APB Dual Timer IP

User Guide

FPGA-IPUG-02250-1.0

March 2024
Disclaimers
Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer’s responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"), FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language
This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.
Contents

Contents .................................................................................................................................................. 3
Acronyms in This Document .................................................................................................................... 5
1. Introduction ........................................................................................................................................ 6

1.1. Overview of the IP ............................................................................................................................. 6
1.2. Quick Facts ...................................................................................................................................... 6
1.3. Features .......................................................................................................................................... 6
1.4. Licensing and Ordering Information ............................................................................................... 6
1.5. IP Validation Summary .................................................................................................................... 7
1.6. Naming Conventions ....................................................................................................................... 7

1.6.1. Nomenclature ............................................................................................................................... 7
1.6.2. Signal Names ............................................................................................................................... 7

2. Functional Description ..................................................................................................................... 8

2.1. IP Architecture Overview ............................................................................................................. 8
2.2. Clocking ........................................................................................................................................ 8
2.3. Reset ............................................................................................................................................. 8

3. IP Parameter Description ................................................................................................................ 9
4. Signal Description .............................................................................................................................. 10
5. Register Description .......................................................................................................................... 11
6. Designing with the IP ........................................................................................................................ 12

6.1. Generating and Instantiating the IP ................................................................................................. 12
7. Software Driver ................................................................................................................................ 13

7.1. APIs ............................................................................................................................................. 13

7.1.1. apb_dt_init() ............................................................................................................................. 13
7.1.2. apb_dt_send_all_good() ......................................................................................................... 13
7.1.3. apb_dt_send_jtag_on() ............................................................................................................ 13
7.1.4. apb_dt_clear_event() ................................................................................................................ 13
7.1.5. apb_dt_read_source() ............................................................................................................. 14

8. Debugging ....................................................................................................................................... 15

8.1. System Clock Input ....................................................................................................................... 15
8.2. Reset Input .................................................................................................................................... 15
8.3. Timer_event Output ........................................................................................................................ 15

Appendix A. Resource Utilization ........................................................................................................ 16
References ............................................................................................................................................. 17
Technical Support Assistance ............................................................................................................... 18
Revision History ................................................................................................................................. 19
Figures
Figure 2.1. APB Dual Timer IP Block Diagram .................................................................................................................. 8
Figure 6.1. Module/IP Block Wizard ................................................................................................................................. 12
Figure 6.2. IP Configuration .................................................................................................................................................. 12

Tables
Table 1.1. Summary of the APB Dual Timer IP ................................................................................................................... 6
Table 1.3. IP Validation Level .................................................................................................................................................. 7
Table 4.1. APB Dual Timer IP Signal Description .............................................................................................................. 10
Table 5.1. List of Accessible Registers ................................................................................................................................ 11
Table A.1. Resource Utilization .............................................................................................................................................. 16
Acronyms in This Document

A list of acronyms used in this document.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>APB</td>
<td>Advanced Peripheral Bus</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>SoC</td>
<td>System on Chip</td>
</tr>
</tbody>
</table>
1. Introduction

1.1. Overview of the IP

The APB Dual Timer IP provides two different timers that can each generate a timer event output to indicate if there is an error to the system. This allows the IP to perform a restart when necessary. The 3-minute timer starts immediately after reset is released. You can write to the control register to stop the 3-minute timer from expiring. The 2-hour timer is reset to zero at startup and held at zero. You can initiate the 2-hour timer by writing to the control register. When started, this timer cannot be reset. A system reset or power cycle is needed to disable this timer.

If either of the timers expires, the timer event output is activated. This signal is used by the system to take the appropriate action. A status register can be read to determine which timer caused the interrupt if the system allows this to happen after the event.

In the Lattice Sentry™ solution, the two timers are used for the following purpose:

- The 3-minute timer starts automatically as soon as reset is released. If the firmware is authenticated and is able to run, the firmware sends an all good message through an API to the dual timer, thus switching off the timer. If the firmware is not able to run, the all good message is not sent and the 3-minute timer expires, indicating to the system that there is a problem with the firmware.
- The 2-hour timer is used to time the JTAG debugging mode. For security reasons, JTAG debugging is only enabled for a 2-hour window. When the firmware puts the device into JTAG debugging mode, the 2-hour timer is set. The 2-hour timer cannot be reset or canceled when it is started. After two hours, the timer_event output indicates to the system that the 2-hour JTAG debugging time has expired.

The APB Dual Timer IP has an APB bus interface that can be connected to the APB switch block in a Propel™ Builder SoC design.

1.2. Quick Facts

Table 1.1. Summary of the APB Dual Timer IP

<table>
<thead>
<tr>
<th>IP Requirements</th>
<th>Supported FPGA Family</th>
<th>Mach™-NX</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Version</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>Resource Utilization</td>
<td>Targeted Devices</td>
<td>Refer to Table A.1</td>
</tr>
<tr>
<td>Supported User Interface</td>
<td>APB</td>
<td></td>
</tr>
<tr>
<td>Design Tool Support</td>
<td>Lattice Implementation</td>
<td>IP Core v 1.0.0 – Lattice Propel software 1.1, Lattice Diamond™ software 3.12</td>
</tr>
<tr>
<td></td>
<td>Synthesis</td>
<td>Synopsys® Synplify Pro for Lattice</td>
</tr>
<tr>
<td></td>
<td>Simulation</td>
<td>No simulation support</td>
</tr>
</tbody>
</table>

1.3. Features

Key features of the APB Dual Timer IP include:

- 3-minute timer
- 2-hour timer
- APB interface
- 1-wire output signal, timer event
- Readable register with timer event data

1.4. Licensing and Ordering Information

The APB Dual Timer IP is provided at no additional cost with the Diamond software.
1.5. IP Validation Summary

The IP is validated with the following synthesis tool:
- Synplify Pro U-2023.03L-SP1, Build 153R, Aug 10 2023

Table 1.2 shows the validation status for the APB Dual Timer IP core. The ✓ mark indicates whether the IP has been validated for Simulation, Timing, or with Hardware.

Table 1.2. IP Validation Level

<table>
<thead>
<tr>
<th>Device Family</th>
<th>IP Version</th>
<th>Validation Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Simulation</td>
</tr>
<tr>
<td>Mach-NX</td>
<td>1.0</td>
<td>✓</td>
</tr>
</tbody>
</table>

1.6. Naming Conventions

1.6.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.6.2. Signal Names
- _n are active low (asserted when value is logic 0)
- _i are input signals
- _o are output signals
2. Functional Description

2.1. IP Architecture Overview
The APB Dual Timer IP performs the following functions:
• Triggers the timer_event output 3 minutes after reset has been released and boot sequence is started, if the 3-minute timer is not cleared before then.
• Triggers the timer_event output 2 hours after the 2-hour timer is set

The timers are set with the APB interface. The CPU can send commands to start the 2-hour timer, to clear the 3-minute timer, or to read the status register to determine which timer event has occurred.
Figure 2.1 shows the IP block diagram.

2.2. Clocking
The APB Dual Timer is designed to work in a system with a 50 MHz clock. Different input clock speeds result in different lengths of time for the two timers.
The APB Dual Timer has one input clock source.

2.3. Reset
The APB Dual Timer has an active low reset input. Upon reset, the interrupt registers and the timers are cleared. The timer_event output is set to 0. After reset, the 3-minute timer starts automatically.
The 2-hour timer must be set by the RISC-V CPU.
3. IP Parameter Description

You do not need to set any IP parameters for this IP.
4. Signal Description

The APB bus interface signals are grouped as one connection on the Propel Builder block that is connected in the SoC design. Table 4.1 shows the list of signals.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clock and Reset</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>clk_i</td>
<td>In</td>
<td>1</td>
<td>System clock (50 MHz)</td>
</tr>
<tr>
<td>rstn_i</td>
<td>In</td>
<td>1</td>
<td>Asynchronous reset (Active Low)</td>
</tr>
<tr>
<td><strong>APB Interface</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>apb_penable_i</td>
<td>In</td>
<td>1</td>
<td>APB enable</td>
</tr>
<tr>
<td>apb_psel_i</td>
<td>In</td>
<td>1</td>
<td>APB select, high when in address range</td>
</tr>
<tr>
<td>apb_paddr_i</td>
<td>In</td>
<td>32</td>
<td>APB address, only lower bits are used</td>
</tr>
<tr>
<td>apb_pwdata_i</td>
<td>In</td>
<td>2</td>
<td>APB write data, only lower bits are used</td>
</tr>
<tr>
<td>apb_prdata_o</td>
<td>Out</td>
<td>1</td>
<td>APB ready, indicates read data is valid</td>
</tr>
<tr>
<td>apb_prdata_o_i</td>
<td>Out</td>
<td>32</td>
<td>APB read data</td>
</tr>
<tr>
<td><strong>Miscellaneous</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| timer_event    | Out | 1     | • Active high signal to indicate to the system that one of the timers has expired.  
|                |     |       | • This signal is used for both the 3-minute and 2-hour timers.  
|                |     |       | • The RISC-V CPU can make an API call to read the status register and determine which timer has expired. |
5. Register Description

Table 5.1 shows the list of registers that are accessible to the processor.

<table>
<thead>
<tr>
<th>Register</th>
<th>Offset from Base</th>
<th>Width</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Good Reg</td>
<td>0x00</td>
<td>–</td>
<td>W</td>
<td>Writing to this register halts the three-minute timer</td>
</tr>
<tr>
<td>JTAG Time On</td>
<td>0x04</td>
<td>–</td>
<td>W</td>
<td>Writing to this register starts the two-hour timer</td>
</tr>
<tr>
<td>Status Reg</td>
<td>0x0C</td>
<td>2 bits</td>
<td>RW</td>
<td>Bit 0 is set by the 3-minute timer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Bit 1 is set by 2-hour timer</td>
</tr>
</tbody>
</table>

Notes:
1. Addresses 0x00 and 0x04 are write-only registers. The first two registers are one-time events. Additional writes have no effect.
2. Writes to address 0x0C will reset the timer_event output to 0. Bits 1 and 0 in the status register are also set to 0.
3. Reads from any register address not defined in this table will return all 1’s.
6. Designing with the IP

6.1. Generating and Instantiating the IP

The APB Dual Timer IP appears in the IP local list as APB_DUAL_TIMER.

To generate the APB Dual Timer IP:

1. In the IP Catalog tab, double-click the APB_DUAL_TIMER IP. The Module/IP Block Wizard opens as shown in Figure 6.1. Enter a name for the IP block in the Component name text box and click Next.

![Figure 6.1. Module/IP Block Wizard](image)

2. There are no configurable items for this IP. In the next Module/IP Block Wizard window, click Generate as shown in Figure 6.2.

3. Click Finish on the next window.

![Figure 6.2. IP Configuration](image)

4. Accept or modify the instance name for the block and click OK to complete the addition of this block to an SoC design.

5. Drag and drop a connection from the APB input of the IP block to a corresponding APB bus interface port. Optionally, connect the timer_event output to a corresponding interrupt input on an appropriate CPU or Interrupt controller block.
7. Software Driver

Two software driver files are provided:

- `apb_dual_timer.h`
- `apb_dual_timer.c`

You need to include the `.h` file in any software file that needs to control the timers.

7.1. APIs

The following subsections list the functions that allow the proper control of the two timers in the APB Dual Timer IP.

7.1.1. `apb_dt_init()`

This API initializes the dual timer.

```c
unsigned char apb_dt_init (struct apb_dual_timer_instance *this_timer, unsigned int base_addr)
```

<table>
<thead>
<tr>
<th>In/Out</th>
<th>Parameter</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>this_timer</td>
<td>Handle of the apb_dual_timer_instance structure.</td>
<td>0: success, 1: failure</td>
</tr>
<tr>
<td>In</td>
<td>base_addr</td>
<td>Base address to be assigned to the timer.</td>
<td></td>
</tr>
</tbody>
</table>

7.1.2. `apb_dt_send_all_good()`

This API sends the all good signal to the timer to indicate that the firmware has booted successfully and clears the 3-minute timer.

```c
void apb_dt_send_all_good (struct apb_dual_timer_instance *this_timer)
```

<table>
<thead>
<tr>
<th>In/Out</th>
<th>Parameter</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>this_timer</td>
<td>Handle of the apb_dual_timer_instance structure.</td>
<td>–</td>
</tr>
</tbody>
</table>

7.1.3. `apb_dt_send_jtag_on()`

This API starts the 2-hour timer to allow a 2-hour window for JTAG debugging.

```c
void apb_dt_send_jtag_on (struct apb_dual_timer_instance *this_timer)
```

<table>
<thead>
<tr>
<th>In/Out</th>
<th>Parameter</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>this_timer</td>
<td>Handle of the apb_dual_timer_instance structure.</td>
<td>–</td>
</tr>
</tbody>
</table>

7.1.4. `apb_dt_clear_event()`

This API clears the timer event output and clears the dual timer interrupt flags.

```c
void apb_dt_clear_event (struct apb_dual_timer_instance *this_timer, unsigned int value)
```

<table>
<thead>
<tr>
<th>In/Out</th>
<th>Parameter</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>this_timer</td>
<td>Handle of the apb_dual_timer_instance structure.</td>
<td>–</td>
</tr>
<tr>
<td>In</td>
<td>value</td>
<td>A 2-bit value to be written to the register:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: clear 3-minute timer interrupt</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x2: clear 2-hour timer interrupt</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x3: clear both timer interrupts</td>
<td></td>
</tr>
</tbody>
</table>
7.1.5. apb_dt_read_source()

This API polls the dual timer interrupt register to determine if a timer_event signal is generated by the 3-minute timer or the 2-hour timer.

```c
unsigned char apb_dt_read_source (struct apb_dual_timer_instance *this_timer)
```

<table>
<thead>
<tr>
<th>In/Out</th>
<th>Parameter</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>this_timer</td>
<td>Handle of the apb_dual_timer_instance structure.</td>
<td>0: 3-minute timer expired</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1: 2-hour timer expired</td>
</tr>
</tbody>
</table>
8. Debugging

Methods for debugging each element of the APB Dual Timer IP are listed below.

8.1. System Clock Input

Ensure that the input clock is 50 MHz. Test and measure this clock signal with external equipment to verify.

8.2. Reset Input

Ensure that an active low system reset is the input signal to the APB Dual Timer IP.

8.3. Timer_event Output

- 3-minute timer:
  - In firmware, ensure that send_all_good API is commented out or not called. Boot the system and set a stopwatch for three minutes. After three minutes, the short timer should expire, and the system should try to boot from the backup configuration image.
  - If this behavior is not observed, test the timer_event output on an I/O pin or LED to narrow down whether the problem is with the timer_event output or with the system’s response.
- 2-hour timer:
  - Enable JTAG debug mode in firmware. Ensure the JTAG enable API calls the dual timer send_jtag_on() API. Set an external timer for two hours. After two hours, JTAG debug mode should be disabled and the system should reboot.
  - If this behavior is not observed, test the timer_event output on an I/O pin or LED to narrow down whether the problem is with the timer_event output or with the system’s response.
Appendix A. Resource Utilization

Table A.1 shows a sample resource utilization of the APB Dual Timer IP Core on a Mach-NX device.

Table A.1. Resource Utilization

<table>
<thead>
<tr>
<th>Registers</th>
<th>LUTs</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>60</td>
</tr>
</tbody>
</table>
References

- Lattice Diamond FPGA design software
- Lattice Solutions IP Cores web page
- Lattice Propel Design Environment web page
- Lattice Diamond Software User Guide
- Lattice Insights for Lattice Semiconductor training courses and learning plans
Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, please refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.
# Revision History

**Revision 1.0, March 2024**

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>