BOSCH
Invented for life

GTM-IP

Application Note AN009
Integration Test Suite

Date: 09.12.2011
(Released)

Robert Bosch GmbH
Automotive Electronics (AE)
Engineering Integrated Circuits Systems (EIY2)

LEGAL NOTICE

The content of this document is subject to continuous developments and improvements. All particulars and its use contained in this document are given by BOSCH in good faith.

NO WARRANTIES: TO THE MAXIMUM EXTENT PERMITTED BY LAW, NEITHER THE INTELLECTUAL PROPERTY OWNERS, COPYRIGHT HOLDERS AND CONTRIBUTORS, NOR ANY PERSON, EITHER EXPRESSLY OR IMPLICITLY, WARRANTS ANY ASPECT OF THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO, INCLUDING ANY OUTPUT OR RESULTS OF THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO UNLESS AGREED TO IN WRITING. THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO IS BEING PROVIDED "AS IS", WITHOUT ANY WARRANTY OF ANY TYPE OR NATURE, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND ANY WARRANTY THAT THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO IS FREE FROM DEFECTS.

ASSUMPTION OF RISK: THE RISK OF ANY AND ALL LOSS, DAMAGE, OR UNSATISFACTORY PERFORMANCE OF THIS SPECIFICATION (RESPECTIVELY THE PRODUCTS MAKING USE OF IT IN PART OR AS A WHOLE), SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO RESTS WITH YOU AS THE USER. TO THE MAXIMUM EXTENT PERMITTED BY LAW, NEITHER THE INTELLECTUAL PROPERTY OWNERS, COPYRIGHT HOLDERS AND CONTRIBUTORS, NOR ANY PERSON EITHER EXPRESSLY OR IMPLICITLY, MAKES ANY REPRESENTATION OR WARRANTY REGARDING THE APPROPRIATENESS OF THE USE, OUTPUT, OR RESULTS OF THE USE OF THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO IN TERMS OF ITS CORRECTNESS, ACCURACY, RELIABILITY, BEING CURRENT OR OTHERWISE. NOR DO THEY HAVE ANY OBLIGATION TO CORRECT ERRORS, MAKE CHANGES, SUPPORT THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO, DISTRIBUTE UPDATES, OR PROVIDE NOTIFICATION OF ANY ERROR OR DEFECT, KNOWN OR UNKNOWN. IF YOU RELY UPON THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO, YOU DO SO AT YOUR OWN RISK, AND YOU ASSUME THE RESPONSIBILITY FOR THE RESULTS. SHOULD THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL LOSSES, INCLUDING, BUT NOT LIMITED TO, ANY NECESSARY SERVICING, REPAIR OR CORRECTION OF ANY PROPERTY INVOLVED TO THE MAXIMUM EXTEND PERMITTED BY LAW.

DISCLAIMER: IN NO EVENT, UNLESS REQUIRED BY LAW OR AGREED TO IN WRITING, SHALL THE INTELLECTUAL PROPERTY OWNERS, COPYRIGHT HOLDERS OR ANY PERSON BE LIABLE FOR ANY LOSS, EXPENSE OR DAMAGE, OF ANY TYPE OR NATURE ARISING OUT OF THE USE OF, OR INABILITY TO USE THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO, INCLUDING, BUT NOT LIMITED TO, CLAIMS, SUITS OR CAUSES OF ACTION INVOLVING ALLEGED INFRINGEMENT OF COPYRIGHTS, PATENTS, TRADEMARKS, TRADE SECRETS, OR UNFAIR COMPETITION.

INDEMNIFICATION: TO THE MAXIMUM EXTEND PERMITTED BY LAW YOU AGREE TO INDEMNIFY AND HOLD HARMLESS THE INTELLECTUAL PROPERTY OWNERS, COPYRIGHT HOLDERS AND CONTRIBUTORS, AND EMPLOYEES, AND ANY PERSON FROM AND AGAINST ALL CLAIMS, LIABILITIES, LOSSES, CAUSES OF ACTION, DAMAGES, JUDGMENTS, AND EXPENSES, INCLUDING THE REASONABLE COST OF ATTORNEYS' FEES AND COURT COSTS, FOR INJURIES OR DAMAGES TO THE PERSON OR PROPERTY OF THIRD PARTIES, INCLUDING, WITHOUT LIMITATIONS, CONSEQUENTIAL, DIRECT AND INDIRECT DAMAGES AND ANY ECONOMIC LOSSES, THAT ARISE OUT OF OR IN CONNECTION WITH YOUR USE, MODIFICATION, OR DISTRIBUTION OF THIS SPECIFICATION, SOFTWARE RELATED THERETO, CODE AND/OR PROGRAM RELATED THERETO, ITS OUTPUT, OR ANY ACCOMPANYING DOCUMENTATION.

GOVERNING LAW: THE RELATIONSHIP BETWEEN YOU AND ROBERT BOSCH GMBH SHALL BE GOVERNED SOLELY BY THE LAWS OF THE FEDERAL REPUBLIC OF GERMANY. THE STIPULATIONS OF INTERNATIONAL CONVENTIONS REGARDING THE INTERNATIONAL SALE OF GOODS SHALL NOT BE APPLICABLE. THE EXCLUSIVE LEGAL VENUE SHALL BE DUESSELDORF, GERMANY.

MANDATORY LAW SHALL BE UNAFFECTED BY THE FOREGOING PARAGRAPHS.

INTELLECTUAL PROPERTY OWNERS/COPYRIGHT OWNERS/CONTRIBUTORS: ROBERT BOSCH GMBH, ROBERT BOSCH PLATZ 1, 70839 GERLINGEN, GERMANY AND ITS LICENSORS.

Revision History

Issue	Date	Remark
1.2	27.09.2010	Initial version
1.3	15.12.2010	Updated version for Release Candidate 1
1.4	29.03.2011	Updated version for Release Candidate 1.1
1.4.1	31.05.2011	Update References for GTM-IP v1.4.1
1.4.2	29.07.2011	Update References for GTM-IP v1.4.2
1.4.4	23.09.2011	Update References for GTM-IP v1.4.4
1.5.0	09.12.2011	Update to GTM-IP specification v1.5.0, Additional power tests

Tracking of major changes

Changes between revision 1.3 and 1.4
Added three (3) new GTM-IP integration tests.
Changes between revision 1.4.4 and 1.5.0
Added eight (8) power tests

Conventions

The following conventions are used within this document.

ARIAL BOLD CAPITALS	Names of signals
Arial bold	Names of files and directories
`Courier bold`	Command line entries
`Courier`	Extracts of files

References

This document refers to the following documents.

Ref	Authors(s)	Title
1	AE/EIY2	GTM-IP Specification v1.5.0
2	AE/EIY2	GTM Testbench Users guide

Terms and Abbreviations

This document uses the following terms and abbreviations.

Term	Meaning
GTM	Generic Timer Module

Table of Contents

1 Overview

This document describes integration tests and power analysis tests for the GTM-IP. This integration tests can be used to evaluate the proper integration of the GTM-IP into a SoC. The integration tests are written in C and can be compiled partially for the target system.
The integration tests run also in the GTM-IP IFS testbench delivered together with the GTM-IP and GTM-RM. There, the customer can see the valid behaviour of the tests. Furthermore, for some of the integration tests, the IFS testbench is needed to generate the stimuli for the GTM.
To run the tests within the IFS testbench environment following command files are necessary:

- sw_gtm_mem_test.cmd
- sw_gtm_irq_test.cmd
- sw_tom_port_test.cmd
- sw_atom_port_test.cmd
- sw_gtm_reg_test
- sw_tim_port_test
- sw_gtm_reg_test_gtm_halt
- pwr_basic_test
- pwr_modclokc_test
- pwr_cmutbu_test
- pwr_tom_test
- pwr_atom_test
- pwr_fifoarubrc_test
- pwr_fulload_test
- pwr_usecase_test

The command files can be found in the delivery in the directory
`<GTM_IP_XXX_Y>\design\gtm\vXXX\sim_data\ifscmd\integration_tests`
and
`<GTM_IP_XXX_Y>\design\gtm\vXXX\sim_data\ifscmd\power`

The command files use the IFS command

SWI RUN_SW <sw_test>

For a detailed description of the IFS and its software interface please refer to the GTM Testbench guide [2].

The integration test suite consists of seven integration tests. Table 1.1 shows the integration tests and their test field.

Name	Purpose
gtm_mem_test.cpp	Tests read/write functionality for all physical memories connected to the GTM-IP.
gtm_irq_test.cpp	Stimulates each of the available GTM-IP interrupt lines by using the FORCINT register. The Level interrupt mode is used throughout the test. Nevertheless, an interrupt mode change should not cause any issue.
tom_port_test.cpp	This test creates a high-low and further low-high edge at each TOM channel available inside the GTM-IP.
atom_port_test.cpp	This test creates a high-low and further low-high edge at each ATOM channel available inside the GTM-IP.
gtm_reg_test.cpp	Tests reading and writing to the register map of the GTM.
tim_port_test.cpp	Tests the TIM input channels. For this test the TOM submodule is used to generate the TIM input stimuli signals. The connection of TOM → TIM is done via the IFS testbench.
gtm_reg_test_ gtm_halt.cpp	This test stresses the GTM debugging feature to read/write GTM-IP registers while the GTM input clock is stopped.

Table 1.1: Overview of the GTM-IP integration test suite.

Following tests are useful to stimulate GTM-IP while measuring current consumption of GTM-IP.

Name	Purpose
pwr_atom_testcpp	Power analysis while ATOM generates PWM
pwr_basic_test.cpp	Power analysis after power and clock on
pwr_cmutbu_test.cpp	Power analysis while CMU and TBU are enabled
pwr_fifoarubrc_test.cpp	Power analysis while FIFO-ARU-BRC are transferring data
pwr_fullload_test.cpp	Power analysis while full load of GTM-IP
pwr_modclock_test.cpp	Power analysis while different clock modes are enabled
pwr_tom_test.cpp	Power analysis while TOM generates PWM
pwr_usecase_test.cpp	Power analysis while application use case is running

Table 1.2: Overview of the GTM-IP power analysis test suite.

For a correct function of the integration tests, additional files are needed as shown in Table 1.3.

Name	Purpose
gtm.h	Main header file includes subsequent header files, gtm_hal.h, virtual_ptr.h and register_map.h.
gtm_hal.h	Makes the link between the GTM AEI interface and the IFS testbench.
virtual_ptr.h	Implements the register manipulation operations used in the integration tests.
register_map.h	Contains the addresses of the GTM-IP registers.
functions.cpp	Contains some helper functions used by integration tests.
functions.h	The header file of file functions.cpp
pwr_functions.cpp	Contains some helper functions used by power tests
pwr_functions.h	The header file of file pwr_functions.cpp

Table 1.3: Files supporting the GTM Integration test suite and power analysis test suite.

1.1 Header file adaptation

There is a minor adaptation necessary for the above mentioned header files for reuse in a SoC environment.
There, only the two header files

- gtm.h
- register_map.h

are necessary.
The header file gtm.h has to be adapted by the user for the SoC environment.

Original code snippet:

```
#ifdef GTM_RTL
#    include "hal_if.h"
#    include "virtual_ptr.h"
typedef virtual_ptr<hal_if, unsigned int> gtm_ptr;
#elif GTM_REF
#    include "hal_if.h"
#    include "virtual_ptr.h"
typedef virtual_ptr<hal_if, unsigned int> gtm_ptr;
#else
#    error "Define a preprocessor variable GTM_REF or GTM_RTL."
#endif
```

Please remove the above code snippet and define the `gtm_ptr` in the file gtm.h by

```
typedef unsigned int * gtm_ptr;
```

Secondly, the address offset for the GTM-IP in the file register_map.h is defined as 0x000xxxxx. This offset has to be changed to point to the correct GTM-IP base address within the SoC environment.

2 Integration test suite

This section contains a description of the integration tests that can be used in a SoC environment to test the integration of the GTM_IP.

2.1 GTM Memory test

The GTM Memory test is located in the `gtm_mem_test.cpp` file. The test can be used to test the correct connection of the RAM's to the GTM-IP. For the submodule DPLL the RAM's 1a, 1bc and 2 are tested. For the MCS both RAM's are tested. Finally, the FIFO RAM is tested via direct RAM accesses.

The main entry point of the memory test is the function:

```
int gtm_mem_test(void)
```

The test writes first the value `0xFFFFFFFF` in each of the addressable RAM's and reads back expected values according to the RAM width and writeability. Thus, for the FIFO the values read back have to be 29 bit wide while the MCS RAM should deliver the whole 32 bits and the DPLL RAM's should deliver 24 bits back.

The memory tests are self checking which means that the RAM read is compared to the expected values for the RAM as described above. If one of the tests fails the function `gtm_mem_test()` returns a -1.

2.2 GTM Interrupt test

The GTM Interrupt test can be used to stimulate the GTM interrupt lines and to show the connection to the SoC interrupt controller. The test stimulates each of the GTM-IP internal interrupt lines by using the register ***_IRQ_FORCINT**.

The main entry point for the interrupt test is the function:

```
int gtm_irq_test(void)
```

The test uses the GTM Level interrupt mode for the interrupt signal generation. The interrupts are enabled one by one by first setting the interrupt bit in the ***_IRQ_EN** register and then forcing the interrupt by writing a '1' to the corresponding bit of the ***_IRQ_FORCINT** register.

The interrupt handling is done within the function

```
int gtm_irq_isr(int number)
```

This function is called from the IFS command file with the coded interrupt number for a dedicated interrupt source. The interrupt number encoding can be obtained from the GTM Testbench User Guide.
When called, the ***_IRQ_NOTIFY** register is cleared by writing to the expected interrupt bit that should occur. Since the internal interrupt lines are bundled inside the submodules and some of the interrupts furthermore are bundled inside the ICM (TOM and ATOM submodules) outside of the GTM-IP the bundled interrupts occur on the same interrupt line. Therefore, the interrupt service routine is called several times.
The software itself checks after all interrupts were forced if the GTM internal ***_IRQ_NOTIFY** registers were cleared by the interrupt service routines.
This is done within the functions

```
int *_result(…)
```

were * is replaced by the submodule names (e.g. fifo).

The interrupt test will return a -1 if one of the tests detects a ***_IRQ_NOTIFY** register content unequal to zero after the interrupts were stimulated and set back by the interrupt service routines.

For correct checking of the interrupt test in a SoC environment at the IP integrator, the IP integrator can either integrate the above mentioned code from the function `gtm_irq_isr()` into his interrupt handler or he has to write an interrupt service routine according to his interrupt system on his own. In any case the self checking mechanism has to be disabled (comment out the function calls `*_result(…)`). If he does so, he has to ensure that the interrupts are visible at his interrupt controller by his own test strategy.

2.3 TOM Port test

The TOM Port test can be used to generate signals at the output ports of the GTM-IP with the TOM submodule. The tests use the SL bit inside the **TOM[i]_CH[x]_CTRL** register of each individual TOM channel. When a TOM channel is disabled and also the output port is disabled, the output signal level can nevertheless be changed by writing to the SL bit within the channel **TOM[i]_CH[x]_CTRL** register.

This feature is used by the test by writing a ‘1’ and subsequently a ‘0’ to toggle the output twice, one time from high to low and the second time back from low to high. The output is monitored by the CHKSIG IFS within an IFS command file.

The main entry point for the interrupt test is the function:

```
int tom_port_test(void)
```

Checks are only done within the IFS command file.

2.4 ATOM Port test

The ATOM Port test can be used to generate signals at the output ports of the GTM-IP with the ATOM submodule. The tests use the SL bit inside the **ATOM[i]_CH[x]_CTRL** register of each individual ATOM channel. When an ATOM channel is disabled and also the output port is disabled, the output signal level can nevertheless be changed by writing to the SL bit within the channel **ATOM[i]_CH[x]_CTRL** register.
This feature is used by the test by writing a '1' and subsequently a '0' to toggle the output twice, one time from high to low and the second time back from low to high. The output is monitored by the CHKSIG IFS within an IFS command file.

The main entry point for the ATOM port test is the function:

```
int atom_port_test(void)
```

Checks are only done within the IFS command file by using the CHKSIG IFS.

2.5 GTM Register test

This integration test tries to read the GTM-IP registers and tests if the register reset values are aligned to the GTM-IP specification. In a second step, the registers are written and it is tested if the written data is visible within the addressed registers, while taking into account read only and reserved bits.
The main entry point for the GTM register test is the function:

```
int gtm_reg_test(void)
```

2.6 TIM port test

The TIM port test examines the GTM input channels to work properly, when input signal edges occur. The input signal edges are generated with the TOM output channels. These signals are connected with the IFS CHKSIG module to the TIM input ports establishing a loop from the TOM outputs to the TIM inputs.
The main entry point for the TIM port test is the function:

```
int tim_port_test(void)
```

2.7 GTM-Halt feature test

This test does the same tests like the GTM register test described in section 2.5. In addition the IFS testbench asserts the *gtm_halt* signal line (see [1]), while the GTM registered are read.
The main entry point for the TIM port test is the function:

```
int gtm_reg_test_gtm_halt(void)
```

3 Power analysis test suite

This section contains a description of the power analysis tests that can be used in a SoC environment to test the power consumption of the GTM_IP.

3.1 Power ATOM test

```
int pwr_atom_test(char*);
```

3.2 Power basic test

```
int pwr_basic_test(char*);
```

3.3 Power CMU TBU test

```
int pwr_cmutbu_test(char*);
```

3.4 Power FIFO-ARU-BRC test

```
int pwr_fifoarubrc_test(char*);
```

3.5 Power full load test

```
int pwr_fullload_test(char*);
```

3.6 Power mode clock test

```
int pwr_modclock_test(char*);
```

3.7 Power TOM test

```
int pwr_tom_test(char*);
```

3.8 Power of typical use case

```
int pwr_usecase_test(char*);
```