Application Report
SPRAAS5A–December 2007–Revised October 2009

TMS320C6472/TMS320TCI6486 Power-On Self Test

Vivek Kangralkar

ABSTRACT

This Power-On Self Test (POST) is designed to verify the operation of the TMS320TCI6486/TMS320C6472. Ten modules are included in this test: Chk6xTest, MemoryEdmaTest, TimerTest, TsipTest, I2cTest, SrioTest, EmacTest, MdioTest, and MultigemTest. These modules check the proper operation of the CPU cores, internal memory, cache operations, and several on-chip peripherals: EDMA3, Timers, TSIP, PLLs, Serial RapidIO®, Ethernet MAC, MDIO, EMIF, and I2C.

It is important to note that this program only provides a confidence check. It is not as comprehensive as the tests done at production that thoroughly check the device's logic, performance, and electrical parameters.

Contents

List of Figures

List of Tables

1 Introduction

This Power-On Self Test (POST) is designed to verify the operation of the TMS320TCI6486/TMS320C6472. Ten modules are included in this test: Chk6xTest, MemoryEdmaTest, TimerTest, TsipTest, I2cTest, SrioTest, Emac Test, MdioTest, and MultigemTest. These modules check the proper operation of the CPU core, internal memory, external memory, cache operations, and several on-chip peripherals: EDMA3, Timers, TSIP, PLLs, Serial RapidIO, Ethernet MAC, EMIF, and I2C. The TMS320TCI6486/TMS320C6472 POST needs to be performed when the TCI6486/C6472 DSP is in its initial state (after reset). All modules consist of C-callable functions.

The C64x+ CPU test module (Chk6xTest) is based on the *TMS320C62x Self-Check Program Applications Brief: Version 1.0* (SPRA635). The TMS320TCI6486/TMS320C6472 POST is an extension to the *TMS320C6416 Power-On Self Test* (SPRA838).

The Chip Support Library (CSL3) is used when testing cache operations, enhanced direct-memory access (EDMA3), EMAC, TSIP, and Timer. The CSL provides a C-language interface for configuring and controlling on-chip peripherals.

It is important to note that this program only provides a confidence check. It is not as comprehensive as the tests done at device's logic, performance, and electrical parameters. This program is not capable of detecting all potential faults.

1.1 System Requirements

To use the TMS320TCI6486/TMS320C6472 POST, the following is required:

- A board with the TMS320TCI6486/TMS320C6472 processor on which to run the POST test.
- A suitable host PC to support TI tools.
- Code Composer Studio™ 3.3, or later revision.
- Emulator tools (XDS560™, etc).

1.2 Test Structure

The test contains the following modules:

- Module 1. Chk6xTest ()
  Verifies the CPU instruction set. This module uses the following assembly files to perform the CPU test:
  - alu_64x.asm: arithmetic operations
  - alu40.asm: 40-bit arithmetic
  - basic_64x.asm: basic operations
  - bit.asm: bit management
  - circular_64x.asm: circular addressing instructions
  - cond.asm: branch and conditional instructions
  - mult_64x: multiplier operations
  - sat_64x: saturation instructions
  - spbuffer_b/le0/1.asm: software pipelined loop buffer
- Module 2. MemoryEdmaTest ()
  Verifies the EDMA3 and on-chip level 1 and level 2 memories, shared memory, and external memory by writing, moving, comparing, and restoring internal data memory through the EDMA.
- Module 3. EmacTest()
  Verifies both EMAC0 and EMAC1, separately.
- Module 4. MdioTest()
  Checks read/write accesses to the MDIO registers.
- Module 5. I2cTest()
  Verifies I2C by comparing transmitted data with the received data through digital loopback.
- Module 6. TimerTest()
  Verifies the timers by waiting until timer interrupt is generated. It checks whether the timer counter is

incremented.

- Module 7. SrioTest()
  Checks if the transmitted data is received on the same port in digital-loopback mode.
- Module 8. TsipTest()
  All 3 TSIPs are configured for digital-loopback mode. This module tests all 3 TSIPs in the system.

When one module fails, the test skips the remaining modules and enters an infinite loop keeping the error value of the first failed module in a variable called Error (see Figure 1).

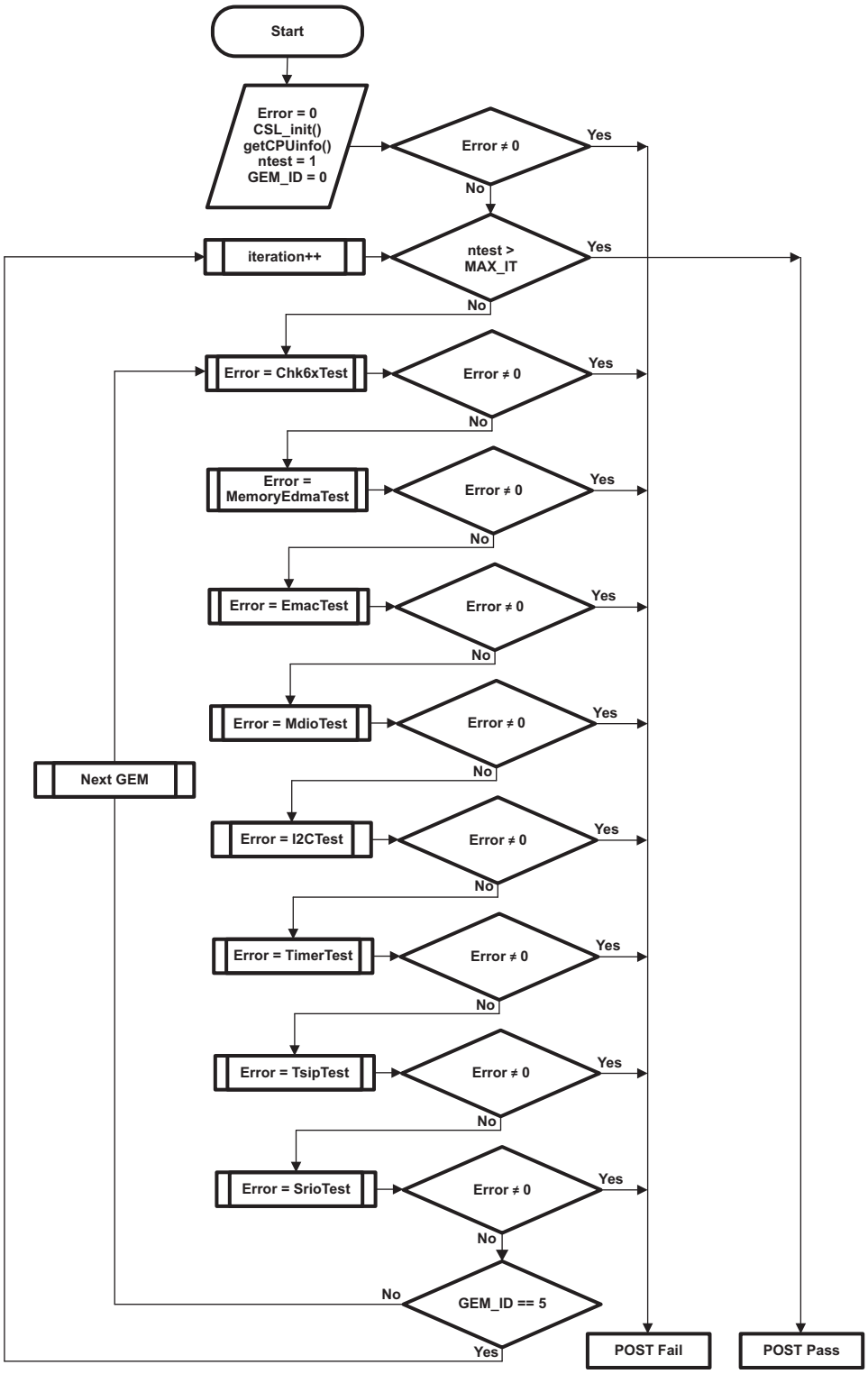

Figure 1. POST Execution Flowchart

1.3 Configuring the TMS320TCI6486/TMS320C6472 POST

The POST main screen file contains a number of constant definitions dedicated to the POST configuration (see Table 1). These constants are defined in TCI6486/C6472screen.h. In case of multi-core execution, use the parallel debugger for loading and execution.

Table 1. POST Configuration Definitions

Constant	Value Range	Default	Definition
PLLC_PLLM_CLK	[0-31]	30	PLL1 multiplier bits. Defines the frequency multiplier of the input reference.
ISTP_VALUE	Memory-mapped addresses	0x800000	Interrupt Service Table Pointer (Needs to be aligned with .vectors section)
SINGLE_CORE_RUN	0, 1	1	1 = run the test in any DSP core and loop until MAX_IT. This is a single core run.
			0 = run the test in Core0 → Core1 → Core2 and loop until MAX_IT. This is a multi-core run.
TEST_INTERNAL_MEM	0	1	Set to select internal memory through EDMA.
TEST_EXTERNAL_MEM	0, 1	0	Set to select external memory through EDMA.
TEST_EMAC	0, 1	1	Set to select EMAC test.
TEST_MDIO	0, 1	1	Set to select MDIO test.
TEST_TSIP	0, 1	1	Set to select TSIP test.
TEST_I2C	0, 1	1	Set to select I2C test.
TEST_SRIO	0, 1	1	Set to select SRIO test.
TEST_TIMER	0, 1	1	Set to select Timer test.
TO_DISPLAY	0, 1	1	Set to display how many current iterations test is going through.
VERBOSE	0, 1	0	Set to print all debug statements in code.
TSIP_DATA_RATE	0, 8, 16, 32, 64	64	Sets different data rates as follows:
			0 = No TSIP
			8 = 8 MHz
			16 = 16 MHz
			32 = 32 MHz
			64 = 64 MHz
TSIP_ACTIVE_LANES	2, 4, 8	2	Denotes number of active links/lanes used in the system.
MAX_IT	Non-zero	10	Sets maximum number of iterations the test should run on each core.

1.4 Changing the TMS320TCI6486/TMS320C6472 POST Endianess

There is one project file (TCI6486/C6472POST.pjt) along with two different project configurations settings:

- post_le - for Little-Endian Mode. In this configuration, a LITTLE_ENDIAN compilation switch is added as some files are using this switch for module setup required for little-endian mode.
- post_be - for Big-Endian Mode.

The user can select any of these settings from the pop-down menu (visible above the project name pane).

1.5 Multi-Core Test Environment

The multi-core test environment is built with Inter-Processor Communication (IPC) interrupt. In case of multi-core execution, use the parallel debugger for loading and execution. Initially, CPU0 will start the test execution and CPU1, 2, 3, 4, 5 are in IDLE state waiting for IPC interrupt. Once CPU0 has completed test execution, it gives an IPC interrupt to CPU1 and goes into IDLE state. Similarly, CPU1 gives control to CPU2, and so on, until CPU5, which completes one iteration or loop. CPU5 then interrupts CPU0 and wakes it from IDLE state. Figure 2 shows the execution process.

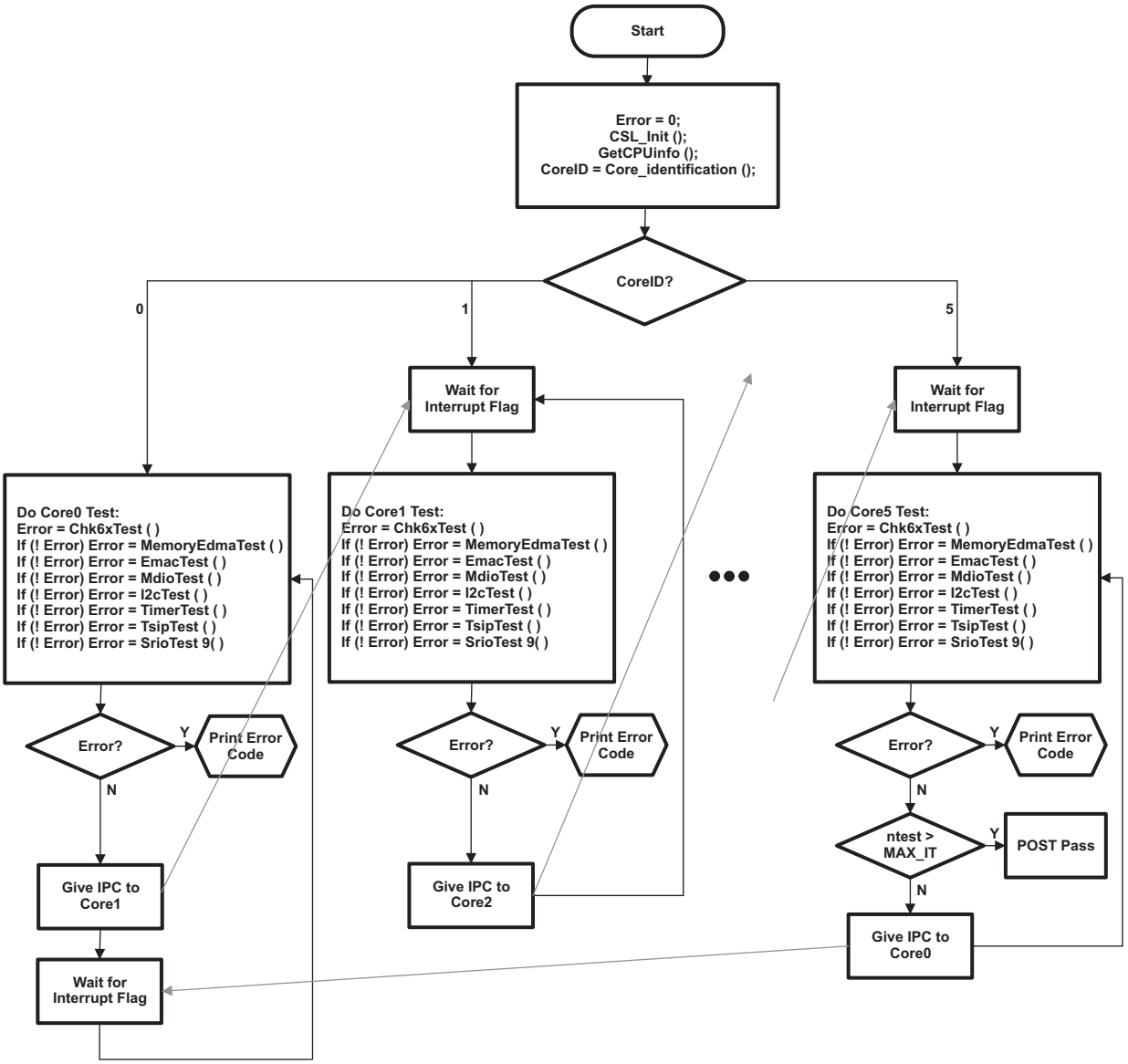

Figure 2. Multi-Core Test Flow

1.6 External Memory Settings

Memory test is extended to use external memory locations. External memory controller registers can be set for two (533 MHz or 400 MHz) DDR2 data and 32- or 16-bit wide mode.

1.7 Test Iteration Time

The memory test extensively tests all the available L1, L2, shared L2, and DDR2 memory locations with memory algorithm technique. One iteration per core 60 seconds (for DDR clock at 533 MHz). The user can set the TO_DISPLAY constant to see which iteration test is currently running.

2 Module Descriptions

2.1 TMS320C64x+ Core Module

This module is based on the *TMS320C62x Self-Check Program Applications Brief: Version 1.0* (SPRA635) and *TMS320C6416 Power-On Self Test* (SPRA838), which verify the proper operation of the TMS320C62x/C64x instructions. Since the TMS320C64x+ provides a superset of the TMS320C62x/C64x architectures, all the TMS320C62x/C64x instructions are tested. Nevertheless, extensions are required to include the additional instructions of the TMS320C64x+.

Table 2 shows the TMS320C64x+ specific instructions. For more details, see the *TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide* (SPRU732).

Table 2. C64x+ Specific Instructions by Functional Unit

No Unit	.L	.M	.S	.D
DINT	ADDSUB	CMPY	CALLP	ADDAB (15-bit const)
SPKERNEL	ADDSUB2	CMPYR	DMV	ADDAH (15-bit const)
SPKERNELR	DPACK2	CMPYR1		ADDAW (15-bit const)
SPLOOP	DPACKX2	DDOTP4		
SPLOOPD	RPACK2	DDOTPH2		
SPLOOPW	SHFL3	DDOTPH2R		
SPMASK	SADDSUB	DDOTPL2		
SPMASKR	SADDSUB2	DDOTPL2R		
SWE	SSUB2	GMPY		
SWENR		MPY		
		2IR		
		MPY32		
		MPY32SU		
		MPY32U		
		MPY32US		
		SMPY32		
		XORMPY		

2.2 Internal/External Memory and EDMA3 Module

The internal and external memory test is non-destructive and makes use of the enhanced direct-memory access (EDMA3) engine. It requires a reserved area, or buffer, to back up existing memory contents, generate memory patterns to be accessed to and from the memory block under test, and compare the read image with the originally-generated pattern. On-chip level 1 and level 2 and shared level 2 memories are configured as SRAM only because their content is submitted to the test. The reserved area is submitted to the same memory test prior to being used as a working buffer.

Figure 3 describes the internal memory test process for Data Block 1.This process is repeated until all 2x8 level 1 and 253 level 2 local blocks have been tested.

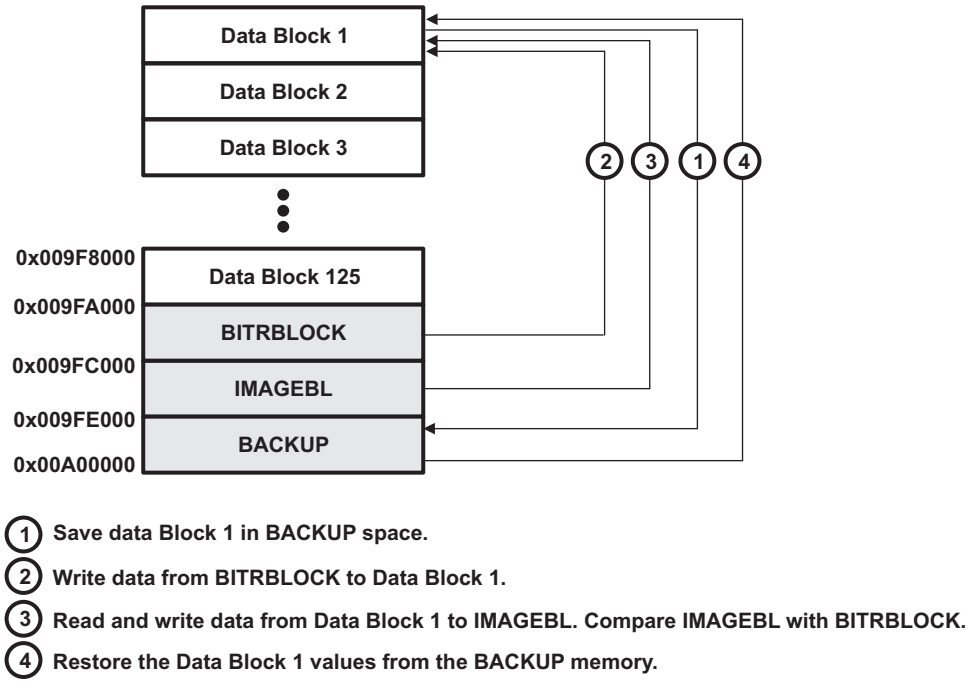

Figure 3. MemoryEdmaTest() Module Flowchart

The address of the buffer under test plays a key role in this memory test technique. The following is a description of the procedure used to generate dedicated patterns for each block under test:

Step 1. Save the 32-bit address number Addna (see Figure 3) at address Addna; when n = 1, Add1a = 0x000FA000.

Step 2. Invert the address number Addna and save it back to address Addnb (Addnb = Addna + 0x4); when n = 1, Add1b = 0x000FA004.

Step 3. Bit-reverse the address number Addna and save it back to address Addnc (Addnc = Addna + 0x8); when n = 1, Add1c = 0x000FA008.

Step 4. Invert the previous bit-reversed word (specified in Step 3) and save it back to address Addnd (Addnd = Addna + 0xC); when n = 1, Add1d = 0x000FA00C.

Step 5. Repeat this process until the end of the buffer is reached (Add1536d).

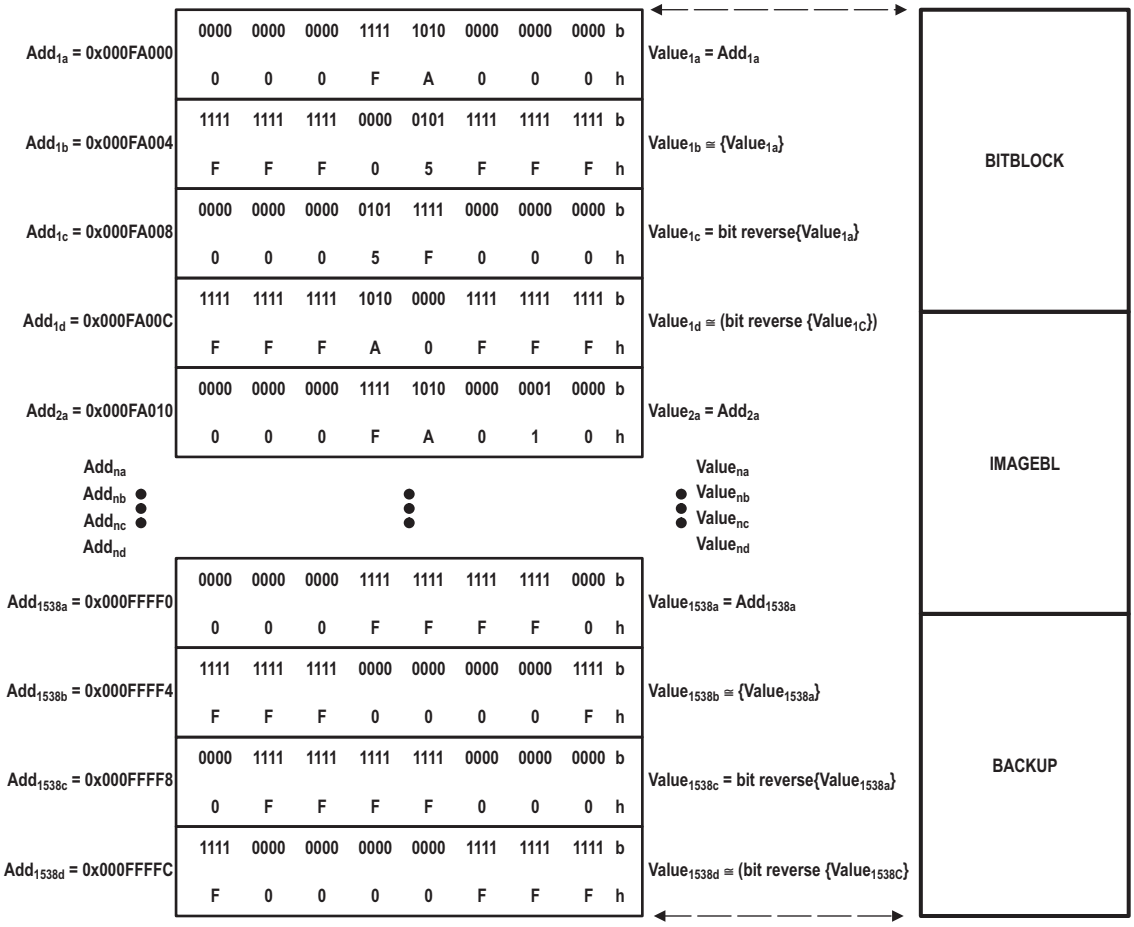

Figure 4. Buffer Test

Once the memory block is tested, the next step is to re-calculate each word and compare it to the values contained in the reserved buffer. If the buffer values do not match the expected number, the test will output an error.

Step 1. Compare the address number Addna (see Figure 3) with the value contained at address Addna (Valuena); when n = 1, Value1a = 0x000FA000.

Step 2. Invert the address number Addna and compare it to the value contained at address Addnb (Valuenb); when n = 1, Value1b = 0xFFF05FFF.

Step 3. Bit-reverse the address number Addna and compare it to the value contained at address Addnc (Valuenc); when n = 1, Value1c = 0x0005F000.

Step 4. Invert the previous bit-reversed word (specified in Step 3) and compare it to the value contained at address Addnd (Valuend); when n = 1, Value1d = 0xFFFA0FFF.

Step 5. Repeated this process until the end of the buffer is reached (Add1536d).

Note that since on-chip level 1 and level 2 memories are configured as SRAM, there is no need to maintain coherency between L1D cache and L2 SRAM when using the EDMA.

2.3 I2C Module

This module tests the functionality of I2C. I2C is configured in loopback mode by the control register. Data is loaded in the ICDXR register and transmitted serially through the I2C transmit shift register (ICXSR).

The transmitted data is serially received in the I2C receive shift register (ICRSR) through internal loopback. Once the ICRSR register receives a whole byte, it will be transferred in the I2C Data Receive Register (ICDRR). In this test, the GEM loads data in the ICDXR register, reads it back from the ICDRR register, and compares it to the data.

2.4 EMAC Module

This test case puts EMAC in the internal loopback mode using the MAC control register. The packets are prepared internally and transmitted. The data is looped back internally to EMAC and received on the same EMAC. This test is performed for both EMAC0 and EMAC1, separately. This test, being a MAC-level internal loopback, is independent of any MII interface.

2.5 MDIO Module

This module confirms the MDIO connectivity at power on. It checks all the MDIO registers at reset.

2.6 Timer Module

This module tests the local and global timers. The timer is configured in dual, unchained 32-bit mode to interrupt the CPU with a period of 0x200. The test verifies whether the timer counter is continuously incremented. When the timer counter register (CNT) register reaches 0x200, the interrupt is triggered and the program counter (PC) jumps to the vector table and then to the Interrupt service routine (ISR) that closes the timer and sets a flag to exit the timer routine. The test is repeated for the global timer.

2.7 SRIO Module

This module tests the Serial RapidIO (SRIO) module. The loopback mode is enabled in the peripheral settings control register and the test makes use of direct I/O mode data transfers. In loopback mode, the transmitted data is received on the same port. Packet data is looped back in the digital domain before the SerDes macros.

2.8 TSIP Module

All three TSIPs (TSIP0, TSIP1, and TSIP2) are configured for digital loopback mode. This module tests all three TSIPs in the system. This module needs the system to drive an external clock and frame sync for TSIP. The number of lanes that TSIP uses need to be configured and the data rate at which the TSIP is intended to test in a particular system. The parameter definitions are explained in Table 3.

Table 3. TSIP Parameter Definitions

Input Parameters	Value	Remarks
TSIP Data Rate	0, 8, 16, 32, 64	0 = No TSIP
		8 = 8 MHz
		16 = 16 MHz
		32 = 32 MHz
		64 = 64 MHz
TSIP Active Lanes Used	2, 4, 8	Denotes number of active links/lanes used in the system

The following is a description of the test:

1. TSIP0 reads the data 0xF00FA55A from the GEM internal memory and then stores it back in the TSIP_DATA variable.
2. TSIP1 reads the TSIP_DATA variable, reads it back through the serial port, and writes it to the TSIP_DATA again.

3. TSIP2 repeats Step 2.
4. CPU compares the TSIP_DATA variable with 0xF00FA55A.

2.9 Multi-Gem Module

All peripheral tests are run by CPU0 followed by CPU1, CPU2, ... CPU5, and this procedure repeats for the iteration number configured.

While one CPU is executing peripheral tests, other CPUs are in IDLE state. Multi-core interrupt is used to wake up the next CPU after it has executed the peripheral tests.

3 Error Codes

Table 4 lists all possible error codes that may be returned by the POST. It also gives the name of the module that generates the error code. It is important to note that the code descriptions identify only potential causes of the error. They should not be taken as absolute. Any number of actual malfunctions could generate a particular error code. For example, a bad memory location would cause every test using it to fail.

Table 4. Error Codes

Code	Description
1h	Internal Mem test failed
2h	External Mem test failed
3h	Tsip operation failed
4h	Srio Operation failed
5h	EMAC0 operation failed
6h	EMAC1 operation failed
7h	MDIO operation failed
8h	I2c operation failed
9h	Local Timer operation failed
Ah	Global Timer6 operation failed
Bh	Global Timer7 operation failed
Ch	Global Timer8 operation failed
Dh	Global Timer9 operation failed
Eh	Global Timer10 operation failed
Fh	Global Timer11 operation failed
11h	LOAD error
12h	STORE error
13h	MV, MVC error
14h	ZERO fail code
15h	MVK, MVKH, MVKLH error
21h	SHIFT AND CMP error
22h	LOGICAL INSTR. Error
23h	ADDITION INSTR. error
24h	SUBTRACTION INSTR. Error
25h	SUBC error
26h	SUB2 error
31h	MPY Error
32h	MPYH & MPYHUS Error
33h	MPYHU & MPYHSU Error
34h	MPYHL Error
35h	MPYLH error

Table 4. Error Codes (continued)

Code	Description
41h	CLR Error
42h	EXTU Error
43h	LMBD Error
44h	NORM Error
45h	SET ERROR
51h	SSHL error
52h	SADD error
53h	SAT error
54h	SSUB error
55h	SMPY(L)(H) error
56h	SMPYHL error
61h	B error
62h	CONDITIONAL instr.
71h	ADDAB error
72h	ADDAH error
73h	ADDAW error
74h	LDW error
81h	ADDU fail code
82h	CMPEQ fail code
83h	SUBU fail code
0x9F001	ABS2 fail code
0x9F002	ADD4 fail code
0x9F003	ANDN fail code
0x9F004	MAX2 fail code
0x9F005	MAX4 fail code
0x9F006	MIN2 fail code
0x9F007	MINU4 fail code
0x9F008	PACKH4 fail code
0x9F009	PACKL4 fail code
0x9F00A	PACKHL2 fail code
0x9F00B	PACKH2 fail code
0x9F00C	PACK2 fail code
0x9F00D	PACKL4 fail code
0x9F00E	PACKHL2 fail code
0x9F00F	PACKLH2 fail code
0x9F010	SHLMB fail code
0x9F011	SHRMB fail code
0x9F012	SUB2 fail code
0x9F013	SUB4 fail code
0x9F014	SUBABS4 fail code
0x9F015	SWAP2 fail code
0x9F016	SWAP4 fail code
0x9F017	UNPKHU4 fail code
0x9F018	UNPKLU4 fail code
0x9F047	ADD2 .S2 fail code
0x9F048	ADD2 .D2 fail code
0x9F049	SUB2 .S2 fail code

Table 4. Error Codes (continued)

Code	Description
0x9F04A	SUB2 .D2 fail code
0x9F04B	PACK2 .S2 fail code
0x9F04C	PACKH2 .S2 fail code
0x9F04D	PACKHL2 .S2 fail code
0x9F04E	PACKLH2 .S2 fail code
0x9F053	SHLMB .S2 fail code
0x9F054	SHRMB .S2 fail code
0x9F055	SHR2 .S2 fail code
0x9F056	SHRU2 .S2 fail code
0x9F057	SWAP2 .S2 fail code
0x9F058	UNPKHU4 .S2 fail code
0x9F059	UNPKLU4 .S2 fail code
0x9F05A	SPACK2 .S2 fail code
0x9F05B	SPACKU4 .S2 fail code
0x9F05C	OR .L2 fail code
0x9F05D	OR .S2 fail code
0x9F05E	OR .D2 fail code
0x9F05F	XOR .S2 fail code
0x9F060	XOR .D2 fail code
0x9F061	MVK .L2 fail code
0x9F062	MVK .S2 fail code
0x9F063	MVK .D2 fail code
0x9F064	AND .L2 fail code
0x9F065	AND .S2 fail code
0x9F066	AND .D2 fail code
0x9F067	ANDN .S2 fail code
0x9F068	ANDN .D2 fail code
0x9F069	ADDAD .D2 fail code
0x9F06F	LDDW & STDW .D2 fail code
0x9F070	LDDW & STDW .D2 fail code
0x9F071	LDNDW & STNDW .D2 fail code
0x9F072	LDNDW & STNDW .D2 fail code
0x9F073	LDNW & STNW .D2 fail code
0x9F074	CMPEQ2 .S2 fail code
0x9F075	CMPGT2 & STNW .D2 fail code
0x9F076	CMPGTU4 .S2 fail code
0x9F077	ADDKPC .S2 fail code
0x9F078	BDEC .S2 fail code
0x9F079	BPOS if branch not taken to Test4 .S2 fail code
0x9F07A	BPOS if branch taken to Test4 .S2 fail code
0x9F019	AVG2 fail code
0x9F01A	AVGU4 fail code
0x9F01B	BITC4 fail code
0x9F01C	BITR fail code
0x9F01D	DEAL fail code
0x9F01E	DOTP2 fail code
0x9F01F	DOTPN2 fail code

Table 4. Error Codes (continued)

Code	Description
0x9F020	DOTPNRSU2 fail code
0x9F021	DOTPNRUS2 fail code
0x9F022	DOTPRUS2 fail code
0x9F023	DOTPRUS2 fail code
0x9F024	DOTPSU4 fail code
0x9F025	DOTPUS4 fail code
0x9F026	DOTPU4 fail code
0x9F027	GMPY4 fail code
0x9F028	GMPY4 fail code
0x9F029	MPY2 fail code
0x9F02A	MPY2 fail code
0x9F02B	MPYHI fail code
0x9F02C	MPYHI fail code
0x9F02D	MPYIH fail code
0x9F02E	MPYIH fail code
0x9F02F	MPYHIR fail code
0x9F030	MPYIHR fail code
0x9F031	MPYLI fail code
0x9F032	MPYLI fail code
0x9F033	MPYIH fail code
0x9F034	MPYIH fail code
0x9F035	MPYLIR fail code
0x9F036	MPYILR fail code
0x9F037	MPYSU4 fail code
0x9F038	MPYSU4 fail code
0x9F039	MPYUS4 fail code
0x9F03A	MPYUS4 fail code
0x9F03B	MPYU4 fail code
0x9F03C	MPYU4 fail code
0x9F03D	MVD fail code
0x9F03E	ROTL fail code
0x9F03F	SHFL fail code
0x9F040	SMPY2 fail code
0x9F041	SMPY2 fail code
0x9F042	SSHVL fail code
0x9F043	SSHVR fail code
0x9F044	XPND2 fail code
0x9F045	XPND4 fail code
0x9F04F	SADD2 error
0x9F050	SADDU4 error
0x9F051	SADDUS2 error
0x9F052	SADDSU2 error
0x2A001	ADDAB (15-bitconst) error
0x2A002	ADDAH (15-bitconst) error
0x2A003	ADDAW (15-bitconst) error
0x2A004	ADDSUB error
0x2A005	ADDSUB2 error

Table 4. Error Codes (continued)

Code	Description
0x2A006	CALLP error
0x2A007	CMPY error
0x2A008	CMPY error
0x2A009	CMPYR error
0x2A00A	CMPYR1 error
0x2A010	DDOTP4 error
0x2A011	DDOTP4 error
0x2A012	DDOTPH2 error
0x2A013	DDOTPH2 error
0x2A014	DDOTPH2R error
0x2A015	DDOTPL2 error
0x2A016	DDOTPL2 error
0x2A017	DDOTPL2R error
0x2A018	DINT error
0x2A019	DMV error
0x2A01A	DMV error
0x2A01B	DPACK2 error
0x2A01C	DPACK2 error
0x2A01D	DPACKX2 error
0x2A01E	DPACKX2 error
0x2A01F	GMPY error
0x2A020	MPY2IR error
0x2A021	MPY32 error
0x2A022	MPY32 error
0x2A023	MPY32 error
0x2A024	MPY32 error
0x2A025	MPY32SU error
0x2A026	MPY32SU error
0x2A027	MPY32U error
0x2A028	MPY32U error
0x2A029	MPY32US error
0x2A02A	MPY32US error
0x2A02B	RPACK2 error
0x2A02C	SADDSUB error
0x2A02D	SADDSUB error
0x2A02E	SADDSUB error
0x2A02F	SADDSUB2 error
0x2A030	SADDSUB2 error
0x2A031	SHFL3 error
0x2A032	SHFL3 error
0x2A033	SMPY32 error
0x2A034	SMPY32 error
0x2A035	SPBUFFER error
0x2A036	SPBUFFER error
0x2A037	SSUB2 error
0x2A038	SSUB2 error
0x2A039	SWE error

Table 4. Error Codes (continued)

Code	Description
0x2A03A	SWE error
0x2A03B	SWENR error
0x2A03C	SWENR error
0x2A03D	XORMPY error

4 References

1. *TMS320C62x Self-Check Program Applications Brief: Version 1.0* (Literature number SPRA635)
2. *TMS320C6000 CPU and Instruction Set Reference Guide* (Literature number SPRU189)
3. *TMS320C6000 DSP Peripherals Overview Reference Guide* (Literature number SPRU190)
4. *TMS320C6000 Chip Support Library API Reference Guide* (Literature number SPRU401)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265