AMD Graphics Tool (AGT) User's Guide

Publication:	**56003**	Revision:	**0.73**
Issue Date:	**July 2017**		

Contents

List of Figures

List of Tables

Revision History

Date	Revision	Changes
July 2017	0.73	• Added new command –xgmiphysetup in Section 2.4 AGT Command Line Options for PHY Eye Diagram Functions (step needs to be run before capturing PHY eye for XGMI lanes). • Removed Section 3.4.1 Using HDT for Link Width Change (no need to use HDT to change linkwidth). • Removed the option –xgmisetuptxeqtune in Section 3.5 Using AGT for XGMI Tuning (step no longer needed for XGMI tuning).
May 2017	0.72	• Added –xgmisetuptxeqtune option (updated Table 2. Command Line Options for xGMI Tuning; Added Section 3.5.2 Selecting the XGMI Lanes).
April 2017	0.71	• Added Chapter 3.
March 2017	0.70	• Initial release.

Chapter 1 Overview

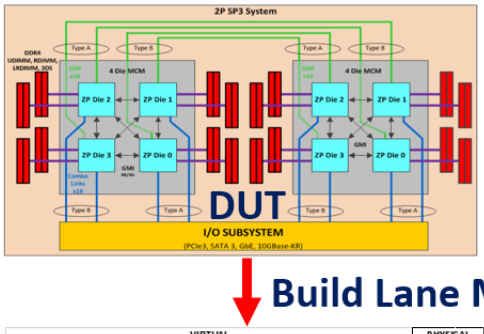
AMD Graphics Tool (AGT) is a console mode utility for AMD GPU/APU/CPU device in a diagnostics environment, providing control of clocks, voltage, Powerplay, PCIe®, and memory setting. It covers margining on xGMI with target-based data capture and analysis.

This document covers only the margining feature set. The validation step described in this document is recommended to provide broad system level coverage and ensure sufficient margin for the variables encountered when a system design is in production. It can also help identify specific configurations, boards, and CPUs.

1.1 Margining Tool Process

The margining process consists of the following basic steps:

- Build Lane Map (see Figure 1)
- Run AGT (see Figure 2 on page 9)
- Report output (see Figure 3 on page 9)



Build Lane Map

VIRTUAL										PHYSICAL	
xGMI		PCIE		SATA		XGBE		WAFL		CHIPLET	LANE
LN	ADDR	LN	ADDR	LN	ADDR	LN	ADDR	LN	ADDR		
0	1200_0000h	32	1220_0000h	64	1240_0000h					A0	0
1		33		65							1
2	1202_0000h	34	1222_0000h	66	1242_0000h					A1	0
3		35		67							1
4	1204_0000h	36	1224_0000h	68	1244_0000h	72	1254_0000h			A2	0
5		37		69		73	1264_0000h				1
6		38		70		74	1274_0000h				2
7		39		71		75	1284_0000h				3
8	1206_0000h	40	1226_0000h							A3	0
9		41									1
10		42									2
11		43									3
12	1208_0000h	44	1228_0000h							A4	0
13		45									1
14		46									2
15		47									3
16	1210_0000h	48	1230_0000h							B0	0
17		49									1
18	1212_0000h	50	1232_0000h							B1	0
19		51									1
20	1214_0000h	52	1234_0000h							B2	0
21		53									1
22		54									2
23		55									3
24	1216_0000h	56	1236_0000h							B3	0
25		57									1
26		58									2
27		59									3
28	1218_0000h	60	1238_0000h							B4	0
29		61									1
30		62									2
31		63									3
								76	11E0_0000h	W0	0
								77	11F0_0000h		1

Figure 1. Build Lane Map

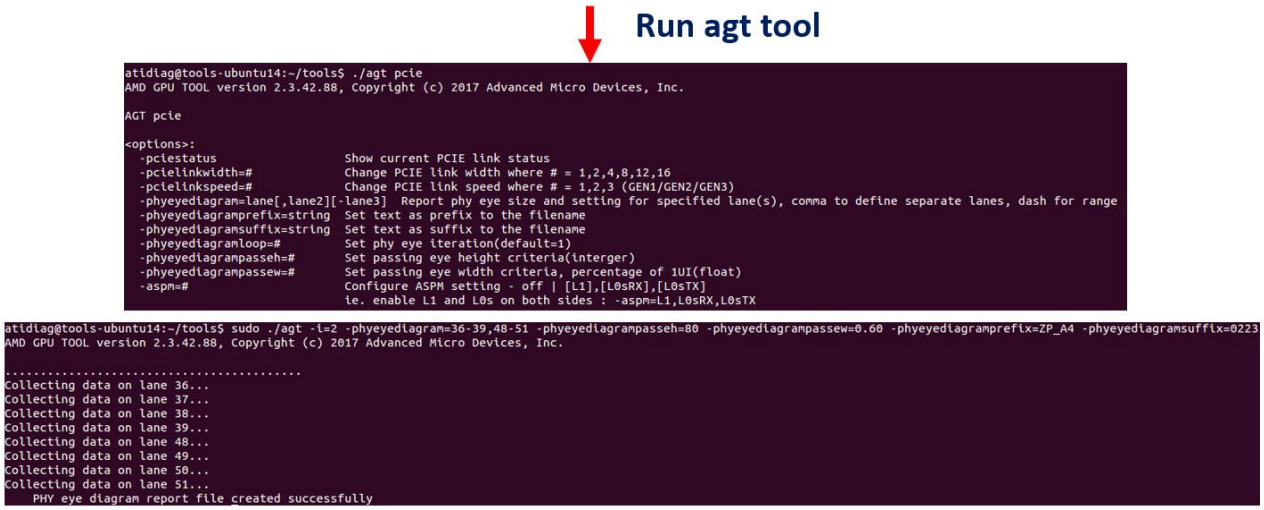


Figure 2. Run AGT Tool

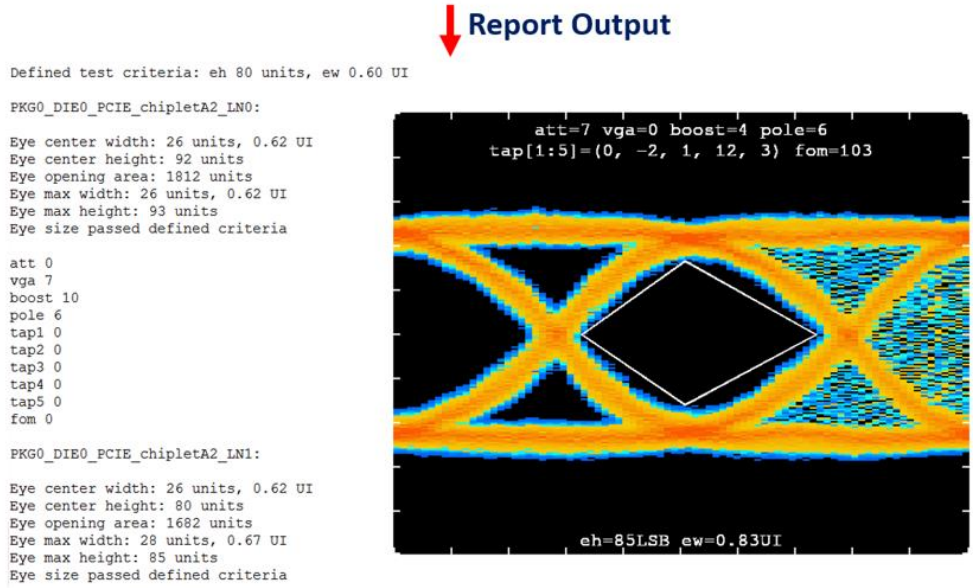


Figure 3. Report Output

Margin is used in normal operation at the receiver side:

- When a margin test is performed on an xGMI link, the link is used in normal operation.
- The Receiving Device margins voltage and timing parameters while receiving the incoming data.

Chapter 2 Setup and Execution

2.1 System Requirements

- Linux® -- Preferably Ubuntu 16.04
- HDT Wombat Hardware

2.2 Release Area

- Tool released via NDA Site
  - /Software/Family 17h (AM4, SP3, SP4)/BIOS-Software-Tools/Tools

2.3 Setup

2.3.1 Tool Installation Setup

To install the tool:

- Unzip the tar.gz file to any folder.

2.3.1.1 About TrueType Font Files in the OS

AGT "-phyeyediagram" command draws the text on image with TrueType font in the OS.

For example, it uses "/usr/share/fonts/truetype/freefont/FreeMono.ttf" in Ubuntu Linux®.

If the OS you installed does not have the TrueType font file in system folder, please copy a TrueType font file into the folder where AGT is located.

2.4 AGT Command Line Options for PHY Eye Diagram Functions

2.4.1 AGT Command Help Option

Type "./agt –h" and "./agt pcie" for tool argument. The help options display (see Figure 4 on page 8.

<options>:	
-i	Displays a list of all present AMD devices
-i=<#>	Select a target device to apply commands.
	Supports both simple index (reported by -i) and PCI location (-i=PCI:<bn>.<dn>.<fn>)
-phyeyediagram=lane[,lane2][-lane3]	Report phy eye size and setting for specified lane(s),
	Comma to define separate lanes, dash for range
-phyeyediagramprefix=string	Set text as prefix to the filename
-phyeyediagramsuffix=string	Set text as suffix to the filename
-phyeyediagramloop=#	Set phy eye iteration
-phyeyediagrampasseh=#	Set passing eye height criteria(integer)
-phyeyediagrampassew=#	Set passing eye width criteria, percentage of 1UI(float)
-xgmiphysetup	Set up XGMI lanes for capturing PHY eye diagram

```
Lane Assignment
0 - XGMI_chipletA0_LN0, 1 - XGMI_chipletA0_LN1, 2 - XGMI_chipletA1_LN0, 3 - XGMI_chipletA1_LN1, 4 - XGMI_chipletA2_LN0, 5 - XGMI_chipletA2_LN1
6 - XGMI_chipletA2_LN2, 7 - XGMI_chipletA2_LN3, 8 - XGMI_chipletA3_LN0, 9 - XGMI_chipletA3_LN1, 10 - XGMI_chipletA3_LN2, 11 - XGMI_chipletA3_LN3
12 - XGMI_chipletA4_LN0, 13 - XGMI_chipletA4_LN1, 14 - XGMI_chipletA4_LN2, 15 - XGMI_chipletA4_LN3, 16 - XGMI_chipletB0_LN0, 17 - XGMI_chipletB0_LN1
18 - XGMI_chipletB1_LN0, 19 - XGMI_chipletB1_LN1, 20 - XGMI_chipletB2_LN0, 21 - XGMI_chipletB2_LN1, 22 - XGMI_chipletB2_LN2, 23 - XGMI_chipletB2_LN3
24 - XGMI_chipletB3_LN0, 25 - XGMI_chipletB3_LN1, 26 - XGMI_chipletB3_LN2, 27 - XGMI_chipletB3_LN3, 28 - XGMI_chipletB4_LN0, 29 - XGMI_chipletB4_LN1
30 - XGMI_chipletB4_LN2, 31 - XGMI_chipletB4_LN3, 32 - PCIE_chipletA0_LN0, 33 - PCIE_chipletA0_LN1, 34 - PCIE_chipletA1_LN0, 35 - PCIE_chipletA1_LN1
36 - PCIE_chipletA2_LN0, 37 - PCIE_chipletA2_LN1, 38 - PCIE_chipletA2_LN2, 39 - PCIE_chipletA2_LN3, 40 - PCIE_chipletA3_LN0, 41 - PCIE_chipletA3_LN1
42 - PCIE_chipletA3_LN2, 43 - PCIE_chipletA3_LN3, 44 - PCIE_chipletA4_LN0, 45 - PCIE_chipletA4_LN1, 46 - PCIE_chipletA4_LN2, 47 - PCIE_chipletA4_LN3
48 - PCIE_chipletB0_LN0, 49 - PCIE_chipletB0_LN1, 50 - PCIE_chipletB1_LN0, 51 - PCIE_chipletB1_LN1, 52 - PCIE_chipletB2_LN0, 53 - PCIE_chipletB2_LN1
54 - PCIE_chipletB2_LN2, 55 - PCIE_chipletB2_LN3, 56 - PCIE_chipletB3_LN0, 57 - PCIE_chipletB3_LN1, 58 - PCIE_chipletB3_LN2, 59 - PCIE_chipletB3_LN3
60 - PCIE_chipletB4_LN0, 61 - PCIE_chipletB4_LN1, 62 - PCIE_chipletB4_LN2, 63 - PCIE_chipletB4_LN3, 64 - SATA_chipletA0_LN0, 65 - SATA_chipletA0_LN1
66 - SATA_chipletA1_LN0, 67 - SATA_chipletA1_LN1, 68 - SATA_chipletA2_LN0, 69 - SATA_chipletA2_LN1, 70 - SATA_chipletA2_LN2, 71 - SATA_chipletA2_LN3
72 - XGBE_chipletA2_LN0, 73 - XGBE_chipletA2_LN1, 74 - XGBE_chipletA2_LN2, 75 - XGBE_chipletA2_LN3, 76 - WAFL_chipletW0_LN0, 77 - WAFL_chipletW0_LN1
```

Figure 4. Command Help Options

"./agt -i"	to list available devices
"./agt –i -xgmiphysetup"	set up XGMI lanes for capturing PHY eye diagram
"./agt -i -phyeyediagram=#,#,#-#"	to create PHY eye diagrams for the selected lanes
"./agt -i -phyeyediagram=#,#,#-# -phyeyediagramloop=x"	to create PHY eye diagrams for the selected lanes with each point read x times
"./agt -i -phyeyediagram=#,#,#-# -phyeyediagrampasseh=x –phyeyediagrampassew=y"	to create PHY eye diagrams for the selected lanes and check if eye size passing criteria or not

Figure 5. Sample Commands

2.4.2 Running Different Lanes Simultaneously

PHY eye diagram data on multiple lanes can be sampled simultaneously by defining lane numbers through "-phyeyediagram=" argument (see Figure 6). User can use "-" to define range of lanes, and use "," to separate single or range of lanes.

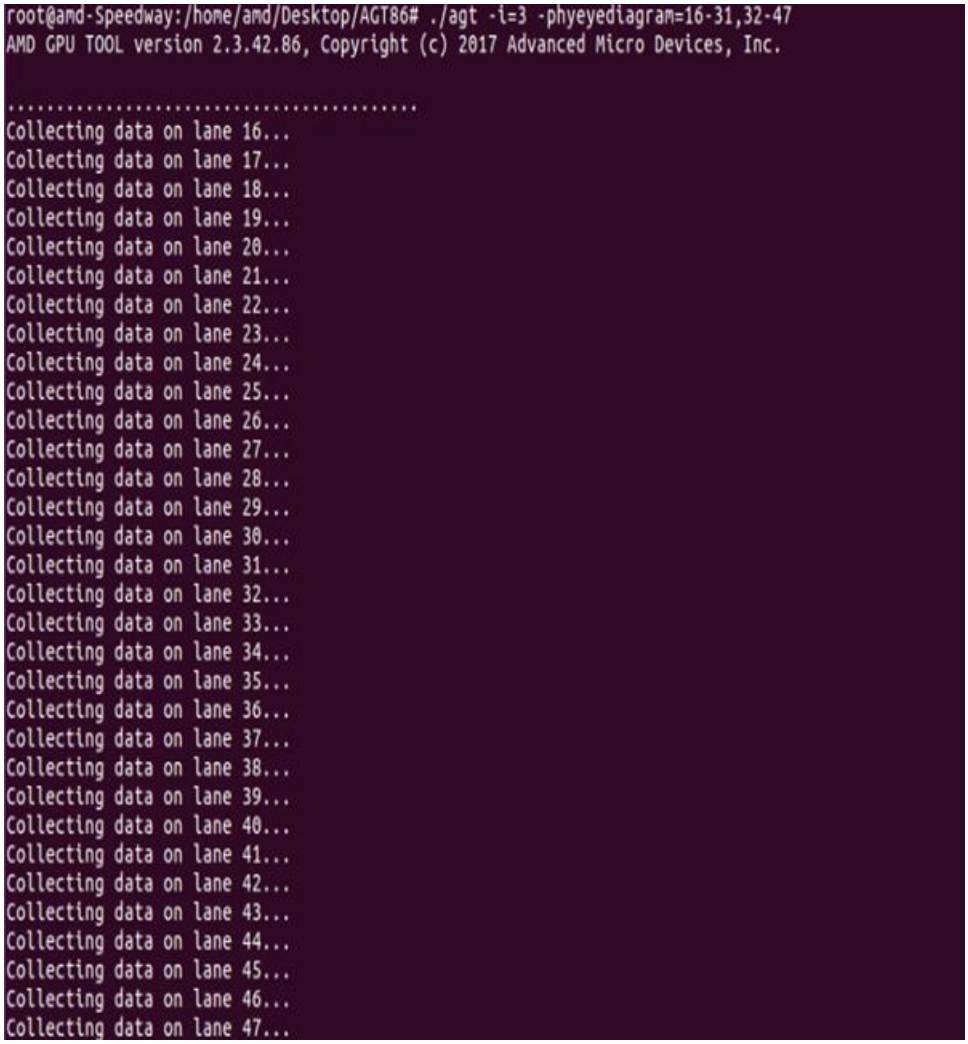


Figure 6. Running Multiple Lanes

2.4.3 Running Different Lanes and Socket/Die Simultaneously

Besides command "-phyeyediagram=" to define lanes, user can use "-i=" argument to define multiple dies so that PHY eye diagram on multiple dies and lanes can be generated simultaneously (see Figure 7 on page 8).

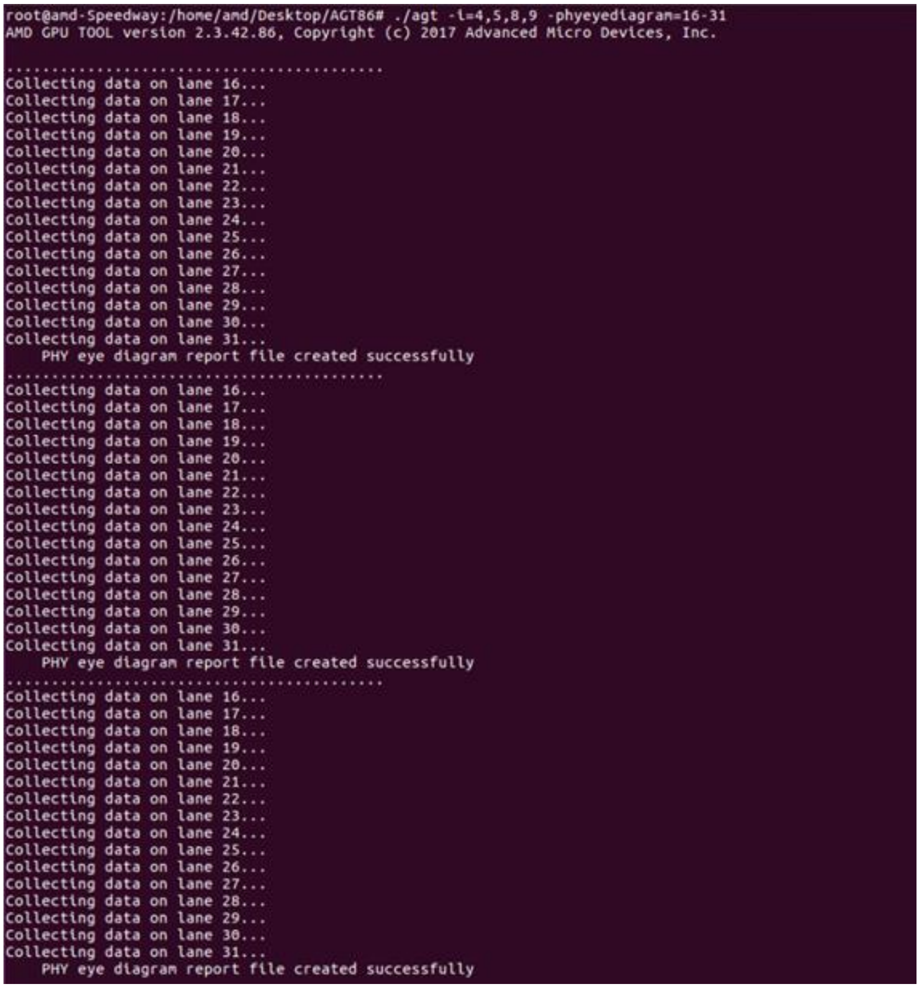


Figure 7. Running Multiple Lanes and Socket/Die Simultaneously

2.5 Results

Results are captured in a picture(.png) and text file for each lane. A summary report file is also generated for each run which has the information of eye size and phy configuration in each lane.

Optional argument "-phyeyediagrampasseh=" and "-phyeyediagrampassew=" can be used to define eye size passing criteria.

Figure 8 on page 8 shows the sample output in summary report file when "-phyeyediagrampasseh=80" and "-phyeyediagrampassew=0.60" are added.

```
Defined test criteria: eh 80 units, ew 0.60 UI

PKG0_DIE0_PCIE_chipletA2_LN0:

Eye center width: 26 units, 0.62 UI
Eye center height: 92 units
Eye opening area: 1812 units
Eye max width: 26 units, 0.62 UI
Eye max height: 93 units
Eye size passed defined criteria

att 0
vga 7
boost 10
pole 6
tap1 0
tap2 0
tap3 0
tap4 0
tap5 0
fom 0

PKG0_DIE0_PCIE_chipletA2_LN1:

Eye center width: 26 units, 0.62 UI
Eye center height: 80 units
Eye opening area: 1682 units
Eye max width: 28 units, 0.67 UI
Eye max height: 85 units
Eye size passed defined criteria

att 0
vga 7
boost 10
pole 6
tap1 0
tap2 0
tap3 0
tap4 0
tap5 0
fom 0
```

Figure 8. Sample Output

Chapter 3 XGMI

3.1 Zeppelin System Topology

In Zeppelin 2P system, XGMI connections between the sockets use x16 links. The lower XGMI lanes for Socket 0 are always connected to the higher lanes of Socket 1 and vice versa. As shown in Figure 10 on page 16, the lower Type A lanes (0-7) of Socket 0 Die 0 are connected to higher Type B lanes (15-8) of Socket 1 Die 2.

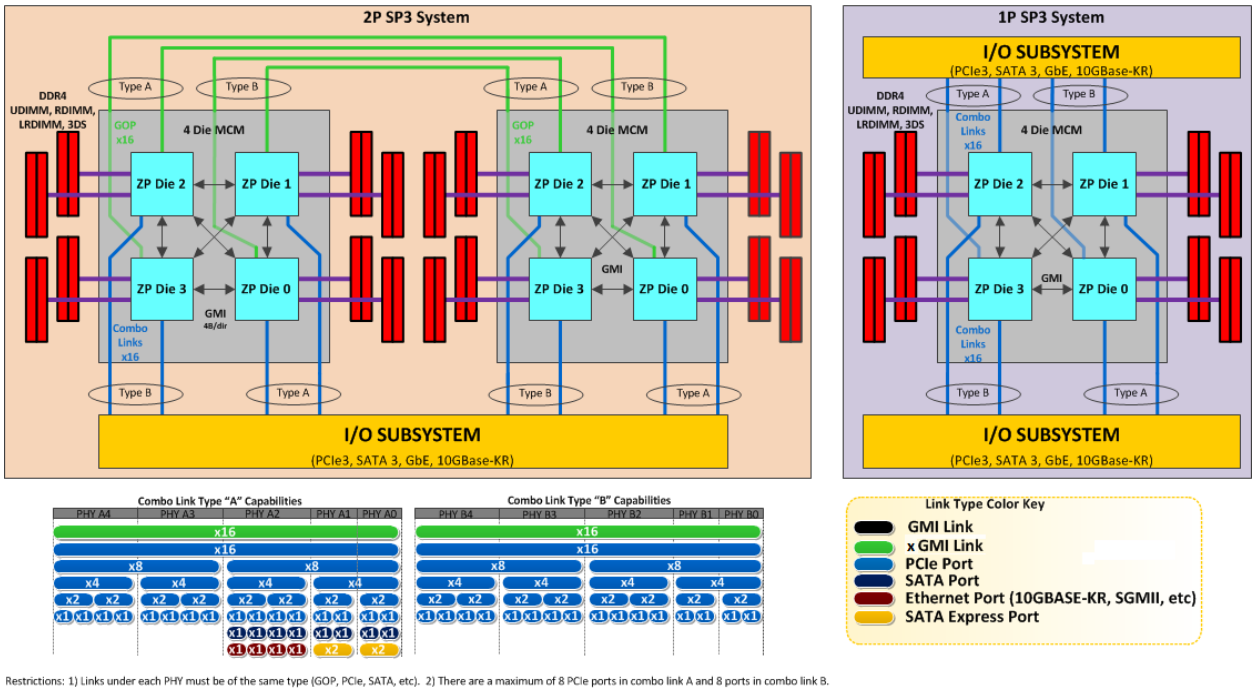


Figure 9. Zeppelin XGMI Topology

The XGMI tuning process is carried out on a live system booted to Linux. Tuning cannot be carried out on lanes which are actively used by the system. So optimal settings for only part of the lanes can be found be after a single tuning step. The step has to be repeated for the rest of the lanes for deriving the full and final optimal settings of a die at a particular speed.

3.2 xGMI Pass/Fail Criteria

The xGMI Pass/Fail criteria are shown in Table 1.

Note: *The tests can only be performed after the system is loaded with the recommended TXEQ Coefficient.*

xGMI margin should be tested with the following:

- phyeyediagramloop =10 (changed from the default of 1) through command line
- Multiple platforms (to check margin across board manufacturing)

Table 1. xGMI Pass/Fail Criteria

Rx	Eye height or Mask at Sampling Point after CTLE	25mVpp
Rx	Eye width or Mask at Sampling Point after CTLE	0.3UI@BER=1e-12

3.3 Link Tuning

XGMI TXEQ tuning is recommended to provide broad system level coverage and ensure optimal margin for the variables encountered when a system design is in production and can also help to identify specific configurations, boards, and CPUs.

3.4 Link Width Change

The XGMI tuning process is carried out on a live system booted to Linux. Tuning cannot be carried out on lanes which are actively used by the system. So we need to first take offline the lanes which need to be tuned. In Zeppelin 2P system, lower lanes of Socket 0 are connected to upper lanes of Socket 1 and vice versa.

To perform tuning on upper lanes of Socket 0, make Socket 0 as Link Master and take offline the upper lanes. Similarly, make Socket 1 as Link Master to perform tuning on lower lanes of Socket 0 and upper lanes of Socket 1.

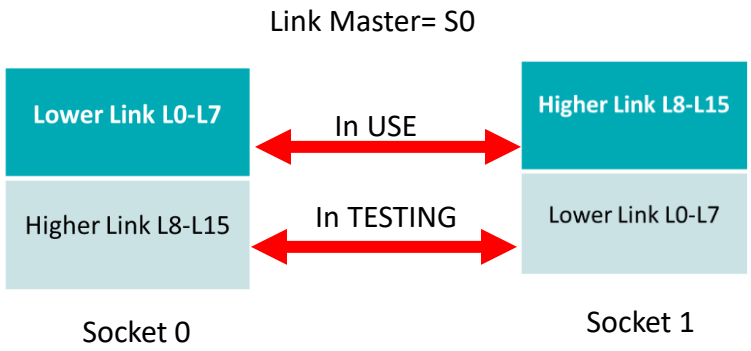


Figure 10. XGMI Tuning: Socket 0 Link Master

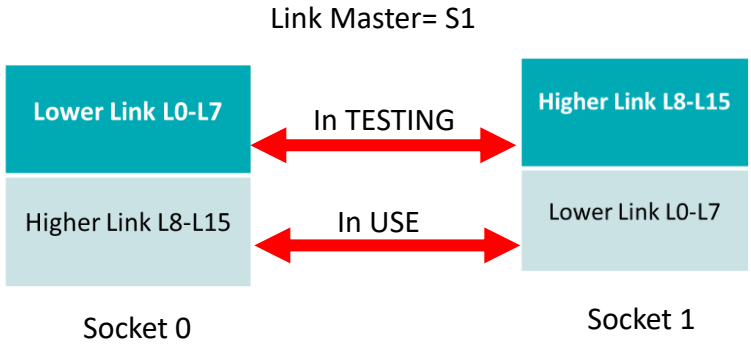


Figure 11. XGMI Tuning: Socket 1 Link Master

3.5 Using AGT for XGMI Tuning

AGT has the following command line options meant for xGMI tuning.

Table 2 . Command Line Options for xGMI Tuning

-i=<#>	Select the number of target dies on which tuning should be carried out. If there is more than one die, give a comma separated list like -i=2,3,4
-xgmitxeqtune=lane1-lane2	Tune TXEQ, and report phy eye size and setting for specified xGMI lanes in the range lane1-lane2. Valid range of lanes is from 0 to 15.
-xgmidatarate=rate	xGMI Link speed at which tuning should be done.
-xgmigenapcbfile	Generate xGMI settings in APCB format.
Notes 1. *AGT should always be run as a root user or in sudo mode.* 2. *All of device selection [-i option], lane selection [-xgmitxeqtune] and XGMI speed selection command line options should be given for performing XGMI tuning.*	

3.5.1 Selecting the Dies

Use the -i option in AGT to select the dies for XGMI tuning. AGT supports XGMI tuning only at die level and not at socket level.

Run the command

```
# sudo ./agt -i
```

This command lists all the devices available as shown in Figure 12. Note the instances of devices having the name '*AMD Zeppelin CPU Pkg[x] Die[y]'*. Only these device indexes should be given in the target die list for XGMI Tuning.

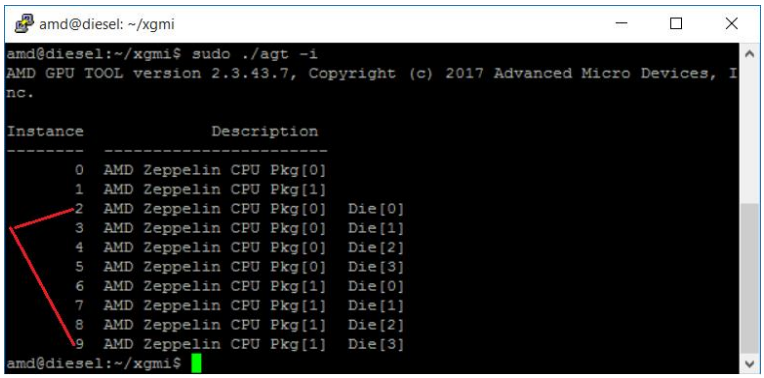


Figure 12. AGT: List Devices

Important to note is that for a selected die, AGT does XGMI tuning on both the transmitter die and receiver die. As an example, if '*-i=2'* selected, AGT does XGMI tuning after enabling transmitter settings on device=8 (Socket 1, Die 2) and receiver settings on device=2 (Socket 0, Die0. Refer to Figure 9 on page 8) and also in the reverse direction where device=2 is transmitting and device=8 is receiving.

So, the following device selection is enough to do tuning on all dies in a 2P system for selected lanes.

```
# sudo ./agt -i=2,3,4,5
```

3.5.2 Selecting the XGMI Lanes

The lanes to be tuned are specified using the command option.

-xgmitxeqtune=<lane1-lane2>	lane1=Start Lane number, lane2=End Lane number • Valid Lane numbers are between 0 and 15 • Both Start Lane and End Lane are mandatory. • Start Lane number should be less than End Lane.

XGMI link width on a Zeppelin system is x16 and the lanes are numbered from 0 to 15. For example, to tune the lower x8 lanes of (Socket 0, Die 0) which has device index 2, use the command.

```
# sudo ./agt -i=2 -xgmitxeqtune=0-7 -xgmidatarate=8
```

As noted before, the command does tuning on Socket 0 Die 0 lane 0-7 and also on Socket 1 Die 2 lane 8-15. The lower lanes of Socket 0 Die 0 are connected to upper lanes of Socket 1 Die 2.

The following command does XGMI tuning on all the lower lanes of Socket 0 dies and upper lanes of Socket 1 dies at XGMI speed 8.6 Gbps (see Section 3.5.3).

```
# sudo ./agt -i=2,3,4,5 -xgmitxeqtune=0-7 -xgmidatarate=8
```

3.5.3 Selecting XGMI Speed

The following command line option selects the XGMI speed at which tuning is performed.

-xgmidatarate=<speed> Speed options are:

Speed	*Actual XGMI Speed*
5	5.3GT/s
6	6.4GT/s
7	7.5GT/s
8	8.5GT/s
9	9.6GT/s
10	10.6GT/s

For example, to run XGMI tuning at 8.5 Gbps speed on upper lanes of Socket 0 and lower lanes of Socket 1, use the command:

```
# sudo ./agt -i=2,3,4,5 -xgmitxeqtune=8-15 -xgmidatarate=8
```

3.5.4 XGMI Tuning Output

After tuning, AGT generates a tuning report and intermediate files needed for APCB file format (see Section 3.5.5). It also generates the eye image diagrams for each (EqMain, EqPre, EqPost) setting applied during tuning.

As the user can run XGMI tuning at different XGMI Speeds (see Section 3.5.3), report folders and output folders are named based on XGMI Speed at which tuning is performed. The folders are created in the current directory. The following table gives a summary of the various output folders and files generated by the tool.

Table 3. Output Folders and Files Generated by the Tool

Name	Type	Description
xgmi_tuning_report_datarate_<xgmispeed>	Report Directory	This directory is created in the current directory from which AGT is run. As an example –xgmispeed=8 creates the report directory called *'xgmi_tuning_report_datarate_8'* This contains the XGMI Tuning reports: xgmi_tuning_report_s0h_s1l xgmi_tuning_report_s1h_s0l
xgmi_tuning_report_s0l_s1h_<timestamp>	Report File	This is the tuning report file generated when lanes 0-7 of Socket 0 dies are selected for tuning. The file name is appended with timestamp. This is created in the report directory.
xgmi_tuning_report_s0h_s1l_<timestamp>	Report File	This is the tuning report file generated when lanes 8-15 of Socket 0 dies are selected for tuning. The file name is appended with timestamp. This is created in the report directory.
xgmi_apcb	APCB Interim Files Directory	This directory is created in the current directory from which AGT is run. This contains the files needed for generating the final APCB output. Modifying the contents/Deleting this directory causes incorrect APCB file generation or no APCB file output.
PKG#xDIE#y_l#l1_#l2_r#s	Data Eye Image Directory	This directory is created in the current directory from which AGT is run. #x: Socket Number #y: Die Number #l1: Start Lane #l2: End lane #s: XGMI speed For example, tuning done on Socket 0, Die 0 lanes 0-7 at XGMI speed 8 creates the directory named *PKG0DIE0_l0_7_r8.*

Name	Type	Description
		This directory contains data eye image files.
datarate_#s_PKG#x_DIE#y_XGMI_chiplet_#c_#l_aabbcc.png	Data Eye Image File	This file is generated in the Data Eye image directory. #s: XGMI speed #x: Socket Number #y: Die Number #c: Chiplet name can be A1-A3 or B1-B3 #l: Lane number within the chiplet, value from 0-4 aa: EqMain bb: EqPost cc: EqPre An example image file name: *datarate_09_PKG0_DIE0_XGMI_chipletB4_LN3_400000.png*

3.5.5 APCB File Generation

AGT can be used to generate TXEQ settings in APCB format. After completing the tuning process, use the following command line option to generate APCB files.

```
# sudo ./agt -xgmigenapcbfile
```

AGT looks for the presence of a folder named *'xgmi_apcb'* and uses the files within the folder to generate APCB report. No file is generated if the folder is not found. The output files are generated in the current directory.

AGT generates a separate APCB file based on XGMI speed. The format of output APCB file name is:

xgmi_apcb_settings_#s.h	#s: Settings for XGMI speed s. For example *xgmi_apcb_settings_8.h* will have TXEQ settings in APCB format for XGMI speed 8.

Note: *Modifying the contents or deleting the files within APCB folder generates incorrect APCB report.*

A minimum of two runs -- the first one for lower lanes and the second for higher lanes -- are needed to find the optimal TXEQ settings for a particular speed. AGT scans *'xgmi_apcb'* directory and collates the tuning reports of lower and upper lanes to give the consolidated APCB output file for a particular XGMI speed.

Each APCB file needs to be pulled into BIOS via APCB (documented in AGESA specification).

Only the COMPACT macros are used in APCB output file. For more details on APCB macros, refer AGESA specification.

See Appendix A for format of generated APCB file.

Chapter 4 PCIe Gen4 Lane Margining

4.1 Lane Margining at Receiver

This feature is based on the lane margining procedure as mentioned in PCI Express Base Specification Rev 4. As described in the specification, this tool feature can be used to get the margin information of a given receiver while the link is in L0 state. The feature gives the margin information for timing from the current receiver position in both directions – left or right.

4.2 Lane Margining Requirements

PCI Express Specification requires the following conditions to be met for lane margining:

- The current data rate of the link must be 16 GT/s (Gen 4).
- The link should be in L0 state.
- Upstream port to be programmed to a D state that link should not enter L1 state.
- ASPM should be disabled in both the Upstream and Downstream port through Link Control register.
- The Hardware Autonomous Speed Disable bit of the Link Control register must be set to 1b in both the Upstream and Downstream port.
- The Hardware Autonomous Width Disable bit of the Link Control register must be set to 1b in both the Upstream and Downstream port.

Note: *User must ensure that all the above conditions are satisfied. For other limitations, refer to* **Section 4.3.6 Known Limitations** *on page 28.*

4.3 Using AGT for PCI Lane Margining

AGT has the following command line options meant for Lane Margining.

Table 4 . Command Line Options for Lane Margining

-i=<#>	Select the target gpu of the link. Only one link can be margined at a time so only one die should be selected.
-listport	This lists the ports/links on the target die. It lists all the ports available regardless of Gen1/2/3/4 speed. Only links at Gen4 speed should be selected for margining.
-marginport=<port number>,<receiver number>	Select the port number and receiver for PCI 1D margining. • Port number = Port number as obtained by –listport command • Receiver Number = 6. Only receiver 6 is supported presently.
-lanes=<lane1-lane2>	Select the range of lanes that should be margined. Lane numbers range from 0 to 15. This option should be used only with –margiport option • Lane1 = Start lane number • Lane2 = End lane number
Notes 3. *AGTshould always be run as a root user or in sudo mode.* 4. *User must ensure that the lanes are available when lanes option is selected. As an example, user should not try to margin lane number 15 when the current link width is only x8.*	

4.3.1 Selecting the Target Die

Use the -i option in AGT to select the gpu for Lane Margining. Run the command

```
# sudo ./agt-i
```

```
root@mi100-benchlab04:~# ./agt -i
AMD GPU TOOL version 2.5.0.5, Copyright (c) 2019 Advanced Micro Devices, Inc.

Instance              Description
--------  ------------------------------------------------------------
       0  VendorID: 0x1002 DeviceID: 0x67df SSID: 0x0b37 (0000:07:00.0)
       1  VendorID: 0x1002 DeviceID: 0x7388 SSID: 0x0834 (0000:0a:00.0)
```

Figure 136. PCI Margin: List Devices

This command lists all the devices available. Note the instances of devices having the name '*Instance VendorID: 0xXXXX DeviceID: 0xXXXX SSID: 0xXXXX (SEG:BUS:DEV:FN)'*. Only these device indexes should be given in the target device list. Only one link can be selected for margin at a time, so only one device index should be specified.

4.3.2 List the Root Ports/Links

-listport	This option is used to list all the available root ports/link on the selected device

```
root@mi100-benchlab04:~# ./agt -i=1 -listport
AMD GPU TOOL version 2.5.0.5, Copyright (c) 2019 Advanced Micro Devices, Inc.

 Port Num: 0 Bus: 0x8 Device: 0x0 Function: 0x0 | Link Width = x8 |  Link Speed = 4
```

Figure 147. PCI Margin: List Ports

A target device should be selected with -i option. As shown in Figure 14, this option shows the following information.

- Port Number
- PCI Bus, Device, Function number
- Current Link Speed – 1, 2, 3, 4 indicates Gen 1/2/3/4 speed, respectively
- Current Link Width

Only links at 16 GT/s (Gen4) speed and lanes available within current link width are expected to be margined. User should use the above information while selecting the port/link and lanes for margining.

4.3.3 Select the Lanes

The lanes to be tuned are specified using the command option.

-lanes=<lane1-lane2>	lane1=Start Lane number lane2=End Lane number • Valid Lane numbers are between 0 and 15. • Both Start Lane and End Lane are mandatory. • Start Lane number should be less than End Lane. • The lane numbers should fall within the current link width. As an example, user should not include lanes 8-15 for a x8 link.

This option should be used along with –marginport option to select the lanes for margining. For example, to margin all the 16 lanes of port number 0 of (Socket 1) with device index 1, use the command.

```
# sudo ./agt -i=1 -marginport=0,6 -lanes=0-15
```

Note: *User must make sure that only valid lane numbers are specified. If a lane number beyond the current link width is mentioned, the link cannot be margined further. It will need a system reboot.*

4.3.4 Margin Link

The links to be margined are specified using the following command option.

-marginport=<port no>,<rcvr no>	port no = Port selected for margining rcvr no = Receiver number • Port number should be one of the numbers got using –listport option. • Only receiver number 6 is supported presently. • Both port and receiver numbers are mandatory.

Note: *User must make sure that only root ports/links which are having Gen4 speed are selected for margining (for 1D lane margin).*

```
root@mi100-benchlab04:~# ./agt -i=1 -marginport=0,6
AMD GPU TOOL version 2.5.0.5, Copyright (c) 2019 Advanced Micro Devices, Inc.

 Starting Margining ....
 Margining Lane 0
 Margining Right....
..........
 Margining Left....
........
 Margining Lane 1
 Margining Right....
........
 Margining Left....
....
 Margining Lane 2
 Margining Right....
..........
 Margining Left....
..........
```

Figure 158. PCI Margin: Margin Link

Figure 15 shows an example of how to use margin option in AGT after selecting the port and lanes. This command will exit immediately if the link doesn't meet the margining requirements as mentioned in Section 4.2.

4.3.5 Lane Margining Reports

After margining, AGT generates margining reports under 'pcie_margin_reports' directory for 1D PCI margin. The 'pcie_margin_reports' directory is created in the current directory of AGT tool.

The report files are of .csv format. They are named as pcie_margin_results_B<bus number>_D<device number>_F<function number>.csv based on the root port/link selected for margining.

Table 5. Output Folders and Files Generated for Gen4 Lane Margining

Name	Type	Description

Name	Type	Description
pcie_margin_reports (for 1D)	Report Directory	This directory is created in the current directory from which AGT is run for 1D.
pcie_margin_results_B<bus number>_D<device number>_F<function number>.csv	Report File	This file contains the margin results of the port/link selected. The file is named after the pci bus, device and function numbers of the port selected. The results of successive runs on the same port are appended to the same file.

4.3.6 Known Limitations

- AGT does not identify a device by the physical slot number. User is expected to identify the right End point device and slot based on the PCI bus/device/function numbers of root port and End point device.
- User should make sure that the current link state meets the preconditions required for margining as mentioned in Section 4.2 on page 23. AGT checks if margining requirements are met, but the tool does not transition the link to the required state if margining requirements are not met.
- During margining, AGT checks if the margining requirements are met before applying a new timing offset. In case the link goes to a state other than L0 or if a speed downgrade happens, the tool does not take any corrective action to restore link state. The tool will exit immediately in the case of these kinds of errors.
- AGT can only margin one link at a time, and lane margining is done sequentially. The tool does not margin the lanes in parallel even if the link is capable of it as reported under Report MaxLanes capability.
- AGT margins only Upstream Port Receiver, identified as receiver number 6, of GPU pcie ports. Margining of Retimer receivers are not supported presently.

Appendix A XGMI Tuning Output Files

This section only gives an overview of the expected format of output files from XGMI tuning. The values shown in the sample output files in this section are only representative and do not represent the actual results or full results after XGMI tuning.

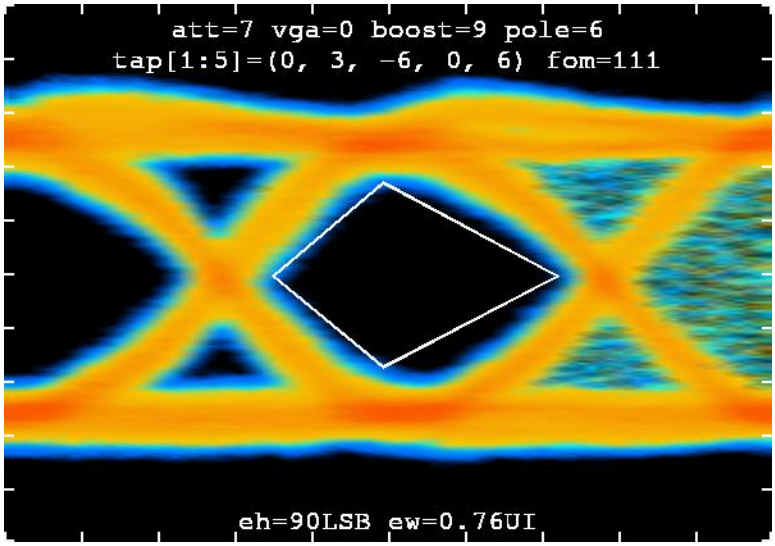
A.1 XGMI Tuning Report Format

Sample XGMI tuning report having the TXEQ optimal settings for Socket 0 Die 0 (upper lanes) and Socket 1 Die 2 (lower lanes).

```
tx_eq = [
[ #PKG1DIE2
                [ 40, 0, 0 ] # main, post, pre, lane 7
                [ 40, 0, 0 ] # main, post, pre, lane 6
                [ 40, 0, 0 ] # main, post, pre, lane 5
                [ 40, 0, 0 ] # main, post, pre, lane 4
                [ 40, 0, 0 ] # main, post, pre, lane 3
                [ 40, 0, 0 ] # main, post, pre, lane 2
                [ 40, 0, 0 ] # main, post, pre, lane 1
                [ 40, 0, 0 ] # main, post, pre, lane 0
]
[ #PKG0DIE0
                [ 28, 0, 0 ] # main, post, pre, lane 15
                [ 28, 0, 0 ] # main, post, pre, lane 14
                [ 28, 0, 0 ] # main, post, pre, lane 13
                [ 28, 0, 0 ] # main, post, pre, lane 12
                [ 28, 0, 0 ] # main, post, pre, lane 11
                [ 28, 0, 0 ] # main, post, pre, lane 10
                [ 28, 0, 0 ] # main, post, pre, lane 9
                [ 28, 0, 0 ] # main, post, pre, lane 8
]
]
```

A.2 Data Eye Image File Format

The image shows the eye generated and the RX Adaptation settings.



A.3 Sample APCB Output File Format

APCB file generated for -xgmidatarate=10, after performing tuning on Socket 0 Die0 and Socket 1 Die 2. *This file only shows partial data (only for one set of lanes). Actual output will have data for all 16 lanes.*

```
////////////////////////////////////////////////////////////////////////////////
/// ADVANCED MICRO DEVICES, INC.
/// This an auto generated file. Do not modify.
////////////////////////////////////////////////////////////////////////////////

MAKE_TX_EQ_FREQ_TABLE_COMPACT(XGMI_TX_EQ_SPEED_106,
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET0_DIE0_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 37, 0, 12 ),
        MAKE_TX_EQ_LANE_DATA( 38, 0, 8 ),
        MAKE_TX_EQ_LANE_DATA( 40, 0, 0 ),
        MAKE_TX_EQ_LANE_DATA( 40, 0, 0 ),
        MAKE_TX_EQ_LANE_DATA( 40, 0, 0 ),
        MAKE_TX_EQ_LANE_DATA( 38, 0, 8 ),
        MAKE_TX_EQ_LANE_DATA( 40, 0, 0 ),
        MAKE_TX_EQ_LANE_DATA( 37, 0, 12 )
    ),
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET1_DIE2_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 23, 11, 5 ),
        MAKE_TX_EQ_LANE_DATA( 26, 5, 0 ),
        MAKE_TX_EQ_LANE_DATA( 25, 0, 8 ),
        MAKE_TX_EQ_LANE_DATA( 27, 0, 2 ),
        MAKE_TX_EQ_LANE_DATA( 25, 5, 5 ),
        MAKE_TX_EQ_LANE_DATA( 28, 0, 0 ),
        MAKE_TX_EQ_LANE_DATA( 24, 5, 8 ),
        MAKE_TX_EQ_LANE_DATA( 26, 5, 0 )
    )
),
```

A.4 Margin output files

Margin report CSV file, i.e marginData_<TImeStamp>.csv.

A.5 PCI Gen4 Lane Margining Sample Output

```
root@mi100-benchlab04:~# cat pcie_margin_reports/pcie_margin_results_B8_D0_F0.csv
   Lane Number,  Right Margin,   Left Margin,  Total Margin,     Pass/Fail
             0 ,        0.29UI,       -0.26UI,        0.56UI,        Pass
             1 ,        0.24UI,       -0.15UI,        0.38UI,        Pass
             2 ,        0.29UI,       -0.32UI,        0.62UI,        Pass
             3 ,        0.29UI,       -0.24UI,        0.53UI,        Pass
             4 ,        0.29UI,       -0.18UI,        0.47UI,        Pass
             5 ,        0.26UI,       -0.32UI,        0.59UI,        Pass
             6 ,        0.29UI,       -0.24UI,        0.53UI,        Pass
             7 ,        0.26UI,       -0.26UI,        0.53UI,        Pass
```