rfDatetime

© 2023 Robert Finch

Table of Contents

[Overview 3](#_Toc128457950)

[Features 3](#_Toc128457951)

[Clocks 3](#_Toc128457952)

[Config Space 4](#_Toc128457953)

[Registers 5](#_Toc128457954)

[Time Register (offset 00h) 6](#_Toc128457955)

[Date-time Register (offset 04h) 6](#_Toc128457956)

[Alarm Register (offset 08h) 6](#_Toc128457957)

[Alarm Register (offset 10h) 6](#_Toc128457958)

[Control Register (offset 14h) 6](#_Toc128457959)

[Snapshot Register (offset 18h) 7](#_Toc128457960)

[I/O Ports 9](#_Toc128457961)

[WISHBONE Compatibility Datasheet 11](#_Toc128457962)

# Overview

Datetime is a date and time keeping core.

# Features

* 32-bit BCD format
* Mars timekeeping option
* programmable time-of-day frequency (50/60/100Hz)
* external time-of-day clock required

# Clocks

The Datetime core uses independent bus and time-of-day (tod) clocks. The tod clock is run through a two-stage synchronizer and edge detector before being used to increment the time and date.

# Config Space

A 256-byte config space is supported. Most of the config space is unused. The only configuration is for the I/O address of the register set and the interrupt line.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Regno | Width | R/W | Moniker | Description |  |  |
| 000 | 32 | RO | REG\_ID | Vendor and device ID |  |  |
| 004 | 32 | R/W |  |  |  |  |
| 008 | 32 | RO |  |  |  |  |
| 00C | 32 | R/W |  |  |  |  |
| 010 | 32 | R/W | REG\_BAR0 | Base Address Register |  |  |
| 014 | 32 | R/W | REG\_BAR1 | Base Address Register |  |  |
| 018 | 32 | R/W | REG\_BAR2 | Base Address Register |  |  |
| 01C | 32 | R/W | REG\_BAR3 | Base Address Register |  |  |
| 020 | 32 | R/W | REG\_BAR4 | Base Address Register |  |  |
| 024 | 32 | R/W | REG\_BAR5 | Base Address Register |  |  |
| 028 | 32 | R/W |  |  |  |  |
| 02C | 32 | RO |  | Subsystem ID |  |  |
| 030 | 32 | R/W |  | Expansion ROM address |  |  |
| 034 | 32 | RO |  |  |  |  |
| 038 | 32 | R/W |  | Reserved |  |  |
| 03C | 32 | R/W |  | Interrupt |  |  |
| 040 to  0FF | 32 | R/W |  | Capabilities area |  |  |

REG\_BAR0 defaults to $FEF30001 which is used to specify the address of the controller’s registers in the I/O address space.

The controller will respond with a decode mask of 0x00FF0000 when BAR0 is written with all ones.

Parameters

CFG\_BUS defaults to zero

CFG\_DEVICE defaults to seven

CFG\_FUNC defaults to zero

Config parameters must be set correctly. CFG device and vendors default to zero.

# Registers

The Datetime uses a 32-bit address decode. Register set selection is determined by the config space BAR0 register.

Registers are holding registers. The date-time is not updated until a write snapshot is indicated in the snapshot register. A read snapshot will read the current date-time into the register holding area. Performing both a read and a write snapshot will swap the holding area with the current date-time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| reg | Bits | R/W | Brief |  |
| 0 | HHMMSSss | RW | time in BCD format |  |
| 1 | YYYYMMDD | RW | date |  |
| 2 | HHMMSSss | RW | alarm time in BCD Format |  |
| 3 | YYYYMMDD | RW | alarm date |  |
| 4 | m ff e mmmmmmmm | W | |  |  |  | | --- | --- | --- | | Bit | Brief |  | | [7:0] | alarm care bits |  | | [8] | tod enable |  | | [10:9] | 00 = 100 Hz,  01 = 60 Hz,  10 = 50 Hz |  | | [16] | 1=Mars timekeeping |  | |  |  |  | |  |  |  | |  |
| 5 |  | W | take a snapshot   |  |  | | --- | --- | | Bit | Brief | | 0 | 1=snapshot read | | 1 | 1=snapshot write | | 8 | 1=snapshot write alarm | |  |

## Time Register (offset 00h)

This register is a record of the time in BCD format. The register may be written anytime. The value written to the holding register will not transfer to current values unless a write snapshot operation is performed.

## Date-time Register (offset 04h)

This register is a record of the date in BCD format. The register may be written anytime. The value written to the holding register will not transfer to current values unless a write snapshot operation is performed.

## Alarm Register (offset 08h)

This register contains the alarm time in the same BCD format as the time register. When the alarm date-time matches the date-time an alarm signal is set. Components of the date-time to match are set by the match bytes in the control register.

## Alarm Register (offset 10h)

This register contains the alarm date in the same BCD format as the date register. When the alarm date-time matches the date-time an alarm signal is set. Components of the date-time to match are set by the match bytes in the control register.

## Control Register (offset 14h)

Bits 0-7 indicate which bytes of the datetime and alarm datetime registers to compare. For instance, an hourly alarm may be set by clearing the year-month-day, and hours bytes.

7 – century match

6 – year match

5 – month match

4 – day match

3- hours match

2 – minutes match

1 – seconds match

0 – jiffies match

Bit 8 – ‘1’ enable time-of-day tracking, 0 disables the time-of-day updates.

Bit 10,9 – specifies the frequency of the time-of-day clock

00 = 100 Hz time-of-day input clock (default)

01 = 60 Hz time of day input clock

10 = 50 Hz time of day input clock

Bit 16 – ‘1’ = keep track of Martian time and date, ‘0’ = Earth date and time.

## Snapshot Register (offset 18h)

The snapshot register allows updates to occur in a synchronized fashion. It is not necessary to disable the clock or interrupts.

Writing to the snapshot register causes a snapshot of the current date and time to be taken. The snapshot may read the current datetime values into holding registers, or write the holding registers to the current datetime, or both operations may be performed at the same time.

The snapshot register is also used to update the alarm datetime.

Code Sample:

DATETIME EQU 0xFEF30000

;------------------------------------------------------------------------------

;------------------------------------------------------------------------------

DisplayDatetime:

subui sp,sp,#32

sm [sp],r1/r2/r3/lr

call CursorOff

lc r2,CursorRow

lc r3,CursorCol

lw r1,#1

outw r1,DATETIME+5 ; trigger a snapshot

lw r1,#46 ; move cursor down to last display line

sc r1,CursorRow

lw r1,#64

sc r1,CursorCol

inw r1,DATETIME ; get the snapshotted date and time

call DisplayWord ; display on screen

sc r2,CursorRow ; restore cursor position

sc r3,CursorCol

call CalcScreenLoc

call CursorOn

lm [sp],r1/r2/r3/lr

ret #32

# I/O Ports

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Wid | I/O | Description |  |
| rst\_i | 1 | I | This is the active high reset signal |  |
| clk\_i | 1 | I | system bus clock |  |
| cs\_config\_i | 1 | I | Circuit select for config space |  |
| cs\_io\_i | 1 | I | Circuit select for I/O register space |  |
| cyc\_i | 1 | I | cycle active |  |
| stb\_i | 1 | I | data strobe |  |
| ack\_o | 1 | O | data transfer acknowledge |  |
| we\_i | 1 | I | write cycle |  |
| sel\_i | 4 | I | byte lane selects |  |
| adr\_i | 32 | I | decode / register address |  |
| dat\_i | 32 | I | data input |  |
| dat\_o | 32 | O | data output |  |
| tod | 1 | I | tod pulse input (eg 100 Hz) |  |
| irq\_o | 32 | O | Interrupt (alarm match) output |  |
|  |  |  |  |  |
|  |  |  |  |  |

# WISHBONE Compatibility Datasheet

The Datetime core may be directly interfaced to a WISHBONE compatible bus.

|  |  |  |
| --- | --- | --- |
| WISHBONE Datasheet  WISHBONE SoC Architecture Specification, Revision B.3 | | |
|  |  | |
| Description: | Specifications: | |
| General Description: | Datetime – date and time keeping | |
| Supported Cycles: | SLAVE, READ / WRITE  SLAVE, BLOCK READ / WRITE  SLAVE, RMW | |
| Data port, size:  Data port, granularity:  Data port, maximum operand size:  Data transfer ordering:  Data transfer sequencing | 32 bit  32 bit  32 bit  Little Endian  any (undefined)  must write register #3 before reading #0 | |
| Clock frequency constraints: | 50/60/100 Hz time of day clock | |
| Supported signal list and cross reference to equivalent WISHBONE signals | Signal Name:  ack\_o  adr\_i(63:0)  clk\_i  dat\_i(63:0)  dat\_o(63:0)  cyc\_i  stb\_i  we\_i | WISHBONE Equiv.  ACK\_O  ADR\_I()  CLK\_I  DAT\_I()  DAT\_O()  CYC\_I  STB\_I  WE\_I |
| Special Requirements: |  | |