



BCM5700/BCM5701

Diagnostic User's Guide

Revision 0.9 · Date 10/18/01

Prepared by: Austin Hui
Updated by: Paul Nguyen

Copyright © 2000, 2001 Broadcom Corporation
All Rights Reserved

No part of this document may be reproduced, in any form or by any means, without permission in writing from Broadcom Corporation.

Broadcom Corporation reserves the right to make changes to the products or information contained in this document without notice. No liability is assumed as a result of their use or application. No rights under any patent accompany the sale of any such products or information.

Epigram, InsideLine, and iLine10 are trademarks of Broadcom Corporation.

Broadcom Corporation
16125 Alton Parkway
Irvine, CA 92619-7013
www.broadcom.com

TABLE OF CONTENTS

1	INTRODUCTION.....	1
2	PREREQUISITES.....	2
3	DIAGNOSTIC TESTS.....	3
3.1	TEST NAMES.....	3
3.2	ERROR CODES.....	3
3.3	TEST DESCRIPTIONS.....	5
3.3.1	A1. Indirect Register Test.....	5
3.3.2	A2. Control Register Test.....	5
3.3.3	A3. BIST.....	16
3.3.4	A4. Interrupt Test.....	17
3.3.5	B1. Scratch Pad Test.....	17
3.3.6	B2. BD SRAM Test.....	17
3.3.7	B3. DMA SRAM Test.....	18
3.3.8	B4. MBUF SRAM Test.....	18
3.3.9	B5. External SRAM Test.....	18
3.3.10	B6. MBUF SRAM via DMA Test.....	18
3.3.11	B7. MBUF Special Test.....	19
3.3.12	C1. EEPROM Test.....	19
3.3.13	C2. CPU Test.....	19
3.3.14	C3. DMA Test.....	19
3.3.15	C4. MII Test.....	20
3.3.16	C5. VPD Test.....	20
3.3.17	C6. ASF Test.....	21
3.3.18	D1. Mac Loopback Test.....	21
3.3.19	D2. Phy Loopback Test.....	21
3.3.20	D3. RJ45 Loopback Test.....	21
4	COMMAND LINE OPTION PARAMETERS.....	23
5	EEPROM.TXT FORMAT.....	30
6	USER INTERFACE COMMANDS.....	32
7	SPECIAL INSTRUCTION.....	35
8	TEST AND FUNCTIONS DESCRIPTION.....	36
8.1	VPDWRITE.....	36
8.2	VPDREAD.....	36
8.3	VPDTEST.....	36
8.4	SEMODE.....	37
8.5	SERREAD.....	37
8.6	SEWRITE.....	38
8.7	SEPRG.....	38
8.8	SECFG.....	39
8.9	SETEST.....	40
8.10	CPUDRT.....	40
8.11	CPUDTT.....	41

8.12	CPUTEST	41
8.13	DMAR	42
8.14	DMAW	43
8.15	DMA_H	44
8.16	DMA_D	45
8.17	DMATEST	46
8.18	TXCFG.....	46
8.19	TXPKT	47
8.20	RXCFG.....	48
8.21	STSBLK.....	49
8.22	STATUSBLK.....	50
8.23	RESET.....	51
8.24	PHYCTRL	51
8.25	MACLPB.....	52
8.26	MREAD.....	52
8.27	MWRITE	53
8.28	MDEV	53
8.29	MIIMODE	53
8.30	MIITEST	54
8.31	READ	54
8.32	WRITE.....	55
8.33	MEMTEST	56
8.34	PMDCFG.....	58
8.35	PMPD	58
8.36	INTR.....	59
8.37	INTRTEST	59
8.38	MACHALT	60
8.39	ADDMC	60
8.40	DELMC.....	60
8.41	FTQ.....	60
8.42	MBUF.....	61
8.43	LOADDRV	61
8.44	UNLOADDRV.....	62
8.45	LOADFW	62
8.46	NICTEST	62
8.47	NICSTATS.....	65
8.48	REGTEST	67
8.49	DEBUG.....	67
8.50	PCISCAN.....	68
8.51	DIAGCFG.....	68
8.52	LOG.....	70
8.53	NOLOG	71
8.54	RADIX	71
8.55	EXIT, QUIT	71
8.56	BLAST	71
8.57	GPIOWRITE.....	73
8.58	GPIOREAD	73
8.59	VERSION.....	74
8.60	RINGINDEX.....	74
8.61	DOS.....	74
8.62	PXECPY	75
8.63	PCIINIT	75
8.64	INTRCTRL	75
8.65	UPGFRM	76

8.66	PKTTEST	76
8.67	TESTE	76
8.68	TESTD.....	77
8.69	LBERTRAM.....	77
8.70	DBERTRAM	77
8.71	BERTSTATS.....	78
8.72	BUSTEST	78

1 Introduction

This program runs in two modes: Manufacturing mode and Engineering mode. The mode is determined with the command line option or the configuration file. When the program is running in manufacturing mode, it starts to run all tests in the configuration. If it detects an error, it displays an error and exits the program. When the program is in engineering mode, it prompts user to enter commands. The commands are explained in the later chapters. This document provides the information on configuration file specification, command line options and engineering diagnostic commands on Broadcom Gigabit Ethernet adapter, in particular to check out the functionality of the BCM5700/5701 and its related components.

In general, this program has a set of default configuration. It is overwritten by configuration file. The command line option overwrites both default and the configuration files.

2 Prerequisites

The engineering diagnostic is executed under DOS protected mode, this requires dos4gw.exe to be placed in the same directory of the b57diag.exe.

OS: Dos 6.22

Software: b57diag.exe, dos4gw.exe, and cpu.bin.

Input File List: The following files should be found in the same location of the b57diag.exe.

ctrlreg.txt (mac registers test input file)
ctrlreg.reg (mac registers input file for engineering mode)
mac_pref.txt (mac address prefix number input file)
miireg.txt (mii registers test input file)
miireg.reg (mii registers input file for engineering mode)
pcicfg.reg (PCI configuration registers input file for engineering mode)
vpdwrite.txt (write vpd input file)
seprg.txt (program SEEPROM input file)
physcript.txt (Phy speed select)
wol.txt (Power Down MAC pattern input file)
firmware.bin (TX & RX CPUs Firmware file)
eeprom.bin (Serial EEPROM config input file)
cpu.bin (CPU test)

Output File List:

The following files will be generated in run time.

diagcfg.bin
b57diag.log

3 Diagnostic Tests

The tests are divided into four groups: Register Tests, Memory Tests, Miscellaneous Tests, and Data Tests. They numbered as group 'A', 'B', 'C', and 'D'.

3.1 Test Names

Group A.

- A1. Indirect Register Test
- A2. Control Register Test
- A3. BIST
- A4. Interrupt Test

Group B.

- B1. Scratch Pad Test
- B2. BD SRAM Test
- B3. DMA SRAM Test
- B4. MBUF SRAM Test
- B5. External SRAM Test
- B6. MBUF SRAM via DMA Test
- B7. MBUF Special Test

Group C.

- C1. EEPROM Test
- C2. CPU Test
- C3. DMA Test
- C4. MII Test
- C5. VPD Test
- C6. ASF Test

Group D.

- D1. Mac Loopback Test
- D2. Phy Loopback Test
- D3. RJ45 Loopback Test

3.2 Error Codes

Code	Message
------	---------

- | | |
|----|---|
| 1. | Got 0x%08x @ 0x%08x. Expected 0x%08x |
| 2. | Cannot run test while chip is running |
| 3. | Invalid NIC device |
| 4. | Read only bit %s got changed after writing zero at offset 0x%X |
| 5. | Read only bit %s got changed after writing 1's at offset 0x%X |
| 6. | Read/Write bit %s did not get cleared after writing zero at offset 0x%X |
| 7. | Read/Write bit %s did not get set after writing 1's at offset 0x%X |
| 8. | BIST failed |

9. Could not generate interrupt
10. Test aborted by user
11. Tx DMA:Got 0x%08x @ 0x%08x. Expected 0x%08x
12. Rx DMA:Got 0x%08x @ 0x%08x. Expected 0x%08x
13. Tx DMA failed
14. Rx DMA failed
15. Data error, got 0x%08X at 0x%08X, expected 0x%08X
16. Second read error, got 0x%08X at 0x%08X, expected 0x%08X
17. Failed writing EEPROM at 0x%04X
18. Failed reading EEPROM at 0x%04X
19. EEPROM data error, got 0x08X at 0x04X, expected 0x%08X
20. Cannot open file %s
21. Invalid CPU image file %s
22. Invalid CPU image size %d
23. Cannot allocate memory
24. Cannot reset CPU
25. Cannot release CPU
26. CPU test failed
27. Invalid Test Address Range
Valid NIC address is 0x%08x-0x%08x and exclude 0x%08x-0x%08x
28. DMA:Got 0x%08x @ 0x%08x. Expected 0x%08x
29. Unsupported PhyId %04X:%04X
30. Too many registers specified in the file, max is %d
31. Cannot write to VPD memory
32. VPD data error, got %08X @ 0x04X, expected %08X
33. No good link! Check Loopback plug
34. Cannot TX Packet!
35. Requested to Tx %d. Only %d is transmitted
36. Expected %d packets. Only %d good packets are received
%d unknown packets have been received.
%d bad packets have been received.
37. %c%d is an invalid Test
38. EEPROM checksum error
39. Error in reading WOL/PXE
40. Error in writing WOL/PXE
41. No external memory detected
42. DMA buffer %04X is large, size must be less than %04X
43. File size %d is too big, max is %d
44. Invalid %s
45. Failed writing 0x%x to 0x%x
46. *1
47. *1
48. *1
49. *1
50. Cannot perform task while chip is not running. (need driver)

- 51 Cannot open register define file or content is bad
- 52 ASF Reset bit did not self-cleared
- 53 ATTN_LOC %d cannot be mapped to %cX CPU event bit %d
- 54 %s Register is not cleared to zero after reset
- 55 Cannot start poll_ASF Timer
- 56 poll_ASF bit did not get reset after acknowledged
- 57 Timestamp Counter is not counting
- 58 %s Timer is not working
- 59 Cannot clear bit %s in %cX CPU event register
- 60 Invalid "EEPROM_FILENAME" file size, expected %d but only can read %d bytes
- 61 Invalid magic value in %s, expected %08x but found %08x
- 62 Invalid manufacture revision, expected %c but found %c
- 63 Invalid Boot Code revision, expected %d.%d but found %d.%d
- 64 Cannot write to EEPROM
- 65 Cannot read from EEPROM
- 66 Invalid Checksum
- 67 Invalid Magic Value
- 68 Invalid MAC address, expected %02X-%02X-%02X-%02X-%02X-%02X
- 69 Slot error, expected an UUT to be found at location %02X:%02X:00
- 70 Adjacent memory has been corrupted while testing block 0x%08x-0x%08x
Got 0x%08x @ address 0x%08x. Expected 0x%08x

*1 Internal Use. Program will not generate this error.

3.3 Test Descriptions

3.3.1 A1. Indirect Register Test

Command: regtest -i

Function: Using indirect addressing method, writing increment data into MAC hash Register table and read back for verification. The memory read/write is done 100 times while increment test data.

Default: Test Enabled

3.3.2 A2. Control Register Test

Command: regtest

Function: Each Register specified in the configuration contents read only bit and read/write bit defines. The test writing zero and one into the test bits to insure the read only bits are not changed, and read/write bits are changed accordingly.

Default: Test Enabled.

Default Register table

The test will try to read the register configuration file 'ctrlreg.txt' for the register defines. If the file does not exist, the following table is used:

Offset	R/O Mask	R/W Mask
0x00000400	0x00000000	0x007fff8c
0x00000404	0x03800107	0x00000000
0x00000408	0x00000000	0x07c01400
0x0000040c	0x00000000	0xc000007f
0x00000410	0x00000000	0x0000ffff
0x00000414	0x00000000	0xffffffff
0x00000418	0x00000000	0x0000ffff
0x0000041c	0x00000000	0xffffffff
0x00000420	0x00000000	0x0000ffff
0x00000424	0x00000000	0xffffffff
0x00000428	0x00000000	0x0000ffff
0x0000042c	0x00000000	0xffffffff
0x00000430	0x00000000	0xffffffff
0x00000434	0x00000000	0x0fffffff
0x0000043c	0x00000000	0x0000ffff
0x00000440	0x00000000	0x001fffffff
0x00000444	0x00000000	0x0000ffff
0x00000448	0x0000ffff	0x00000000
0x0000044c	0x00000000	0x0fffffff
0x00000450	0x00000000	0x00000001
0x00000454	0x00000000	0x00008013
0x00000458	0x00000000	0x00000000
0x0000045c	0x00000000	0x00000070
0x00000460	0x00000009	0x00000000
0x00000464	0x00000000	0x00003fff
0x00000468	0x00000000	0x000007fc
0x0000046c	0x00000001	0x00000000
0x00000470	0x00000000	0xffffffff
0x00000474	0x00000000	0xffffffff
0x00000478	0x00000000	0xffffffff
0x0000047c	0x00000000	0xffffffff
0x00000480	0x00000000	0xffffffff
0x00000484	0x00000000	0xffffffff
0x00000488	0x00000000	0xffffffff
0x0000048c	0x00000000	0xffffffff
0x00000490	0x00000000	0xffffffff
0x00000494	0x00000000	0xffffffff
0x00000498	0x00000000	0xffffffff
0x0000049c	0x00000000	0xffffffff

0x000004a0	0x00000000	0xffffffff
0x000004a4	0x00000000	0xffffffff
0x000004a8	0x00000000	0xffffffff
0x000004ac	0x00000000	0xffffffff
0x000004b0	0x00000000	0xffffffff
0x000004b4	0x00000000	0xffffffff
0x000004b8	0x00000000	0xffffffff
0x000004bc	0x00000000	0xffffffff
0x000004c0	0x00000000	0xffffffff
0x000004c4	0x00000000	0xffffffff
0x000004c8	0x00000000	0xffffffff
0x000004cc	0x00000000	0xffffffff
0x000004d0	0x00000000	0xffffffff
0x000004d4	0x00000000	0xffffffff
0x000004d8	0x00000000	0xffffffff
0x000004dc	0x00000000	0xffffffff
0x000004e0	0x00000000	0xffffffff
0x000004e4	0x00000000	0xffffffff
0x000004e8	0x00000000	0xffffffff
0x000004ec	0x00000000	0xffffffff
0x000004f0	0x00000000	0xffffffff
0x000004f4	0x00000000	0xffffffff
0x000004f8	0x00000000	0xffffffff
0x000004fc	0x00000000	0xffffffff
0x00000500	0x00000000	0x000000f8
0x00000c00	0x00000000	0x00000006
0x00000c04	0x00000004	0x00000000
0x00000c08	0x00000000	0x00000003
0x00000c0c	0x00000000	0x00ffffff
0x00000c80	0xffffffff	0x00000000
0x00000c84	0xffffffff	0x00000000
0x00000c88	0xffffffff	0x00000000
0x00000c8c	0xffffffff	0x00000000
0x00000c90	0xffffffff	0x00000000
0x00000c94	0xffffffff	0x00000000
0x00000c98	0xffffffff	0x00000000
0x00000c9c	0xffffffff	0x00000000
0x00000ca0	0xffffffff	0x00000000
0x00000ca4	0xffffffff	0x00000000
0x00000ca8	0xffffffff	0x00000000
0x00000cac	0xffffffff	0x00000000
0x00000cb0	0xffffffff	0x00000000
0x00000cb4	0xffffffff	0x00000000
0x00000cb8	0xffffffff	0x00000000
0x00000cbc	0xffffffff	0x00000000

```

0x00000cc0 0xffffffff 0x00000000
0x00000cc4 0xffffffff 0x00000000
0x00000cc8 0xffffffff 0x00000000
0x00000ccc 0xffffffff 0x00000000
0x00000cd0 0xffffffff 0x00000000
0x00000cd4 0xffffffff 0x00000000
0x00000cd8 0xffffffff 0x00000000
0x00000cdc 0xffffffff 0x00000000
0x00001000 0x00000000 0x00000002
0x00001400 0x00000000 0x00000006
0x00001404 0x00000004 0x00000000
0x00001408 0x0000ffff 0x00000000
0x00001440 0x0000000f 0x00000000
0x00001444 0x0000000f 0x00000000
0x00001448 0x0000000f 0x00000000
0x0000144c 0x0000000f 0x00000000
0x00001450 0x0000000f 0x00000000
0x00001454 0x0000000f 0x00000000
0x00001458 0x0000000f 0x00000000
0x0000145c 0x0000000f 0x00000000
0x00001460 0x0000000f 0x00000000
0x00001464 0x0000000f 0x00000000
0x00001468 0x0000000f 0x00000000
0x0000146c 0x0000000f 0x00000000
0x00001470 0x0000000f 0x00000000
0x00001474 0x0000000f 0x00000000
0x00001478 0x0000000f 0x00000000
0x0000147c 0x0000000f 0x00000000
0x00001800 0x00000000 0x00000006
0x00001804 0x00000004 0x00000000
0x00001808 0xffffffff 0x00000000
0x0000180c 0xffffffff 0x00000000
0x00001810 0xffffffff 0x00000000
0x00001814 0xffffffff 0x00000000
0x00001818 0xffffffff 0x00000000
0x0000181c 0xffffffff 0x00000000
0x00001820 0xffffffff 0x00000000
0x00001824 0xffffffff 0x00000000
0x00001828 0xffffffff 0x00000000
0x0000182c 0xffffffff 0x00000000
0x00001830 0xffffffff 0x00000000
0x00001834 0xffffffff 0x00000000
0x00001838 0xffffffff 0x00000000
0x0000183c 0xffffffff 0x00000000
0x00001840 0xffffffff 0x00000000

```

```

0x00001844 0xffffffff 0x00000000
0x00001c00 0x00000000 0x00000002
0x00002000 0x00000000 0x0000001e
0x00002004 0x0000001c 0x00000000
0x0000200c 0xffffffff 0x00000000
0x00002010 0x00000000 0x00003fff
0x00002014 0x00000000 0x00000003
0x00002100 0xffffffff 0x00000000
0x00002104 0xffffffff 0x00000000
0x00002108 0xffffffff 0x00000000
0x00002110 0xffffffff 0x00000000
0x00002114 0xffffffff 0x00000000
0x00002118 0xffffffff 0x00000000
0x00002120 0xffffffff 0x00000000
0x00002124 0xffffffff 0x00000000
0x00002128 0xffffffff 0x00000000
0x00002130 0xffffffff 0x00000000
0x00002134 0xffffffff 0x00000000
0x00002138 0xffffffff 0x00000000
0x00002140 0xffffffff 0x00000000
0x00002144 0xffffffff 0x00000000
0x00002148 0xffffffff 0x00000000
0x00002150 0xffffffff 0x00000000
0x00002154 0xffffffff 0x00000000
0x00002158 0xffffffff 0x00000000
0x00002160 0xffffffff 0x00000000
0x00002164 0xffffffff 0x00000000
0x00002168 0xffffffff 0x00000000
0x00002170 0xffffffff 0x00000000
0x00002174 0xffffffff 0x00000000
0x00002178 0xffffffff 0x00000000
0x00002180 0xffffffff 0x00000000
0x00002184 0xffffffff 0x00000000
0x00002188 0xffffffff 0x00000000
0x00002190 0xffffffff 0x00000000
0x00002194 0xffffffff 0x00000000
0x00002198 0xffffffff 0x00000000
0x000021a0 0xffffffff 0x00000000
0x000021a4 0xffffffff 0x00000000
0x000021a8 0xffffffff 0x00000000
0x000021b0 0xffffffff 0x00000000
0x000021b4 0xffffffff 0x00000000
0x000021b8 0xffffffff 0x00000000
0x000021c0 0xffffffff 0x00000000
0x000021c4 0xffffffff 0x00000000

```

```

0x000021c8 0xffffffff 0x00000000
0x000021d0 0xffffffff 0x00000000
0x000021d4 0xffffffff 0x00000000
0x000021d8 0xffffffff 0x00000000
0x000021e0 0xffffffff 0x00000000
0x000021e4 0xffffffff 0x00000000
0x000021e8 0xffffffff 0x00000000
0x000021f0 0xffffffff 0x00000000
0x000021f4 0xffffffff 0x00000000
0x000021f8 0xffffffff 0x00000000
0x00002200 0x000003ff 0x00000000
0x00002204 0x000003ff 0x00000000
0x00002208 0x000003ff 0x00000000
0x0000220c 0x000003ff 0x00000000
0x00002210 0x000003ff 0x00000000
0x00002214 0x000003ff 0x00000000
0x00002218 0x000003ff 0x00000000
0x0000221c 0x000003ff 0x00000000
0x00002220 0x000003ff 0x00000000
0x00002224 0x000003ff 0x00000000
0x00002228 0x000003ff 0x00000000
0x0000222c 0x000003ff 0x00000000
0x00002230 0x000003ff 0x00000000
0x00002234 0x000003ff 0x00000000
0x00002238 0x000003ff 0x00000000
0x0000223c 0x000003ff 0x00000000
0x00002240 0x000003ff 0x00000000
0x00002244 0x000003ff 0x00000000
0x00002248 0x000003ff 0x00000000
0x0000224c 0x000003ff 0x00000000
0x00002250 0x000003ff 0x00000000
0x00002254 0x000003ff 0x00000000
0x00002258 0x000003ff 0x00000000
0x00002400 0x00000000 0x0000001e
0x00002404 0x0000001c 0x00000000
0x00002408 0x00000000 0x0000ffff
0x00002440 0x00000000 0xffffffff
0x00002444 0x00000000 0xffffffff
0x0000244c 0x00000000 0xffffffff
0x00002450 0x00000000 0xffffffff
0x00002454 0x00000000 0xffffffff
0x0000245c 0x00000000 0xffffffff
0x00002460 0x00000000 0xffffffff
0x00002464 0x00000000 0xffffffff
0x0000246c 0x00000000 0xffffffff

```



```

0x00002470 0xffffffff 0x00000000
0x00002474 0xffffffff 0x00000000
0x00002478 0xffffffff 0x00000000
0x00002480 0xffffffff 0x00000000
0x00002484 0xffffffff 0x00000000
0x00002488 0xffffffff 0x00000000
0x0000248c 0xffffffff 0x00000000
0x00002490 0xffffffff 0x00000000
0x00002494 0xffffffff 0x00000000
0x00002498 0xffffffff 0x00000000
0x0000249c 0xffffffff 0x00000000
0x000024a0 0xffffffff 0x00000000
0x000024a4 0xffffffff 0x00000000
0x000024a8 0xffffffff 0x00000000
0x000024ac 0xffffffff 0x00000000
0x000024b0 0xffffffff 0x00000000
0x000024b4 0xffffffff 0x00000000
0x000024b8 0xffffffff 0x00000000
0x000024bc 0xffffffff 0x00000000
0x000024c0 0xffffffff 0x00000000
0x00002800 0x00000000 0x00000002
0x00002c00 0x00000000 0x00000004
0x00002c04 0x00000004 0x00000000
0x00002c14 0x00000000 0xffffffff
0x00002c18 0x00000000 0xffffffff
0x00002c1c 0x00000000 0xffffffff
0x00003000 0x00000000 0x00000006
0x00003004 0x00000004 0x00000000
0x00003008 0x00000000 0x0000000f
0x0000300c 0x00000000 0x0000001f
0x00003010 0x00000000 0x0000003f
0x00003400 0x00000000 0x00000004
0x00003404 0x00000004 0x00000000
0x00003800 0x00000000 0x00000006
0x00003804 0x00000004 0x00000000
0x00003c00 0x00000000 0x00000002
0x00003c04 0x00000004 0x00000000
0x00003c08 0x00000000 0xffffffff
0x00003c0c 0x00000000 0xffffffff
0x00003c10 0x00000000 0xffffffff
0x00003c14 0x00000000 0xffffffff
0x00003c18 0x00000000 0xffffffff
0x00003c1c 0x00000000 0xffffffff
0x00003c20 0x00000000 0xffffffff
0x00003c24 0x00000000 0xffffffff

```

0x00003c28	0x00000000	0xffffffff
0x00003c30	0x00000000	0xffffffff
0x00003c34	0x00000000	0xffffffff
0x00003c38	0x00000000	0xffffffff
0x00003c3c	0x00000000	0xffffffff
0x00003c40	0xffffffff	0x00000000
0x00003c44	0xffffffff	0x00000000
0x00003c80	0x00000000	0x000007ff
0x00003c84	0x00000000	0x000007ff
0x00003c88	0x00000000	0x000007ff
0x00003c8c	0x00000000	0x000007ff
0x00003c90	0x00000000	0x000007ff
0x00003c94	0x00000000	0x000007ff
0x00003c98	0x00000000	0x000007ff
0x00003c9c	0x00000000	0x000007ff
0x00003ca0	0x00000000	0x000007ff
0x00003ca4	0x00000000	0x000007ff
0x00003ca8	0x00000000	0x000007ff
0x00003cac	0x00000000	0x000007ff
0x00003cb0	0x00000000	0x000007ff
0x00003cb4	0x00000000	0x000007ff
0x00003cb8	0x00000000	0x000007ff
0x00003cbc	0x00000000	0x000007ff
0x00003cc0	0x00000000	0x000001ff
0x00003cc4	0x00000000	0x000001ff
0x00003cc8	0x00000000	0x000001ff
0x00003ccc	0x00000000	0x000001ff
0x00003cd0	0x00000000	0x000001ff
0x00003cd4	0x00000000	0x000001ff
0x00003cd8	0x00000000	0x000001ff
0x00003cdc	0x00000000	0x000001ff
0x00003ce0	0x00000000	0x000001ff
0x00003ce4	0x00000000	0x000001ff
0x00003ce8	0x00000000	0x000001ff
0x00003cec	0x00000000	0x000001ff
0x00003cf0	0x00000000	0x000001ff
0x00003cf4	0x00000000	0x000001ff
0x00003cf8	0x00000000	0x000001ff
0x00003cfc	0x00000000	0x000001ff
0x00004000	0x00000000	0x03ffffff
0x00004004	0x00000000	0x00000000
0x00004008	0x00000000	0x001fffff
0x0000400c	0x00000000	0x001fffff
0x00004400	0x00000000	0x00000000
0x00004404	0x00000014	0x00000000

```

0x00004408 0x00000000 0x007ffffff
0x0000440c 0x00000000 0x007ffffff
0x00004410 0x00000000 0x0000ffffff
0x00004414 0x00000000 0x0000ffffff
0x00004418 0x00000000 0x0000ffffff
0x0000441c 0x00000000 0x0000ffffff
0x00004420 0xffffffff 0x00000000
0x00004424 0x00000000 0x0000ffffff
0x00004428 0xffffffff 0x00000000
0x0000442c 0xffffffff 0x00000000
0x00004430 0xffffffff 0x00000000
0x00004434 0x00000000 0x000001ff
0x00004438 0x00000000 0x000001ff
0x00004440 0xffffffff 0x00000000
0x00004448 0xffffffff 0x00000000
0x0000444c 0xffffffff 0x00000000
0x00004450 0xffffffff 0x00000000
0x00004454 0xffffffff 0x00000000
0x00004800 0x00000000 0x000003fc
0x00004804 0x00000000 0x00000000
0x00004c00 0x00000000 0x000003fc
0x00004c04 0x000003fc 0x00000000
0x00005800 0x00000000 0xffffffff
0x00005804 0x00000000 0xffffffff
0x00005808 0x00000000 0xffffffff
0x0000580c 0x00000000 0xffffffff
0x00005810 0x00000000 0xffffffff
0x00005814 0x00000000 0xffffffff
0x00005818 0x00000000 0xffffffff
0x0000581c 0x00000000 0xffffffff
0x00005820 0x00000000 0xffffffff
0x00005824 0x00000000 0xffffffff
0x00005828 0x00000000 0xffffffff
0x0000582c 0x00000000 0xffffffff
0x00005830 0x00000000 0xffffffff
0x00005834 0x00000000 0xffffffff
0x00005838 0x00000000 0xffffffff
0x0000583c 0x00000000 0xffffffff
0x00005840 0x00000000 0xffffffff
0x00005844 0x00000000 0xffffffff
0x00005848 0x00000000 0xffffffff
0x0000584c 0x00000000 0xffffffff
0x00005850 0x00000000 0xffffffff
0x00005854 0x00000000 0xffffffff
0x00005858 0x00000000 0xffffffff

```

```

0x0000585c 0x00000000 0xffffffff
0x00005860 0x00000000 0xffffffff
0x00005864 0x00000000 0xffffffff
0x00005868 0x00000000 0xffffffff
0x0000586c 0x00000000 0xffffffff
0x00005870 0x00000000 0xffffffff
0x00005874 0x00000000 0xffffffff
0x00005878 0x00000000 0xffffffff
0x0000587c 0x00000000 0xffffffff
0x00005880 0x00000000 0xffffffff
0x00005884 0x00000000 0xffffffff
0x00005888 0x00000000 0xffffffff
0x0000588c 0x00000000 0xffffffff
0x00005890 0x00000000 0xffffffff
0x00005894 0x00000000 0xffffffff
0x00005898 0x00000000 0xffffffff
0x0000589c 0x00000000 0xffffffff
0x000058a0 0x00000000 0xffffffff
0x000058a4 0x00000000 0xffffffff
0x000058a8 0x00000000 0xffffffff
0x000058ac 0x00000000 0xffffffff
0x000058b0 0x00000000 0xffffffff
0x000058b4 0x00000000 0xffffffff
0x000058b8 0x00000000 0xffffffff
0x000058bc 0x00000000 0xffffffff
0x000058c0 0x00000000 0xffffffff
0x000058c4 0x00000000 0xffffffff
0x000058c8 0x00000000 0xffffffff
0x000058cc 0x00000000 0xffffffff
0x000058d0 0x00000000 0xffffffff
0x000058d4 0x00000000 0xffffffff
0x000058d8 0x00000000 0xffffffff
0x000058dc 0x00000000 0xffffffff
0x000058e0 0x00000000 0xffffffff
0x000058e4 0x00000000 0xffffffff
0x000058e8 0x00000000 0xffffffff
0x000058ec 0x00000000 0xffffffff
0x000058f0 0x00000000 0xffffffff
0x000058f4 0x00000000 0xffffffff
0x000058f8 0x00000000 0xffffffff
0x000058fc 0x00000000 0xffffffff
0x00005900 0x00000000 0xffffffff
0x00005904 0x00000000 0xffffffff
0x00005908 0x00000000 0xffffffff
0x0000590c 0x00000000 0xffffffff

```

0x00005910	0x00000000	0xffffffff
0x00005914	0x00000000	0xffffffff
0x00005918	0x00000000	0xffffffff
0x0000591c	0x00000000	0xffffffff
0x00005920	0x00000000	0xffffffff
0x00005924	0x00000000	0xffffffff
0x00005928	0x00000000	0xffffffff
0x0000592c	0x00000000	0xffffffff
0x00005930	0x00000000	0xffffffff
0x00005934	0x00000000	0xffffffff
0x00005938	0x00000000	0xffffffff
0x0000593c	0x00000000	0xffffffff
0x00005940	0x00000000	0xffffffff
0x00005944	0x00000000	0xffffffff
0x00005948	0x00000000	0xffffffff
0x0000594c	0x00000000	0xffffffff
0x00005950	0x00000000	0xffffffff
0x00005954	0x00000000	0xffffffff
0x00005958	0x00000000	0xffffffff
0x0000595c	0x00000000	0xffffffff
0x00005960	0x00000000	0xffffffff
0x00005964	0x00000000	0xffffffff
0x00005968	0x00000000	0xffffffff
0x0000596c	0x00000000	0xffffffff
0x00005970	0x00000000	0xffffffff
0x00005974	0x00000000	0xffffffff
0x00005978	0x00000000	0xffffffff
0x0000597c	0x00000000	0xffffffff
0x00005980	0x00000000	0xffffffff
0x00005984	0x00000000	0xffffffff
0x00005988	0x00000000	0xffffffff
0x0000598c	0x00000000	0xffffffff
0x00005990	0x00000000	0xffffffff
0x00005994	0x00000000	0xffffffff
0x00005998	0x00000000	0xffffffff
0x0000599c	0x00000000	0xffffffff
0x000059a0	0x00000000	0xffffffff
0x000059a4	0x00000000	0xffffffff
0x000059a8	0x00000000	0xffffffff
0x000059ac	0x00000000	0xffffffff
0x000059b0	0x00000000	0xffffffff
0x000059b4	0x00000000	0xffffffff
0x000059b8	0x00000000	0xffffffff
0x000059bc	0x00000000	0xffffffff
0x000059c0	0x00000000	0xffffffff

```

0x000059c4 0x00000000 0xffffffff
0x000059c8 0x00000000 0xffffffff
0x000059cc 0x00000000 0xffffffff
0x000059d0 0x00000000 0xffffffff
0x000059d4 0x00000000 0xffffffff
0x000059d8 0x00000000 0xffffffff
0x000059dc 0x00000000 0xffffffff
0x000059e0 0x00000000 0xffffffff
0x000059e4 0x00000000 0xffffffff
0x000059e8 0x00000000 0xffffffff
0x000059ec 0x00000000 0xffffffff
0x000059f0 0x00000000 0xffffffff
0x000059f4 0x00000000 0xffffffff
0x000059f8 0x00000000 0xffffffff
0x000059fc 0x00000000 0xffffffff
0x00005a04 0x00000000 0xffffffff
0x00005a0c 0x00000000 0xffffffff
0x00005c10 0x00000000 0x00000005
0x00005c14 0xffffffff 0x00000000
0x00005c24 0xffffffff 0x00000000
0x00005c34 0xffffffff 0x00000000
0x00005c44 0xffffffff 0x00000000
0x00005c54 0xffffffff 0x00000000
0x00005c64 0xffffffff 0x00000000
0x00005c74 0xffffffff 0x00000000
0x00005c84 0xffffffff 0x00000000
0x00005c94 0xffffffff 0x00000000
0x00005ca4 0xffffffff 0x00000000
0x00005cb4 0xffffffff 0x00000000
0x00005cc4 0xffffffff 0x00000000
0x00005cd4 0xffffffff 0x00000000
0x00005ce4 0xffffffff 0x00000000
0x00005cf4 0xffffffff 0x00000000
0x00005d04 0xffffffff 0x00000000
0x00005d14 0xffffffff 0x00000000
0x00006000 0x00000000 0x0000007e
0x00006400 0x00000000 0x00000002
0x00006800 0x00000000 0xffffffff7
0x00006804 0x0001e000 0x00000ffe
0x00006808 0x00000000 0x03ffc008

```

3.3.3 A3. BIST

Command: bist

Function: Hardware Built-In-Self-Test (BIST). This test initiates BIST, and wait for the test result returned by hardware.

Default: Due to the intermittent failure, this test is currently disabled by default

3.3.4 A4. Interrupt Test

Command: intrtest

Function: This test verifies the interrupt functionality. It enables interrupt and waits for interrupt to occur. It waits for 500ms and reports error if could not generate interrupts.

Default: Enabled

3.3.5 B1. Scratch Pad Test

Command: memtest -s

Function: This test tests the scratch pad SRAM on board. The following tests are performed:

Data Pattern Test: Write test data into SRAM, read back to ensure data is correct. The test data used is 0x00000000, 0xffffffff, 0xaa55aa55, and 0x55aa55aa.

Alternate Data Pattern Test: Write test data into SRAM. Write complement test data into next address. Read back both data to insure the data is correct. After the test, the program reads back data one more time to insure the data stays correct. The test data used is 0x00000000, 0xffffffff, 0xaa55aa55, and 0x55aa55aa.

Address Test: Write each address with unique increment data. Read back data to insure data is correct. After fill the entire data with the unique data, the program reads back data again to insure data stays the same.

WalkingOne bit Test: For each address. Data one is written and read back for testing. Then shift the data left one bit, so the data becomes two and do the same test again. It repeats for 32 times until the test bit is shifted out of test data. The same is test is repeated for entire test range.

Pseudo Random Data Test: A pre-calculated pseudo random data is used to write a unique data into each test RAM. After the first pass the test, the program reads back one more time to insure data stays correct.

Default: Enabled

3.3.6 B2. BD SRAM Test

Command: memtest -b

Function: This test tests the BD SRAM. This performs exact the same way of testing as described in B1. Scratch Pad Test.

Default: Enabled

3.3.7 B3. DMA SRAM Test

Command: memtest -d

Function: It tests DMA SRAM by performing the tests described in test B1. The Scratch Pad Test.

Default: Enabled

3.3.8 B4. MBUF SRAM Test

Command: memtest -m

Function: It tests DMA SRAM by performing the tests described in test B1. The Scratch Pad Test.

Default: Enabled

3.3.9 B5. External SRAM Test

Command: memtest -e

Function: It tests DMA SRAM by performing the tests described in test B1. The Scratch Pad Test.

Default: Disabled

3.3.10 B6. MBUF SRAM via DMA Test

Command: memtest -x

Function: Eight test pattern data are used in the test. They are described below. A 0x1000 sized data buffer is used for this test. Before each pattern test, the buffer is initialized and filled with the test pattern. It then, performs size 0x1000 transmit DMA from host buffer to NIC MBUF memory. Verify the data integrity in MBUF against host memory and repeat the DMA for the entire MBUF buffer. Then it performs receive DMA from NIC to host. The 0x1000-byte test buffer is cleared to zero before each receive-DMA. Verify the data integrity and test is repeated for the entire MBUF SRAM range.

Test Pattern	Description
"16 00's 16 FF's"	Full the entire host DMA buffer with 16 bytes of 00's and then 16 bytes of FF's.

“16 FF's 16 0's”	Full the entire host DMA buffer with 16 bytes of 00's and then 16 bytes of FF's.
“32 00's 32 FF's”	Full the entire host DMA buffer with 32 bytes of 00's and then 32 bytes of FF's.
“32 FF's 32 00's”	Full the entire host DMA buffer with 32 bytes of FF's and then 32 bytes of 00's.
“00000000's”	Full the entire host DMA buffer with all zeros.
“FFFFFFFF's”	Full the entire host DMA buffer with all FF's.
“AA55AA55's”	Full the entire host DMA buffer with data 0xAA55AA55.
“55AA55AA's”	Full the entire host DMA buffer with data 0xAA55AA55.

Default: Enabled

3.3.11 B7. MBUF Special Test

Command: memtest -c

Function: Initially, full up the entire test block with the inverted test pattern. Then, for every 128-byte boundary, rewrite the test pattern. Verify all test memories contains the correct test data, as it should be.

Default: Disabled

3.3.12 C1. EEPROM Test

Command: setest

Function: An increment test data is used in EEPROM test. It fills the test data into the test range and read back to verify the content. After the test, it fills data with zero to clear the memory.

Default: Enabled

3.3.13 C2. CPU Test

Command: cputest

Function: This test opens the file cpu.bin. If file exists and content is good, it loads code to rx and tx CPU and verifies CPU execution.

Default: Enabled

3.3.14 C3. DMA Test

Command: dmatest

Function: Both high and low priorities DMA are tested. It moves data from host memory to NIC SRAM, verifies data, and then moves data back to host memory again to verify data.

Default: Enabled

3.3.15 C4. MII Test

Command: miitest

Function: The function is identical to A2. Control Register Test. Each Register specified in the configuration contents read only bit and read/write bit defines. The test writing zero and one into the test bits to insure the read only bits value are not changed, and read/write bits are changed accordingly.

Default: Test Enabled.

Default Register table

The test will try to read the register configuration file 'miireg.txt' for the register defines. If the file does not exists, the following table is used:

Offset	R/O Mask	R/W Mask
0x00	0x0000	0x7180
0x02	0xffff	0x0000
0x03	0xffff	0x0000
0x04	0x0000	0xffff
0x05	0xffff	0x0000
0x06	0x0001	0x0000
0x07	0x0800	0xb7ff
0x08	0xffff	0x0000
0x09	0x0000	0xff00
0x0a	0x7c00	0x0000
0x10	0x0000	0xffbf
0x11	0x3300	0x0000
0x19	0x001f	0x0000
0x1e	0x0000	0xffff
0x1f	0x0000	0xffff

3.3.16 C5. VPD Test

Command: vpdtest

Function: It saves the content of VPD first before perform the test. Once it is done, it writes one of the five pattern test data, 0xff, 0xaa, 0x55, increment data, or decrement data, into VPD memory. By default, increment data pattern is used. It writes and reads back the data for the entire test range, and then restores the original content.

Default: Disabled

3.3.17 C6. ASF Test

Command: asftest

Function:m

1. Reset test.

Setting reset bit, poll for self-clearing. Verify reset value of registers.

2. Event Mapping Test

Setting SMB_ATTEN bit. By changing ASF_ATTEN LOC bits, verify the mapping bits in TX_CPU or RX_CPU event bits.

3. Counter Test

Clear WG_TO, HB_TO, PA_TO, PL_TO, RT_TO bits by setting those bits. Make sure the bits clear.

Clear Timestamp Counter. Writing a value 1 into each PL, PA, HB, WG, RT counters. Set TSC_EN bit.

Poll each PA_TO bit and count up to 50 times. Check if PL_TO gets set at the end of 50 times. Continue to count up to 200 times. Check if all other TO bits are set and verify Timestamp Counter is incremented.

3.3.18 D1. Mac Loopback Test

Command: pkttest -m

Function: This is internal loopback data transmit/receive test. It initializes MAC into internal loopback mode, and transmits 100 packets. The data should be routed back to receive channel and receive by the receive routine, which verifies the integrity of data. One Giga bit rate is used for this test.

Default: Enabled

3.3.19 D2. Phy Loopback Test

Command: pkttest -p

Function: This test is same as D1. Mac Loopback Test except, the data is routed back via physical layer device. One Giga bit rate is used for this test.

Default: Enabled

3.3.20 D3. RJ45 Loopback Test

Command: pkttest -e

Function: This is external loopback test. From the UUT point of view, no loopback mode is configured. The data expected to be routed back by RJ45 loopback connector. 100M/s and 1000M/s are used for this test.

Default: Disabled

4 Command line option parameters

When user invoke this program, a set of option parameter can be used to overwrite the configuration file or the default configuration. This section summarizes the options. The options are case sensitive.

-c <num> specify UUT device number

When more than one device is in the system, the devices are number starting from zero. For example, if there is three devices detected, the device is numbered as 0,1,and 2. In this case, by entered the parameter `-c 2` will select the last found device as default UUT.

In manufacture testing mode, by default, all devices are tested; however, if this option is used, only that selected device is tested.

Example: `-c 2`

-coe Continue on error

Enter this option will allow the test to continue even if with error. The software will reset the iteration value as well as the error code.

Example: `-coe`

-l <file> log file

All diagnostic output can be saved in a log file. Type log file name is specified by this option. The default is no log file.

Example: `-l mylogfile.txt`

-w <value> enable WOL programming in manufacture mode

After a successful manufacturing testing, the program will program WOL to either enable or disable mode.

Value = 0 to disable WOL and else to enable it.

When `-f` is entered, software uses eeprom.bin's content for WOL setting.

When `-w` is entered with `-f`, `-w` is ignored.

Example: `-w 1`

-x <value> enable PXE in manufacture mode

After a successful manufacturing testing, the program will program PXE to either enable or disable mode.

Value = 0 to disable PXE and else to enable it.

When -f is entered, software uses eeprom.bin's content for PXE setting.

When -x is entered with -f, -x is ignored.

Example: -x 0

-t <id> disable test

-T <id> enable test

A certain test is enabled or disabled by default. User can overwrite the enabling status by those options. The test id must start with a letter 'A', 'B', 'C', or 'D' to indication the group and followed by test numbers. Each digit of number represents the sub-test number. For example, if the user wants to disable test A1 and A3. The option -t A13 should be entered. If no test numbers entered, all tests in that group are selected. For the tests not specified, the default setting will be used.

Example -t A15BC1 -T C4 -t D2

 This disables A1, A5, B1, B2, B3, B4, B5, B6, C1, D2 and enables C4

Default Setting:

Enabled Tests:

- A1. Indirect Register Test
- A2. Control Register Test
- A4. Interrupt Test
- B1. Scratch Pad Test
- B2. BD SRAM Test
- B3. DMA SRAM Test
- B4. MBUF SRAM Test
- B6. MBUF SRAM via DMA Test
- C1. EEPROM Test
- C1. CPU Test
- C2. DMA Test
- C3. MII Test
- D1. Mac Loopback Test
- D2. Phy Loopback Test

Disabled Tests:

- A3. BIST
- B5. External SRAM Test
- C4. VPD Test
- C6. ASF Test
- D3. RJ45 Loopback Test

-I <num> iteration number

Use this option to specify the number of times the tests to be run. The default is run one time. A number zero indicates loop forever. A control-C or control-break key can be used to break the loop. Any error detected will also stop testing after reporting the error.

Example: -I 5
Run tests five times.

-ver display current version number

If this option is entered, it displays the software version number/silkscreen revision and then exits the program.

-srev display internal silicon version number

If this option is entered, it displays the software version number/silicon version and then exits the program.

-e <code> Encryption Code

This option is required to use option -geneep, -f, -m, -n, and -s.

-geneep Generate eeprom.bin file from eeprom.txt

A password is needed to run this option. With this option, it updates eeprom.bin with the specifications defined in eeprom.txt.

Keywords	Argument
MAC_PREFIX	Three 8-bit hex numbers separated by column
POWER_DISSIPATED	Four decimal number ranges 0 to 255 separated by column
POWER_CONSUMED	Four decimal number ranges 0 to 255 separated by column
SUBSYSTEM_VENDOR_ID	A 16-bit hex number
SUBSYSTEM_DEVICE_ID	A 16-bit hex number
PXE	Disable or Enable
PXE_SPEED	Auto, 10hd, 10fd, 100hd, 100fd, 1000hd or 1000fd
WOL	Disable or Enable
PRODUCT_NAME	Max. 64-byte string
PART_NUMBER	Max. 11-byte string
ENGINEERING_CHANGE	Max. 10-byte string
MANUFACTURING_ID	A 16-bit hex number
ASSET_TAG	Max. 16-byte string

```
VOLTAGE_SOURCE      1.3 or 1.8
FORCE_PCI           Disable or Enable
```

Example of eeprom.txt

```
MAC_PREFIX          = 00:10:18
POWER DISSIPATED    = 10:0:0:100
POWER_CONSUMED      = 10:0:0:100
SUBSYSTEM_VENDOR_ID = 14e4
SUBSYSTEM_DEVICE_ID = 1644
PXE                 = Disable
PXE_SPEED           = Auto
WOL                 = Disable
PRODUCT_NAME        = Broadcom Gigabit Ethernet Controller
PART_NUMBER         = BCM95700A6
ENGINEERING_CHANGE  = 107070-16
MANUFACTURING_ID    = 14e4
ASSET_TAG           = XYZ01234567
VOLTAGE_SOURCE      = 1.3
FORCE_PCI           = Disabled
```

-bus bb:dd Test UUT location

If this option is specified, the program will only test the UUT at the specified bus number and device number. This option is ignored if `-c` option is entered.

-dpmi Use DPMI memory allocation

Use DPMI memory allocation method to allocate memory instead of `malloc()` or `free()`

-f Program eeprom.bin

This option requires the file `eeprom.bin` in the same working directory. The program programs the content into EEPROM before testing.

-m Program MAC address

If this option is entered, the program will prompt user for a new MAC address before starts testing. Prompt user for the MAC address. With this option, user must enter/scan the MAC address before testing. The program also checks for the file "mac_pref.txt" This is a text file should contain six digits of ASCII MAC three-byte-prefix address. Any of the following formats are supported:

Example:


```
001018  
  
00 10 18  
  
00  
10  
18
```

If this file exists, user has option to enter/scan 7 digit hex number. The first digit will be ignored and the last 6 digits will be used for the lower part of MAC address. Combine with the prefix, it creates 6 byte (12 digit) hex number. If this file does not exist, the whole 12-digit number must be entered for the MAC address. For readability, when entering MAC address, a space character is allowed between each byte. For example, any of the following examples are valid.

```
001018010203  
00 10 18 01 02 03  
1010203      (currently, the scanner uses this format)
```

-n Run program in Manufacturing Loop mode.

With this option, the `-I`, iteration number option, is ignored. The program will run in manufacturing loop mode. Power on/off is supported. After each test, the program will prompt user to exchange the UUT before starts another testing.

-s Skip eeprom programming process.

With this option, the program will skip the eeprom programming process. However, it will check for the eeprom content and print a warning message if the content is not valid.

The -m and -f combination will create the following behavior:

With both -f and -m:

Program will not validate the eeprom content and go ahead to prompt user for the MAC address. It programs MAC address and EEPROM content and then checks the validity of eeprom content at the end of programming.

```
Loading EEPROM content from eeprom.bin: passed  
Programming EEPROM from eeprom.bin....: passed  
Checking EEPROM content.....: passed
```

-f only:

Program will check the validity of eeprom. If it is not valid, it will act as a), -f -m option. If it is good, it saves the MAC address from eeprom, program new eeprom.bin content into EEPROM and then restores the original MAC address. It checks the validity of eeprom content once more at the end of programming.

```
Checking EEPROM content.....: passed
Loading EEPROM content from eeprom.bin: passed
Programming EEPROM from eeprom.bin....: passed
Checking EEPROM content.....: passed
```

or

```
Checking EEPROM content.....: invalid
Loading EEPROM content from eeprom.bin: passed
Programming EEPROM from eeprom.bin....: passed
Checking EEPROM content.....: passed
```

-m only:

Program will check the validity of EEPROM. If it is not valid, it will act as a), -f -m option. If it is good, the program will prompt the user for a new MAC address and program the MAC address only. It checks the validity of EEPROM content once more at the end of programming.

```
Checking EEPROM content.....: passed
Programming MAC address.....: passed
Checking EEPROM content.....: passed
```

or

```
Checking EEPROM content.....: invalid
Loading EEPROM content from eeprom.bin: passed
Programming EEPROM from eeprom.bin....: passed
Checking EEPROM content.....: passed
```

d) no -m and -f options

Program will check the validity of EEPROM. If it is not valid, it will act as a), -f -m option. If it is good, it proceeds to normal diagnostics.

```
Checking EEPROM content.....: passed
```

or

```
Checking EEPROM content.....: invalid
Loading EEPROM content from eeprom.bin: passed
Programming EEPROM from eeprom.bin....: passed
Checking EEPROM content.....: passed
```

5 EEPROM.TXT format

A set of commands is defined to allow user to change EEPROM.BIN content. To update EEPROM.BIN, user must enter `-e <code> -geneep` options at the command prompt. A password must be entered to run this option.

```
MAC_PREFIX = xx:xx:xx
POWER_DISSIPATCHED = d:d:d:d
POWER_CONSUMED = d:d:d:d
SUBSYSTEM_VENDOR_ID = xxxx
SUBSYSTEM_DEVICE_ID = xxxx
PXE = {enable, disable}
PXE_SPEED = {auto, 10hd, 10fd, 100hd, 100fd, 1000hd, 1000fd}
WOL = {enable, disable}
PRODUCT_NAME = string
PART_NUMBER = string
ENGINEERING_CHANGE = string
MANUFACTURING_ID = string
ASSET_TAG = string
```

All reserved words are not case sensitive. A `;`, `//` can be used at the beginning of line as comment.

Example:

```
; This comment line
// This also can be used as comment line

// Blank line is also allowed

// This is Broadcom's MAC prefix
MAC_PREFIX = 00:10:18

POWER_DISSIPATCHED = 10:0:0:100
POWER_CONSUMED = 10:0:0:100
SUBSYSTEM_VENDOR_ID = 14e4
SUBSYSTEM_DEVICE_ID = 1644
pxe = disable
PXE_Speed = 100fd
WOL = enable
Product_name = Broadcom Gigabit Ethernet Controller
PART_NUMBER = BCM95700A6
ENGINEERING_CHANGE = 106679-15
MANUFACTURING_ID = 14e4
Asset_Tag = XYZ1234567
```


6 User Interface Commands

The commands are summarized in twelve groups: vpd, seeprom, cpu, dma, packet, mii, mem, test, power, irq, mac, and misc.

Command Group vpd

vpdwrite	Write VPD Memory
vpdread	Read VPD Memory

Command Group seeprom

semode	Set SEEPROM Mode to Manual or Auto Mode
seread	Read SEEPROM
sewrite	Write SEEPROM
secfg	Configure SEEPROM
seprg	Program SEEPROM
sever	Display Serial EEPROM Version
sechksum	Check/Update Serial EEPROM checksum

Command Group cpu

cpudtt	Dump Debugging Trace of TX CPU
cpudrt	Dump Debuggin Trace of RX CPU

Command Group dma

dmaw	DMA from NIC to Host Memory
dmar	DMA from Host to NIC Memory
dma_h	Dump DMA Entries
dma_d	Dump DMA Entries with Decode

Command Group packet

maclpk	Configure MAC loopback
nicstats	Display NIC statistics
ringindex	Dump Ring Index
blast	Blast Packts in Poll Mode
tprot	Blast with TPROT packets
txpkt	Transmit Packet
phyctrl	Configure Speeds/Duplex Mode
statusblk	Dump Statistics Block
reset	Reset Chip
txcfg	Configure protocol packets for transmission
rxcfg	Configure Rx parameters

Command Group mii

mread	Read PHY register via MII
mwrite	Write PHY registers via MII

mdev	Select current PHY to be accessed
miimode	Select Mode of MII Access
lbertram	Load data to PHY BIST RAM
dbertram	Dump PHY BIST RAM
bertstats	Dump PHY BIST Statistics
rm	Scan/List for phy devices

Command Group mem

memsearch	Search a Data Pattern in Memory
read	Read Memory
write	Write Memory

Command Group test

vpdtest	Run VPD Memory Test
regtest	Run Register Tests
miitest	Run MII Register Test
msi	Run MSI Test
memtest	Run Memory Tests
setest	Run SEEPROM Test
bist	Run BIST
nictest	Run a set of NIC Tests
intrtest	Run Interrupt Test
pktest	Run Packets Tests
cputest	Run CPU Test
dmatest	Run DMA Test
bustest	Run PCI Bus Test

Command Group power

pmdcfg	Dump Power Management Info
pmpcfg	Add/Del Pattern
pmpd	Power Down MAC

Command Group irq

intrctrl	Mask/Unmask Current Interrupt
intr	Dump Interrupt Info

Command Group mac

loadfw	Load Firmware to Tx & Rx CPUs
mbuf	Dump Content of Mbufs
loaddrv	Load Driver
unloaddrv	Unload Driver
machalt	Halt MAC Controller
ftq	Dump FTQ
addmc	Add Multicast MAC
delmc	Delete Multicast MAC

Command Group misc

quit, exit	Exit the System
debug	Debug Functions
gpiowrite	Write a Value into GPIO pin
gpioread	Read GPIO Value
pxecpy	Load PXE code to MBUF Memory
upgfrm	Upgrade PXE or Boot code from a File
device	Show or Switch Device
version	Display Program Version
help, ?	Displays the Commands Available
radix	Change System Radix
nolog	Close the Current Logfile
log	Open Logfile
pciinit	Initialize PCI Configuration Registers
pciscan	Scan for All PCI Devices
dos	Enter DOS Shell
diagcfg	Configure Diagnostics

7 Special Instruction

1. Mac register test:

Unload MAC driver before running test.

2. Memory test:

Unload MAC driver before running test.

3. DMA test:

Unload MAC driver before running test.

4. TX RX packets:

TX sides need to be configured (txcfg).

RX sides need to be configured (rxcfg).

Configure MAC and PHY loop back.

Call txpkt to transmit packets.

5. The following tests need to setup test configuration before running.

To setup test configuration, run "diagcfg". Diag config can be saved in system for future use.

Test:

Memory test

NIC test

6. Unload driver before power down NIC card.

7. Load driver after power up NIC card.

8. Blast Test:

Load MAC driver before running test.

8 Test and Functions Description

8.1 vpdwrite

cmd: vpdwrite

Description: Write data to VPD storage.

Syntax: vpdwrite <start[-end | len] value> | <filename>

File format:

Address range: 0x00 – 0xFF

num_bytes: 256 (max)

If only one argument is entered, filename is assumed. Otherwise, 'start [len] value' format must be used.

Example:

8.2 vpdread

cmd: vpread (Not support in A0)

Description: Read data from VPD storage

Syntax: vpdread <begin_addr> [- end_addr | num_bytes]

Address range : 0x00 – 0xFF

num_byte : 256 (max)

Example:

8.3 vpdtest

cmd: vpdtest (Not support in A0, A1)

Description: Write designed pattern to VPD storage. Then read back and compare with designed pattern.

Syntax: vpdtest [iteration]

Options:

-b<n> : Pattern to test.

- 0 - Increment (def);
- 1 - Decrement ;1 - 0's
- 2 - FF's
- 3 - AA55
- 4 - 55AA
- d : Force Destructive Test

Example:

8.4 semode

cmd: semode

Description: Configure Serial EEPROM to either Auto (I²C) or Manual (Bit-Bang) Mode.

Syntax: semode <auto> | <man> | <> for help

Example:

1. Set Serial EEPROM mode to Auto (I²C).
0:> semode auto
2. Set Serial EEPROM mode to Manual (Bit-Bang).
0:> semode man
3. Display Current mode
0:> semode
Current semode is auto

8.5 seread

cmd: seread

Description: Read content of designated block of Serial EEPROM.

Syntax: seread <begin_addr> [- end_addr | num_bytes]

Address range: 0x00 – 0x1000

Num_bytes: 4097

Example:

1. Set number base to hex, then read and display serial eeprom locations from 0x00 to 0x20

```
0:> radix 16
0:> seread 0-20
*** Dump Serial EEPROM (Auto Mode) ***
000000: 669955aa 08000000 00000069 00000200 d97b07d0 00000000 00000000
00000000

000020: 00000000
```

2. Set number base to hex then read location 0x18 of serial eeprom.

```
0:> radix 16
0:> seread 18 1
*** Dump Serial EEPROM (Auto Mode) ***
000018: 000000ff
```

8.6 sewrite

cmd: sewrite

Description: Write data to designated block of Serial EEPROM.

Syntax: sewrite <begin_addr>[- end_addr | value]

Address range: 0x0000 – 0x1000

Example:

1. Set number base to hex, write 0x55AA to serial eeprom from locations 0x30 to 0x35

```
0:> radix 16
0:> sewrite 30-35 55AA
*** Write Serial EEPROM (Auto Mode) ***
```

2. Set number base to hex, write 0x2 to serial eeprom location 0x25

```
0:> radix 16
0:> sewrite 25 2
*** Write Serial EEPROM (Auto Mode) ***
```

8.7 seprg

cmd: seprg

Description: Program designated Serial EEPROM block via an input file. Input file need to be found in the same location as b57diag.exe.

Syntax: seprg <file_name>

File_name:

Example:

1. Program Serial EEPROM via input file seprg.bin

```
0:> seprg seprg.bin
```

8.8 secfg

cmd: secfg

Description: Configure Serial EEPROM content.

If selected program with defaults (-f=1), eeprom.bin must be found in the same directory of b57diag.exe.

Syntax: secfg

Options:

-a<n> : Verbose Level

-f : Force to program with defaults

Example:

1 Program Serial EEPROM with defaults value and set verbose level to 0.

```
0:> secfg -fv=0
```

```
Reading current serial eeprom ... OK
```

```
1. MAC Address           : 00:00:00:00:00:00
2. Part Number          : BCM5700
3. Revision             : P0
4. Power Dissipated (D0:D1:D2:D3) : 0:0:0:0
5. Power Consumed (D0:D1:D2:D3)   : 0:0:0:0
0. Exit
20. Save and Exit
```

```
Enter your choice (option=paramter) :0
```

2. Set verbose level= 1 to display detail of the Serial EEPROM configuration.

```
0:> secfg -v=1
```

```
Reading current serial eeprom ... OK
```

```
*****
Magic Number   : 0x669955aa
Boot Code Info (start,length,offset): 0x08000000,0x69,0x0200
Code Directory (start,length,offset)
Dir#0 : 0x0,-4,0x000000ff   Dir#1 : 0x1,8,0x00000000
Dir#2 : 0x0,0,0x00000000   Dir#3 : 0x0,0,0x00000000
Dir#4 : 0x0,0,0x00000000   Dir#5 : 0x0,0,0x00000000
Dir#6 : 0x0,0,0x00000000   Dir#7 : 0x0,0,0x00000000
*****
```

```

1. MAC Address           : 00:00:00:00:00:00
2. Part Number          : BCM5700
3. Revision             : P0
4. Power Dissipated (D0:D1:D2:D3) : 0:0:0:0
5. Power Consumed (D0:D1:D2:D3)  : 0:0:0:0
0. Exit
20. Save and Exit

Enter your choice (option=paramter) : 0

```

8.9 setest

cmd: setest

Description: Serial EEPROM read write test

Syntax: setest [iteration]

Example:

1. Run Serial EEPROM read write test.

```
0:seeprom> setest 2
```

```
Iteration 1 of 2
```

```
  C1. EEPROM Test.....: Passed
```

```
Iteration 2 of 2
```

```
  C1. EEPROM Test.....: Passed
```

2. Display Help.

```
0:> setest ?
```

```
Usage : setest [iteration]
```

```
Description:
```

```
The default iteration is 1. 0 means run forever
```

8.10 cpudrt

cmd: cpudrt

Description: Read and display RX CPU trace

Syntax: cpudrt <begin_addr>[- end_addr | num_bytes]

Address range: 0x00 – 0x80

Example:

1. Read and display RX CPU trace from location 0x00 to 0x04.

```
0:> cpudrt 0-5
```

```
000 MainCpuA t00000030 164414e4 e1000004 00000000 164414e4 00000000
001 *BUpCpuA t00000032 00000000 08000034 00440400 00001c40 00000000
002 *BUpCpuA t00000001 00000001 08000034 00440000 00000000 00000000
003 t00000000 00000000 00000000 00000000 00000000 00000000
004 t00000000 00000000 00000000 00000000 00000000 00000000
```

2. Read and display 4 locations of RX CPU trace from start from location 0x00.

```
0:> cpudrt 0 5
000 t00000030 164414e4 e1000004 00000000 164414e4 00000000
001 t00000032 00000000 08000034 00440400 00001c40 00000000
002 t00000001 00000001 08000034 00440000 00000000 00000000
003 t00000000 00000000 00000000 00000000 00000000 00000000
```

8.11 cpudtt

cmd: cpudtt

Description: Read and display TX CPU trace

Syntax: cpudtt <begin_addr>[- end_addr | num_bytes]

Address range: 0x00 – 0x80

Example:

1. Read and display TX CPU trace from location 0x00 to 0x04.

```
0:> cpudtt 0-5
000 t0000002f c0000000 00000000 00000000 00000000 00000000
001 t00000000 00000000 00000000 00000000 00000000 00000000
002 t00000000 00000000 00000000 00000000 00000000 00000000
003 t00000000 00000000 00000000 00000000 00000000 00000000
004 t00000000 00000000 00000000 00000000 00000000 00000000
```

2. Read and display 4 locations of TX CPU trace from start from location 0x00.

```
0:> cpudtt 0 5
000 MainCpuB t0000002f c0000000 00000000 00000000 00000000 00000000
001 t00000000 00000000 00000000 00000000 00000000 00000000
002 t00000000 00000000 00000000 00000000 00000000 00000000
003 t00000000 00000000 00000000 00000000 00000000 00000000
```

8.12 cputest

cmd: cputest

Description: TX / RX CPU Test. This test needs an input CPU file in the same location as b57diag.exe. The default file name is cpu.bin unless specified by -f option.

Syntax: cputest [iteration]

-f : input filename

Example:

1. Running CPU Test two times.

```
0:> cputest 2
Iteration 1 of 2
    C2. CPU Test...:Passed

Iteration 2 of 2
    C2. CPU Test...:Passed
```

8.13 dmar

cmd: dmar

Description: Setup DMA Host Memory to NIC memory

Syntax: dmar

Options:

- a<u32> NIC address to DMA data to
- l<u32> Length of DATA in bytes to DMA
- p<u32> Pattern of Data.
 - 0 - increment
 - 1- decrement
 - 2 - FF's
 - 3 - 00's
 - 4 - AA 55...
 - 5 - 55 AA...
 - 6 - FFFFFFFF 00000000 FFFFFFFF 00000000
 - 7 - FFFFFFFFFFFFFFFFFF 0000000000000000 FFFFFFFFFFFFFFFFFF
 - 8 - FFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFF 000000000000...
- h Use high priority DMA Read.
- b Byte Swap
- w Word Swap
- f Force to 32-bit bus

Example:

1. Get valid NIC address, then set up DMA host memory to NIC memory. Using high priority DMA Read and enable byte swap.


```

0:> dmar

Valid NIC address is 0x00000000-0x0001ffff and exclude 0x00002000-
0x000020c0
0:> dmar -a=0 -l=100 -p=4 -h -b -w -6
Host Address : 0x001422a0
NIC Address  : 0x00000000
Length       : 0x0100
Priority      : High
Byte Swap    : Yes
Word Swap    : Yes
DMAing from Host memory to NIC memory ... OK

```

2. Sup DMA host memory to NIC memory. Using low priority DMA Read and disable byte swap.

```

0:> dmar -a=0 -l=100
Host Address : 0x001422a0
NIC Address  : 0x00000000
Length       : 0x0100
Priority      : Low
Byte Swap    : No
Word Swap    : No

```

3. Display Help.

```

0:> dmaw -?
Usage : dmaw
Options:
  -a<u32> : NIC address to DMA data to.
  -l<u32> : Length of DATA in bytes to DMA.
  -p<u32> : Pattern of Data.
           0 - byte increment ; 1- byte decrement
           2 - FF's ; 3 - 00's ; 4- AA 55 ... ; 5 - 55 AA ...
           6 - FFFFFFFF 00000000 FFFFFFFF 00000000
           7 - FFFFFFFFFFFFFFFFFF 0000000000000000 FFFFFFFFFFFFFFFFFF
           8 - FFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFF 000000000000...
           9 - 00000000000000000000000000000000 FFFFFFFFFFFFF...
           a - Word Increment ; b - Dword Increment
           c - Word Decrement ; d - Dword Decrement
  -h : Use high priority DMA Read.
  -b : Byte Swap
  -w : Word Swap
  -f : Force to use 32-bit

```

8.14 dmaw

cmd: dmaw

Description: Setup DMA NIC Memory to HOST memory

Syntax: dmaw

Options:

-a<u32> NIC address to DMA data from

-l<u32> Length of DATA in bytes to DMA

-h Use high priority DMA Write

-b Byte Swap

-w Word Swap

-d Dump content of Host Memory

-f Force to use 32-bit bus

Example:

1. Setup DMA NIC Memory to HOST memory. Using high priority DMA Read and enable byte swap and disable detail display.

```
0:> dmaw -a=1 -l=10 -hbdf
Host Data :
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

2. Setup DMA NIC Memory to HOST memory. Using low priority DMA Read and disable byte swap and enable detail display.

```
0:> dr maw -a=0 -l=10
Host Address : 0x003421f8
NIC Address : 0x00000000
Length : 0x0010
Priority : Low
Byte Swap : No
Word Swap : No
DMAing from NIC memory to Host memory ... OK
Host Data :
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

3. Display Help.

```
0:> dmaw ?
Usage : dmaw
Options:
-a<u32> NIC address to DMA data from.
-l<u32> Length of DATA in bytes to DMA.
-h Use high priority DMA Write.
-b Byte Swap
-w Word Swap
-d Dump content of Host Memory
-f Force to use 32-bit bus
```

8.15 dma_h

cmd: dma_h

Description: Display DMA entries in HEX

Address range: 0x2000 – 0x3FFFF

Syntax: dma_h <begin_addr>[- end_addr | num_bytes]

Example:

1. Read DMA content start from location 0x2000 to 0x2040.

```
0:> dma_h 2000-2040
002000: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000
002020: 00000000 003421f8 00000000 10070010 00020004 deadbeef deadbeef
deadbeef
002040: 00000000
```

2. Read 40 bytes DMA content start from location 0x2000.

```
0:> dma_h 2000 40
002000: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000
002020: 00000000
```

8.16 dma_d

cmd: dma_d

Description: Display DMA entries with decode

Syntax: dma_d <begin_addr>[- end_addr | num_bytes]

Example:

1. Read and decode DMA content from location 0x2000 to 0x2040.

```
0:> dma_d 2000-2040

**** DMA entry @ 0x2000 ****
Host Address : 00000000:00000000
NIC Address  : 0x00000000
Complete Q   : Unknown
Source Q     : Unknown
Length       : 0
Flags        : 0x00000000
Opaque Data  : 0x00000000 0x00000000 0x00000000

**** DMA entry @ 0x2020 ****
Host Address : 00000000:003421F8
NIC Address  : 0x00000000
Complete Q   : Rx Data Complete Q
Source Q     : DMA High Priority WQ
Length       : 16
Flags        : 0x00020004
Opaque Data  : 0xdeadbeef 0xdeadbeef 0xdeadbeef
```

2. Read and decode 40 bytes DMA content start from location 0x2000.

```
0:> dma_d 2000 40

**** DMA entry @ 0x2000 ****
Host Address : 00000000:00000000
NIC Address  : 0x00000000
Complete Q   : Unknown
Source Q     : Unknown
Length       : 0
Flags        : 0x00000000
Opaque Data  : 0x00000000 0x00000000 0x00000000
```

8.17 dmatest

cmd: dmatest

Description: DMA Test

Syntax: dmatest [iteration]

Options:

```
-l<u32>      Length of data to do DMA
-a<u32>      NIC address
-f           Force to use 32-bit bus
```

Example:

1. Run DMA test.

```
0:> dmatest
**** Testing low priority DMA ***
DMAing HOST (@0x003421f8) to NIC (@0x02100) length = 0x400 ... OK
DMAing NIC (@0x02100) to HOST (@0x003421f8) length = 0x400 ... OK
Checking data contents ... OK
**** Testing High priority DMA ***
DMAing HOST (@0x003421f8) to NIC (@0x00002100) length = 0x400 ... OK
DMAing NIC (@0x02100) to HOST (@0x003421f8) with length = 0x400 ...
OK
Checking data contents ... OK
```

2. Display Help.

```
0:> dmatest -?
Usage : dmatest [iteration]
Options:
  -l<u32>      Length of data to do DMA
  -a<u32>      NIC address
  -f           Force to use 32-bit bus
```

8.18 txcfg

cmd: txcfg

Description: Configure transmits packet protocol

Syntax: txcfg

Example:

```
0:> txcfg
 1. Source MAC                : 00:01:02:03:04:05
 2. Destination MAC          : 10:11:12:13:14:15
 3. Length (14-1518)         : 1514
 4. Packet Type { EthV2(1), 802.3(2), SNAP(3)} : Ethernet II
 5. Protocol Field { IP(800) } : IP
 6. Source IP                 : 10.2.1.1
 7. Destination IP           : 10.2.1.2
 8. IP Protocol Field { UDP(17), TCP(6) }       : UDP
    80. Source Port           : 100
    81. Destination Port     : 200
 9. IP Option Length (32-bit Words)             : 0
10. TCP Option Length (32-bit Words)            : 0
11. Pattern { Inc(1), Random(2), 0s(3), FFs(4) : Increment (00,01,02 ...)
12. IP Checksum Offload{ YES(1), NO(2) }        : NO
13. TCP/UDP Checksum Offload { YES(1), NO(2) }  : NO
14. TCP/UDP Pseudo Checksum Only { YES(1), NO(2) } : NO
15. Insert VLAN Tag { YES(1), NO(2) }          : NO
16. VLAN Tag                                : 1
 0. Exit
Enter your choice (option=paramter) :
```

8.19 txpkt

cmd: txpkt

Description: Transmit Packets. Driver must be loaded.

Syntax: txpkt [num_packet]

Options:

- f<u32> Max Number of Fragments
- j Random number of Fragments
- k Use Random Packet Length
- z<u32> Minimum Fragment Size
- r<u32> Tx Ring Number
- o Insert Fragment Count and Fragment Size into the Packet
- i Incremental Length

- s<u32> Start Packet Length
- d<u32> Interpacket Delay in Microseconds
- g<u32> Tx Flags
- m Use multiple Tx Ring Test
- p<u32> Number of Tx Rings to Use in Multiple Ring Test
- q<u32> Number of Packets per Ring
- b<u32> Burst Length
- l Don't Initialize Packets
- c Clear Statistics
- x Display Real Time NIC Statistics

Example:

1. Transmitting one packet. Driver must be loaded.

```

0:> loaddrv
Reinitializing PCI Configuration Space
Bus Number      : 0
Device/Funtion  : 14/0
Base Address    : 0xf4000004
IRQ             : 10
Bringing up MAC driver ... OK
0:> txpkt 1
***** Blasting Packets *****
Packets to Transmit : 1          Tx Ring Number      : 0
Source MAC          : 00:01:02:03:04:05
Destination MAC    : 10:11:12:13:14:15
Packet Type        : Ethernet II
Pattern            : Increment (00,01,02 ...)
Interpacket Delay  : 0          Fragment Size      : 1514
Incremental Length : Y          Start Length       : 60
IP Checksum Offload : N          TCP/UDP Chksum Offload : N
Pseudo-Chksum Only : N
VLAN Tag Insertion : N          VLAN Tag          : 1

*****
Transmitting packet :              1 ( 60 bytes)
    
```

8.20 rxcfg

cmd: rxcfg

Description Configure RX parameters.

Syntax: rxcfg

Example:

```
0:> rxcfg
 1. Host Loopback { Enable(1), Disable(2) }      : Disable
 2. Modify Rx Packet { Enable(1), Disable(2) }   : Disable
 3. Dump Rx Packet { None(1),Hex(2), Decode(3) } : None
 4. Dump Rx Length                               : 64
 5. Tx Fragment Length                           : 1518
 6. Tx Flags                                     : 0000
 7. Tx VLAN Tag                                 : 0000
 8. Tx Ring Number                              : 0
 9. Tx Generate CRC { Enable(1), Disable(2) }   : Enable
10. Capture Rx Pacpkt { Enable(1), Disable(2) } : Enable
 0. Exit
```

8.21 stsblk

cmd: stsblk

Description: Display Statistics Block.

Syntax: stsblk

Example:

```
0:> stsblk
***** STATISTICS Block @ 0x0027c0c0 *****
ifHCInOctets          :          0   etherStatsFragments      :
0
ifHCInUcastPkts      :          0   ifHCInMulticastPkts   :
0
ifHCInBroadcastPkts :          0   d3StatsFCSErrors     :
0
d3StatsAlignmentErrors :          0   xonPauseFramesReceived :
0
xonPauseFramesReceived :          0   macControlFramesReceived:
0
macControlFramesReceived :          0   dot3StatsFramesTooLong :
0
dot3StatsFramesTooLong :          0
etherStatsJabbers    :          0   etherStatsUndersizePkts :
0
etherStatsUndersizePkts :          0
inRangeLengthError   :          0   outRangeLengthError     :
0
outRangeLengthError  :          0
etherStatsPkts64Octets :          0   etherStatsPkts65-127   :
0
etherStatsPkts65-127 :          0
etherStatsPkts128-255 :          0   etherStatsPkts256-511  :
0
etherStatsPkts256-511 :          0
etherStatsPkts512-1023 :          0   etherStatsPkts1024-1522 :
0
etherStatsPkts1024-1522 :          0
etherStatsPkts1523-2047 :          0   etherStatsPkts2048-4095 :
0
etherStatsPkts2048-4095 :          0
etherStatsPkts4096-8191 :          0   etherStatsPkts8192-9022 :
0
etherStatsPkts8192-9022 :          0
ifHCOutOctets        :          0   etherStatsCollisions   :
0
etherStatsCollisions  :          0
```

```

outXonSent           :           0      outXoffSent           :
0
flowControlDone      :           0      d3StatsInt1MacTxErrors :           0
d3StatsSingleColFrames :           0      d3StatsMultipleColFrames:           0
dt3StatsDeferredTx   :           0      d3StatsExcessiveCol    :           0
d3StatsLateCol       :           0      d3Collided2Times      :
0
d3Collided3Times     :           0      d3Collided4Times      :           0
d3Collided5Times     :           0      d3Collided6Times      :           0
d3Collided7Times     :           0      d3Collided8Times      :
0
d3Collided9Times     :           0      d3Collided10Times     :           0
d3Collided11Times    :           0      d3Collided12Times     :           0
d3Collided13Times    :           0      d3Collided14Times     :           0
d3Collided15Times    :           0      ifHCOutUcastPkts     :           0
d3StatsCarSenseErrors :           0      ifOutDiscards         :
0
COSIfHCInPkts[00]   :           0      COSIfHCInPkts[01]    :           0
COSIfHCInPkts[02]   :           0      COSIfHCInPkts[03]    :           0
COSIfHCInPkts[04]   :           0      COSIfHCInPkts[05]    :           0
COSIfHCInPkts[06]   :           0      COSIfHCInPkts[07]    :           0
COSIfHCInPkts[08]   :           0      COSIfHCInPkts[09]    :           0
COSIfHCInPkts[10]   :           0      COSIfHCInPkts[11]    :           0
COSIfHCInPkts[12]   :           0      COSIfHCInPkts[13]    :           0
COSIfHCInPkts[14]   :           0      COSIfHCInPkts[15]    :           0
COSFrmsDxDueToFilters :           0      nicDmaWriteQueueFull :
0
nicDmaWrHiPQFull    :           0      nicNoMoreRxBds       :           0
ifInDiscards        :           0      ifInErrors            :           0
nicRecvThresholdHit :           0      nicDmaReadQueueFull  :           0
COSIfHCOutPkts[00]  :           0      COSIfHCOutPkts[01]   :           0
COSIfHCOutPkts[02]  :           0      COSIfHCOutPkts[03]   :           0
COSIfHCOutPkts[04]  :           0      COSIfHCOutPkts[05]   :           0
COSIfHCOutPkts[06]  :           0      COSIfHCOutPkts[07]   :           0
COSIfHCOutPkts[08]  :           0      COSIfHCOutPkts[09]   :           0
COSIfHCOutPkts[10]  :           0      COSIfHCOutPkts[11]   :           0
COSIfHCOutPkts[12]  :           0      COSIfHCOutPkts[13]   :
0
COSIfHCOutPkts[14]  :           0      COSIfHCOutPkts[15]   :           0
nicDmaRdHPQueueFull :           0      nicSendDataCompQueueFull:
0
nicRingSetSdPIDx    :           0      nicRingStatusUpdate   :           0
nicInterrupts       :           0      nicAvoidedInterrupts  :           0
nicSendThresholdHit :           0

```

8.22 statusblk

cmd: statusblk

Description: Display Status Block

Syntax: statusblk

Example:

```
0:> statusblk
```

```
***** STATUS Block @ 0x0027c040 *****
Status : 0x0000
```



```

Rx Standard CIdx : 0      Rx Jumbo CIdx : 0      Rx Mini CIdx : 0
Rx PIdx[00]   : 0          Send CIdx[00]  : 0
Rx PIdx[01]   : 0          Send CIdx[01]  : 0
Rx PIdx[02]   : 0          Send CIdx[02]  : 0
Rx PIdx[03]   : 0          Send CIdx[03]  : 0
Rx PIdx[04]   : 0          Send CIdx[04]  : 0
Rx PIdx[05]   : 0          Send CIdx[05]  : 0
Rx PIdx[06]   : 0          Send CIdx[06]  : 0
Rx PIdx[07]   : 0          Send CIdx[07]  : 0
Rx PIdx[08]   : 0          Send CIdx[08]  : 0
Rx PIdx[09]   : 0          Send CIdx[09]  : 0
Rx PIdx[10]   : 0          Send CIdx[10]  : 0
Rx PIdx[11]   : 0          Send CIdx[11]  : 0
Rx PIdx[12]   : 0          Send CIdx[12]  : 0
Rx PIdx[13]   : 0          Send CIdx[13]  : 0
Rx PIdx[14]   : 0          Send CIdx[14]  : 0
Rx PIdx[15]   : 0          Send CIdx[15]  : 0

```

8.23 reset

cmd: reset

Description: Reset Chip

Syntax: reset

Example:

```

0:> reset
Global Resetting CHIP ... OK

```

8.24 phyctrl

cmd: phyctrl

Description: Configure Phy Speed

Syntax: phyctrl

Options:

- s<u32> 0:10 Mbps, 1:100 Mbps, 2:1000 Mbps, 3:Auto Negotiation.
- h Force Half Duplex
- r Reset PHYs
- f<string> File contains initialization scripts

Example:

1. Configure Phy into Auto Negotiation, full duplex mode.

```
0:> physctrl -s=3 -i=1
Resetting PHY ... OK
PHY ID      : 0x0020 - 0x6051
PHY Description : BCM5401 Rev#1
Initializing registers (work-around for BCM5401) ... Done
Configure MAC and PHY to ... MII/Full/Auto Negotiation mode
```

8.25 maclpb

cmd: maclpb (B0 only)

Description: Enable or Disable MAC loop back

Syntax: maclpb <0|1>

0 to disable. Otherwise enable

Example:

1. Driver must be loaded before configure.

```
0:> loaddrv
Reinitializing PCI Configuration Space
Bus Number      : 0
Device/Funtion  : 14/0
Base Address    : 0xf4000004
IRQ             : 10
Bringing up MAC driver ... OK
```

2. Enable MAC loop back.

```
0:> mcaclpb 1
Enabling MAC loopback ... OK
```

2. Disable MAC loop back.

```
0:> maclpb 0
Disabling MAC loopback ... OK
```

8.26 mread

cmd: mread

Description: Read PHY registers via MII

Syntax: mread <begin_addr>[-<end_addr> | <len>]

Address range: 0x00 – 0x1F

Example:

1. Read MII register 0

```
0:> mread 0
00: 1100
```

2 Read MII registers 0 to 10

```
0:> mread 0-10
00: 1100 7949 0020 6051 01e1 0000 0004 2001
08: 0000 0300 0000 0000 0000 0000 0000 3000
10: 0002
```

3. Read 5 MII registers start from register 0

```
0:> mread 0 5
00: 1100 7949 0020 6051 01e1
```

8.27 mwrite

cmd: mwrite

Description: Write PHY registers via MII

Syntax: mwrite <addr > <value>

Address range: 0x00 – 0x1F

Example:

1. Write 0x15 to MII register 2

```
0:> mwrite 2 15
```

8.28 mdev

cmd: mdev

Description: Current Phy Selection. The default device ID is 0x01. If no parameter is entered, it displays current phy address setting.

Syntax: mdev [<phy_id>]

Example:

```
0:> mdev 1
Phy Address = 1
```

8.29 miimode

cmd: miimode

Description: MII auto or manual mode select

Syntax: miimode <1|0>

Example:

```
0:> miimode 0
Setting MII auto mode to OFF
0:> miimode 1
Setting MII auto mode to ON
0:> miimode
```

8.30 miitest

cmd: miitest [iteration]

Description: PHY registers read write test

Syntax: miitest

Example:

1. Running MII Test.

```
0:> miitest

C4. MII Test.....:Passed
```

8.31 read

cmd: read

Description: Generic Memory Read

Syntax: read [@|#|^|s|x|m]<begin_addr> [- end_addr | num_bytes]

@ = Configuration space (address range: 0x00 – 0xFF)

= Registers (default) (address range: 0x00 – 0x1FFFF)

* = SRAM (address range: 0x00 –0x1FFFF)

\$ = Serial EEPROM

% = Parallel EEPROM

m = MII Registers

^ = internal scratchpad (address range: 0x3000 – 0x37FFFF)

l = direct access (dword)

s = direct access (word)

x = direct access (byte)

Example:

1. Read from Configuration space

```
0:> read @10
000010: f4000004
```

2. Read from Register

```
0:> read #10
000010: f4000004
```

3. Read from SRAM

```
0:> read *10
000010: 00010001
```

4. Read from internal scratchpad

```
0:> read ^00
000000: 000312ae
```

8.32 write

cmd: write

Description: Generic Memory Write

Syntax: write [**@**|**#**|*****|**\$**|**%**|**m**|**^**|**l**|**s**|**x**]**<begin_addr>** [- **end_addr**] **<value>**

@ = Configuration space (address range: 0x00 – 0xFF)

= Registers (default) (address range: 0x00 – 0x1FFFF)

***** = SRAM (address range: 0x00 – 0x1FFFF)

\$ = Serial EEPROM

% = Parallel EEPROM

m = MII Registers

^ = internal scratchpad (address range: 0x3000 – 0x37FFFF)

l = direct access (dword)

s = direct access (word)

x = direct access (byte)

Example:

1. Write to configuration space.

```
0:> write @10 f4000004
```

2. Write to register.

```
0:> write #10 f4000004
```

3. Write to SRAM

```
0:> write *10 10001
```

4. Write to internal scratchpad

```
0:> write ^10 f4000004
```

8.33 memtest

cmd: memtest

Description: Test memory blocks such as scratch pad, BD sram, DMA sram, Mbuf, external SRAM. Running “diagcfg” can configure memory block ranges. See “diagcfg” for detail. Driver must be unloaded.

Syntax: memtest [iteration]

Options:

-s : Test Scratch Pad (0x30000-0x37fff)

-b : Test BD SRAM (0x0000-0x0fff and 0x4000-0x7fff)

-d : Test DMA SRAM (0x2000-0x3fff)

-m : Test MBUF SRAM (0x8000-0x1ffff)

-e : Test External Memory (0x20000-0XXXXXXXX)

-x: Test MBUF SRAM via DMA

-c: Test MBUF special test

Example :

1. Unload driver if driver loaded, run memory test 1 time. Scratch Pad, DMA blocks had been selected in this example.

```
0:> unloaddrv
Unloading MAC driver ... OK
0:> memtest
```

B1. Scratch Pad Test

Data Read/Write Test:

Data Pattern 0x00000000.....: Passed

Data Pattern 0xFFFFFFFF.....: Passed

Data Pattern 0xAA55AA55.....: Passed

```

    Data Pattern 0x55AA55AA.....: Passed
Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Address Test.....: Passed
Walking bit.....: Passed
Pseudo Random Data.....: Passed
B2. BD SRAM Test
Data Read/Write Test:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Address Test.....: Passed
Walking bit.....: Passed
Pseudo Random Data.....: Passed
Data Read/Write Test:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Address Test.....: Passed
Walking bit.....: Passed
Pseudo Random Data.....: Passed
B4. MBUF SRAM Test
Data Read/Write Test:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
Address Test.....: Passed
Walking bit.....: Passed
Pseudo Random Data.....: Passed

```

8.34 pmdcfg

cmd: pmdcfg

Description: Display Power Management Info

Syntax: pmdcfg

Example:

```
0:> pmdcfg
PMCSR          : 0x2100
PM Capability   : 0xc002
Data Scale     : 1
D0 Power Consumed (00) : 0x00
D1 Power Consumed (01) : 0x00
D2 Power Consumed (02) : 0x00
D3 Power Consumed (03) : 0x00
D0 Power Dissipated (04) : 0x00
D1 Power Dissipated (05) : 0x00
D2 Power Dissipated (06) : 0x00
D3 Power Dissipated (07) : 0x00
Common Power Cons. (08) : 0x00
Reserved       (09) : 0x00
Reserved       (10) : 0x00
Reserved       (11) : 0x00
Reserved       (12) : 0x00
Reserved       (13) : 0x00
Reserved       (14) : 0x00
Reserved       (15) : 0x00
```

8.35 pmpd

cmd: pmpd

Description: Power Down MAC. Input file wol.txt should be found in the same location of b57diag.exe. The input file contains patterns. If the file name is not specified, data zero will be used.

Syntax: pmpd [filename]

Options:

- a<u32> 0 to add a pattern; 1 means otherwise
- o<u32> Offset
- m<1:0> 1 enables Magic MAC detection
- i<1:0> 1 enables ACPI Packet Match
- v<1:0> Verbose level

Example:

1. Power down MAC

```
0:> pmpd -a=0 -o=0 -m=1 -i=1 -v=1
No input file sepcified... Data Pattern contains zeros

Wake-On-Lan Patterns :

Halting MAC ... OK
Programming patterns to H/W ... OK
Programming ACPI Registers Buf @ 0x00000800 offset = 0 length = 128.
Enable ACPI Pattern Match
Enable Magic MAC detection
```

8.36 intr**cmd:** intr**Description:** Display Interrupt Info**Syntax:** intr**Example:**

```
0:> intr
Interrupt Count : 3
IPC MASK          : 0xb8 0x0b
IPC IS1 IS2       : 0x00 0x00
IPC IRR1 IRR2     : 0x00 0x00
IPC ILCR1 ILCR2   : 0x00 0x0e
```

8.37 intrtest**cmd:** intrtest**Description:** Interrupt Test**Syntax:** intrtest [iteration]**Example:**

1. Running interrupt test 2 times.

```
0:> intrtest 2
Iteration 1 of 2:
  A4. Interrupt Test...:Passed

Iteration 2 of 2:
  A4. Interrupt Test...:Passed
```

8.38 machalt

cmd: machalt

Description: Halt MAC controller

Syntax: machalt

Example:

```
0:> machalt
Halting MAC ... OK
```

8.39 addmc

cmd: addmc

Description: Add Multicast MAC

Syntax: addmc address0 [address1...]

Example:

```
0:> addmc FF:FF:00:0A:00:00
```

8.40 delmc

cmd: delmc

Description: Delete Multicast MAC

Syntax: delmc address0 [address1...]

Example:

```
0:> delmc FF:FF:00:0A:00:00
```

8.41 ftq

cmd: ftq

Description: Display FTQ info

Syntax: ftq

Example:

```
0:> ftq

***** Dump FTQ Peak/Write (Control,Full Counter, Write/Peak) *****
DMA Read FTQ (1)      : 00000000 00000000 20000000
DMA High Read FTQ (2) : 00000000 00000000 60002160
DMA Write FTQ (6)     : 00000000 00000000 20000000
DMA High Write FTQ (7) : 00000000 00000000 20000000
```

```

DMA Complete Dx FTQ (3)      : 00000000 00000000 20000000
Send BD Comp. FTQ (4)      : 00000000 00000000 20000000
Send Data Init FTQ (5)     : 00000000 00000000 20000000
Send Data Comp. FTQ (9)    : 00000000 00000000 20000000
Rx BD Complete FTQ (13)    : 00000000 00000000 60002160
Rx Data Complete FTQ (16)  : 00000000 00000000 20000000
S/W Type 1 FTQ (8)        : 00000000 00000000 20000000
Host Coalescing FTQ (10)   : 00000000 00000000 2000:00000000
MAC TX FTQ (11)           : 00000000 00000000 2000:00000000
Mbuf Cluster Free FTQ (12) : 00000000 00000000 2000:00000000
RX List Placement FTQ (14) : 00000000 00000000 2000:00000000
RX Data Initiator FTQ (15) : 00000000 00000000 2000:00000000

S/W Type 2 FTQ (17)       : 00000000 00000000 2000:00000000
    
```

8.42 mbuf

cmd: mbuf

Description: Display Content of Mbufs

Syntax: mbuf <chain|info|cluster|hdr|ckhdr|dump [<mbuf number>]|workaround>

Options:

-m : Mbuf number to display/decode. 0:decode, 1: in hex

Example:

1. Display Mbuf.

```

0:> mbuf dump -m1
**** Mbufs 0x100 @ 0x00008000 ****
00008080 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
Chain : 0      Frame : 0      Next Mbuf : 0x0101      Len : 0
Next Frame Pointer : 0x00000000
Data:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000
    
```

2. Display Mbuf chain.

```

0:> mbuf chain
->143->144->145->146->147->148->149->14a->14b->14c->14d->14e->14f->150
->151->152->153->154->155->156->157->158->159->15a->15b->15c->15d->15e
->15f->160->161->162->163->164->165->166->167->168->169->16a->16b->16c
->16d->16e->16f->170->171->172->173->174->175->176->177->178->179->17a
    
```

8.43 loadrv

cmd: loadrv

Description: Load NIC driver

Syntax: loaddrv

Example:

```
0:> loaddrv
Reinitializing PCI Configuration Space
Bus Number      : 0
Device/Funtion  : 14/0
Base Address    : 0xf4000004
IRQ             : 10
Bringing up MAC driver ... OK
```

8.44 unloaddrv

cmd: unloaddrv

Description: Unload NIC driver

Syntax: unloaddrv

Example:

```
0:> unloaddrv
Unloading MAC driver ... OK
```

8.45 loadfw

cmd: loadfw

Description: Load Firmware to TX & RX CPUs

Syntax: loadfw <-f=filename | -r | -t> | <-?> for help

-f : firmware file

-r : Load firmware to RX-CPU

-t : Load firmware to TX-CPU

Example:

```
0:> loadfw -?
Usage : loadfw <t|r> <filename>
```

Use 't' option to load firmware to TX CPU and 'r' to RX CPU. File name also must be specified.

8.46 nictest

cmd: nictest

Description: NIC test includes memory test, serial eeprom test, interrupt test, packet exchange, MAC registers test, Mii registers test, cpu test, dma test. This test need to be configured by running “diagcfg”. See “diagcfg” for detail.

Syntax: nictest [iteration]

Example:

```
0:> nictest
```

Testing Device:

```
0:PCI BUS#3:BCM5700(B0),0xfda00004,IRQ 11,Conventional PCI/66MHz/64-bit
```

```
Manufacturing revision      : C
Boot Code revision         : 0.8
```

Group A. Register Tests

```
A1. Indirect Register Test.....: Passed
A2. Control Register Test.....: Passed
A4. Interrupt Test.....: Passed
```

Group B. Memory Tests

B1. Scratch Pad Test

Data Read/Write Test:

```
Data Pattern 0x00000000.....: Passed
Data Pattern 0xFFFFFFFF.....: Passed
Data Pattern 0xAA55AA55.....: Passed
Data Pattern 0x55AA55AA.....: Passed
```

Alternate Pattern:

```
Data Pattern 0x00000000.....: Passed
Data Pattern 0xFFFFFFFF.....: Passed
Data Pattern 0xAA55AA55.....: Passed
Data Pattern 0x55AA55AA.....: Passed
```

```
Address Test.....: Passed
Walking bit.....: Passed
Pseudo Random Data.....: Passed
```

B2. BD SRAM Test

Data Read/Write Test:

```
Data Pattern 0x00000000.....: Passed
Data Pattern 0xFFFFFFFF.....: Passed
Data Pattern 0xAA55AA55.....: Passed
Data Pattern 0x55AA55AA.....: Passed
```

Alternate Pattern:

```
Data Pattern 0x00000000.....: Passed
Data Pattern 0xFFFFFFFF.....: Passed
Data Pattern 0xAA55AA55.....: Passed
Data Pattern 0x55AA55AA.....: Passed
```

```
Address Test.....: Passed
Walking bit.....: Passed
Pseudo Random Data.....: Passed
```

Data Read/Write Test:

```
Data Pattern 0x00000000.....: Passed
Data Pattern 0xFFFFFFFF.....: Passed
Data Pattern 0xAA55AA55.....: Passed
```

```

    Data Pattern 0x55AA55AA.....: Passed
    Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
    Address Test.....: Passed
    Walking bit.....: Passed
    Pseudo Random Data.....: Passed
B3. DMA SRAM Test
    Data Read/Write Test:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
    Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
    Address Test.....: Passed
    Walking bit.....: Passed
    Pseudo Random Data.....: Passed
B4. MBUF SRAM Test
    Data Read/Write Test:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
    Alternate Pattern:
    Data Pattern 0x00000000.....: Passed
    Data Pattern 0xFFFFFFFF.....: Passed
    Data Pattern 0xAA55AA55.....: Passed
    Data Pattern 0x55AA55AA.....: Passed
    Address Test.....: Passed
    Walking bit.....: Passed
    Pseudo Random Data.....: Passed
B6. MBUF SRAM via DMA Test
    Testing pattern 16 00's 16 FF's ..: Passed
    Testing pattern 16 FF's 16 00's ..: Passed
    Testing pattern 32 00's 32 FF's ..: Passed
    Testing pattern 32 FF's 32 00's ..: Passed
    Testing pattern 00000000's .....: Passed
    Testing pattern FFFFFFFF's .....: Passed
    Testing pattern AA55AA55's .....: Passed
    Testing pattern 55AA55AA's .....: Passed
Group C. Miscellaneous Tests
C1. EEPROM Test.....: Passed
C2. CPU Test.....: Passed
C3. DMA Test
    Low Priority.....: Passed
    High Priority.....: Passed
C4. MII Test.....: Passed
C5. VPD Test.....: Passed

```

Group D. Data Tests

D1. Internal Loopback Test.....: Passed
D2. Phy Loopback Test.....: Passed

8.47 nicstats

cmd: nicstats

Description: Display NIC test statistics

Syntax: nicstats <-c>

-c : Clear Statistics

Example: Load driver if driver is not loaded.

```
0:> loaddrv
Reinitializing PCI Configuration Space
Bus Number      : 0
Device/Funtion  : 14/0
Base Address     : 0xf4000004
IRQ             : 10
Bringing up MAC driver ... OK
0:> nicstats
```

	Total	Rate
	=====	=====
Txed Packets (Ring#0) :	0	0
Txed Packets (Ring#1) :	0	0
Txed Packets (Ring#2) :	0	0
Txed Packets (Ring#3) :	0	0
Tx Packets Enqed (Ring#0) :	0	0
Tx Packets Enqed (Ring#1) :	0	0
Tx Packets Enqed (Ring#2) :	0	0
Tx Packets Enqed (Ring#3) :	0	0
Rxed Packets (Ring00) :	0	0
Rxed Packets (Ring01) :	0	0
Rxed Packets (Ring02) :	0	0
Rxed Packets (Ring03) :	0	0
Rxed Packets (Ring04) :	0	0
Rxed Packets (Ring05) :	0	0
Rxed Packets (Ring06) :	0	0
Rxed Packets (Ring07) :	0	0
Rxed Packets (Ring08) :	0	0
Rxed Packets (Ring09) :	0	0

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

Total Rate

```

=====
Rxed Packets (Ring#10)      :          0          0
Rxed Packets (Ring#11)      :          0          0
Rxed Packets (Ring#12)      :          0          0
Rxed Packets (Ring#13)      :          0          0
Rxed Packets (Ring#14)      :          0          0
Rxed Packets (Ring#15)      :          0          0
Rxed CRC-32 Errors          :          0          0
Out of Memory                :          0          0
Too Many Frag Pkt          :          0          0
=====

```

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

CHIP Statistics

```

=====
ifHCInOctets                :          0  etherStatsFragments      :          0
ifHCInUcastPkts             :          0  ifHCInMulticastPkts      :          0
ifHCInBroadcastPkts         :          0  d3StatsFCSErrors         :          0
d3StatsAlignmentErrors      :          0  xonPauseFramesReceived   :          0
xonPauseFramesReceived      :          0  macControlFramesReceived :          0
xonStateEntered             :          0  dot3StatsFramesTooLong   :          0
etherStatsJabbers           :          0  etherStatsUndersizePkts  :          0
inRangeLengthError          :          0  outRangeLengthError      :          0
etherStatsPkts64Octets      :          0  etherStatsPkts65-127     :          0
etherStatsPkts128-255       :          0  etherStatsPkts256-511    :          0
etherStatsPkts512-1023      :          0  etherStatsPkts1024-1522  :          0
etherStatsPkts1523-2047     :          0  etherStatsPkts2048-4095  :          0
etherStatsPkts4096-8191     :          0  etherStatsPkts8192-9022  :          0
ifHCOutOctets               :          0  etherStatsCollisions     :          0
outXonSent                  :          0  outXoffSent               :          0
flowControlDone             :          0  d3StatsInt1MacTxErrors   :          0
d3StatsSingleColFrames      :          0  d3StatsMultipleColFrames :          0
dt3StatsDeferredTx          :          0  d3StatsExcessiveCol      :          0
=====

```

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

CHIP Statistics

```

=====
d3StatsLateCol              :          0  d3Collided2Times        :          0
d3Collided3Times           :          0  d3Collided4Times        :          0
d3Collided5Times           :          0  d3Collided6Times        :          0
d3Collided7Times           :          0  d3Collided8Times        :          0
d3Collided9Times           :          0  d3Collided10Times       :          0
d3Collided11Times          :          0  d3Collided12Times       :          0
d3Collided13Times          :          0  d3Collided14Times       :          0
d3Collided15Times          :          0  ifHCOutUcastPkts        :          0
d3StatsCarSenseErrors       :          0  ifOutDiscards            :          0
COSIfHCInPkts[00]          :          0  COSIfHCInPkts[01]       :          0
COSIfHCInPkts[02]          :          0  COSIfHCInPkts[03]       :          0
COSIfHCInPkts[04]          :          0  COSIfHCInPkts[05]       :          0
COSIfHCInPkts[06]          :          0  COSIfHCInPkts[07]       :          0
COSIfHCInPkts[08]          :          0  COSIfHCInPkts[09]       :          0
=====

```



```

COSIfHCInPkts[10]      :          0  COSIfHCInPkts[11]      :          0
COSIfHCInPkts[12]      :          0  COSIfHCInPkts[13]      :          0
COSIfHCInPkts[14]      :          0  COSIfHCInPkts[15]      :          0
COSFrmsDxDueToFilters  :          0  nicDmaWriteQueueFull   :          0
nicDmaWrHiPQFull       :          0  nicNoMoreRxBDs         :          0
    
```

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

CHIP Statistics

```

=====
ifInDiscards           :          0  ifInErrors             :          0
nicRecvThresholdHit    :          0  nicDmaReadQueueFull    :          0
COSIfHCOutPkts[00]    :          0  COSIfHCOutPkts[01]    :          0
COSIfHCOutPkts[02]    :          0  COSIfHCOutPkts[03]    :          0
COSIfHCOutPkts[04]    :          0  COSIfHCOutPkts[04]    :          0
Rxed Packets (Ring#05) :          0  Rxed Packets (Ring#05) :          0
Rxed Packets (Ring#06) :          0  Rxed Packets (Ring#06) :          0
Rxed Packets (Ring#07) :          0  Rxed Packets (Ring#07) :          0
Rxed Packets (Ring#08) :          0  Rxed Packets (Ring#08) :          0
Rxed Packets (Ring#09) :          0  Rxed Packets (Ring#09) :          0
    
```

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

8.48 regtest

cmd: regtest

Description: MAC registers read write test. Driver must be unloaded.

Syntax: regtest [<iteration>]

Example:

1. Running register test.

```

0:> unloaddrv
Unloading MAC driver ... OK
0:> regtest
    
```

A2. Control Register Test ...:Passed

8.49 debug

cmd: debug

Description: Display debugs information

Syntax: debug <n>

- 1: Dump TX / RX Stats
- 2: Dump Clock Scale info
- 3: Clear worst interrupt latency
- 4: Toggle indirect access flag

Example:

1. Display debug information.

```
0:> debug 1
Tx Packets Enqueued      :          0
Tx Packet Complete      :          0
Tx Packet Complete Error :          0
Rx Packets               :          0
Rx Unknown Packets      :          0
Rx Bad Packets          :          0
Rx Good Packets         :          0
```

8.50 pciscan

cmd: pciscan

Description: Scan for all PCI Devices

Syntax: pciscan

Example:

```
0:> pciscan
Scanning PCI devices ...
Bus Dev Func  Vendor ID Device ID      Class      Base/IO Address      IRQ
====  =====  =====  =====  =====  =====  =====
      0  0  0      8086      7190      06:00:00      00000000:F8000008  0
      0  1  0      8086      7191      06:04:00      00000000:00000000  0
      0  7  0      8086      7110      06:01:00      00000000:00000000  0
      0  7  1      8086      7111      01:01:80      00000000:00000000  0
      0  7  2      8086      7112      0C:03:00      00000000:00000000  9
      0  7  3      8086      7113      06:80:00      00000000:00000000  0
10    0 14  0          12AE          0003          02:00:00          00000000:F4000004
11    1  0  0          1002          4742          03:00:00          00009001:F5000000
```

8.51 diagcfg

cmd: diagcfg

Description: Configure diagnostics parameter for Memory tests and Manufacturing test (NIC test).

Syntax: diagcfg

Example:

0:misc> diagcfg

Diagnostics Configuration Menu

1. Memory Test Configuration Menu
2. Manufacturing Test Configuration Menu
3. Driver Configuration Menu
4. Abort On Failure {Yes(1), No(0)} : Yes
5. Verbose Level (0..6) : 2

Enter your choice (option=paramter/save/cancel) :

Memory Test Configuration Menu

1. SRAM BD1 Start (0x00000000-0x00000fff) : 00000000
2. SRAM BD1 End (0x00000000-0x00000fff) : 00000fff
3. SRAM BD2 Start (0x00004000-0x00007fff) : 00004000
4. SRAM BD2 End (0x00004000-0x00007fff) : 00007fff
5. SRAM DMA Start (0x00002000-0x00003fff) : 00002000
6. SRAM DMA End (0x00002000-0x00003fff) : 00003fff
7. SRAM MBUF Start (0x00008000-0x0001ffff) : 00008000
8. SRAM MBUF End (0x00008000-0x0001ffff) : 0001ffff
9. SRAM SPAD Start (0x00030000-0x00037fff) : 00030000
10. SRAM SPAD End (0x00030000-0x00037fff) : 00037fff
11. Ext. SRAM Start (0x00020000-0x00ffffff) : 00020000
12. Ext. SRAM End (0x00020000-0x00ffffff) : 00ffffff
0. Exit to previous menu

Enter your choice (option=paramter) :

Diagnostics Configuration Menu

1. Memory Test Configuration Menu
2. Manufacturing Test Configuration Menu
3. Driver Configuration Menu
4. Abort On Failure {Yes(1), No(0)} : Yes
5. Verbose Level (0..6) : 2

Enter your choice (option=paramter/save/cancel) :

Manufacturing Test Configuration Menu

1. Register Test { Enable(1), Disable(2) } : Enable
2. Memory Test { Enable(1), Disable(2) } : Enable
3. Serial EEPROM Test { Enable(1), Disable(2) } : Enable
4. Interrupt Test { Enable(1), Disable(2) } : Enable
5. CPU Test { Enable(1), Disable(2) } : Enable

- 6. DMA Test { Enable(1), Disable(2) } : Enable
- 7. VPD Test { Enable(1), Disable(2) } : Disable
- 8. MII Test { Enable(1), Disable(2) } : Enable
- 9. Packet Tx/Rx { Enable(1), Disable(2) } : Disable
- 0. Exit to previous menu

Enter your choice (option=paramter) :

Diagnostics Configuration Menu

- 1. Memory Test Configuration Menu
- 2. Manufacturing Test Configuration Menu
- 3. Driver Configuration Menu
- 4. Abort On Failure {Yes(1), No(0)} : Yes
- 5. Verbose Level (0..6) : 2

Enter your choice (option=paramter/save/cancel) :

Driver Configuration Menu

- 1. Rx Coalescing Ticks : 1000
- 2. Rx Coalescing Ticks During Intr : 0
- 3. Rx Coalescing Frames : 1
- 4. Rx Coalescing Frames During Intr : 0
- 5. Tx Coalescing Ticks : 1000
- 6. Tx Coalescing Ticks During Intr : 0
- 7. Tx Coalescing Frames : 1
- 8. Tx Coalescing Frames During Intr : 0
- 9. Statistics Coalescing Ticks : 1000000
- 10. Tx Packet Descriptor Count : 50
- 11. Rx Standard Packet Count : 100
- 12. Rx Jumbo Packet Count : 50
- 13. Queue Rx Packets : 1
- 14. Tx Copy Buffer Size : 64
- 15. External Memory Exists : 0
- 16. MBUF Base : 0x008000
- 17. MBUF Length : 0x018000
- 18. MBUF WorkAround : 0
- 0. Exit to previous menu

8.52 log

cmd: log

Description: Save all output to log file (tp.log)

Syntax: log

Example:

```
0:> log
started logfile 'b57diag.log'
```

8.53 nolog

cmd: nolog

Description: Closes Log file

Syntax: nolog

Example:

```
0:> nolog
logfile closed
```

8.54 radix

cmd: radix

Description: Set the base of input number.

Syntax: radix <2 | 8 | 10 | 16>

Example: Set base of input number to hex

```
0:> radix 16
0:> radix
current input radix = 16
```

8.55 exit, quit

cmd: exit, quit

Description: Exit System

Syntax: exit

Example:

```
0:> exit
```

8.56 blast

cmd: blast

Description: Blast Packets in Poll Mode and display statistics. Load MAC driver before running the test.

Syntax: blast -l=<length> -t -r -h -c=<num_buf> -k -s

-l : Length of Tx packet (Default = 64)

-t : Enable Tx

- r : Enable Rx
- h : Enable Host Loopback *
- c : Number of Tx buffer (Default = 100)
- k : Applies CRC-32 check on Rx path
- s : Stop on Failure

*Workaround A1 memory Problem: In order to use -h option do the followings.

1. loaddrv
2. mbuf -w=1
3. blast -h
4. Start SmartBit to inject traffic. Watch for CRC Error indication.

Example:

1. Load MAC driver and enable transmission .

```
0:packet> loaddrv
PHY ID      : 0x0020 - 0x6051
PHY Description : BCM5401 Rev#1
Configuring BCM5401 ... Done
Reinitializing PCI Configuration Space
Bus Number   : 0
Device/Funtion : 14/0
Base Address  : 0xf400004
IRQ          : 10
Bringing up MAC driver ... OK
0:packet> blast -t
PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit
```

	Total	Rate
	=====	=====
Txed Packets (Ring#0) :	1007609	507523
Txed Packets (Ring#1) :	0	0
Txed Packets (Ring#2) :	0	0
Txed Packets (Ring#3) :	0	0
Tx Packets Enqed (Ring#0) :	0	0
Tx Packets Enqed (Ring#1) :	0	0
Tx Packets Enqed (Ring#2) :	0	0
Tx Packets Enqed (Ring#3) :	0	0
Rxed Packets (Ring#00) :	0	0
Rxed Packets (Ring#01) :	0	0
Rxed Packets (Ring#02) :	0	0
Rxed Packets (Ring#03) :	0	0
Rxed Packets (Ring#04) :	0	0
Rxed Packets (Ring#05) :	0	0
Rxed Packets (Ring#06) :	0	0
Rxed Packets (Ring#07) :	0	0
Rxed Packets (Ring#08) :	0	0
Rxed Packets (Ring#09) :	0	0

PageUP/PageDN to scroll. Ins/Del toggles refresh. ESC to exit

2. Display Help.

```
0:packet> blast -?
Usage : blast -l=<length> -t -r -h -c=<num_buf> -k -s
  -l : Length of Tx packet (Default = 64)
  -t : Enable Tx
  -r : Enable Rx
  -h : Enable Host Loopback
  -c : Number of Tx buffer (Default = 100)
  -k : Applies CRC-32 check on Rx path
  -s : Stop on Failure
```

8.57 gpiowrite

cmd: gpiowrite

Description: Control Output of GPIO Pin

Syntax: gpiowrite <GPIO_num> <1 | 0>

Valid value for <GPIO_num> is 0-2, <value> is 0 or 1.

Example:

1. Write 1 to GPIO#1 Pin

```
0:> gpiowrite 1 1

Writing 1 to GPIO#1
```

8.58 gpioread

cmd: gpioread

Description: Get Input of GPIO Pin

Syntax: gpioread

Example:

1. Read GPIO Pins

```
0:> gpioread
GPIO#0 : 1
GPIO#1 : 1
GPIO#2 : 0
```

8.59 version

cmd: version

Description: Display Diagnostics Version

Syntax: version

8.60 ringIndex

cmd:

Description: Dump Ring Index. Load Mac driver before running.

Syntax: ringindex t | r | rt

Example:

1 Load MAC driver and display TX and RX Ring Index.

```
0:> loaddrv
PHY ID          : 0x0020 - 0x6051
PHY Description : BCM5401 Rev#1
Configuring BCM5401 ... Done
Bringing up MAC driver ... OK
0:> ringindex rt
```

	Mailbox	RBDI	RBDC	HC	StsBlk	Driver
	=====	=====	=====	=====	=====	=====
RxStdPidx	100	100	100	---	---	100
RxStdCidx	---	---	---	000	000	000
RetRPidx#00	---	---	---	000	---	---
RetRCidx#00	000	---	---	---	---	000

	Mailbox	SBDI	SBDSEL	HC	StsBlk	Driver
	=====	=====	=====	=====	=====	=====
SendHostPidx#00	000	000	---	---	---	000
SendHostCidx#00	---	---	000	000	000	000
SendHostPidx#01	000	000	---	---	---	000
SendHostCidx#01	---	---	000	000	000	000
SendHostPidx#02	000	000	---	---	---	000
SendHostCidx#02	---	---	000	000	000	000
SendHostPidx#03	000	000	---	---	---	000
SendHostCidx#03	---	---	000	000	000	000

8.61 dos

cmd: dos

Description: Enter to Dos shell

Syntax: dos

Example:

```
0:> dos
```

8.62 pxecpy

cmd: pxecpy

Description: Copy PXE code to MBUF memory

Syntax: pxecpy <file>

Example:

```
0:> pxecpy -?
```

```
Usage : pxecpy <filename>
```

Description:

The file name must be specified in the parameter.

8.63 pciinit

cmd: pciinit

Description: Initialize PCI configuration registers

Syntax: pciinit

Example:

```
0:misc> pciinit
Initializing PCI Configuration Space
Bus Number      : 0
Device/Funtion  : 14/0
Base Address     : 0xf4000004
IRQ              : 10
Broadcom 5700 NIC is detected
```

8.64 intrctrl

cmd: intrctrl

Description: Control Interrupt Controller

Syntax: intrctrl u|m

u : unmask current interrupt

m : mask current interrupt

Example:

1. Mask current interrupt

```
0:irq> intrctrl m
Masking Interrupt 10
```
2. Unmask current interrupt

```
0:irq> intrctrl u
Unmasking Interrupt 10
```

8.65 upgfrm

cmd: upgfrm

Description: Upgrade boot code firmware or PXE.

Syntax : upgfrm <pxe | boot> <filename>

Examples:

1. Upgrade boot code firmware from eeprom.bin
Upgfrm boot eeprom.bin
2. Upgrade PXE code from b57pxe.bin
Upgfrm pxe b57pxe.bin

8.66 pkttest

Command: pkttest

Description: Perform MAC and/or PHY loopback test. This test will send 100 packets in incremental length and check for contents of loopbacked packets.

Syntax: pkttest [<iteration>]

Options: -m : Perform MAC loopback test.
 -p : Perform PHY loopback test.
 -e : External Loopback Test

Examples:

pkttest -pm 3 -- Perform MAC and PHY loopback in 3 iterations.

8.67 teste

Command: teste

Description: The command enables tests. It effects nictest, regtest, pkttest, and memtest commands. The test must starts with test group alpha (a-d). If no number is entered, all tests in that group are enabled.

Syntax: teste [<tests> [<tests>...]]

Example: teste a12bc -- Enable test a1, a2, all tests in group b and c
 teste ab cd -- Enables all tests
 teste -- Display enabled tests

8.68 testd

Command: testd

Description: The command disables tests. It effects nictest, regtest, pkttest, and memtest commands. The test must starts with test group alpha (a-d). If no number is entered, all tests in that group are disabled.

Syntax: testd [<tests> [<tests>...]]

Example: testd a12bc -- Disable test a1, a2, and all tests in group b and c.
 testd ab cd -- Disables all tests.
 testd -- Display disabled tests.

8.69 lbertram

Command: lbertram

Description: Load data to PHY BIST RAM

Syntax: lbertram [filename]

Options: -f<string> Filename containing BIST data
 -c<u32> Channel Number
 -e Enable BIST

8.70 dbertram

Command: dbertram

Description: Dump PHY BIST RAM

Syntax: dbertram

Options: -c<u32> Channel Number
 -t Dump Tx BIST RAM
 -r Dump Rx BIST RAM
 -b<u32> Begin of BIST RAM
 -e<u32> End of BIST RAM

8.71 bertstats

Command: bertstats

Description: Dump PHY BIST statistics

Syntax: bertstats

8.72 bustest

Command: bustest

Description: Dump PHY BIST statistics

Syntax: bustest

Options: -a NIC address to DMA data to.
 -l Minimum length (default = 256).
 -h Maximum length (default = 1024).
 -n iteration (default = 1)
 -I Number of transactions per pattern (default = 10)
 -s Start of test case (default = 0)
 -e End of test case (default = 259)

There are total 260 test cases (258 unique tests cases) which are described as follows:

Test case#	Pattern
=====	=====
0	ffffffff ffffffff 00000000 00000000
1	ffffffff ffffffff0e 00000000 00000000
2	ffffffff ffffffff0d 00000000 00000000
.	.
.	.
.	.
64	7ffffffff ffffffff 00000000 00000000
65	00000000 00000000 ffffffff ffffffff

```
66          00000000 00000000 ffffffff ffffffffe
67          00000000 00000000 ffffffff ffffffffd
.
.
.
129         00000000 00000000 7fffffff ffffffff
130         00000000 00000000 ffffffff ffffffff (repeat)
131         00000000 00000001 ffffffff ffffffff
132         00000000 00000002 ffffffff ffffffff
.
.
.
194         80000000 00000000 ffffffff ffffffff
195         ffffffff ffffffff 00000000 00000000 (repeat)
196         ffffffff ffffffff 00000000 00000001
197         ffffffff ffffffff 00000000 00000002
.
.
.
259        ffffffff ffffffff 80000000 00000000
```

If you run `bustest` command without any parameters, it will perform DMA testing on all 260 patterns with 10 iterations per pattern and different data length in each iteration. First eight bytes of data are used to store the following info for debug:

```
byte 0-4 : length
byte 5-6 : iteration#
byte 6-7 : test case#
```

9 ERROR MESSAGES

```
/* 0 */ "PASS",
/* 1 */ "Got 0x%08X @ 0x%08X. Expected 0x%08X",
/* 2 */ "Cannot perform task while chip is running",
/* 3 */ "Invalid NIC device",
/* 4 */ "Read only bit %s got changed after writing zero at offset 0x%X",
/* 5 */ "Read only bit %s got changed after writing one at offset 0x%X",
/* 6 */ "Read/Write bit %s did not get cleared after writing zero at offset 0x%X",
/* 7 */ "Read/Write bit %s did not get set after writing one at offset 0x%X",
/* 8 */ "BIST failed",
/* 9 */ "Could not generate interrupt",
/* 10 */ "Aborted by user",
/* 11 */ "Tx DMA:Got 0x%08X @ 0x%08X. Expected 0x%08X",
/* 12 */ "Rx DMA:Got 0x%08X @ 0x%08X. Expected 0x%08X",
/* 13 */ "Tx DMA failed",
/* 14 */ "Rx DMA failed",
/* 15 */ "Data error, got 0x%08X at 0x%08X, expected 0x%08X",
/* 16 */ "Second read error, got 0x%08X at 0x%08X, expected 0x%08X",
/* 17 */ "Failed writing EEPROM at 0x%04X",
/* 18 */ "Failed reading EEPROM at 0x%04X",
/* 19 */ "EEPROM data error, got 0x08X at 0x04X, expected 0x%08X",
/* 20 */ "Cannot open file %s",
/* 21 */ "Invalid CPU image file %s",
/* 22 */ "Invalid CPU image size %d",
/* 23 */ "Cannot allocate memory",
/* 24 */ "Cannot reset CPU",
/* 25 */ "Cannot release CPU",
/* 26 */ "CPU test failed",
/* 27 */ "Invalid Test Address Range\nValid NIC address is 0x%08X-0x%08X and exclude 0x%08X-0x%08X",
/* 28 */ "DMA:Got 0x%08X @ 0x%08X. Expected 0x%08X",
/* 29 */ "Unsupported PhyId %04X:%04X",
/* 30 */ "Too many registers specified in the file, max is %d",
/* 31 */ "Cannot write to VPD memory",
/* 32 */ "VPD data error, got %08X @ 0x04X, expected %08X",
/* 33 */ "No good link! Check Loopback plug",
/* 34 */ "Cannot TX Packet!",
/* 35 */ "Requested to Tx %d. Only %d is transmitted",
/*36*/ "Expected %d packets. Only %d good packet(s) have been received\n%d unknown packets have been
received.\n%d bad packets have been received.",
/* 37 */ "%c%d is an invalid Test",
/* 38 */ "EEPROM checksum error",
/* 39 */ "Error in reading WOL/PXE",
/* 40 */ "Error in writing WOL/PXE",
/* 41 */ "No external memory detected",
/* 42 */ "DMA buffer %04X is large, size must be less than %04X",
/* 43 */ "File size %d is too big, max is %d",
/* 44 */ "Invalid %s",
/* 45 */ "Failed writing 0x%x to 0x%x",
/* 46 */ "",
/* 47 */ "Ambiguous command",
/* 48 */ "Unknown command",
/* 49 */ "Invalid option",
```

```

/* 50 */ "Cannot perform task while chip is not running. (need driver)",
/* 51 */ "Cannot open register define file or content is bad",
/* 52 */ "ASF Reset bit did not self-cleared",
/* 53 */ "ATTN_LOC %d cannot be mapped to %cX CPU event bit %d",
/* 54 */ "%s Register is not cleared to zero after reset",
/* 55 */ "Cannot start poll_ASF Timer",
/* 56 */ "poll_ASF bit did not get reset after acknowledged",
/* 57 */ "Timestamp Counter is not counting",
/* 58 */ "%s Timer is not working",
/* 59 */ "Cannot clear bit %s in %cX CPU event register",
/* 60 */ "Invalid "EEPROM_FILENAME" file size, expected %d but only can read %d bytes",
/* 61 */ "Invalid magic value in %s, expected %08x but found %08x",
/* 62 */ "Invalid manufacture revision, expected %c but found %c",
/* 63 */ "Invalid Boot Code revision, expected %d.%d but found %d.%d",
/* 64 */ "Cannot write to EEPROM",
/* 65 */ "Cannot read from EEPROM",
/* 66 */ "Invalid Checksum",
/* 67 */ "Invalid Magic Value",
/* 68 */ "Invalid MAC address, expected %02X-%02X-%02X-%02X-%02X-%02X",
/* 69 */ "Slot error, expected an UUT to be found at location %02X:%02X:00",
/* 70 */ "Adjacent memory has been corrupted while testing block 0x%08x-0x%08x\nGot 0x%08x @ address 0x%08x.
Expected 0x%08x",
/* 71 */ "The function is not Supported in this chip",
/* 72 */ "Packets received with CRC error",
/* 73 */ "MII error bits set: %04x",
/* 74 */ "CPU does not initialize MAC address register correctly",
/* 75 */ "Invalid firmware file format",
/* 76 */ "Resetting TX CPU Failed",
/* 77 */ "Resetting RX CPU Failed",
/* 78 */ "Invalid MAC address",
/* 79 */ "Mac address registers are not initialized correctly",
/* 80 */ "EEPROM Bootstrap checksum error",

```