|  |  |  |
| --- | --- | --- |
|  | | 芯驿电子科技（上海）有限公司  021-67676997  www.alinx.cn |
| **GTX** 光纤通信测试例程  黑金动力社区 **2021-10-13** | | |
| **1** 实验简介  本实验将介绍通过光纤实现数据通信的传输，测试数据由 FPGA 自身产生, 由 GTX 发送到第一 路光纤口，然后通过光纤线环路到第二路光纤口，通过 GTP/GTX 接收数据进行校验。  **2** 实验原理  **2.1 GTX IP** 设计  XILINX 的 Vivado 软件已经为用户设计好了 GTX IP，用户无需关心 GTX 的内部具体工作就可以 使用 IP 来实现 GTX 的高速的数据收发。下面我具体的 GTX IP 的生成和配置方法：  1. 在 IP Catalog 界面中双击 FPGA Features and Design\IO Interface 目录下的"7 Series FPGAs Transceivers Wizard"图标。   |  | | --- | |  |   2. Component Name 栏输入"gtx"为取名，在 GT selection 界面里无需修改，保留默认。 | | |
| 黑金 FPGA | 版权所有 | **1** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 3. 在 Line Rate, Refclk Selection 界面里，首先设置 GTX 的传输协议位“Aurora 8B10B single lane 4byte“， 我们在前面一章讲过， Xilinx 的 GTX 是支持很多种协议的，Aurora 8B/10B 协议是一个可 扩展的、轻量级的链路层协议，可以用于通过一条或多条串行链路将数据点到点传输。这里我们 用的光模块传输是单路的，所以选择 single lane, 数据接口为 4byte, 就是 32 位数据。再选择 TX 和 RX 的 Line Rate 速度，这个 Line Rate 速度是需要是 GTX 参考时钟的整数倍，开发板上的 GTX 参考 时钟为 125Mhz, 这里我们 Line Rate 为参考时钟的 80 倍，所以 Line Rate 设置为 10Gbps ，如果用户 需要设置其它速率比如 1Gbps,只需要直接修改。Reference Clock 为 125Mhz。 | |
| 黑金 FPGA 开发板 | **2** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 以 AX7Z035 开发板为例, xc7z035ffg676 芯片有 2 个 BANK 的 GTX 收发器，在 AX7Z035 硬件 设计上 BANK111 连接了 4 路 SFP+的光模块，也就是 Quad2。所以这里需要选中 GTX\_X0Y8, GTX\_X0Y9, GTX\_X0Y10, GTX\_X0Y11, 然后选择右边的"Use GTX X0Y8/9/10/11"前面的钩，这样 QPLL 都连接到了 4 个 GTX Channel 模块。    另外 TX Clock Source 和 RX Clock Source 需要都选择 REFCLK0 Q2, 因为电路设计上 125Mhz 的参 考时钟是连接到 BANK117 的 REFCLK0 的管脚上的。 | |
| 黑金 FPGA 开发板 | **3** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 4. 在 Encoding and Clocking 界面里，设置 TX 和 RX 的外部数据宽度， 8B/10B 使能，内部数据宽度 等信息， 这里设置内部数据宽度为 40。因为外部数据宽度是 32 ，通过 8B/1B 转换后为 40。    4. 在 Comma Alignment and Equalization 界面里，选择默认设置。这里选择 Comma 值为 K28.5, K28.5 是一种用以表示 Fibre Channel 操作开始的特殊 10 比特字符。 8B/10B 编码中将 K28.5 作为 K 码的控制字符，称为“comma”，所以可以用 comma 字符指示帧的开始和结束标志，或始终修正和 数据流对齐的控制字符。 | |
| 黑金 FPGA 开发板 | **4** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| |  | | --- | |  |   5. PCIe,SATA, PRBS 页面无需修改， 保持默认设置。 | |
| 黑金 FPGA 开发板 | **5** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| |  | | --- | |  |   6. 在 CB and Sequence 页面中不用修改。 | |
| 黑金 FPGA 开发板 | **6** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| |  | | --- | |  |   7. 在 Summary 界面中检查一下配置情况，点击 OK 完成。 | |
| 黑金 FPGA 开发板 | **7** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 关于更多 GTX IP 的配置信息，大家请参考 Xilinx 提供的文档“pg168-gtwizard.pdf”和“UG476 7 Series FPGAs GTX\_GTH Transceivers User Guide.pdf”，这两个文档上有对 GTX IP 各个配置参数做了 详细的介绍。  8. 配置完成在工程中自动添加刚刚生成 gtx 的 IP。   |  | | --- | |  |   9. 我们来生成 GTX IP 的 example 工程，右键选择 gtx,在下拉菜单里选择"Open IP Example Design..."。 | |
| 黑金 FPGA 开发板 | **8** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 10. 生成的 example 工程如下图所示，在这个例子工程中，程序会在 gt0\_frame\_gen 模块中产生测 试数据进行 GTX 的数据传输，在 gt0\_frame\_check 模块接收并检查是否正确，如果不正确， 错误 统计值增加。   |  | | --- | |  |   关于 example 的工程代码和测试我们这里不做介绍，大家自己去看去测试就可以了。在后面的 GTX 数据传输的软件开发中我们会用到这里 example 工程中的一些文件和 IP。 | |
| 黑金 FPGA 开发板 | **9** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| **2.2** 硬件介绍  在 AX7Z035 开发板上，有 4 路光纤接口 OPT1~OPT4，分别连接到 FPGA 芯片的 GTX 的通道 上。  其中 SFP1 光模块接口连接到 GTX 的 Channel0 上， SFP2 跟 GTX 的 Channel1 相连， SFP3 跟 GTX 的 Channel2 相连， SFP4 跟 GTX 的 Channel3 相连。  光模块的 TX\_Disable 信号连接到 FPGA 的普通 IO 上，SFP1 和 SPF2 用同一个信号控制，SFP3 和 SFP4 用同一个信号控制。TX\_Disable 信号用来使能或者不使能光模块的光发射，如果  TX\_Disable 信号为高，光发射关闭，否则光发送使能， 正常使用的时候需要拉低此信号。 LOSS 信 号这里不使用。TSFP1 光模块硬件原理图如下：    **3** 程序设计  光纤数据传输的 FPGA 程序设计是在 example 的工程代码的基础上添加了一个 TOP 文件和 3 个.v 文件， 工程的逻辑框图如下图所示： | |
| 黑金 FPGA 开发板 | **10** / **20** |

|  |  |  |
| --- | --- | --- |
| **www.alinx.com** |  | |
| GTX接收  GTX发送   |  | | --- | | 200M差分 时钟 |  |  | | --- | | 125M差分 时钟 |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Packet rec.v  Gtx\_exdes.v  Word\_align.v  \_   |  | | --- | | 32位数据对齐模 块 |  |  |  |  | | --- | --- | --- | | |  | | --- | | GTX 数据接收模块 |   GTX收发器IP   |  | | --- | | GTX 数据发送模块 | |   rx\_data\_align  rx0\_data  packet\_cnt\_o  rx0 clk  error\_packet\_cnt\_o  rx\_ctrl\_align  rx0\_kchar  tx1\_data  tx1\_kchar  rx1\_clk   |  | | --- | | ILA |   FPGA   |  | | --- | | PLL |   200mhz  100mhz   |  | | --- | | 接收数据校 验模块 |   Packet\_send.v   |  | | --- | | 测试数据产生模块 |  |  | | --- | | IBUFDS | | | | 光模块2  光纤  光模块1 |
| 程序产生数据使用 GTX IP 发送给外部的光膜块 1，光模块 1 把电信号转换成光信号通过光纤 传输到光模块 2，光模块 2 又把光信号转换成电信号输入到 FPGA 的 GTX 接收，GTX 接收到的数据 需要做一个 32 位数据对齐之后， 并解析出数据信号和控制数据信号，校验模块对这些数据信号和 控制信号进行校验计算判断接收数据和发送数据是否一致。  光纤数据传输设计好的工程如下图所示：    这里工程添加了 ILA 工具可以用来查看接收到的数据。  **1**） **gtx** 数据通信模块  在前面我们已经生成过 gtx IP 的 example 工程， 这里删除了 gt0\_frame\_gen.v 模块和 gt0\_frame\_check.v 模块。 因为这两个是测试数据的产生和检查模块， 本例程用不上。另外对 gtp\_exdes.v 文件进行修改，主要是删除 gt0\_frame\_gen.v 模块和 gt0\_frame\_check.v 模块的例化, 再 | | |
| 黑金 FPGA 开发板 | **11** / **20** | |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 在模块的 Port 处添加以下的 4 个 channel 的用户接口信号，添加后的四个通道的信号如下图所 示：    再把端口定义的四个 Channel 的信号和 gtx\_support 子模块里的信号进行连接。   |  | | --- | |  | | |
| 黑金 FPGA 开发板 | **12** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 下面对在 gtx\_exdes.v 端口中添加的几个 GTX 用户接口信号做一下介绍，以下以 channel0 的 GTX 接口为例：   |  |  |  |  | | --- | --- | --- | --- | | 信号名 | 位数 | 输入输出 | 说明 | | tx0\_clk | 1 | 输出 | 数据发送时钟，也就是第十四部分介绍的 TXUSRCLK2， 频率为 GTX 的参考时钟 250Mhz 。数据在上升沿有效。 | | tx0\_data | 32 | 输入 | GTX 发送数据。 | | tx0\_kchar | 4 | 输入 | GTX 发送的 K 控制字， 用来指示发送的数据是 K 码控制 字符还是正常传输数据。高电平表明是 K 码控制字符， 4 位对应发送数据的 4 个 Byte。  tx0\_kchar [3] 对应 tx0\_data [31:24]  tx0\_kchar [2] 对应 tx0\_data [23:16]  tx0\_kchar [1] 对应 tx0\_data [15:8]  tx0\_kchar [0] 对应 tx0\_data [7:0] | | rx0\_clk | 1 | 输出 | 数据接收时钟，也就是第十四部分介绍的 RXUSRCLK2， 频率为 GTX 的参考时钟 250Mhz 。数据在上升沿有效。 | | rx0\_data | 32 | 输出 | GTX 接收数据。 | | rx0\_kchar | 4 | 输出 | GTX 接收 K 控制字，用来指示接收的数据是 K 码控制字 符还是正常传输数据。高电平表明是 K 码控制字符，4 位 对应接收数据 32 位的 4 个 Byte。  rx0\_kchar [3] 对应 rx0\_data [31:24]  rx0\_kchar [2] 对应 rx0\_data [23:16]  rx0\_kchar [1] 对应 rx0\_data [15:8]  rx0\_kchar [0] 对应 rx0\_data [7:0] | | gt0\_txfsmresetdone | 1 | 输出 | GTX 初始化完成信号。 |   知道了 GTX 用户接口信号的含义，我们就能通过这些用户接口实现光纤数据的收发了。  **2**） **gtx** 数据包发送模块 **packet\_send.v**  在 packet\_send.v 中会由一个状态机来发送测试数据，首先一包数据开始传输前，会先发送同 步包头信号，再发送数据包的序号和控制信号。然后把这测试数据通过 GTX 发送出去。 发送流程 如下图： | |
| 黑金 FPGA 开发板 | **13** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| Tx\_packet\_req 为高   |  | | --- | | SEND\_SEQ  \_NUM |  |  | | --- | | IDLE |   SEND\_HEAD  ER  SEND\_DUMM  Y   |  | | --- | | SEND\_DAT  A\_END |  |  | | --- | | SEND\_DAT  A |  |  | | --- | | SEND\_CHE  CK |  |  | | --- | | SEND\_CORR  ECTION |  |  | | --- | | SEND\_CTR  L |   所有的 GTX 发送的数据位数为 32 位， 在发送数据包之前需要发送 32 位的数据校正控制字， 校正控制字用于消除不同的系统时钟之间造成的频率误差。gt\_tx\_ctrl 信号设置为 1111 ，表明这 32 位数据都是控制字。    这个 32 位的校正字（F7F7F7F7）在 GTX 的 IP 里已经配置好了。所以 GTX 接收数据包的时候 会自动做校正    同步信号包头信号定义为 32 位的“ff\_00\_00\_bc ”，低 8 位"bc"是 K28.5 码控制字符。K 码特 征字定义在 Xilinx 的“7 Series FPGAs GTX\_GTH Transceivers User Guide.pdf”文档里有描述。 | |
| 黑金 FPGA 开发板 | **14** / **20** |

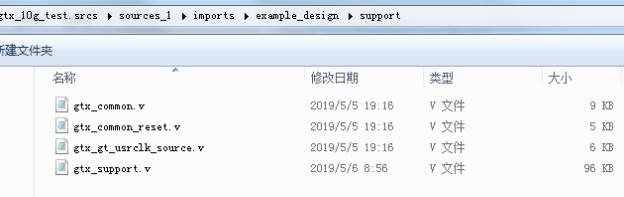
|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 向 GTX 发送 K28.5 码控制字符时， 需要拉高 gt\_tx\_ctrl 信号的对应位， 标示发送数据里的某个 字节位为 K 码控制字。所以这里在向 GTX 发送同步信号时，gt\_tx\_ctrl 信号设置为 0001，发送其 它数据的时候则置为 0000。   |  | | --- | |  |   **3**）位数据对齐模块 **word\_align.v**  GTX 收发器外部用户数据接口的宽度为 32 位， 内部数据宽度为 40 位(8b/10b 转换)。在实际 测试过程中发现，发送的 32 位数据会有可能出现 16 位的数据的移位，就是说发送的数据和接收 到的数据会有 16 位的错位，下表演示 GTX 发送数据和接收数据移位的情况：   |  |  |  |  | | --- | --- | --- | --- | | **GTX** 发送的数据 | | **GTX** 接收的数据 | | | 数据 1 | 11111111 | 数据 1 | 11112222 | | 数据 2 | 22222222 | 数据 2 | 22223333 | | 数据 3 | 33333333 | 数据 3 | 33334444 | | 数据 4 | 44444444 | 数据 4 | 44445555 | | |
| 黑金 FPGA 开发板 | **15** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| |  |  |  |  | | --- | --- | --- | --- | | 数据 5 | 55555555 | 数据 5 | 5555..... | | ..... | ..... | ..... | ..... |   因为我们在 GTX 发送同步信号和无用数据的时候加入了 K 码控制字，并且设置 gt\_tx\_ctrl 信 号为 0001, 如果出现 16 位数据移位的情况， 接收到的同步信号和无用数据时， K 码控制字也会跟 着移位， gt\_tx\_ctrl 的信号就会变成 0100。所以我们在程序可以通过判断 gt\_tx\_ctrl 信号的值来判 断接收到的 GTP 数据是否移位， 如果接收到的 gt\_tx\_ctrl 为 0001，跟我们发送的时候一样， 说明 数据没有移位；如果接收到的 gt\_tx\_ctrl 为 0100，接收到的数据移位，需要重新组合，在  word\_align.v 模块里完成。   |  | | --- | |  |   **4**）**GTX** 数据解析模块 **packet\_rec.v**  因为接收到的 32 位数据中只有一部分是有效的数据，其它的是同步包头，序列数据，控制数 据和 Checksum，在 packet\_rec.v 模块里会计算数据的 checksum, 然后跟接收到的 checksum 值进行 对比，如果不正确， 会产生数据错误信号。  程序的一个功能是检测 GTX 数据中的同步包头信号（数据为 ff\_00\_02\_bc）， 如果接收到同步 包头信号，开始一包数据的接收。   |  | | --- | |  |   程序的另一个功能是判断统计数据的 checksum 并和接收到的 checksum 判断。 | |
| 黑金 FPGA 开发板 | **16** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| |  | | --- | |  |   **5**） 管脚约束  这里的管脚约束是在 gtx IP 的 example 工程中的 gtx\_exdes.xdc 文件中修改而来，比如 GTX 的 参考时钟输入管脚， 这里需要跟开发板上的管脚对应。 另外还需要添加 SFP 光模块的发送控制管 脚的定义如    **4** 光纤数据传输测试  编译项目通过后我们就可以开始光纤数据传输的实验了。光模块插入到 AX7Z035 开发板的 OPT1~OPT2 接口上， 再连接光纤。AX7Z035 硬件连接后如下图所示： | |
| 黑金 FPGA 开发板 | **17** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 再下载 bit 文件到 FPGA,    点击 Run Trigger for this ILA core 按钮。 | |
| 黑金 FPGA 开发板 | **18** / **20** |

|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 我们就可以在 ILA 里看到接收的测试数据了。    如果有接收的数据包错误， error\_packet\_cnt\_o 的值会加 1。  这里为止 GTX 光纤数据传输例程就介绍完了。如果用户需要对工程中的 gtx IP 重新配置的 话，如果只是修改工程里的 gtx IP 配置的话，编译的时候会报错。用户需要在修改工程中的 gtx IP 配置的同时，还是需要重新生成 gtx IP 的 example 文件，然后用 example 工程中生成的文件去替 换工程中的以下文件。 | |
| 黑金 FPGA 开发板 | **19** / **20** |



|  |  |
| --- | --- |
| **www.alinx.com** |  |
| 黑金 FPGA 开发板 | **20** / **20** |