HYGON

成都海光集成电路设计有限公司

Chengdu Haiguang Integrated Circuit Design Co., Ltd.

HYGON HGT User Manual

内容

修订记录

日期	版本	描述
2020/05/06	1.4	初始发布版本
2020/05/23	1.5	修订部分缺陷
2020/05/28	1.6	修订部分缺陷
2020/09/28	2.0	增加 XGMI TX EQ Tune 功能

1. 概述

HGT(Hygon xGMI margin Tool) 是海光公司开发的用于评估和优化海光处理器的 SATA/PCIe/xGMI 信号完整性工具。

1.1 功能描述

HGT 提供了以下子命令：

- eye：绘制并分析 xGMI、PCIe、SATA 等 RX 链路的眼图（PCIe、SATA 链路眼图只用作直观对比参考，不能作为信号测试通过与否的标准）。
- xgmitxeqtune：对 XGMI 链路 TX EQ 参数做 Tune 测试
- xgmigenapcbfile：根据 XGMI TX EQ Tune 结果产生 BIOS 所需 APCB 头文件

1.2 运行环境

HGT 的运行环境要求如下：

硬件	搭配了 海光处理器的主板
处理器	海光 ES 芯片 海光 MP 芯片（需要 unlock）
BIOS PI 版本	不低于 1.0.1.0
操作系统	Linux（Ubuntu server 16.04，CentOS 7.5）
HGT 版本	2.0

特别注意

如需在海光量产版本芯片(MP)上运行 HGT，必须先对处理器进行安全解锁(unlock)操作。**unlock** 工具和使用文档请联系海光 FAE 现场工程师获取，对应工具名称为 HygonSecureUnlockTool。

2. 眼图绘制

本章详细介绍 HGT 的眼图绘制步骤。

2.1 环境需求

- 目标链路应连接了设备。
- 执行 HGT 命令需要 root 权限

2.2 运行方法

2.2.1 确定目标 Die 和 Lane

首先需要确认待测 lane 的编号及 DIE 编号。

DXIO 由 Type A 和 Type B 两部分组成，TYPE A 中的 PHY 是多功能的，支持 SATA/PCIE/ xGMI 三种协议, Type B 下的 PHY 支持 xGMI/PCIE 两种协议。Type A 由 2 个 X2 的 PHY 和 3 个 X4 的 PHY 组成,包含 16 条 Lane， Type B 同样也是由 2 个 X2 的 PHY 和 3 个 X4 的 PHY 组成, 包含 16 条 Lane。

命令： ./hgt eye -s -d 0

Command	Parameter	Notes
eye	-s	显示 DXIO 各个 PHY 的工作模式
	-d	指定 die number

执行命令之后，信息以表格形式输出，如图一所示。“lane”表示 DXIO lane 编号，“mode”表示 lane 的配置模式，“P”表示 PCIe， “S”表示 SATA， “G”表示 xGMI (注：若某条 lane 或者某几条 lane 没有配置，则默认显示为 G)

```
[root@localhost hgt]# ./hgt eye -s -d 0
Die 0
+-------+-------------------------------+-------------------------------+
|       |            Type B             |            Type A             |
+-------+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       |3|3|2|2|2|2|2|2|2|2|2|2|1|1|1|1|1|1|1|1|1|1| | | | | | | | | | |
| lane  |1|0|9|8|7|6|5|4|3|2|1|0|9|8|7|6|5|4|3|2|1|0|9|8|7|6|5|4|3|2|1|0|
+-------+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| mode  |   P   |   P   |   P   | P | P |   P   |   G   |   S   | S | S |
+-------+-------+-------+-------+---+---+-------+-------+-------+---+---+
```

图 1 HGT 显示 lane 配置

2.2.2 采集数据并绘制眼图

命令： ./hgt eye -d 3 -m 100000

Command	Parameter	Notes
eye	-d	指定 die number
	-m	以位的形式指定待测试 lane， 16 进制无符号整数，bit 位为“1”表示测试 lane，为“0”表示不测试

使用-m 参数按位指定某个 Die 的若干 Lane(s)，然后采集数据并绘制眼图。

如图 2 中的“-m 100000”，其中的 100000 是 16 进制数，bit20 被置“1”, 表示测试目标是 lane 20, 可以参考上一章节 2.2.1 中‘-s’配合测试，注意被测的 Lane 要连接正常工作的设备，否则无法测试。

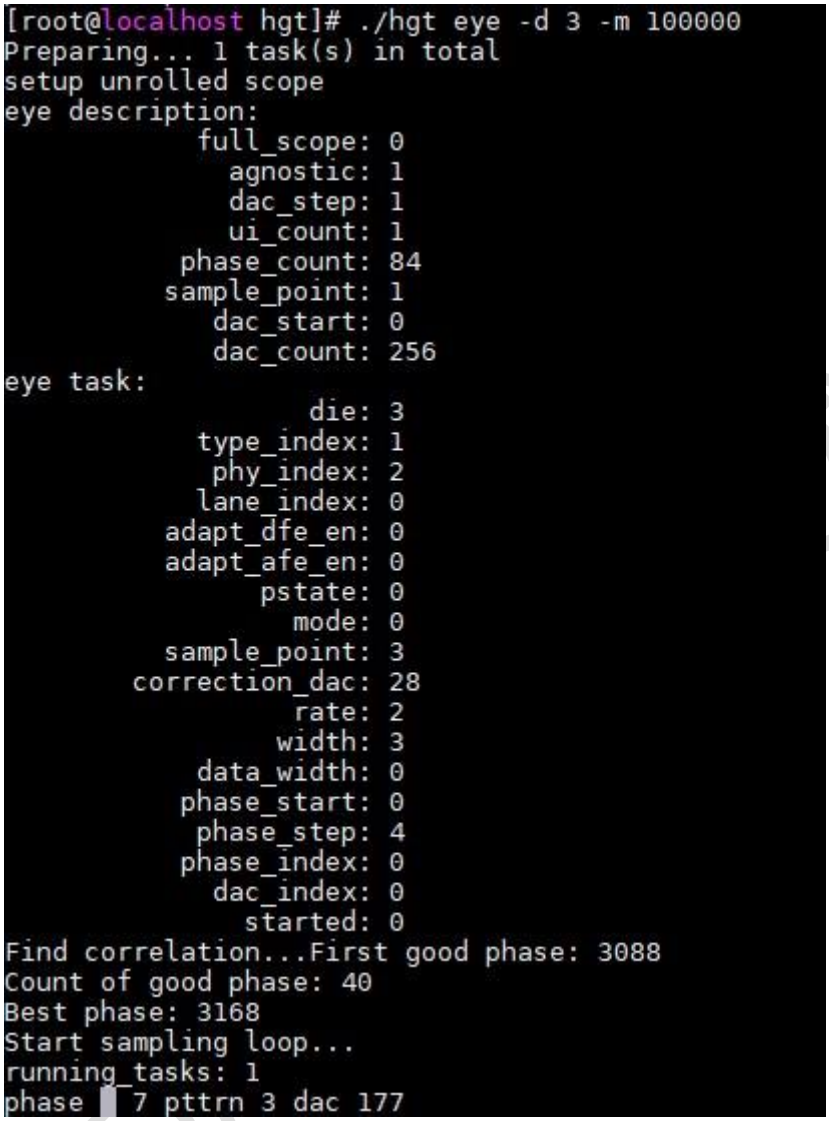

图 2 HGT 绘制眼图

2.2.3 查看结果

眼图绘制完成后，HGT工具会在当前目录下为每条 lane 生成两个文件：*.bmp和*.txt。假设绘制了 Die 3 Lane 20 的眼图，那么会生成D3T1P2L0.bmp 和 D3T1P2L0.txt 两个文件，其中 D3T1P2L0.bmp 是眼图，如图3所示。

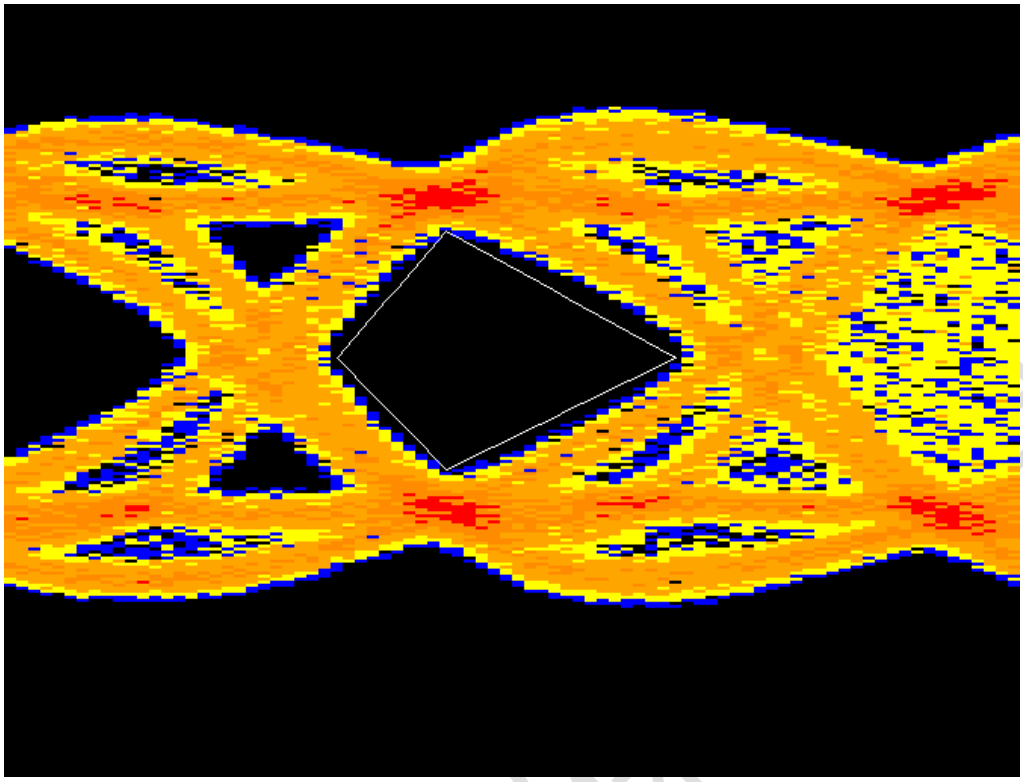

图 3 眼图示意图

D3T1P2L0.txt 中包含了眼图的测量数据和该 Lane 的当前状态，如图4所示：

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
Eye opening area: 1331 units
Eye max width: 29 units, 0.69 UI
Eye max height: 80 units

att 7
vga1 0
vga2 0
boost 5
pole 6
tap1 124
tap2 5
tap3 -7
tap4 14
tap5 14
fom 95
```

图 4 眼图测量数据

2.3 xGMI 测试通过标准

由于 xGMI 是 HYGON CPU 间特有的互联链路，HGT 给出了 xGMI 信号完整性的测试标准。对于 xGMI 测试，满足如下标准即通过。

表格 1 xGMI 测试标准

Max Eye width	Max Eye height
>0.30 UI	>25 units

举例：

在 2.2.3 中介绍了如何查看 HGT 的测试结果，结果中包含了一个 txt 文件用来描述眼图的测量数据，关注下图中红框部分：

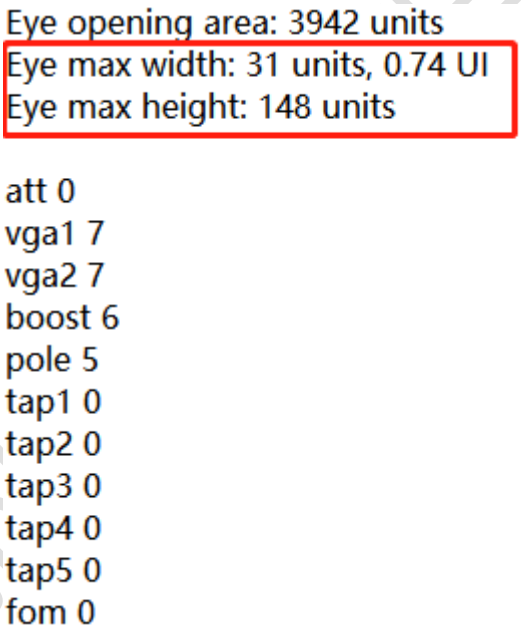

图 5 XGMI 测试举例

可以得知 Max Eye width=0.74 UI, Max Eye height=148 units

那么参考表 1 XGMI 的测试标准：

Max Eye width = 0.74 UI > 0.30UI

Max Eye height = 148 units > 25 units

测试结果表明这条 xGMI lane 测试通过。

3. XGMI TX EQ Tune

XGMI TX EQ Tune 可以测试每条 xGMI lane 的 TX EQ 参数，并以*.h 头文件格式输出测试结果，将测试结果更新到 BIOS 中可以增强 XGMI 信号，改善眼图质量。

3.1 环境需求

- 双路服务器
- 执行 HGT 需要 root 权限

3.2 Tune 测试

XGMI TX EQ Tune测试使用子命令xgmitxeqtune，参数说明如下：

Command	Parameter	Notes
xgmitxeqtune	-d die	specify die index
	-m lanes_bitmask	specify lanes indexed in one die, bit[15:0]: xGMI(default: 0xFFFF)
	-r datarate	specify data rate, 10:10.6G(default) 9: 9.6G 8: 8.5G
	-h	show this message

命令示例：

./hgt xgmitxeqtune -d 1 -m 0xffff -r 10

Tune测试需要在发送端遍历多组EQ参数，在RX端评估眼图质量，每个die 16条lane的测试时间约1.5小时。

3.3 生成 APCB 头文件

XGMI TX EQ Tune测试完成后可以使用子命令xgmigenapcbfile生成XGMI APCB头文件，执行命令如下：无需参数。

./hgt xgmigenapcbfile

执行后会产生如下三个文件：

```
xgmi_apcb_settings_10.h
xgmi_apcb_settings_9.h
xgmi_apcb_settings_8.h
```

将这三个头文件拷贝到 PI 目录

AgesaPkg/Addendum/Apcb/#BoardName#/ApcbData/AGT_Data/Instance0/

然后重新编译集成该 PI 的 BIOS 即可。

XGMI APCB头文件内容示例见附录4.1。

3.4 完整 tune 测试

使用 xgmitxeqtune_all.sh 可以对所有 die 的全部 lane，全部速率进行 tune 测试，并生成 APCB 头文件。7 x x x 双路系统完整 tune 测试时间大约需要 36 小时。

```
#!/bin/bash

DieNum=$(lspci -n -d :1450|wc -l)

for((data_rate = 10; data_rate > 7; data_rate--))
do
        for((die = 0; die < $DieNum; die++))
        do
                ./hgt xgmitxeqtune -d $die -m 0xffff -r $data_rate
        done
done
```

```
./hgt xgmigenapcbfile
```

4. 附录

4.1 XGMI APCB 头文件示例

```
////////////////////////////////////////////////////////////////////////////////
/// Copyright 2020 HYGON, INC. All Rights Reserved.
/// This file was generated by hgt xgmitxeqtune. Do not modify.
////////////////////////////////////////////////////////////////////////////////

MAKE_TX_EQ_FREQ_TABLE_COMPACT(XGMI_TX_EQ_SPEED_107,
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET0_DIE0_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  0
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  1
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  7
        MAKE_TX_EQ_LANE_DATA( 30, 32,  8 ), // lane  8
        MAKE_TX_EQ_LANE_DATA( 32, 32,  0 ), // lane  9
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane 10
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 11
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 12
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane 13
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane 14
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane 15
    ),
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET0_DIE1_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane  0
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  1
```

```
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  7
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  8
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  9
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 10
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane 11
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane 12
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 13
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 14
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane 15
    ),
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET0_DIE2_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  0
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  1
        MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 28, 36, 12 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  7
        MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane  8
        MAKE_TX_EQ_LANE_DATA( 28, 40,  8 ), // lane  9
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane 10
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane 11
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane 12
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 13
```

```
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 14
        MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane 15
    ),
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET0_DIE3_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 28, 48,  0 ), // lane  0
        MAKE_TX_EQ_LANE_DATA( 28, 48,  0 ), // lane  1
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 28, 48,  0 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 28, 40,  8 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  7
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane  8
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  9
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 10
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane 11
        MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane 12
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 13
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane 14
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 15
    ),
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET1_DIE0_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  0
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  1
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  7
```

```
            MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  8
            MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  9
            MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane 10
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 11
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 12
            MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane 13
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 14
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 15
        ),
        MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET1_DIE1_EACH_LANES_COMPACT(),
            MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane  0
            MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane  1
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  2
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  3
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  4
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  5
            MAKE_TX_EQ_LANE_DATA( 29, 32, 12 ), // lane  6
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  7
            MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane  8
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane  9
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 10
            MAKE_TX_EQ_LANE_DATA( 29, 44,  0 ), // lane 11
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 12
            MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane 13
            MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 14
            MAKE_TX_EQ_LANE_DATA( 28, 44,  4 ), // lane 15
        ),
        MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET1_DIE2_EACH_LANES_COMPACT(),
            MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  0
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  1
```

```
        MAKE_TX_EQ_LANE_DATA( 30, 32,  8 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 30, 32,  8 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 32, 32,  0 ), // lane  7
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  8
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  9
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane 10
        MAKE_TX_EQ_LANE_DATA( 30, 40,  0 ), // lane 11
        MAKE_TX_EQ_LANE_DATA( 29, 40,  4 ), // lane 12
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 13
        MAKE_TX_EQ_LANE_DATA( 29, 36,  8 ), // lane 14
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 15
    ),
    MAKE_TX_EQ_LIST_COMPACT(MAKE_TX_EQ_SOCKET1_DIE3_EACH_LANES_COMPACT(),
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  0
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  1
        MAKE_TX_EQ_LANE_DATA( 32, 32,  0 ), // lane  2
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  3
        MAKE_TX_EQ_LANE_DATA( 31, 28,  8 ), // lane  4
        MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane  5
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane  6
        MAKE_TX_EQ_LANE_DATA( 32, 32,  0 ), // lane  7
        MAKE_TX_EQ_LANE_DATA( 32, 32,  0 ), // lane  8
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane  9
        MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 10
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane 11
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane 12
        MAKE_TX_EQ_LANE_DATA( 31, 36,  0 ), // lane 13
```

```
            MAKE_TX_EQ_LANE_DATA( 31, 32,  4 ), // lane 14
            MAKE_TX_EQ_LANE_DATA( 30, 36,  4 ), // lane 15
        ),
    ),
```