



Datasheet

RTC_{MODULE}

Version 1.0.1

INICORE INC.
5600 Mowry School Road
Suite 180
Newark, CA 94560
t: 510 445 1529 f: 510 656 0995 e: info@inicare.com
www.inicare.com

COPYRIGHT © 2003 INICORE INC.

Table of Contents

1 INTRODUCTION.....	4
1.1 Features.....	4
1.2 Implementation Features.....	4
1.2.1 Deliverables.....	5
1.3 Block Diagram.....	5
1.4 Block Description.....	6
1.4.1 System Bus Interface (UPI).....	6
1.4.2 Time Keeper.....	6
1.4.3 Alarm Register.....	6
1.4.4 Interrupt Controller.....	6
2 SIGNAL DESCRIPTIONS.....	7
2.1 Symbol.....	7
2.2 General Inputs.....	7
2.3 Local Bus.....	8
2.4 External Reference Clock.....	8
3 PROGRAMMING MODEL.....	9
3.1 Memory Mapping.....	9
3.2 Register Description.....	10
3.2.1 Configuration Registers.....	10
3.2.2 Time Keeping Register.....	10
3.2.3 Alarm 1/2 Register.....	11
3.2.4 Interrupt Register.....	12
4 TIMING DIAGRAM.....	13
4.1 Local Bus Interface.....	13

Table of Figures

Figure 1: Block Diagram RTCmodule.....	5
Figure 2: Symbol with I/Os.....	7
Figure 3: Local Bus Interface Timing.....	13

1 Introduction

The RTCmodule implements the standard functionality of a Real Time Clock. With a synchronous system interface, the module can be integrated in a wide range of systems, from small CPU to a ARM AMBA system.

1.1 Features

Following special features are available:

- ◆ External or internal reference time
- ◆ Counts milliseconds, seconds, minutes, and hours of the day
- ◆ Counts days, months, and years
 - Automatic end-of-month and leap year recognition
- ◆ 2 alarm interrupts
- ◆ Second, minutes, hour, day, month, and year over-roll interrupts
- ◆ Time and date are counted in BCD format.
- ◆ The time is in 24-hour format
- ◆ Special test mode to increase test coverage and accelerate system test
- ◆ Synchronous bus interface
 - Zero wait-states
 - Supports system bus' such as AMBA APB version 2.0
- ◆ Technology independent

1.2 Implementation Features

Upon synthesis, several configurations are available to optimize the core for the target application:

- ◆ Number of timer alarms
- ◆ CPU readback. Configuration value readback can be disabled to minimize gate count.

1.2.1 Deliverables

- VHDL or Verilog RTL source code
- Simulation testbench
- Timing constraints file
- Synthesis script
- User Guide

1.3 Block Diagram

The top-level block diagram of the RTCmodule is shown in the figure below:

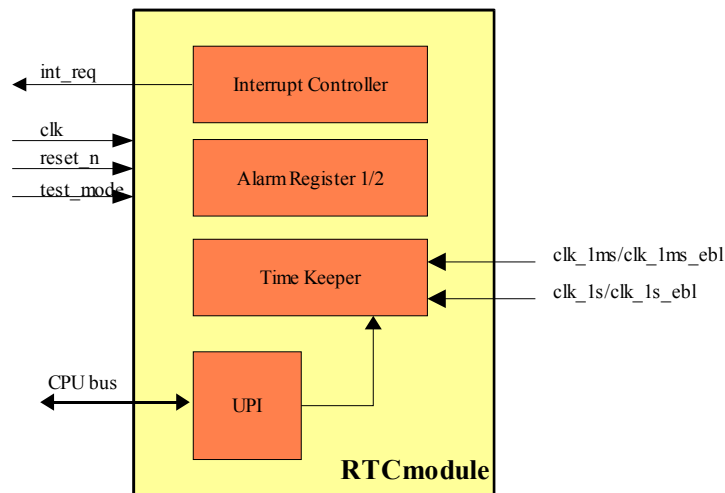


Figure 1: Block Diagram RTCmodule

1.4 Block Description

1.4.1 System Bus Interface (UPI)

A fully synchronous single clock cycle bus interface provides direct access to all local register resources. See the memory map for detailed description of all registers.

1.4.2 Time Keeper

Using either the system clock or an external time reference, the time keeper keeps track of the current time.

1.4.3 Alarm Register

Using the alarm register, a preprogrammed system wake-up event can be programmed allowing the CPU to go into a power-down mode.

1.4.4 Interrupt Controller

The interrupt controller bundles all local interrupt sources together and provides one interrupt request line to the main system.

2 Signal Descriptions

The following paragraph lists the input and output ports of the RTCmodule. Please refer to the chapter 4 for a definition of the interface timings.

2.1 Symbol

The figure below shows all interfaces of the RTCmodule.

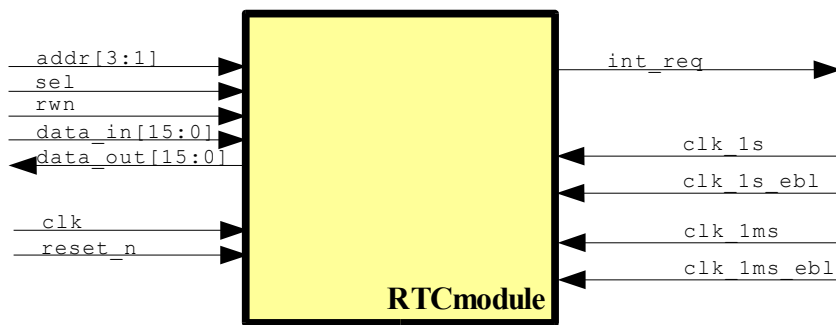


Figure 2: Symbol with I/Os

2.2 General Inputs

The module is initialized with one asynchronous, active low reset input. All registers are clocked with the system clock.

Pin Name	Type	Description
clk	in	System clock
reset_n	in	Asynchronous system reset, active low
test_mode	in	Scan test mode, to accelerate test and increase test coverage, the core can be put into a special test mode.

2.3 Local Bus

A synchronous zero wait-states system bus interface provides access to all status and configuration registers. The timing diagram is shown in chapter 4.

<i>Pin Name</i>	<i>Type</i>	<i>Description</i>
addr[3:1]	in	Address bus input
data_write[15:0]	in	Data bus input
data_read[15:0]	out	Data bus output
sel	in	Module chip select, active high
rwn	in	Read/write control signal '0': Write '1': Read
int_req	out	Interrupt request, active high

2.4 External Reference Clock

The RTCmodule uses either the system clock as a reference time, or it can use an external reference clock.

<i>Pin Name</i>	<i>Type</i>	<i>Description</i>
clk_1s	in	1Hz reference clock
clk_1s_ebl	in	1Hz reference clock enable
clk_1ms	in	1kHz reference clock
clk_1ms_ebl	in	1kHz reference clock enable

3 Programming Model

3.1 Memory Mapping

The table below shows the entire memory mapping of the RTCmodule. A detailed description of the functionalities of these registers can be found in the next paragraph.

Address	Name	R/W	Description
+0x00	rtc_config	r/W	Configuration register
+0x01	rtc_time_low	R/W	Time, low
+0x02	rtc_time_high	R/W	Time, high
+0x03	rtc_date_low	R/W	Date, low
+0x04	rtc_date_high	R/W	Date, high
+0x05	rtc_alarm1_time_low	r/W	Alarm 1 Time, low
+0x06	rtc_alarm1_time_high	r/W	Alarm 1 Time, high
+0x07	rtc_alarm1_date_low	r/W	Alarm 1 Date, low
+0x08	rtc_alarm1_date_high	r/W	Alarm 1 Date, high
+0x09	rtc_alarm2_time_low	r/W	Alarm 2 Time, low
+0x0A	rtc_alarm2_time_high	r/W	Alarm 2 Time, high
+0x0B	rtc_alarm2_date_low	r/W	Alarm 2 Date, low
+0x0C	rtc_alarm2_date_high	r/W	Alarm 2 Date, high
+0x0D	rtc_int_status	R/W	Interrupt status register
+0x0E	rtc_int_ebl	r/W	Interrupt enable register

Following acronyms are used in the table above:

- r: Data readback
- R: Data read
- W: Data write

3.2 Register Description

The on-chip real-time clock generates the system time reference. Upon system reset, the RTC module has to be reinitialized.

3.2.1 Configuration Registers

To prevent overwriting of time and alarm registers, they have an individual write protection.

Address	Name	R/W	Description
+0x00	rtc_config	R/W	RTC configuration [0]: Time and calendar registers write enable '0': Registers write protected '1': Registers write enabled [1]: Alarm 1 write enable '0': Registers write protected '1': Registers write enabled [2]: Alarm 2 write enable '0': Registers write protected '1': Registers write enabled

3.2.2 Time Keeping Register

To set a time, the the time and calendar register write enable in the rtc_config register has to be set to '1'. While this bit is '1', the time keeping is stopped.

The time keeping registers are not affected by the system reset. They continue to keep track of time.

Reading the time and data registers, the current time is shown.

Address	Name	R/W	Description
+0x01	rtc_time_low	R/W	Time register, low word [7:0]: Seconds (0-59) [15:8]: Minutes (0-59)
+0x02	rtc_time_high	R/W	Time register, high word [7:0]: Hours (0-23)
+0x03	rtc_date_low	R/W	Calendar register, low word [7:0]: Day (1-31) [15:8]: Month (1-12)
+0x04	rtc_date_high	R/W	Calendar register, high word [15:0]: Year (range 2000 – 2099)

3.2.3 Alarm 1/2 Register

To set an alarm time, the respective alarm register write enable in the `rtc_config` register has to be set to '1'.

The alarm registers are not affected by the system reset. They continue to keep their previous value.

Address	Name	R/W	Description
+0x05	<code>rtc_alarm1_time_low</code>	r/W	Time alarm 1 register, low word [7:0]: Seconds (0-59) [15:8]: Minutes (0-59)
+0x06	<code>rtc_alarm1_time_high</code>	r/W	Time alarm 1 register, high word [7:0]: Hours (0-23)
+0x07	<code>rtc_alarm1_date_low</code>	r/W	Calendar alarm 1 register, low word [7:0]: Day (1-31) [15:8]: Month (1-12)
+0x08	<code>rtc_alarm1_date_high</code>	r/W	Calendar alarm 1 register, high word [15:0]: Year (range 2000 – 2099)
+0x09	<code>rtc_alarm2_time_low</code>	r/W	Time alarm 2 register, low word
+0x0A	<code>rtc_alarm2_time_high</code>	r/W	Time alarm 2 register, high word
+0x0B	<code>rtc_alarm2_date_low</code>	r/W	Calendar alarm 2 register, low word
+0x0C	<code>rtc_alarm2_date_high</code>	r/W	Calendar alarm 2 register, high word

3.2.4 Interrupt Register

Local interrupt status and enable registers are used to bundle several interrupt sources of the core together and control when the external interrupt request is set.

The interrupt request signal (int_req) is only set if its respective interrupt enable flag and the interrupt status register are set.

Acknowledging an interrupt is done by writing a '1' to its respective interrupt status flag.

Address	Name	R/W	Description
+0x0D	rtc_int_status	R/W	Interrupt status register [0]: Second rollover interrupt [1]: Minute rollover interrupt [2]: Hour rollover interrupt [3]: Day rollover interrupt [4]: Month rollover interrupt [6]: Year rollover interrupt [7]: Alarm 1 interrupt [8]: Alarm 2 interrupt
+0x0E	rtc_int_ebl	r/W	Interrupt enable register [0]: Second rollover interrupt enable [1]: Minute rollover interrupt enable [2]: Hour rollover interrupt enable [3]: Day rollover interrupt enable [4]: Month rollover interrupt enable [6]: Year rollover interrupt enable [7]: Alarm 1 interrupt enable [8]: Alarm 2 interrupt enable

4 Timing Diagram

Timing numbers depend on the chosen device, synthesis options, place&route constraints. Typical numbers will be provided.

4.1 Local Bus Interface

The microprocessor interface is designed to operate on a full synchronous bus with zero wait-states. The internal registers are selected using `cs_n`, `wr_n` or `rd_n` respectively and `addr`. The selected register is written on the rising edge of the system clock. `data_out` is asynchronously generated.

Following figures shows the interface timing diagram.

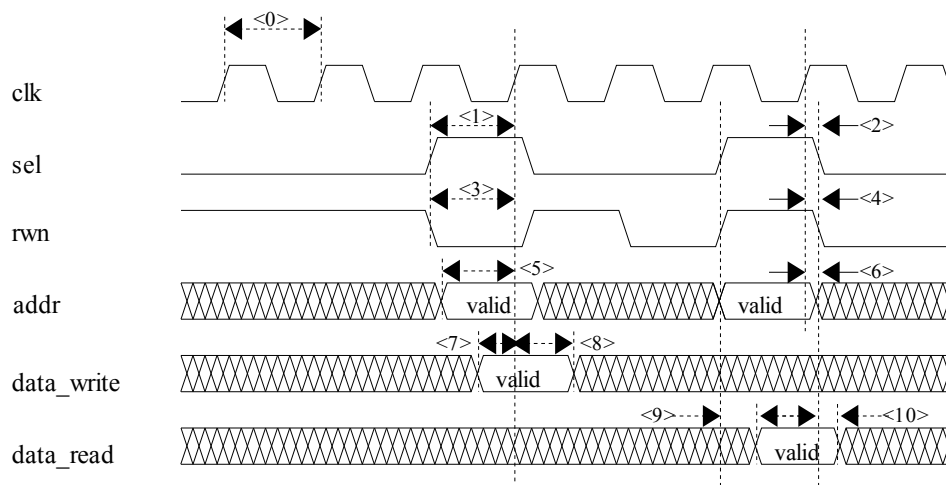


Figure 3: Local Bus Interface Timing

The timing numbers are technology dependent and can only be given once a target technology is selected.

Time nr	t_{min} [ns]	t_{typ} [ns]	t_{max} [ns]	Comment
<0>	tbd			Clock period
<1>		tbd		sel setup time
<2>		0		sel hold time
<3>		tbd		rwn setup time
<4>		0		rwn hold time
<5>		tbd		addr setup time
<6>		0		addr hold time
<7>		tbd		data_write setup time
<8>		0		data_write hold time
<9>		tbd		data_read valid after sel, rwn, addr valid
<10>		0		data_read hold time after sel, rwn, addr invalid



About Inicore

- ◆ FPGA and ASIC Design
- ◆ Easy-to-use IP Cores
- ◆ System-on-Chip Solutions
- ◆ Consulting Services
- ◆ ASIC to FPGA Migration
- ◆ Obsolete ASIC Replacements

Inicore is an experienced system design house providing FPGA / ASIC and SoC design services. The company's expertise in architecture, intellectual property, methodology and tool handling provides a complete design environment that helps customers shorten their design cycle and speed time to market. Our offering covers feasibility study, concept analysis, architecture definition, code generation and implementation. When ready, we deliver you a FPGA or take your design to an ASIC provider, whatever is more suitable for your unique solution.

Customer Advantages

We offer one-stop shopping for everything from the specifications to the chip or module solution. Our experience and fast turnaround time reduces your development costs and increases your returns from the market. Your system is not limited by the level of expertise and standard chip solutions you happen to have in-house. Achieve market success by differentiating and optimizing your product. Reusability builds the basis for further developments in the ever-decreasing product life cycle.

Visit us @ www.inicore.com

INICORE, INC. has made every attempt to ensure that the information in this document is accurate and complete. However, INICORE, INC. assumes no responsibility for any errors, omissions, or for any consequences resulting from the information included in this document or the equipments it accompanies. INICORE, INC. reserves the right to make changes in its products and specifications at any time without notice.

© 2003 INICORE, INC. All rights reserved.