
<code>outSignals</code>	<code>[8 x iref / Signalface]</code>	



Name	Type	Description
powerDown	uint32_t	
rtcCounter	uint32_t	
rtcCounterProgramReg	uint32_t	
rtcScaler	uint32_t	
rtcScalerProgramReg	uint32_t	
softwareReset	uint32_t	
systemFaultStatus	uint32_t	
testControl	uint32_t	
timerControl	uint32_t	
uartChanARxTx	uint32_t	
uartChanBRxTx	uint32_t	
uartStatus	uint32_t	
uarta	iref / <unknown>	
uartb	iref / <unknown>	
waitStateConfig	uint32_t	
wdogProgAndTimeoutAck	uint32_t	
wdogTrapDoorSet	uint32_t	

3.2. Interfaces

Name	Type	Description
DeviceIface	DeviceIface	
IrqClientIface	IrqClientIface	
IrqIface	IrqIface	
MemAccessIface	MemAccessIface	
ResetIface	ResetIface	
SignalIface	SignalIface	Incomming signals
UartAIface	SerialIface	
UartBIface	SerialIface	

3.3. Ports

Prop	Iface	Description
irqControl	IrqClientIface	uart a
uarta	UartAIface	uart a
uartb	UartBIface	uart b



4. Notes

The MEC sets the interrupt pending register bit when an interrupt is raised even when the interrupt is masked. The mask is only applied when evaluating whether to raise an IRQ with the CPU.

5. 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.