PH-ESE-BE

GBT-FPGA User Guide

Version: 1.01

2014.04.15

DRAFT

The GBT-FPGA team: S. Baron (Sophie.Baron@cern.ch) and M. Barros Marin (manoel.barros.marin@cern.ch)

GBT-FPGA support: GBT-FPGA-support@cern.ch

Web site: *https://espace.cern.ch/GBT-Project/GBT-FPGA*

SVN repository: *https://svn.cern.ch/reps/ph-ese/be/gbt_fpga*

Document History

- **v1.01, 2014.04.15:** Minor corrections.
- **v1.00, 2014.04.11:** First draft in the document history.

The GBT-FPGA team

- **Sophie Baron** (Sophie.Baron@cern.ch)
- **Manoel Barros Marin** (manoel.barros.marin@cern.ch)
- **GBT-FPGA-support** (GBT-FPGA-support@cern.ch)

Table of Contents

1. General Introduction

1.1. The GBT and the Versatile Link Projects

The GBTx is a radiation tolerant chip that can be used to implement multipurpose high speed (3.2-4.48 Gbps user bandwidth) bidirectional optical links for high-energy physics experiments.

Logically the link provides three "distinct" data paths for Timing and Trigger Control (TTC), Data Acquisition (DAQ) and Slow Control (SC) information. In practice, the three logical paths do not need to be physically separated and are merged on a single optical link as indicated in Figure 1. The aim of such architecture is to allow a single bidirectional link to be used simultaneously for data readout, trigger data, timing control distribution, and experiment slow control and monitoring. This link establishes a point-to-point, optical, bidirectional (two fibres), constant latency connection that can function with very high reliability in the harsh radiation environment typical of high energy physics experiments at LHC.

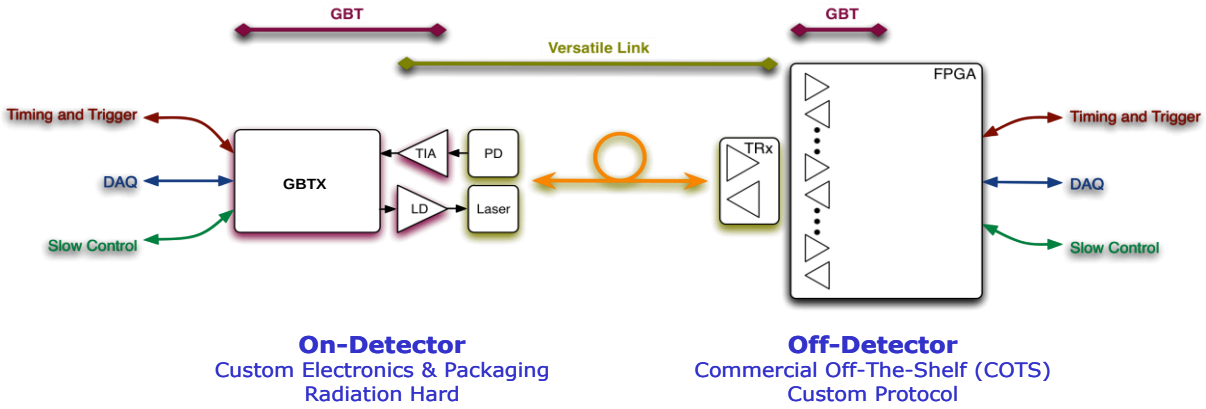


Figure 1: Link architecture with the GBT chip set and the Versatile Link opto-components

The development of the proposed link is conceptually divided into two distinct but complementary parts: the GBT link chips and the Versatile link opto components. The versatile link selects and qualifies appropriate fibres and opto-electronic components for use in radiation. The GBT develops and qualifies the required radiation hard ASICs.

The link is implemented by a combination of custom developed and Commercial-Off-The-Shelf (COTS) components. In the counting room the receiver and transmitters are implemented using COTS components and FPGA's. Embedded in the experiments, the receivers and transmitters are implemented by the GBT chipset and the Versatile Link optoelectronic components. This architecture clearly distinguishes between the counting room and front-end electronics because of the very different radiation environments. The on-detector front-end electronics works in a hostile radiation environment requiring custom made components. The counting room components operate in a radiation free environment and can be implemented by COTS components. The use of COTS components in the counting house allows this part of the link to take full advantage of the latest commercial technologies and components (e.g. FPGA) enabling efficient data concentration and data processing from many front-end sources to be implemented in very compact and cost efficient trigger and DAQ interface systems.

The firmware required to allow these FPGAs to communicate with the GBTx chipset over the versatile link is handled by the GBT-FPGA project.

1.2. The GBT-FPGA Project Mandate

Initiated in 2009 to emulate the GBTx serial link and test the first GBTx prototypes, the GBT-FPGA project developed first to provide the users with a basic "starter kit" allowing them to get used to the GBTx protocol. As the features of the GBTx ASIC inflated during design phase together with the users' requirements, the GBT-FPGA project followed naturally and grew up as well.

This GBT-FPGA core is now a full library, targeting FPGAs from ALTERA and XILINX, allowing the implementation of one or several GBT links of 2 different types: "**Standard**" or "**Latency-Optimized**" (providing low, fixed and deterministic latency either on Tx, Rx or on both). These links can be also configured to provide any encoding mode offered by the GBTx: the "**GBT-Frame**" mode (Reed-Solomon based), the "**8b10b**" mode or the "**Wide-Bus**" mode (no encoding).

The GBT-FPGA core is freely available from SVN, and can be instantiated on Back-End FPGAs, but it can also be used as a GBTx emulator for the serial link. As such, the core is fully configurable in any of the above described options.

For obvious reasons, the GBT-FPGA core will not offer firmware versions for each FPGA type on the market. It targets the main vendors and the main series. The design effort on new series is foreseen to stop during 2014. The evolution of the core to follow-up the technology will thus depend on users' contributions.

2. The GBT-FPGA Core

2.1. Introduction

2.1.1. GBT-FPGA Core Overview

In order to facilitate the in-system implementation and the user support of the GBT-FPGA, the different components of the core are integrated in a single module called "**GBT Bank**" (see Figure 2 and Figure 3).

Figure 2: GBT Bank module

Most of these components are common for the different platforms. The GBT Bank may include several "**GBT Links**" (the maximum number is vendor dependent). Each GBT Link is composed by a GBT Tx, a GBT Rx (both together will be referred to as "GBT Logic") and a Multi-Gigabit Transceiver (MGT). The clocking resources are external to the GBT Bank so the user can connect the different clocks as desired.

```
gbtBank_1: entity work.gbt_bank
   generic map (
      GBT_BANK_ID                => 1)
   port map (
      CLKS_I                     => to_gbtBank1_clks,
      CLKS_O                     => from_gbtBank1_clks,
      ---------------------------
      GBT_TX_I                   => to_gbtBank1_gbtTx,
      GBT_TX_O                   => from_gbtBank1_gbtTx,
      ---------------------------
      MGT_I                      => to_gbtBank1_mgt,
      MGT_O                      => from_gbtBank1_mgt,
      ---------------------------
      GBT_RX_I                   => to_gbtBank1_gbtRx,
      GBT_RX_O                   => from_gbtBank1_gbtRx
   );
```

Figure 3: GBT Bank VHDL instantiation

The number of GBT Links of the GBT Bank as well as the three encoding schemes proposed by the GBTx ASIC ("GBT-Frame" (Reed-Solomon), "Wide-Bus" and "8b10b") and the two types of optimization ("Standard" and "Latency-Optimized"), may be selected at implementation time through a single file (**GBT User Setup File**).

2.1.2. Standard VS Latency-Optimized

Trigger related electronic systems in High Energy Physics (HEP) experiments, such as Timing Trigger and Control (TTC), require a fixed, low and deterministic latency in the transmission of the clock and data to ensure correct event building. On the other hand, other electronic systems that are not time critical, such as Data Acquisition (DAQ), do not need to comply with this requirement. The GBT-FPGA project provides two types of implementation for the transmitter and the receiver: the "**Standard**" version, targeted for non-time critical applications and the "**Latency-Optimized**" version, ensuring a fixed, low and deterministic latency of the clock and data (at the cost of a more complex implementation).

If latency is not an issue, the **Standard version** is strongly recommended.

Table 1: Standard VS Latency-Optimized

	Standard	Latency-Optimized
Latency	Non Fixed, Higher, Non Deterministic	Fixed, Low, Deterministic
Logic Resources Utilization	Low	Low
Clocking Resources Utilization	Low	High
Clock Domain Crossing	Don't Care	Critical
Implementation	Simple	Complex

2.1.3. Data Frame & Encodings

As previously mentioned, the GBT-FPGA supports the three available encoding schemes proposed by the GBTx.

The "**GBT-Frame**", shown in Figure 4, adopts the Reed-Solomon that can correct bursts of bit errors caused by Single Event Upsets (SEU). This encoding scheme can be used for Data Acquisition (DAQ), Timing Trigger & Control (TTC) and Experiment Control (EC).

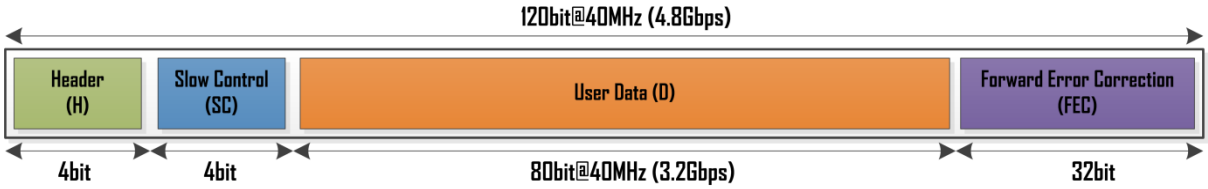


Figure 4: GBT-Frame encoding frame

The "**8b10b**", shown in Figure 5, provides 4bit more than the GBT-Frame to be used by the user at the cost of no error correction and limited error detection capability. This encoding scheme can **only** be used for DAQ and EC **in the uplink direction** (Front-End to Back-End).

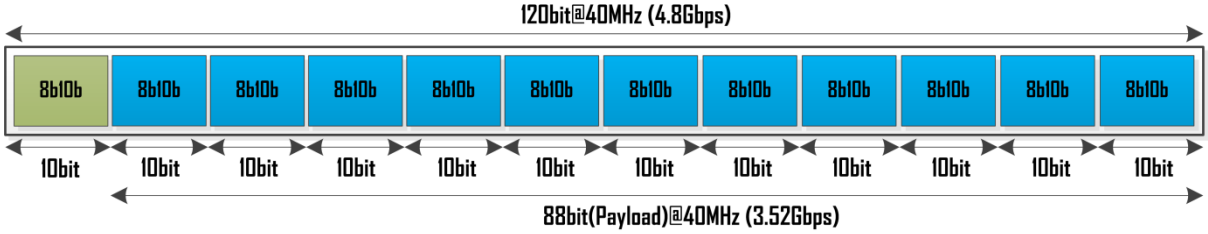


Figure 5: 8b10b encoding frame

For the "**Wide-Bus**", shown in Figure 6, the FEC field is fully replaced by user data at the cost of no error detection nor correction capability. This encoding scheme can **only** be used for DAQ and EC **in the uplink direction**.

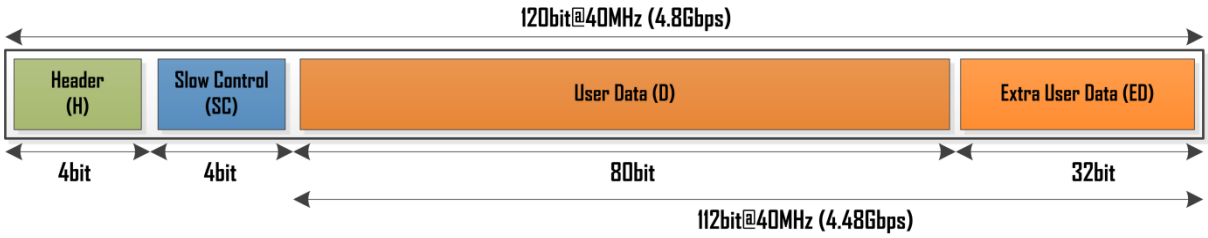


Figure 6: Wide-Bus encoding frame

2.1.4. Multiple Platforms

The GBT-FPGA Core already available for different FPGAs of the two main vendors, Xilinx and Altera.

- Xilinx: Virtex 6, Kintex 7, Virtex 7
- Altera: Cyclone V, Stratix V

The port of the GBT-FPGA Core for other FPGA devices and vendors is under study.

2.1.5. Typical Implementation Resources

The logic resources utilization of one GBT Bank instantiating one GBT Link is shown in Table 2 and Table 3 for Xilinx Kintex 7 and Altera Cyclone V respectively.

Table 2: Logic resources utilization in Xilinx Kintex 7

Xilinx (Kintex7: XC7K325T)		
Resources	**Standard (%)**	**Latency-Optimized (%)**
LUT	2658 (1.30)	2776 (1.36)
FD_LD	817 (0.20)	969 (0.24)
BMEM	10 (1.12)	0 (0.00)
GTX	1 (6.25)	1 (6.25)

A table with clocking resources utilization will be included in a next revision of this user guide

Table 3: Logic resources utilization in Altera Cyclone V

Altera (Cyclone V: 5CGTFD9E5F35C7N)		
Resources	**Standard (%)**	**Latency-Optimized (%)**
ALM	1674 (1.47)	1827 (1.61)
Register	1100 (0.24)	1475 (0.32)
Mem (M10K)	10 (0.81)	2 (0.16)
GT	1 (8.33)	1 (8.33)

The logic resources utilization per GBT Link is very low compared with the amount of resources of the latest FPGAs.

When using the Standard version of the GBT-FPGA core, the clocking resources can be shared among the different GBT Banks. On the other hand, **when using the Latency-Optimized version, clocking resources become a critical factor** due to the high number of clock domains in multilink implementations.

2.1.6. Usual Latency

A preliminary latency study of the Latency-Optimized version has been carried out, obtaining the values showed in Table 4.

Table 4: Preliminary latency measurement results

Latency of the Latency-Optimized GBT Link (Virtex 6)			
	GBT Logic	**MGT (GTX)**	**Total**
GBT Link Tx	29.2ns	18.7ns	47.9ns
GBT Link Rx	54.2ns	28.2ns	82.4ns

Further latency measurements are foreseen, including temperature tests in a climatic chamber.

2.1.7. Links & Other Documents

To be completed

2.2. Entities

2.2.1. GBT Bank

The GBT Bank is the top module of the GBT-FPGA Core. Each GBT Bank may include several GBT Links (up to four in Xilinx FPGAs or up to three in Altera FPGAs). This limitation in the number of GBT Links per GBT Bank is a requirement in order to main the GBT Banks independent in terms of logic and clocking resources. It depends on FPGA architectures. If necessary, the user may instantiate several GBT Banks in parallel. The maximum number of GBT Banks is device dependant.

A simplified block diagram of a GBT Bank instantiating two GBT Links is shown in Figure 7.

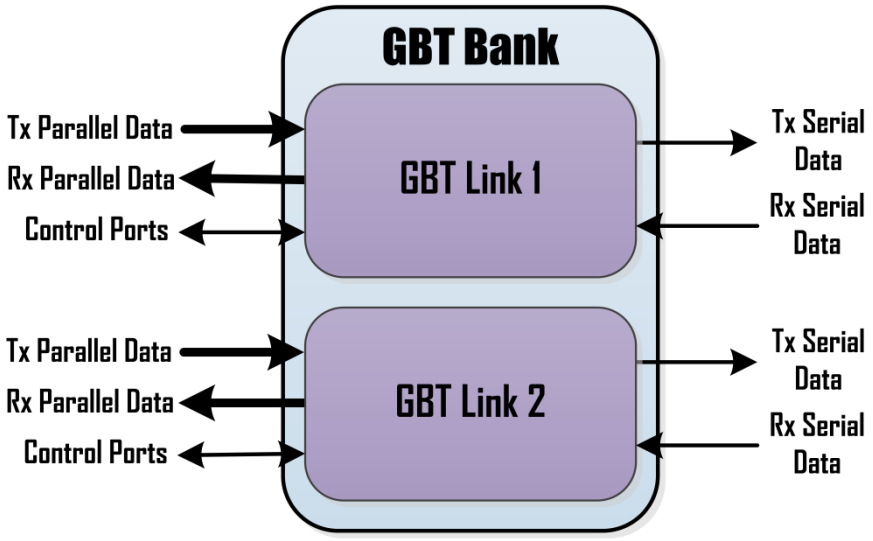


Figure 7: GBT Bank simplified block diagram

A detailed block diagram of the GBT Bank is shown in APPENDIX B:.

2.2.2. GBT Link

The GBT Link is the actual channel of the link. It is composed by a GBT Tx (that scrambles and encodes the transmitted parallel data), a Multi-Gigabit Transceiver (MGT) (that serializes, transmits, receives and de-serializes the data) and a GBT Rx (that aligns, decodes and descrambles the incoming data stream).

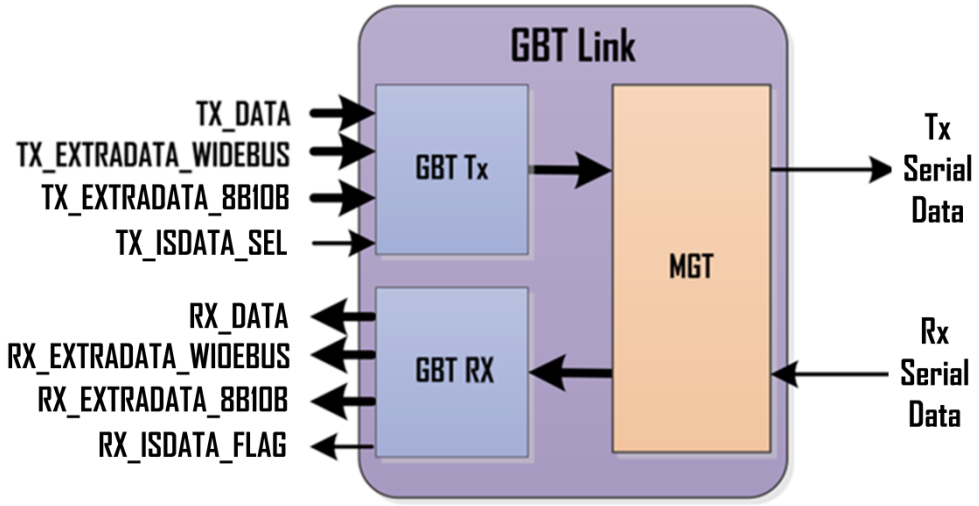


Figure 8: GBT Link simplified block diagram

2.2.3. GBT Tx

A simplified block diagram of the GBT Tx highlighting its main components is shown in Figure 9.

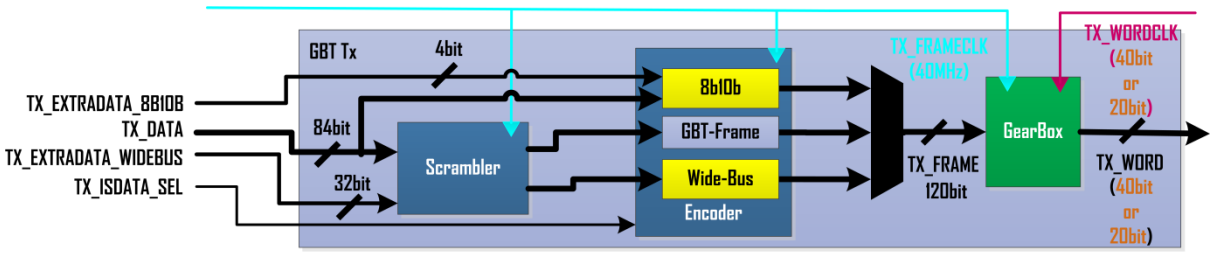


Figure 9: GBT Tx simplified block diagram

2.2.4. GBT Rx

A simplified block diagram of the GBT Rx highlighting its main components is shown in Figure 10.

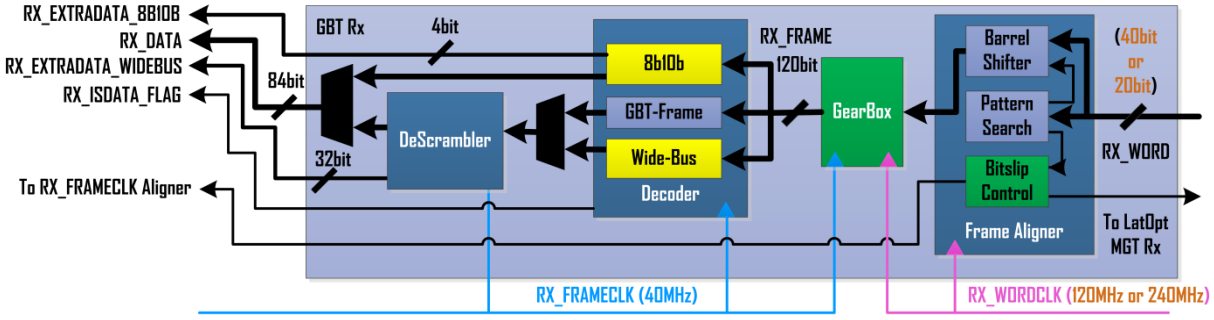


Figure 10: GBT Rx simplified block diagram

2.2.5. Multi-Gigabit Transceiver (MGT)

The Multi-Gigabit Transceiver (MGT) is a vendor and device specific Serializer/Deserializer (SerDes). In order to facilitate the integration with the GBT Logic, the MGT is wrapped with a HDL file (depicted by Figure 11) featuring a common interface with the GBT Logic. Besides this common interface, the MGT also has specific control ports. These ports are grouped in two "records" (a type in VHDL similar to "structures" in C programming), where the inputs named "MGT_I" and the outputs named "MGT_O" are common for all MGTs from the different FPGA vendors. These records are forwarded out from the GBT Bank thus allowing custom configuration of the MGT.

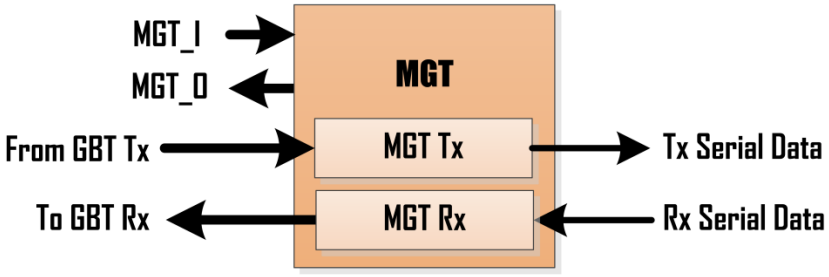


Figure 11: MGT simplified block diagram

2.3. GBT-FPGA Project Architecture

2.3.1. Common VS Device Specific Source Files

The concept behind the design of the GBT-FPGA Core is to provide a common firmware for the most common FPGAs. As a consequence, most of the source files of the GBT-FPGA Core are common to all FPGAs. These files are gathered in the same folder and classified in subfolders per functionality (gbt_rx, gbt_tx, mgt, etc.). The root folder of the common files of the GBT Bank is:

"... \gbt_bank\core_sources\..."

The rest of the files, that are vendor specific, are gathered per FPGA type, in separated folders. These files are also classified in subfolders following the same logic. They can be found in the folder:

"... \gbt_bank\<vendor>_<device>\..." (e.g. "... \gbt_bank\xilinx_v6\...").

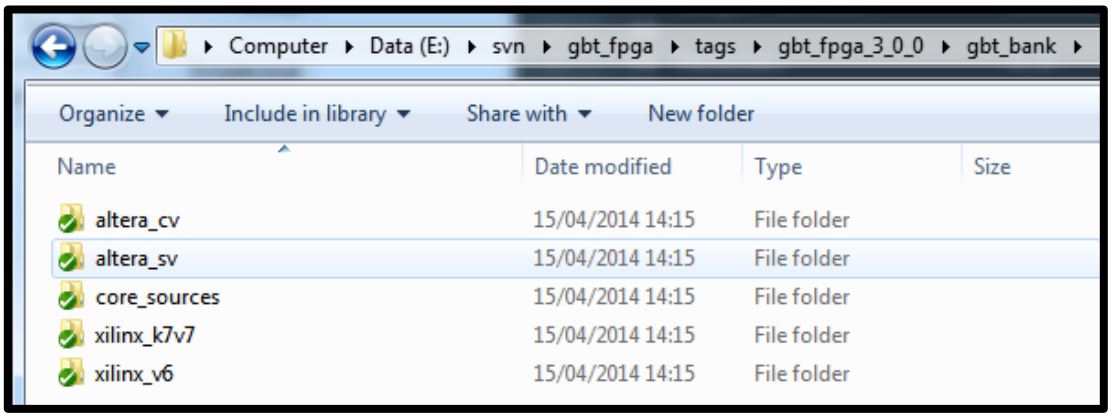


Figure 12: Common (in "core_source") and vendor specific source files folders

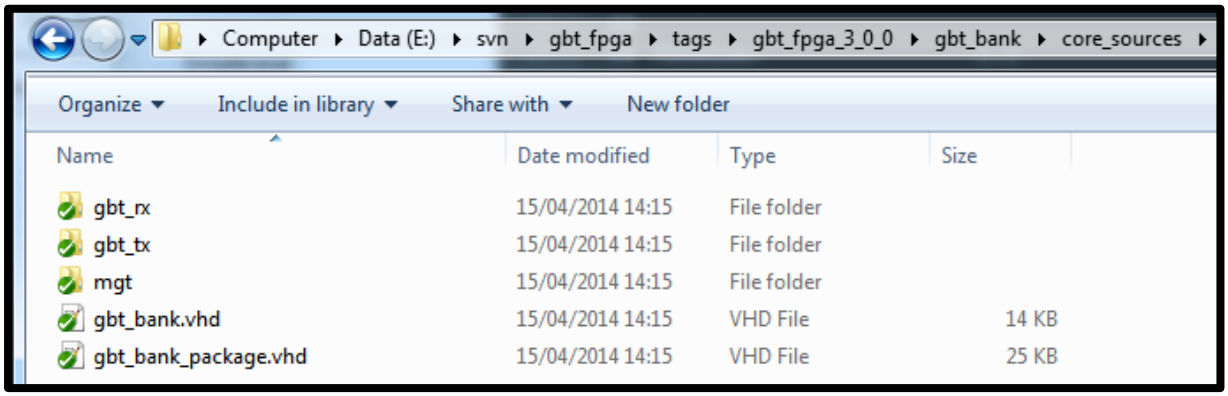


Figure 13: Classification of files both in "core_sources" and in vendor specific folders

2.3.2. The GBT-FPGA SVN Repository

The source files, documentation and TCL scripts of the GBT-FPGA project are available in a CERN subversion (SVN) repository and supported by the members of the GBT-FPGA team.

https://svn.cern.ch/reps/ph-ese/be/gbt_fpga/tags

To access the SVN repository, the user may use any of the numerous open source SVN clients such as TortoiseSVN, shown in Figure 14. The official releases are stored in the "**tags**" folder.

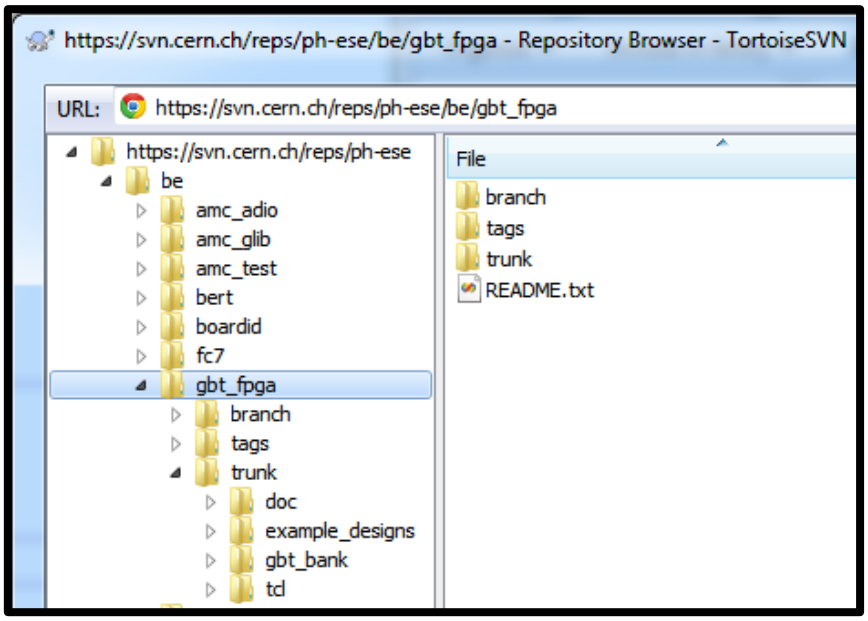


Figure 14: GBT-FPGA SVN repository access through TortoiseSVN

One release (including all the FPGA types and reference designs) requires about 300 Mbytes of disk space.

2.4. Implementation

2.4.1. Getting Started

2.4.1.a. Check the Structure of the Folders

The first step after having downloaded the source files, documentation and TCL scripts of the GBT-FPGA project from the SVN repository as well as having read this user guide should be to go to the folder where the GBT-FPGA Core files were downloaded and check how the different source files are organized.

2.4.1.b. Open One of the GBT-FPGA Example Design Projects

The next step should be to open one of the GBT-FPGA example design projects. The procedure for opening a GBT-FPGA example design project is explained in section 3.

2.4.1.c. Check the Source Files

Once the GBT-FPGA example design project is open (in Altera projects it is also recommended to run the synthesis in order to generate the hierarchy of the source files), check the hierarchy and the content of the source files (which are extensively commented). The most important files from the user's point of view are explained next:

- The top level of the GBT Bank "**gbt_bank.vhd**" and its related package "**gbt_bank_package.vhd**". These two files show the external connectivity of the GBT Bank that is common for all FPGAs. These files can be found in the folder:

  "*... \gbt_bank \core_sources *"

  Note that these two files cannot be modified by the user because they are part of the GBT-FPGA Core

- The GBT Bank package specific for the targeted FPGA "**<vendor>_<device>_gbt_bank_package.vhd**" (e.g. "xlx_v6_gbt_bank_package.vhd"). This file shows the external connectivity of the MGT. It can be found in the folder:

  "*... \gbt_bank \<vendor>_<device> *"

  Note that this file cannot be modified by the user because it is part of the GBT-FPGA Core

- The main file of the GBT-FPGA example design "**<vendor>_<device>_gbt_example_design.vhd**" (e.g. "xlx_v6_gbt_example_design.vhd") that contains the GBT Bank(s) and the rest of the modules. Through this file, it is possible to understand the interaction and connectivity between the GBT Bank and the other modules that compose the example design. This file can be found in the folder:

  "*... \example_designs \<vendor>_<device> \core_sources *"

  (e.g." *... \example_designs \xilinx_v6 \core_sources *")

- The top level of the GBT-FPGA example design "**<hardware_platform>_gbt_example_design.vhd**" (e.g. "glib_gbt_example_design.vhd"). This file is basically a wrapper for interfacing the previously mentioned file ("<vendor>_<device>_gbt_example_design.vhd") with the FPGA-based card. This file is interesting because shows the connectivity between the example design and the hardware components of the FPGA-based card. This file can be found in the folder:

  "*... \example_designs \<vendor>_<device> \<hardware_platform> *"

  (e.g. "... \example_design \xilinx_v6 \glib \")

2.4.1.d. Implement and Run one of the GBT-FPGA Example Design Projects

To be completed

2.4.2. The GBT User Setup File

The user setup file for all the GBT-Banks is the file named: "**<vendor>_<device>_gbt_banks_user_setup.vhd**" (e.g. "xlx_v6_gbt_banks_user_setup.vhd.). This file gathers all the constants defining the setup of all the GBT banks in one file. It can be found in the folder:

"*... \gbt_bank\<vendor>_<device>*"

It is important to mention that this is the only file of the GBT Bank that may be modified by the user

```
--================================================================================================--
--################################   Package Declaration   ########################################--
--================================================================================================--

package gbt_banks_user_setup is

   --================================ GBT Banks parameters ==============================--

   --======================--
   -- Number of GBT Banks --
   --======================--

   -- Comment:   * On Stratix V it is possible to implement up to THREE links per GBT Bank.
   --
   --            * If more links than allowed per GBT Bank are needed, then it is
   --              necessary to instantiate more GBT Banks.

   constant NUM_GBT_BANKS                     : integer := 2;

   --==================--
   -- GBT Banks setup --
   --==================--

   -- Comment: For more information about the record "GBT_BANKS_USER_SETUP" see the file:
   --          "../gbt_bank/altera_sv/alt_sv_gbt_link_package.vhd"

   constant GBT_BANKS_USER_SETUP : gbt_bank_user_setup_R_A(1 to NUM_GBT_BANKS) := (

      -- GBT Bank 1:
      --------------

      1 => (NUM_LINKS                         => 1,                   -- Comment: * 1 to 3
            ----------------------------------
            TX_OPTIMIZATION                   => STANDARD,            --          * (STANDARD or LATENCY_OPTIMIZED)
            RX_OPTIMIZATION                   => LATENCY_OPTIMIZED,   --          * (STANDARD or LATENCY_OPTIMIZED)
            ----------------------------------
            TX_ENCODING                       => GBT_FRAME,           --          * (GBT_FRAME or WIDE_BUS or GBT_8B10B)
            RX_ENCODING                       => GBT_FRAME),          --          * (GBT_FRAME or WIDE_BUS or GBT_8B10B)

      -- GBT Bank 2:
      --------------

      2 => (NUM_LINKS                         => 3,                   -- Comment: * 1 to 3
            ----------------------------------
            TX_OPTIMIZATION                   => LATENCY_OPTIMIZED,   --          * (STANDARD or LATENCY_OPTIMIZED)
            RX_OPTIMIZATION                   => STANDARD,            --          * (STANDARD or LATENCY_OPTIMIZED)
            ----------------------------------
            TX_ENCODING                       => GBT_FRAME,           --          * (GBT_FRAME or WIDE_BUS or GBT_8B10B)
            RX_ENCODING                       => GBT_FRAME)--,        --          * (GBT_FRAME or WIDE_BUS or GBT_8B10B)

      -- GBT Bank 3:
      --------------

--    3 => (NUM_LINKS                         => 4,                   -- Comment: * 1 to 3
--          ----------------------------------
--          TX_OPTIMIZATION                   => STANDARD,            --          * (STANDARD or LATENCY_OPTIMIZED)
--          RX_OPTIMIZATION                   => STANDARD,            --          * (STANDARD or LATENCY_OPTIMIZED)
--          ----------------------------------
--          TX_ENCODING                       => GBT_FRAME,           --          * (GBT_FRAME or WIDE_BUS or GBT_8B10B)
--          RX_ENCODING                       => WIDE_BUS,            --          * (GBT_FRAME or WIDE_BUS or GBT_8B10B)

   );

   --================================================================================--
end gbt_banks_user_setup;
--================================================================================================--
--################################################################################################--
```

Figure 15: Content of the file "gbt_banks_user_setup"

2.4.3. Adding the GBT-FPGA Core to Your Project

The Different files that compose the GBT Bank may be added to the user project by using TCL scripts. Note that clocking resources and the resets scheme files are not added to the user project by these scripts since they are application dependent. The naming of these scripts has the following format:

"<vendor_<device>_gbt_bank.tcl" (e.g. xlx_v6_gbt_bank.tcl)

They can be found in the folder:

"...\tcl\"

2.4.3.a. Running TCL Scripts for Xilinx FPGAs

This tutorial describes the steps to add the source files of the GBT-FPGA Core targeted to Xilinx Virtex 6.

1. Go to the folder where the TCL scripts are located (see Figure 16)

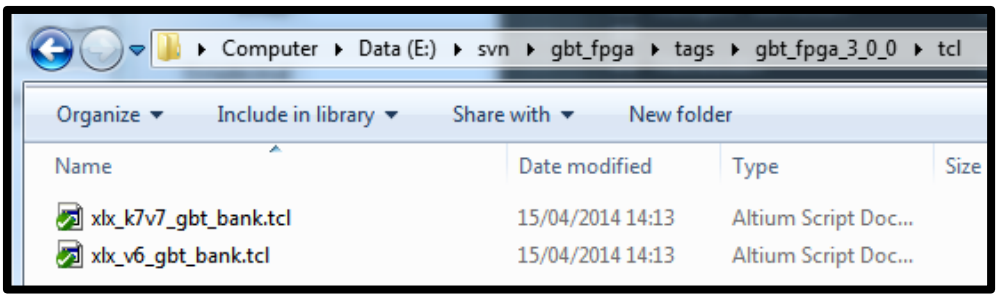


Figure 16: Content of the TCL scripts folder

2. Open the script targeted to the selected FPGA (in this case "xlx_v6_gbt_bank.tcl")

3. Set the variable "SOURCE_PATH" with the absolute path of the root folder of the GBT-FPGA project on your computer (see Figure 17).

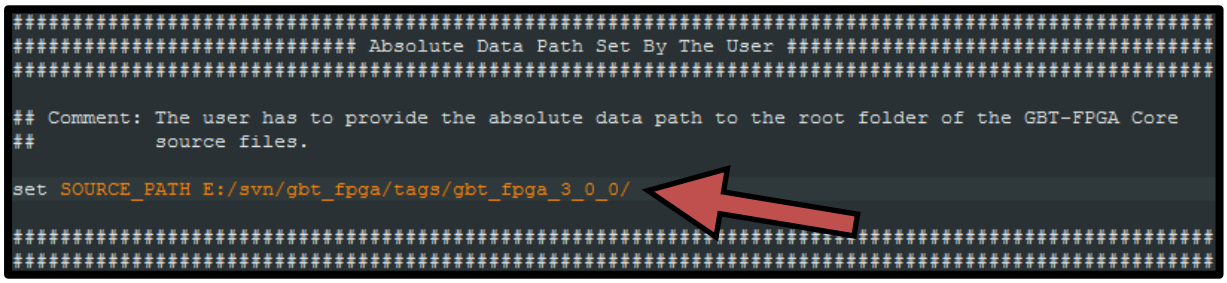


Figure 17: Modifying the absolute data path of the TCL sript file

Save the modified script file and close it.

4. Open ISE (These scripts have not been tested in Vivado).

5. If the TCL console is not visible, go to "View" -> "Panels" and click on "Tcl Console" (See Figure 18).

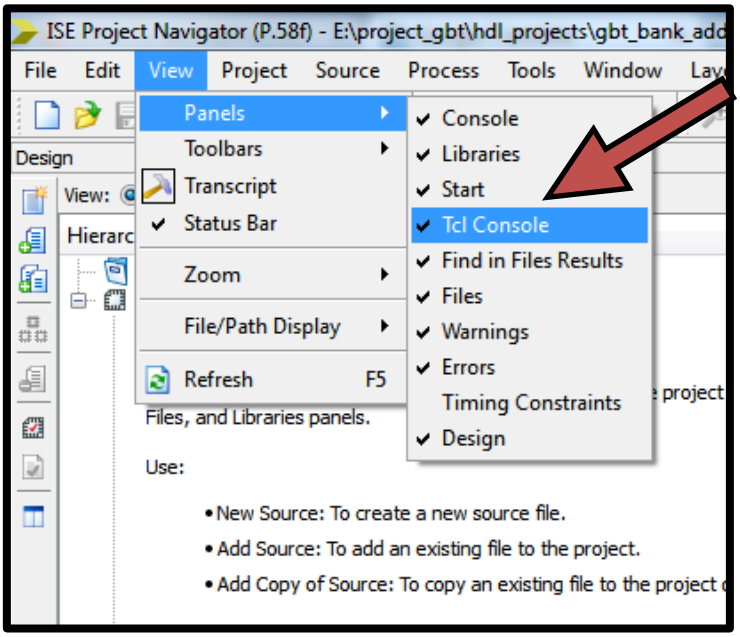


Figure 18: Activating the "Tcl Console" of ISE

6. Type the command for running the TCL script in the Tcl Console (see Figure 19). The command is shown next:

source "<absolute path to the folder of the TCL script >/<vendor>_<device_gbt_bank.tcl>"

Note the **inverted slashes** and the inclusion of **quotes.**

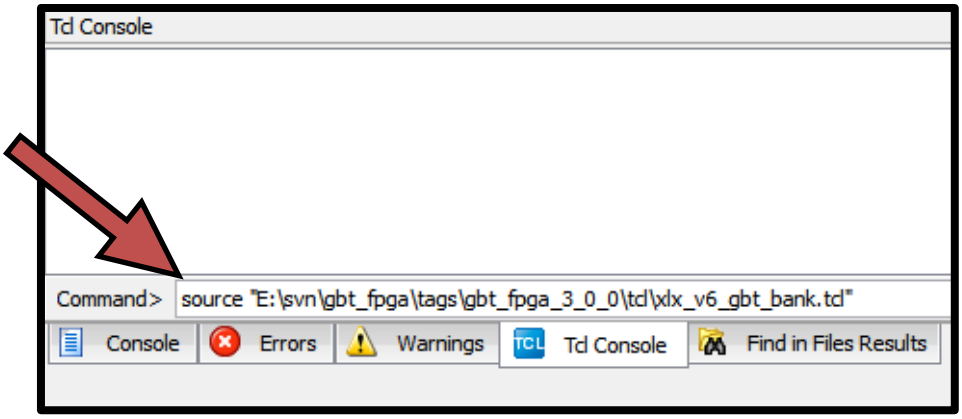


Figure 19: Writing the TCL command for running the TCL script in the Tcl Console of ISE

7. After running the script the source files of the GBT-FPGA core of the selected FPGA appear in the "Hierarchy" window (see Figure 20 and Figure 21).

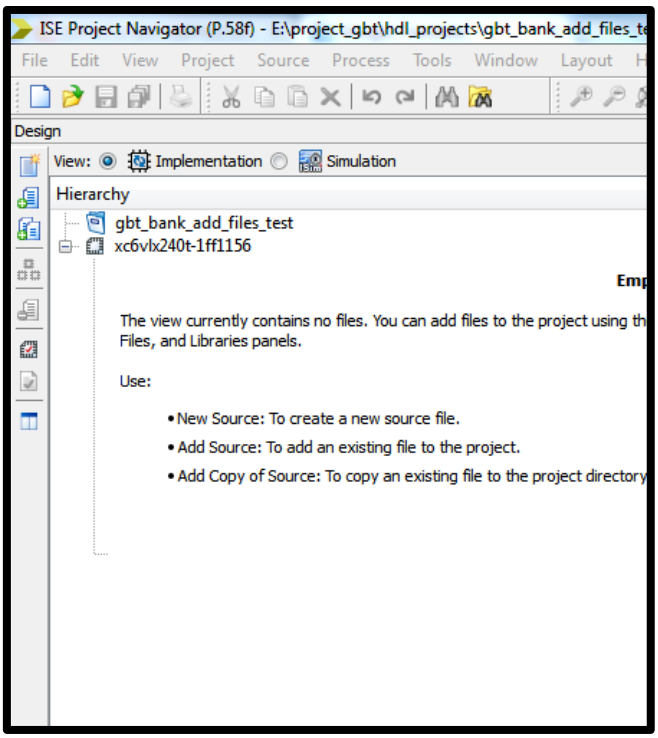


Figure 20: Hierarchy window before running the TCL script.

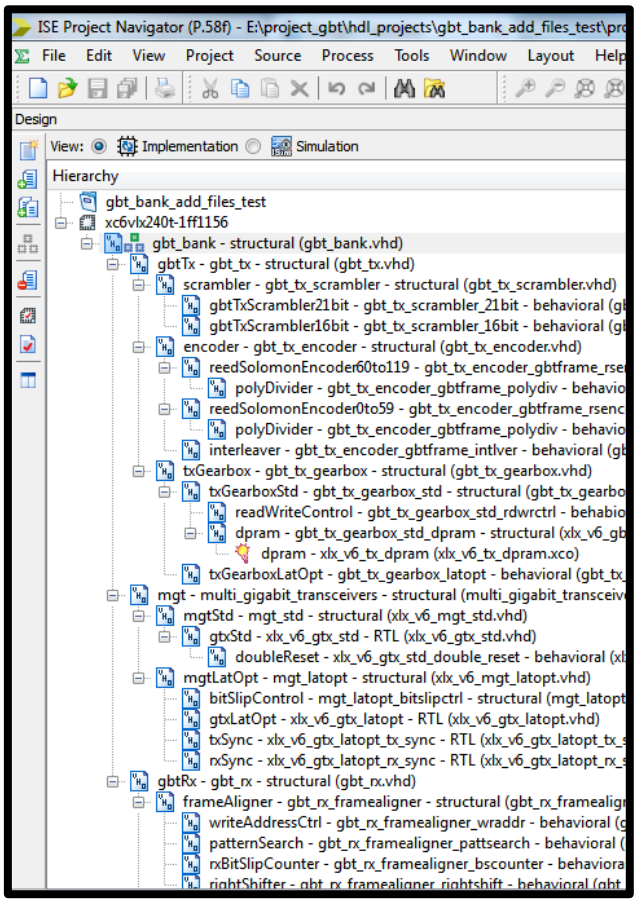


Figure 21: Hierarchy window after running the TCL script.

2.4.3.b.Running TCL Scripts for Altera FPGAs

TCL scripts for Altera FPGAs are not available yet.

2.4.4. The Particularities of the Latency-Optimized Version

To be completed

2.4.5. Multi-Links Instantiation

To be completed

2.4.5.a.Multi-Links Instantiation of Standard Version

To be completed

2.4.5.b.Multi-Links Instantiation of Latency-Optimized Version

To be completed

2.4.6. Timing Closure

To be completed

2.4.6.a.Timing Closure in Xilinx FPGAs

To be completed

2.4.6.b.Timing Closure in Altera FPGAs

To be completed

2.5. Operating the GBT-FPGA Core

2.5.1. Resets Scheme

To be completed

2.5.2. Control

2.5.2.a. Common ports

To be completed

2.5.2.b. Vendor Specific Ports

To be completed

2.5.3. Data

To be completed

3. Example Designs

3.1. Example Design Overview

Besides the GBT-FPGA Core source files, the GBT-FPGA project delivers example designs for some selected FPGA-based cards and the most common FPGA devkits. These example designs follow the same concept of unification as the GBT-FPGA Core, so all of them share the same structure and many of the source files (only vendor specific files are not shared). This scheme facilitates the user support and the port of these example designs to other FPGA-based cards if needed.

Although all these example designs follow the same structure, there are two different types of example designs in terms of number of GBT Links implemented. All of the example designs are single-link except a multi-link example design for the AMC40, an FPGA-based card featuring an Altera Stratix V that has been developed at *Le Centre de Physique des Particules de Marseille* (CPPM).

Table 5: Available Reference Designs per Vendor/device

VENDOR	FPGA type	Reference design	Manufacturer	Available	Configuration
ALTERA	Cyclone V GT	GBTx_SAT board	CERN	Soon	1 GBT bank, 1 GBT link
		Cyclone V GT dekit	ALTERA	Yes	1 GBT bank, 1 GBT link
	Stratix V	AMC40	CPPM – LHCb - ALICE	Yes	2 GBT banks, 4 GBT links (see below)
XILINX	Virtex 6	ML605	Xilinx	Yes	1 GBT bank, 1 GBT link
		GLIB	CERN	Yes	1 GBT bank, 1 GBT link
	Kintex 7	KC705	XILINX	Yes	1 GBT bank, 1 GBT link
		FC7	CERN- CMS	Soon	1 GBT bank, 1 GBT link
	Virtex 7	VC705	XILINX	Yes	1 GBT bank, 1 GBT link

The **single-link example design**, shown in Figure 22, consists of a GBT Bank implementing one GBT Link. The Tx is connected to a pattern generator whilst the Rx is connected to a pattern checker. The serial lanes of the GBT Link may be connected in loopback but also to any other GBT compatible device. Regarding the versions of the GBT-FPGA Core, it is possible to implement the two versions since the example design features the components needed for implementing the Latency-Optimized version and these components are also compatible with the Standard version (just change the option in the GBT User Setup File. Besides the version of the GBT-FPGA Core, the user may also chose the type of encoding (GBT-Frame, Wide-Bus or 8b10b).

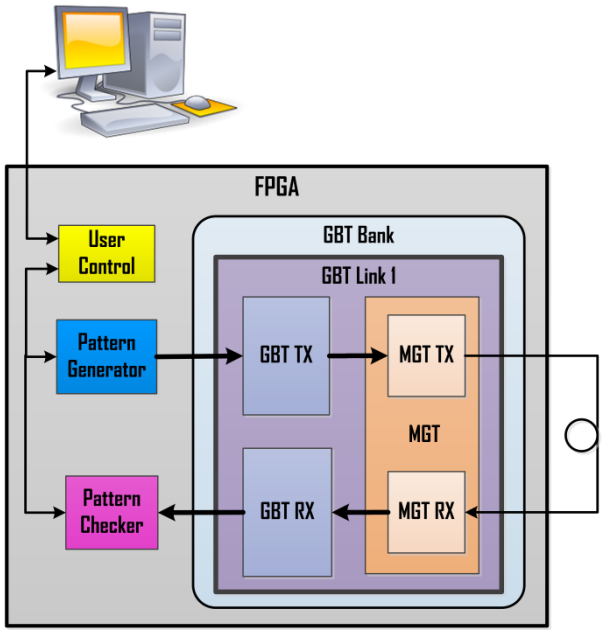


Figure 22: GBT-FPGA single-link example design simplified block diagram

The **multi-link example design** (only available for AMC40 (Stratix V)), shown in Figure 23, consists of four GBT Links implemented into two GBT Banks. The GBT Bank 1 implements one GBT Link configured with Standard Tx and Latency-Optimized Rx whilst the GBT Bank 2 implements three GBT Links configured with Latency-Optimized Txs and Standard Rxs. This example design has been optimized for the previously mentioned configuration, but the GBT Banks will also operate in fully Standard version.

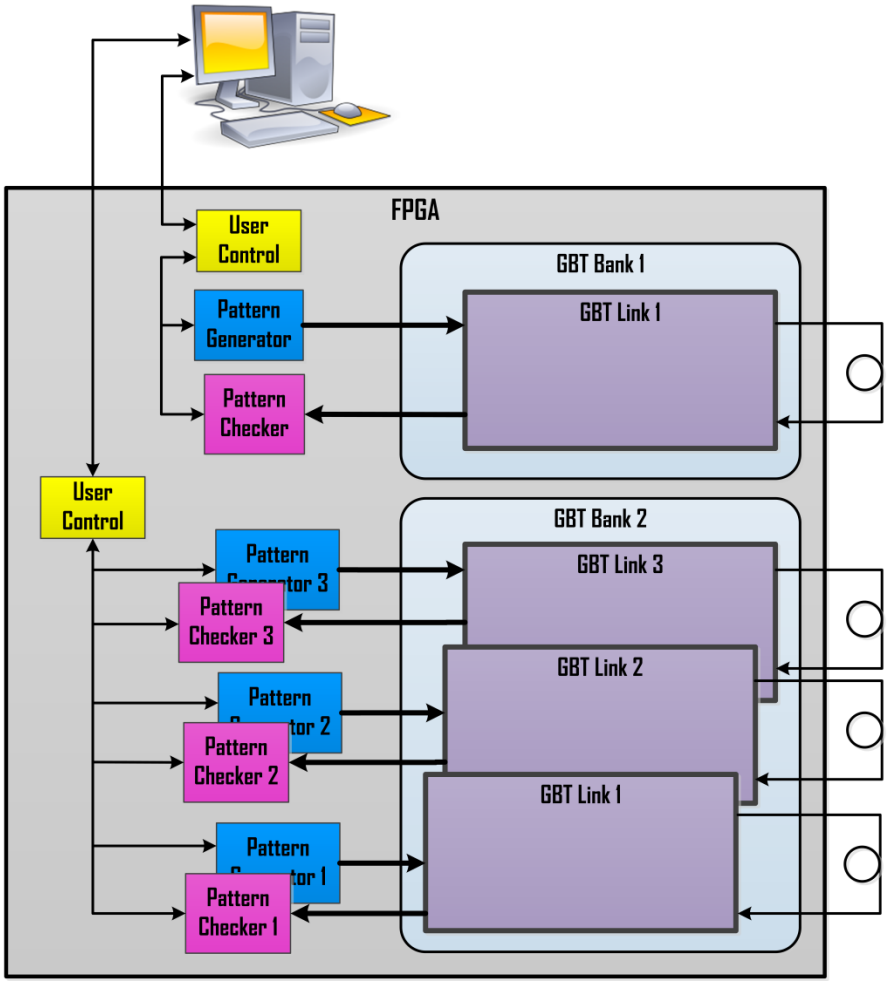


Figure 23: GBT-FPGA multi-link example design simplified block diagram

Like in the single-link example design, the Txs are connected to respective pattern generators whilst the Rxs are connected to respective a pattern checkers. The serial lanes of the GBT Links may be connected in loopback but also to any other GBT compatible devices. Regarding logic and clocking resources, the two GBT Banks are independent, only sharing the MGT reference clock for simplicity reasons. Besides the version of the GBT-FPGA Core, the user may also chose the type of encoding (GBT-Frame, Wide-Bus or 8b10b). It is important to mention that **this example design may be used as a reference for implementing multi-GBT Link systems in other Stratix V FPGA-based cards and even in different model/vendor FPGA-based cards** due to the similarities between the different GBT-FPGA Core implementations.

The control of the example designs is done through a PC using HDL cores and software provided by the FPGA vendors (Xilinx's ChipScope and Altera's In-System Sources and Probes/SignalTap II).

3.2. Implementing the Example Designs

This section describes the steps for implementing example designs for Xilinx FPGAs and for Altera FPGAs.

3.2.1. Implementing the Example Designs in Xilinx FPGAs

The Xilinx EDA tool used by the GBT-FPGA team for developing the GBT-FPGA Core and the associated example designs is **ISE 14.5**. For this reason, the utilization of this EDA tool and version for implementing the example designs is recommended.

In this tutorial, the GBT-FPGA example design implemented is for the GLIB. Nevertheless, the same procedure may be applied to any other GBT-FPGA example design targeted to Xilinx FPGAs-based cards.

8. Open ISE (It may be both, the 32bit or 64bit versions) (see Figure 24).

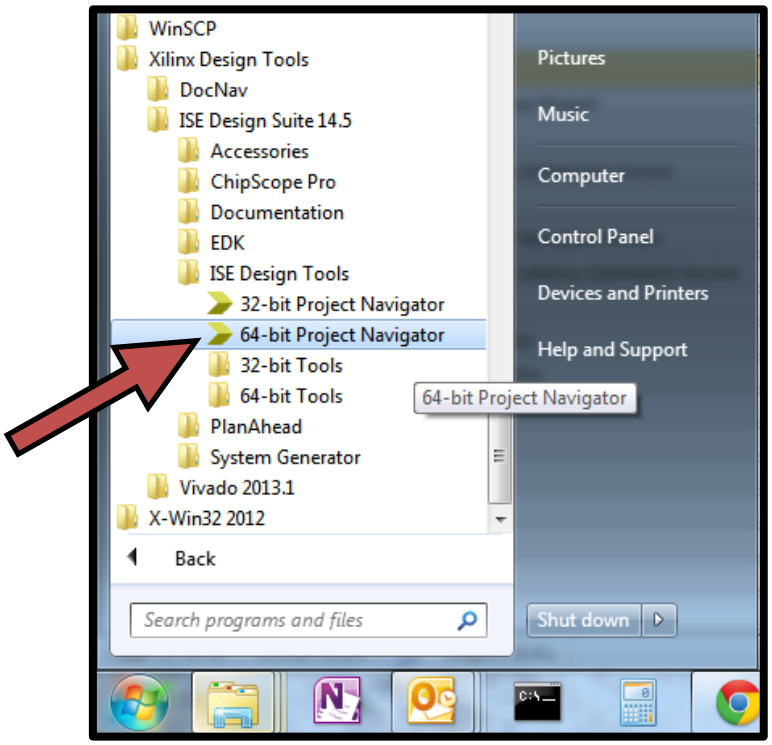


Figure 24: Opening ISE

9. Once ISE is open, click on the button "Open Project..." on the top left corner (see Figure 25).

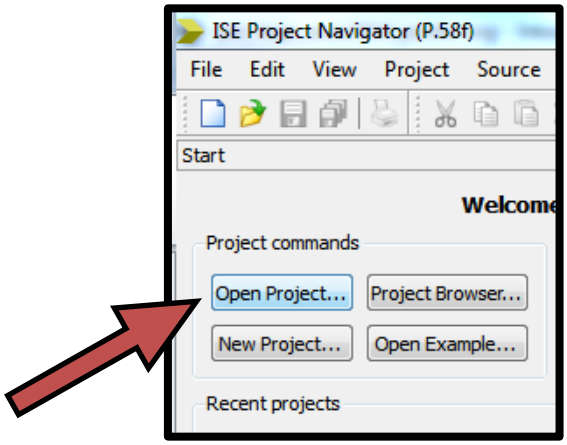


Figure 25: Clicking "Open Project..."

10. Go to the folder where the GBT-FPGA Core source files were downloaded from the GBT-FPGA SVN repository (see Figure 26).

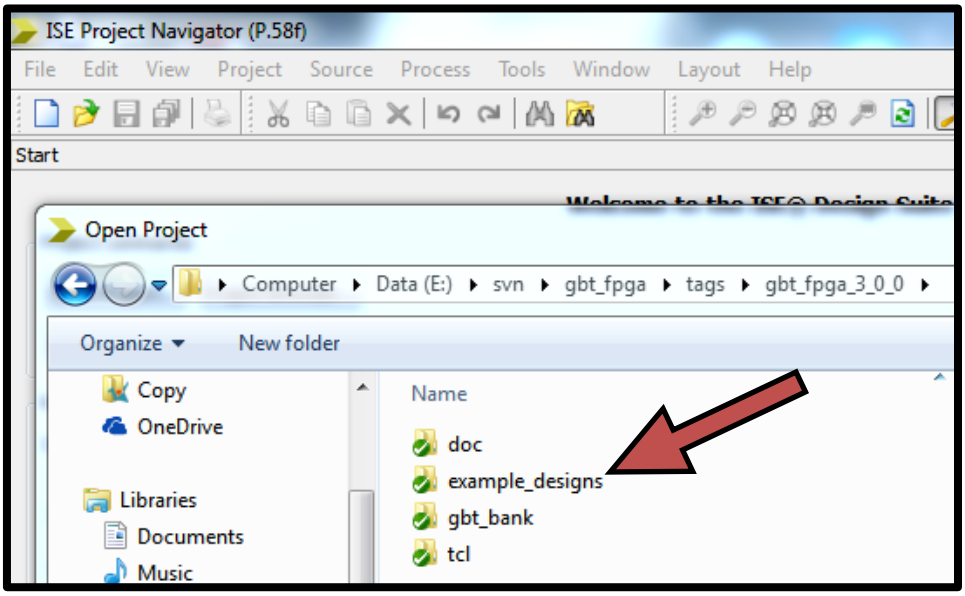


Figure 26: Root folder of the GBT-FPGA source files from the SVN repository

11. Open the folder where the ISE project file of the selected GBT-FPGA example design (in this case "glib_gbt_example_design.xise") is stored and double click on it for opening the project in ISE (if it is the first time that you open the GBT-FPGA example design, the folder only contains the .xise file) (see Figure 27).

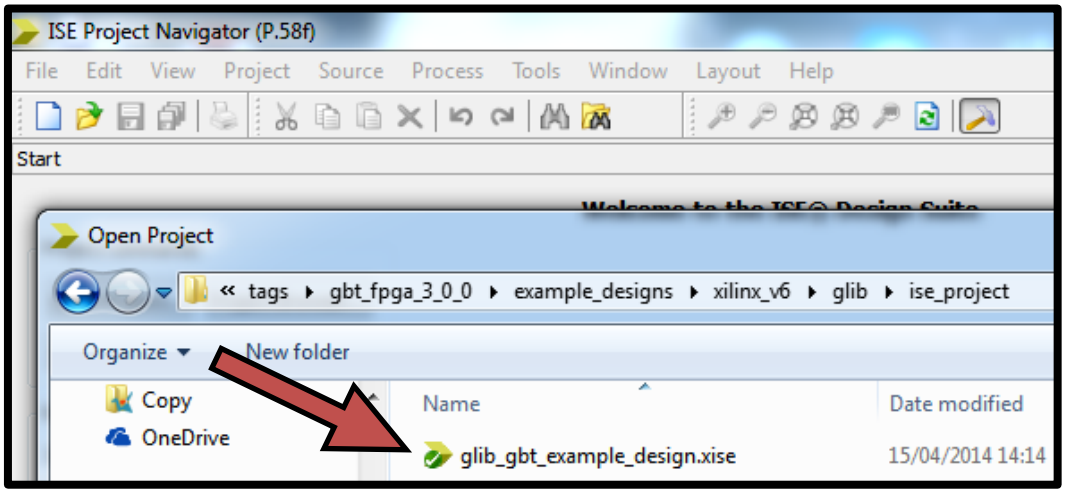


Figure 27: Folder of the ISE project file (.xise)

12. Once the project of the GBT-example design is open in ISE, you can expand the hierarchy of the source files by clicking on the "+" sign near the source file names the in "Hierarchy" pane of the "Design" window (see Figure 28 and Figure 29).

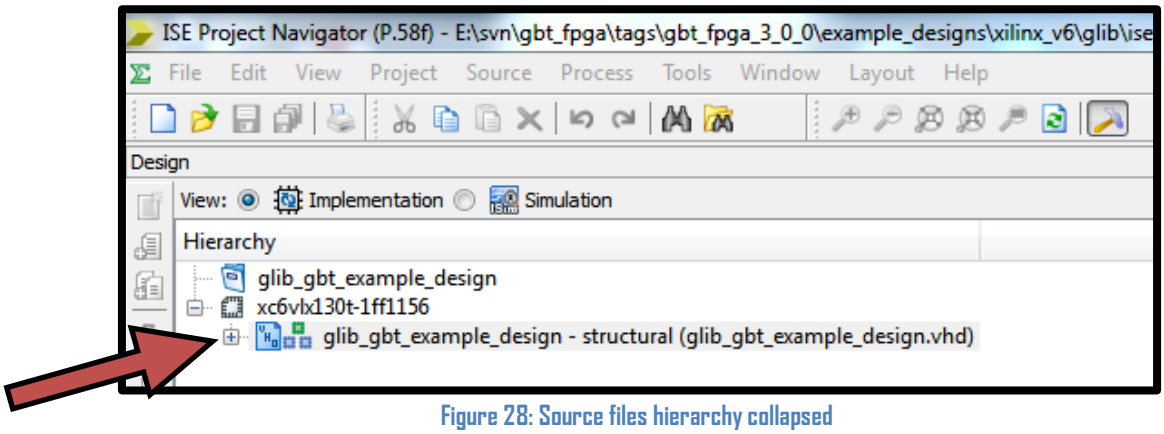


Figure 28: Source files hierarchy collapsed

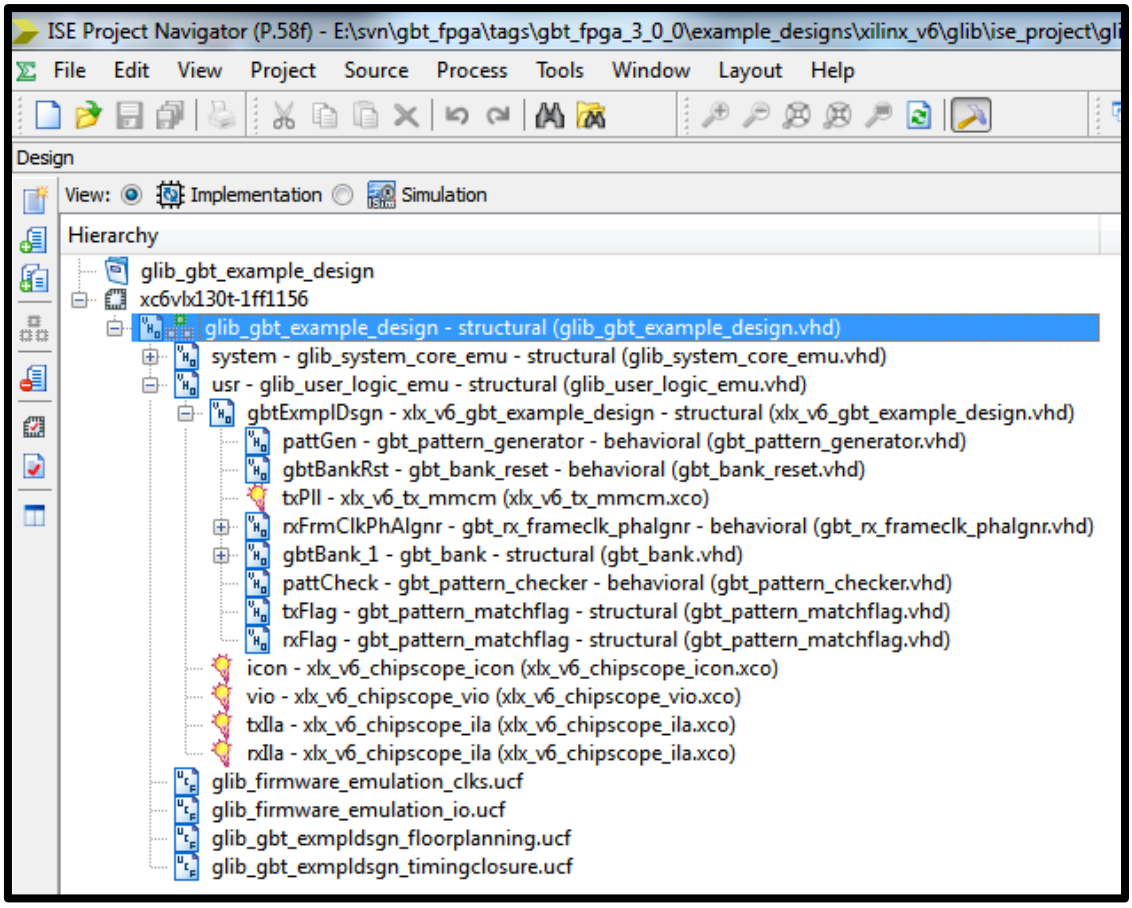


Figure 29: Source files hierarchy partially expanded

13. By clicking on the name of the top level file of the ISE project, the "Synthesize", "Implement Design" and "Generate Program File" processes appear in the "Processes" pane of the "Design" window (see Figure 30 and Figure 31).

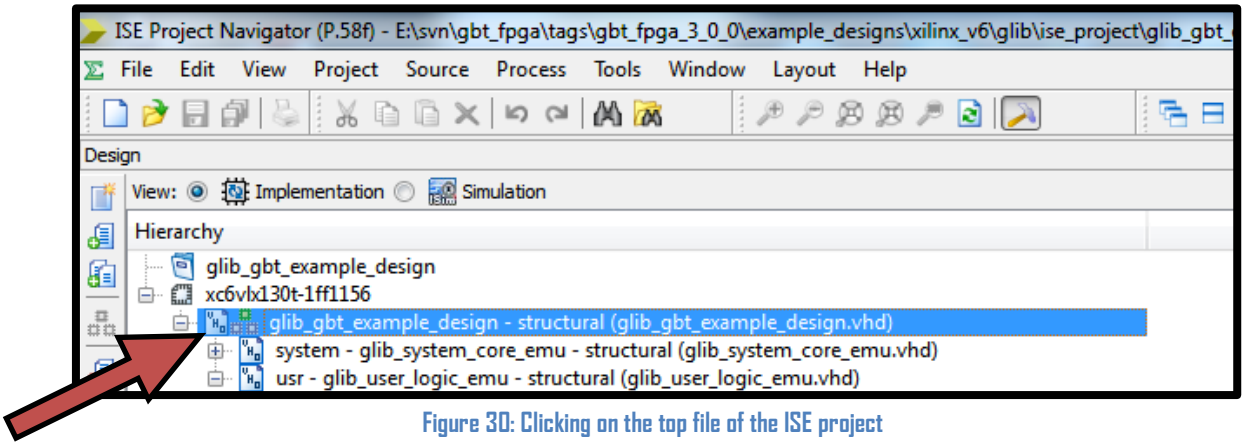


Figure 30: Clicking on the top file of the ISE project

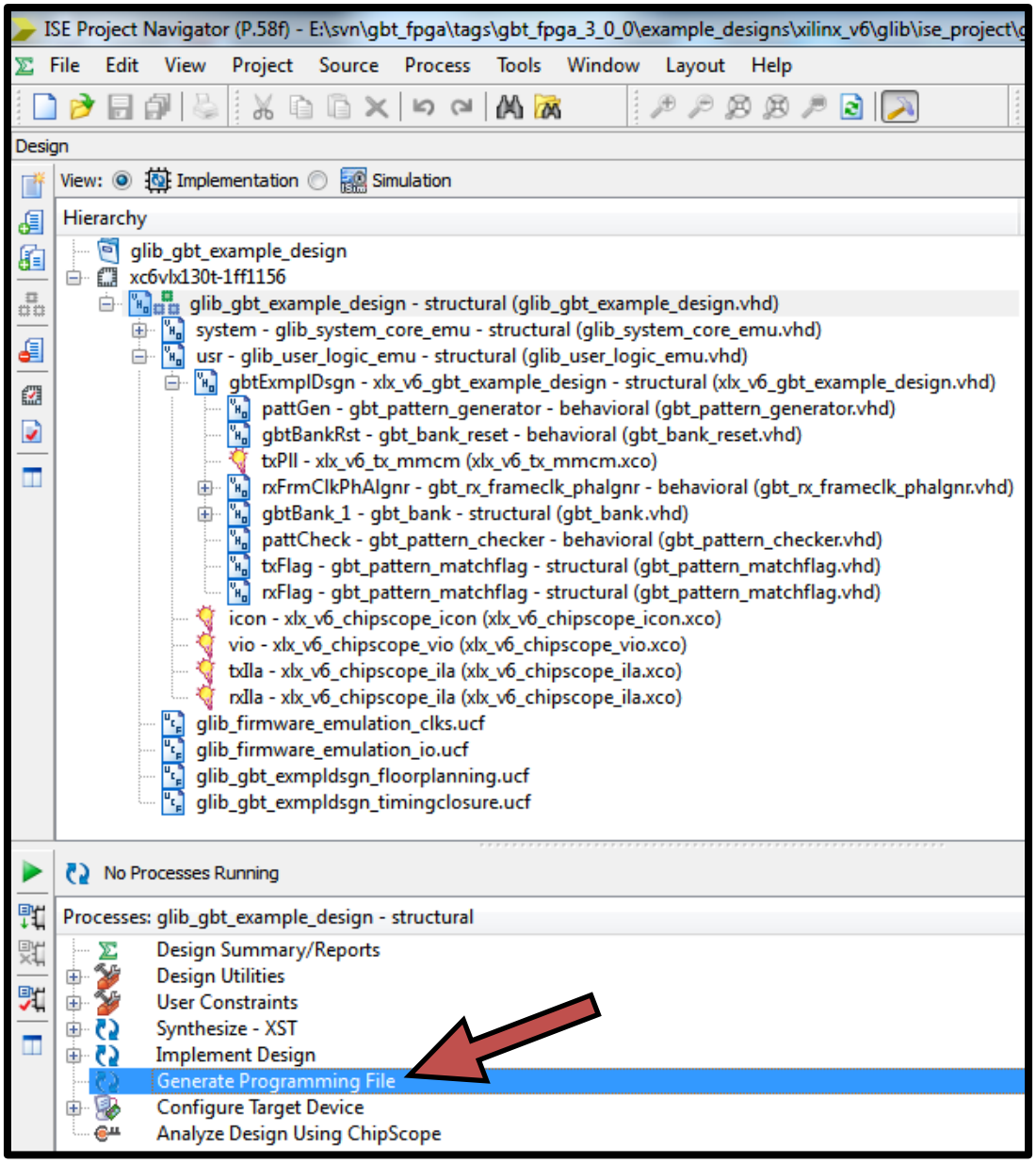


Figure 31: "Synthesize", "Implement Design" and "Generate Programming File"

14. Double click on the "Generate Programming file" process to synthesize and implement the project as well as to generate the programming file (.bit). You should see a blue sphere spinning near the process that is running (see Figure 32).

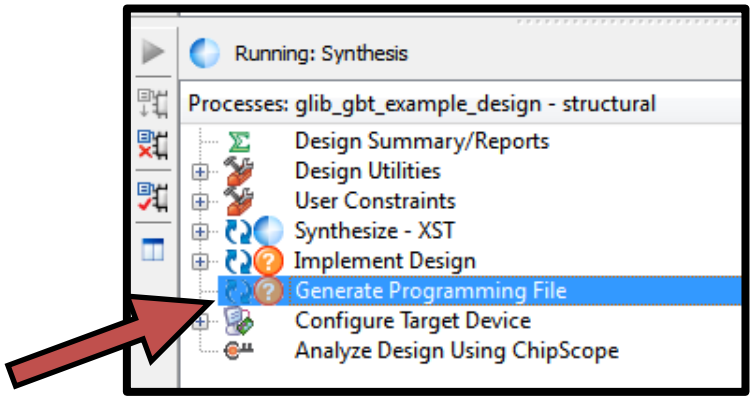


Figure 32: Processes running after clicking on "Genera Programming File"

15. The last step of this tutorial is the verification that all processes were done correctly. This information is shown in the right pane of the "Design Summary" tab, in the section "Design Overview" -> "Summary" (see Figure 33).

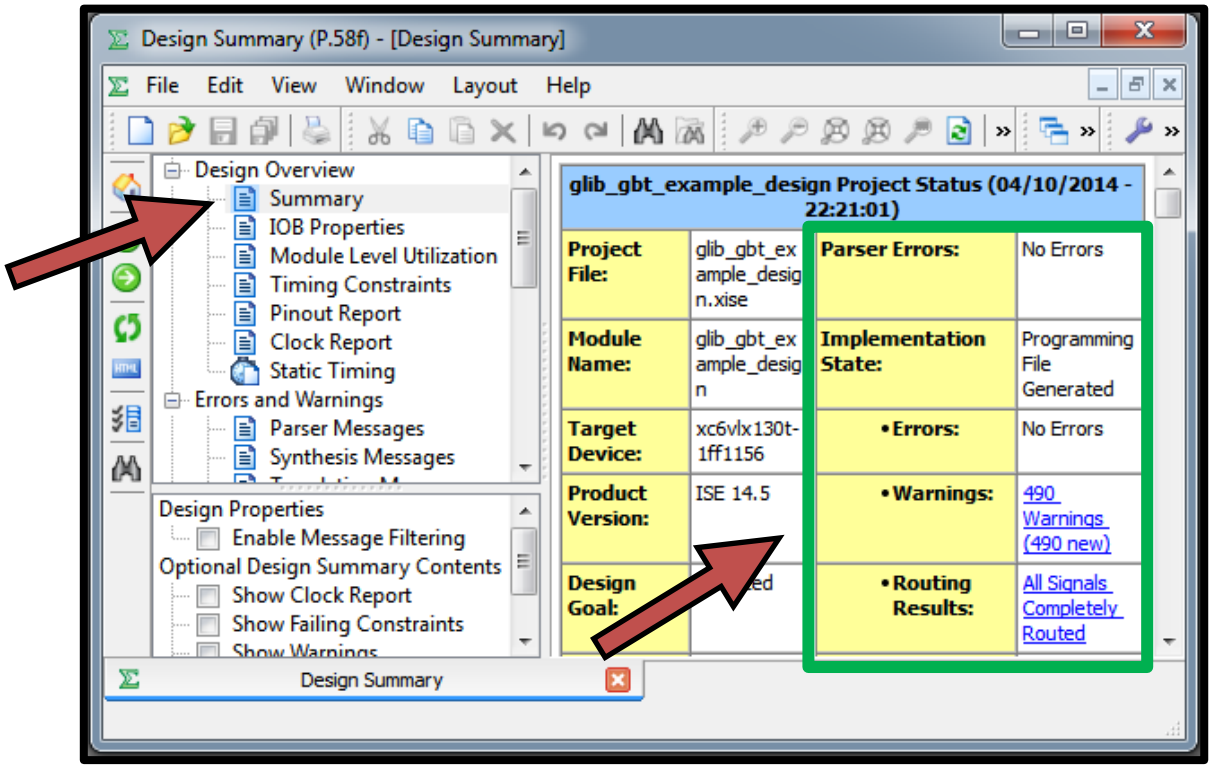


Figure 33: "Design Summary" report

3.2.2. Implementing the Example Designs in Altera FPGAs

The Altera EDA tool used by the GBT-FPGA team for developing the GBT-FPGA Core and the example associated designs is **Quartus II 13.1.** For this reason, the utilization of this EDA tool and version for implementing the example designs is recommended.

In this tutorial, the GBT-FPGA example design implemented is for the Cyclone V GT Devkit. Nevertheless, the same procedure may be applied to any of the other GBT-FPGA example design targeted to Altera FPGAs-based cards.

1. Open Quartus II (It may be both, the 32bit or 64bit versions) (see Figure 34).

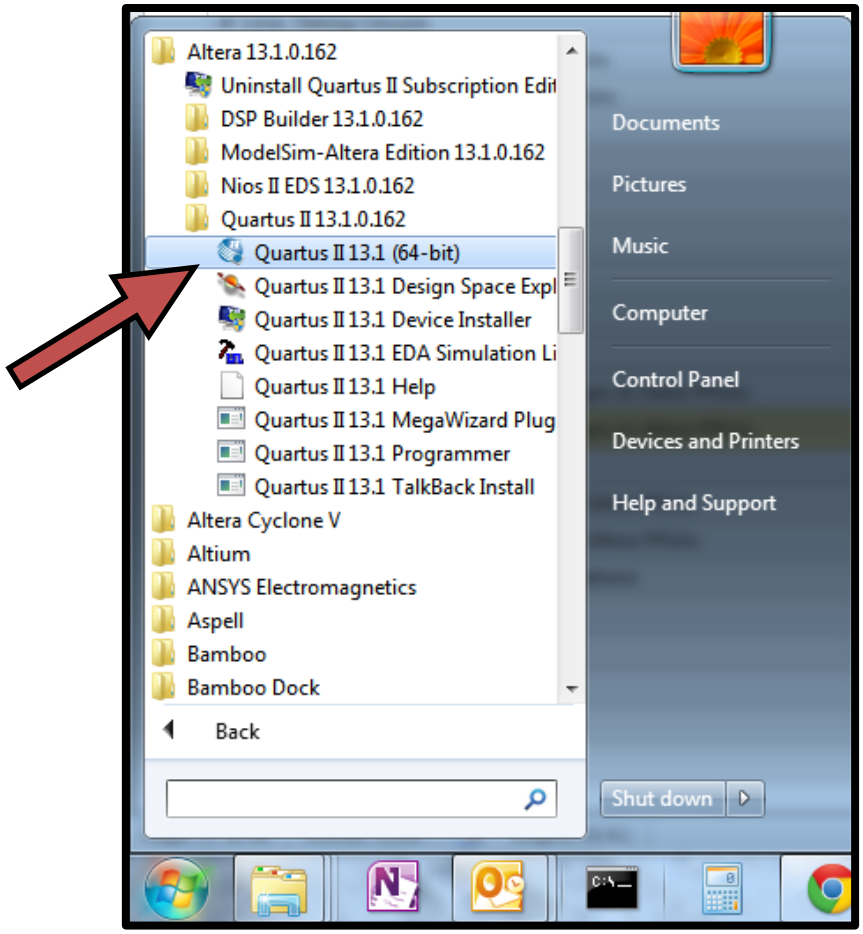


Figure 34: Opening Quartus II

2. Once with Quartus II is open, click on "Open Project…" in the "File" menu on the top left corner (see Figure 35).

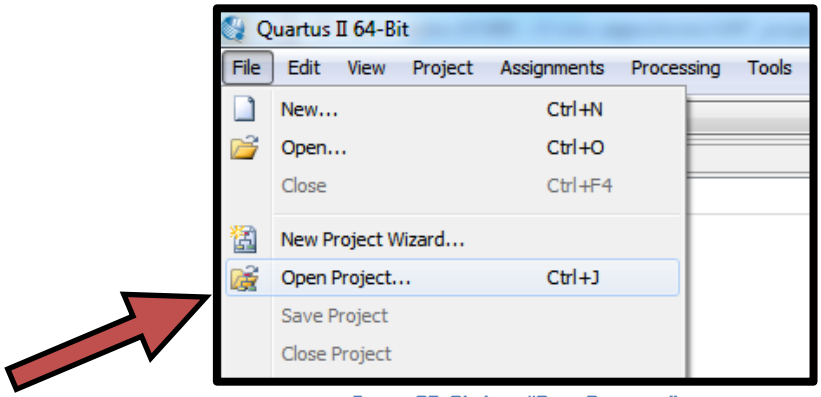


Figure 35: Clicking "Open Project…"

3. Go to the folder where the GBT-FPGA Core source files were downloaded from the GBT-FPGA SVN repository (see Figure 36).

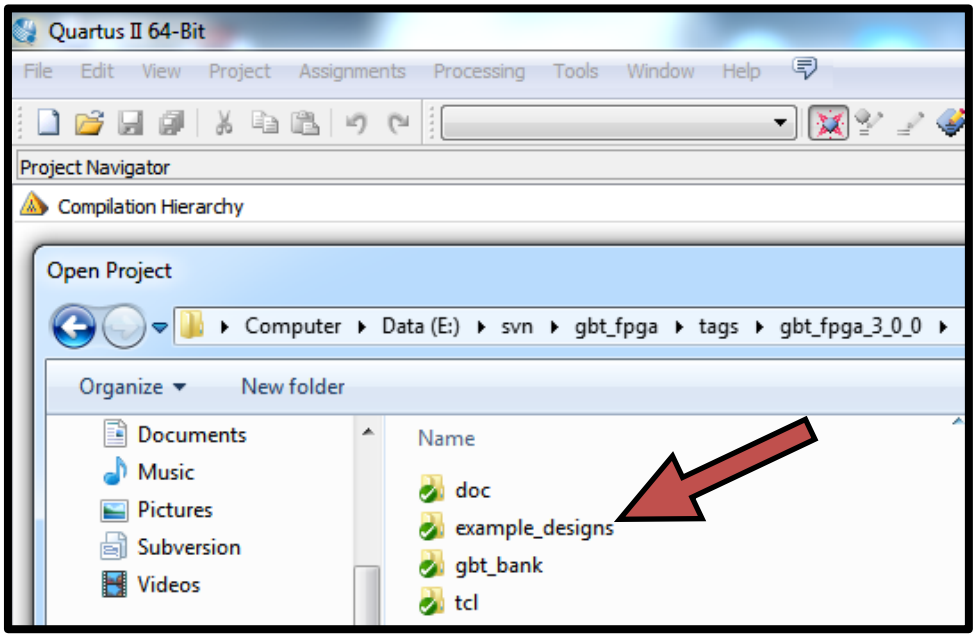


Figure 36: Root folder of the GBT-FPGA source files from the SVN repository

4. Open the "qii_project" folder where the Quartus II project file of the selected GBT-FPGA example design (in this case "cvGtFpgaDevkit_gbt_example_design.qpf") is stored and double click on it for opening the project in Quartus II (if it is the first time that you open the GBT-FPGA example design, the "Open Project" window will only show the .qps file) (see Figure 37).

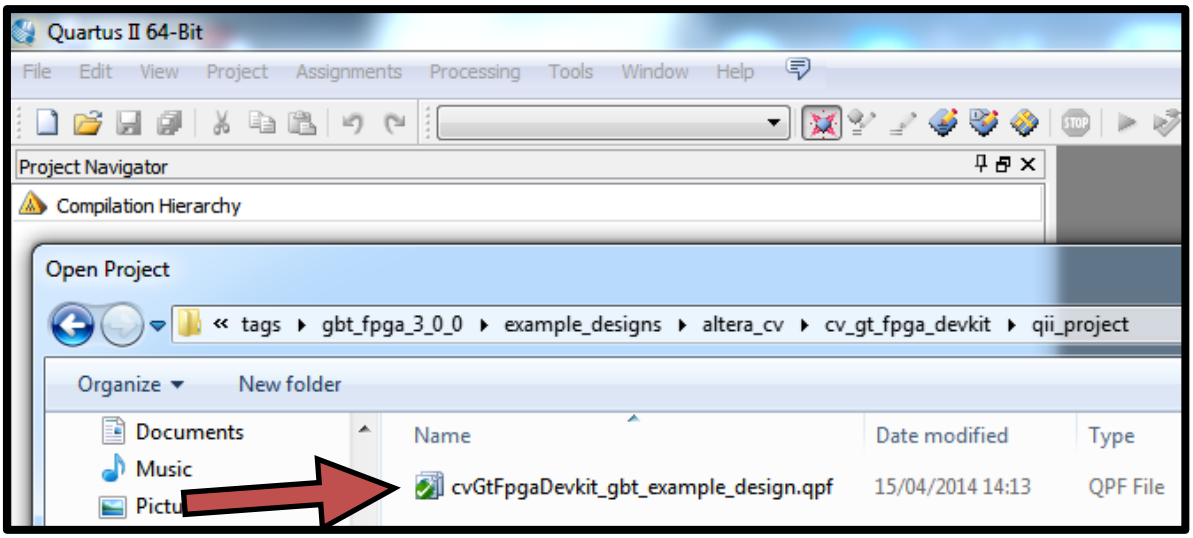


Figure 37: Folder of the Quartus II project file (.qpf)

5. Once the project of the GBT-example design is open in Quartus II, click on the button with a violet triangle for synthesizing, Implementing and generating the programming file (.sof).

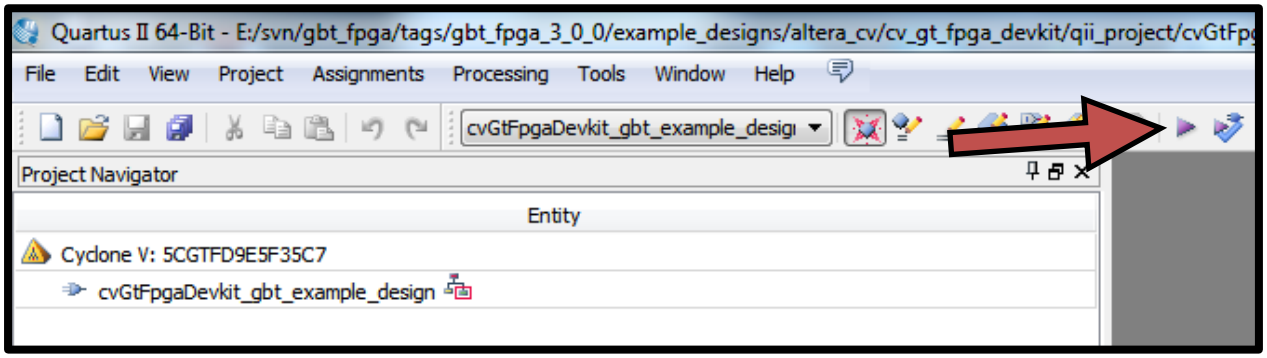


Figure 38: Clicking on the button for synthesizing, implementing and generating the programming file

6. After clicking the violet triangle, several tasks status bars appear in the window "Tasks", indicating how much of the project has been implemented (see Figure 39).

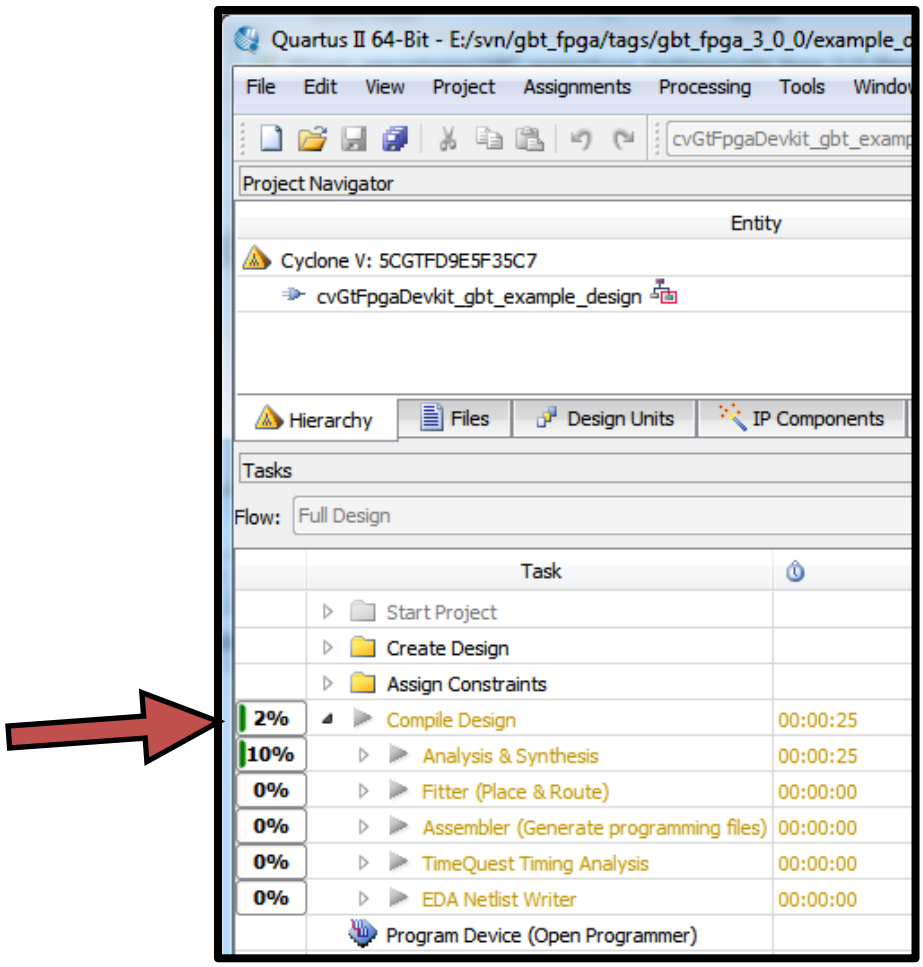


Figure 39: Implementation status bars

7. Once the implementation is completed, the next step is the verification that all tasks were done correctly. The green colour of the different tasks means that all of them were done correctly (see Figure 40). Besides the green colour of the tasks, we can also verify the correct implementation of the project and generation of the programming file through the "Table of Contents" window in the "Compilation Report" tab where no report is highlighted in red colour (see Figure 41)

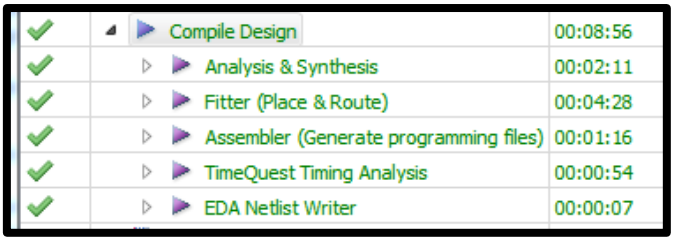


Figure 40: Correctly performed tasks highlighted in green colour

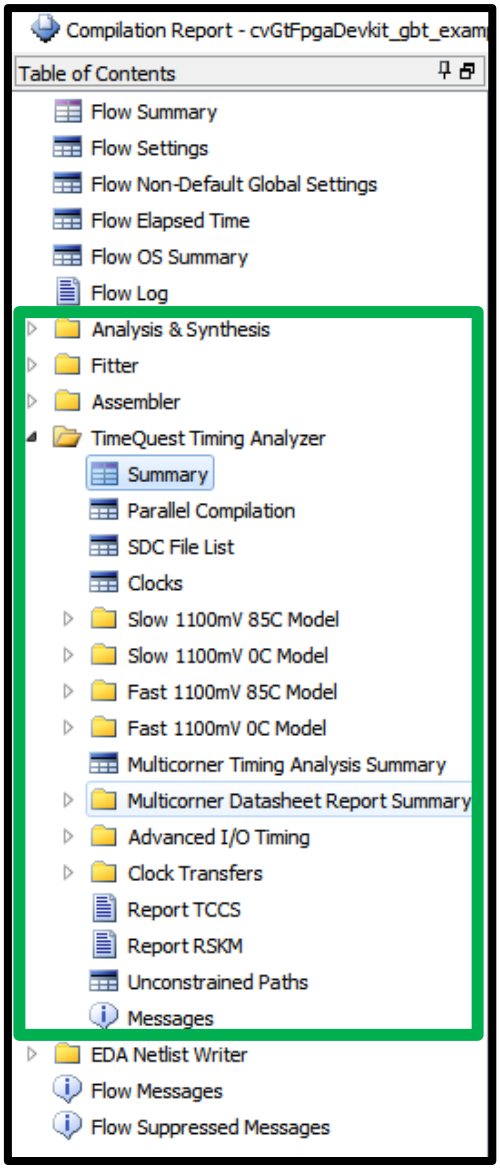


Figure 41: Absence of reports highlighted in red colour after implementation and programming file generation

The last (and optional) step after a correct implementation and programming file generation is clicking on the white triangle sign near the source file names in the "Project Navigator" pane, in order to expand the hierarchy of the source files that has been generated after a correct synthesis of the source files (see Figure 42 and Figure 43).

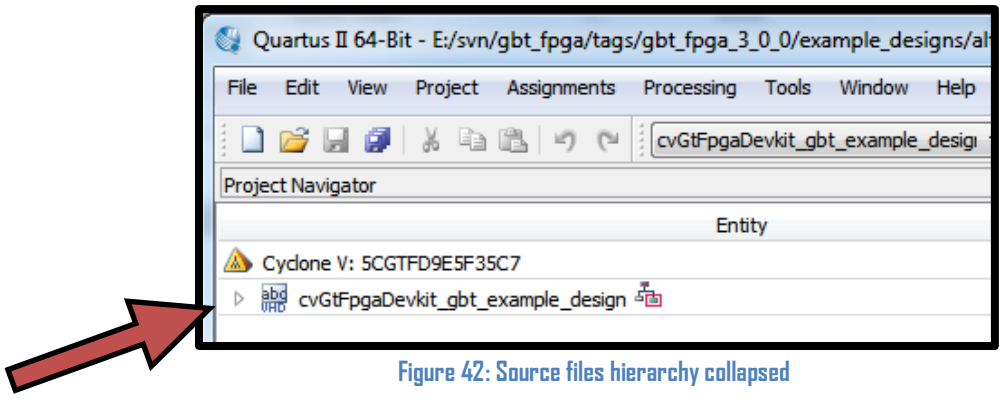


Figure 42: Source files hierarchy collapsed

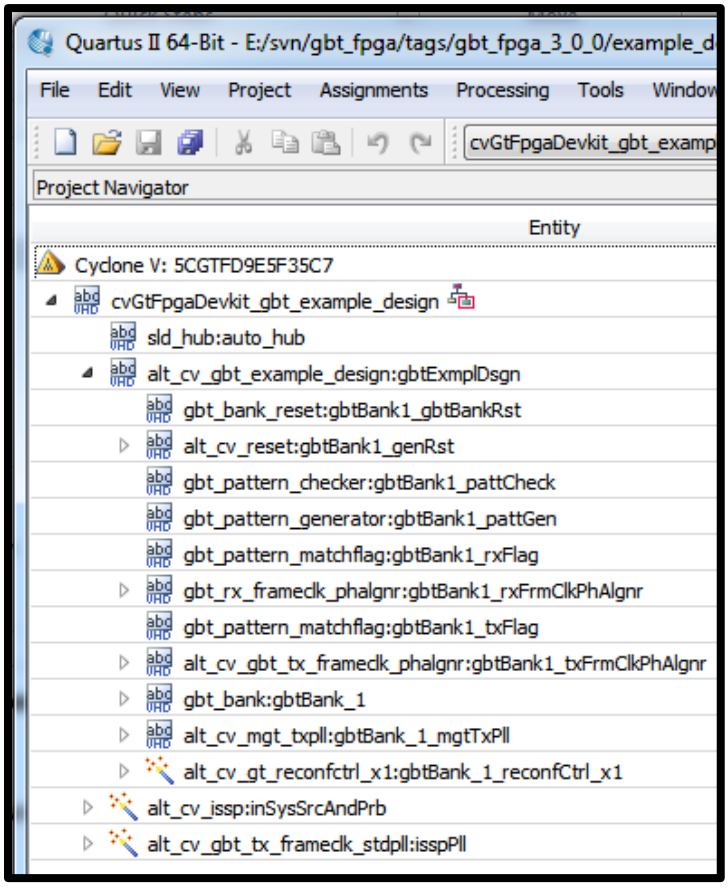


Figure 43: Source files hierarchy partially expanded

3.3. Running the Example Designs

3.3.1. Running the Example Design in Xilinx FPGAs using ChipScope

To be completed

3.3.2. Running the Example Design in Altera FPGAs using In-system Sources and Probes

To be completed

4. FPGA Particularities & Hardware Recommendations

4.1. FPGA Particularities

4.1.1. Xilinx FPGAs

4.1.1.a. Virtex 6

- The frequency of the MGT reference clock is 240MHz.
- When using the Latency-Optimized Rx version, the RX_FRAMECLK must be aligned in order to maintain the correct phase (a RX_FRAMECLK phase aligner is provided with the GBT-FPGA example design).

4.1.1.b. Kintex 7 & Virtex 7

- The frequency of the MGT reference clock is 120MHz.
- When using the Latency-Optimized Rx version, the RX_FRAMECLK must be aligned in order to maintain the correct phase (a RX_FRAMECLK phase aligner is provided with the GBT-FPGA example design).

4.1.2. Altera FPGAs

4.1.2.a. Cyclone V

- The frequency of the MGT reference clock is 120MHz.
- When using the Latency-Optimized Tx version, the phase of the TX_WORDCLK may be 0deg or 180deg with respect to the phase of the MGT REFCLK due to a phase uncertainty issue after MGT reset, present in all Altera GT transceivers of the V series. For that reason it is necessary the utilization of a phase monitoring that resets the MGT Tx when the phase of the TX_WORDCLK is not de one desired (the TX_WORDCLK monitor is already integrated into the Latency-Optimized MGT).
- When using the Latency-Optimized Tx version, the TX_FRAMECLK may need to be aligned in order to find the correct sampling point in the gearbox of the GBT Tx when crossing from TX_FRAMECLK to TX_WORDCLK domains (a TX_FRAMECLK phase aligner is provided with the GBT-FPGA example design)
- When using the Latency-Optimized Rx version, the RX_FRAMECLK must be aligned in order to maintain the correct phase (a RX_FRAMECLK phase aligner is provided with the GBT-FPGA example design).

Note for Latency-Optimized version in Cyclone V: achieving timing closure may be a difficult task in Cyclone V FPGAs (which are low-end devices). For that reason it is not recommended the utilization of Cyclone V FPGA when using the latency-optimized version. For more information about timing closure with Altera MGT see AN580.

4.1.2.b. Stratix V

- The frequency of the MGT reference clock is 120MHz.
- When using the Latency-Optimized Tx version, the phase of the TX_WORDCLK may be 0deg or 180deg with respect to the phase of the MGT REFCLK due to a phase uncertainty issue after MGT reset, present in all Altera GT transceivers of the V series. For that reason it is necessary the utilization of a phase monitoring that resets the MGT Tx when the phase of the TX_WORDCLK is not de one desired (the TX_WORDCLK monitor is already integrated into the Latency-Optimized MGT).
- When using the Latency-Optimized Tx version, the TX_FRAMECLK may need to be aligned in order to find the correct sampling point in the gearbox of the GBT Tx when crossing from TX_FRAMECLK to TX_WORDCLK domains (a TX_FRAMECLK phase aligner is provided with the GBT-FPGA example design)
- When using the Latency-Optimized Rx version, the RX_FRAMECLK must be aligned in order to maintain the correct phase (a RX_FRAMECLK phase aligner is provided with the GBT-FPGA example design).

4.2. Hardware Recommendations

- When using the Latency-Optimized version, the MGT reference clock must have very low jitter. The allowed values may be consulted in the datasheet of the targeted FPGA.
- Achieving timing closure may be complicated in low-end FPGAs (e.g. Cyclone V), mainly when implementing multi-GBT Link systems. For that reason it is recommended the utilisation of high-end FPGA (e.g. Virtex 6) when implementing multi-GBT Link systems.

5. References

- **The GBT Project Home Page:**

  https://espace.cern.ch/GBT-Project

- **The GBTx ASIC User Guide**

  https://espace.cern.ch/GBT-Project/GBTX/Manuals/gbtxManual.pdf

- **The GBT-FPGA Project Home Page**

  https://espace.cern.ch/GBT-Project/GBT-FPGA

- ***The GBT_FPGA: one unified core for multiple users*, ppt, M. Barros Marin, Feb 2014**

  https://svnweb.cern.ch/cern/wsvn/ph-ese/be/gbt_fpga/tags/doc/GBT-FPGA_students-fellows_ManoelBarrosMarin_2014_02_05.pdf

- ***The GBT_FPGA project*, M. Barros Marin, S. Baron, ACES 2014**

  https://svnweb.cern.ch/cern/wsvn/ph-ese/be/gbt_fpga/tags/doc/gbt_fpga_aces2014_poster.pdf

- **The GBT encoding scheme:** "An Error-Correcting Line Coding ASIC for a HEP Rad-Hard Multi-GigaBit Optical Link", G. Papotti, Proc. 2nd Conference on Ph.D. Research in Microelectronics and Electronics (PRIME 2006), Otranto (Lecce), Italy, 12-15 June 2006, pp.225-8.

APPENDIX A: Frequently Asked Questions

- **I am a new user: how do I start?**

See 2.4.1.

- **What is the difference between a GBT Bank and a GBT Link?**

The GBT Bank is the main module of the GBT-FPGA Core. Each GBT Bank may include several GBT Links (up to four in Xilinx FPGAs or up to three in Altera FPGAs).

The GBT Link is the actual channel of the link. It is composed by a GBT Tx (that scrambles and encodes the transmitted parallel data), a Multi-Gigabit Transceiver (MGT) (that serializes, transmits, receives and de-serializes the data) and a GBT Rx (that aligns, decodes and descrambles the incoming data stream).

- **Where can I choose the version I want to implement?**

The different parameters of the GBT Bank, such as optimization or frame type, are set through the GBT User Setup File (see 2.4.2). This file can be found in:

"... \gbt_bank\<vendor>_<device>\<vendor>_<device>_gbt_banks_user_setup.vhd"

(e.g. *".../gbt_bank\altera_cv\alt_cv_gbt_banks_user_setup.vhd"*)

- **Where can I define the frame type?**

The different parameters of the GBT Bank, such as optimization or frame type, are set through the GBT User Setup File (see 2.4.2). This file can be found in:

"... \gbt_bank\<vendor>_<device>\<vendor>_<device>_gbt_banks_user_setup.vhd"

(e.g. *".../gbt_bank\altera_cv\alt_cv_gbt_banks_user_setup.vhd"*)

- **How can I instantiate several GBT links in my FPGA?**

GBT links are grouped in GBT Banks. Each GBT Bank may have up to 4 GBT Links in Xilinx FPGAs or up to 3 GBT Links in Altera FPGAs. The user may instantiate several GBT Banks in parallel. The maximum number of GBT Banks is device dependant.

The example design for Altera Stratix V can be used as a reference of multi-GBT Links implementation. It instantiates one GBT Bank with one GBT Link and a second GBT Bank with three GBT Links.

- **How many GBT links can I fit within an FPGA?**

The maximum number of GBT Links is device dependent.

When implementing the Standard version in Xilinx or Stratix V devices, it is possible to have one GBT Link per MGT of the FPGA. In Cyclone V, at least one MGT Tx has to be used as reference clock by the other MGTs of the FPGA.

When implementing the Latency-Optimized version, the number of GBT Link also depends of the clocking resources available.

- **What is the difference between the Standard and the Latency Optimized versions?**

See 2.1.2.

- **Why the Latency Optimized version is so tricky?**

The use of registers for Cross Domain Crossing instead of FIFOs or DPRAMs and the dynamic realignment of the phase of the clocks in order to maintain the latency low, fixed and deterministic, introduce new technical challenges. Extra logic and clocking resources as well as timing and floorplanning constraints are needed for performing the clock alignments and for monitoring the correct phase relationship between clocks in some critical paths, thus complicating the implementation of the system.

- **What is critical for using a Latency Optimized version?**

The two main critical points when implementing the latency-optimized version of the GBT-FPGA are metastability issues and clocking resources availability.

In order to avoid **metastability issues** due to the register-based Clock Domain Crossing (CDC) of the latency-optimized version, the quality of the reference clocks in terms of jitter is very important. In addition, it is also necessary to ensure the correct relationship between the different clocks in the critical paths (using timing and floorplannig constraints, phase aligners, phase monitoring, etc.). Please note that the drift of the different clocks due to voltage or temperature variations may also lead to metastability issues so further study of the system under different conditions is recommended.

The other critical point when using the latency-optimized version is the intensive use of **clocking resources** due to the numerous clocks domains generated and the need of controlling the phase relationship between them.

- **Can I mix Standard and Latency-Optimized links within one MGT bank?**

No, all GBT Links of the same GBT Bank will share the same configuration.

- **Can I have one GBT link with the Rx in Standard version and the Tx in Latency-Optimized version (or reversely)?**

Yes, it is possible to use Latency-Optimized Tx and Standard Rx and vice versa.

Note that all GBT Links of the same GBT Bank will share the same configuration.

- **How can I integrate only the files I need in my project?**

The GBT-FPGA team provides TCL scripts (see 2.4.3) for adding the GBT-FPGA sources files into the project of the vendor tool (ISE in Xilinx and Quartus II in Altera).

- **How to use the current GBT-FPGA core for an FPGA which is not on the list.**

Most of the HDL files are vendor agnostic so they can be used as is in FPGA devices with enough resources. These files can be found in the folder:

"... \gbt_bank\core_sources\"

The rest of the files, that are vendor specific, must be regenerated for the targeted FPGA. These files can be found in the folder:

"... \gbt_bank\<vendor>_<device>\" (e.g. "... \gbt_bank\xilinx_v6")

- ***How to properly reset the GBT link?***

The GBT Link has to be reset sequentially.

In the **Tx side**, the GBT Tx has to be reset first and then the MGT Tx.

In the **Rx side**, the MGT Rx has to be reset first and then the GBT Rx.

- **What is the best suited FPGA on the GBT link point of view?**

There is not a best suited FPGA on the GBT Link point of view.

When using the **Standard version**, any high-end FPGA (e.g. Virtex 6, Stratix V) is a good candidate for hosting a GBT Link-based system.

When using the **Latency-Optimized version**, the differences between the Multi-Gigabit Transceivers (MGT) and the clocking resources of the different vendors may lead to a preferred candidate for certain GBT Link based applications.

Example 1: Xilinx FPGAs featuring GTX transceivers do not need any monitoring in the Tx side whilst Altera FPGAs featuring GT transceivers need extra logic for monitoring the TX_WORDCLK.

Example 2: The dynamic phase alignment of Altera fPLL allows independent phase values per output clock whilst the dynamic phase alignment of Xilinx MMCMs applies the same phase value to all clock outputs that are shifted.

- **What are the particularities of the Virtex6 for the Latency-Optimized version?**

See 4.1.1.a.

- **What are the particularities of the Kintex 7 for the Latency-Optimized version?**

The same GBT-FPGA core is shared by Kintex 7 and Virtex 7 so the particularities may be applied to both families of FPGAs.

See 4.1.1.b.

- **What are the particularities of the Virtex7 for the Latency-Optimized version?**

The same GBT-FPGA core is shared by Kintex 7 and Virtex 7 so the particularities may be applied to both families of FPGAs.

See 4.1.1.b.

- **What are the particularities of the Cyclone V for the Latency-Optimized version?**

See 4.1.2.a.

- **What are the particularities of the Stratix V for the Latency-Optimized version?**

See 4.1.2.b.

- **Which FPGA will be targeted in the future?**

For obvious reasons, the GBT-FPGA core will not offer firmware versions for each FPGA type on the market. It targets the main vendors and the main series. The design effort on new series is foreseen to stop during 2014. The evolution of the core to follow-up the technology will thus depend on users' contributions. If you aim to port the GBT-FPGA core to an FPGA or a reference design which is not yet in the list of supported devices, and you are willing to share your work, you are very much welcome to contact the GBT-FPGA support team (GBT-FPGA-support@cern.ch): they will try to integrate it to further releases.

- **When will the 8b10b version available?**

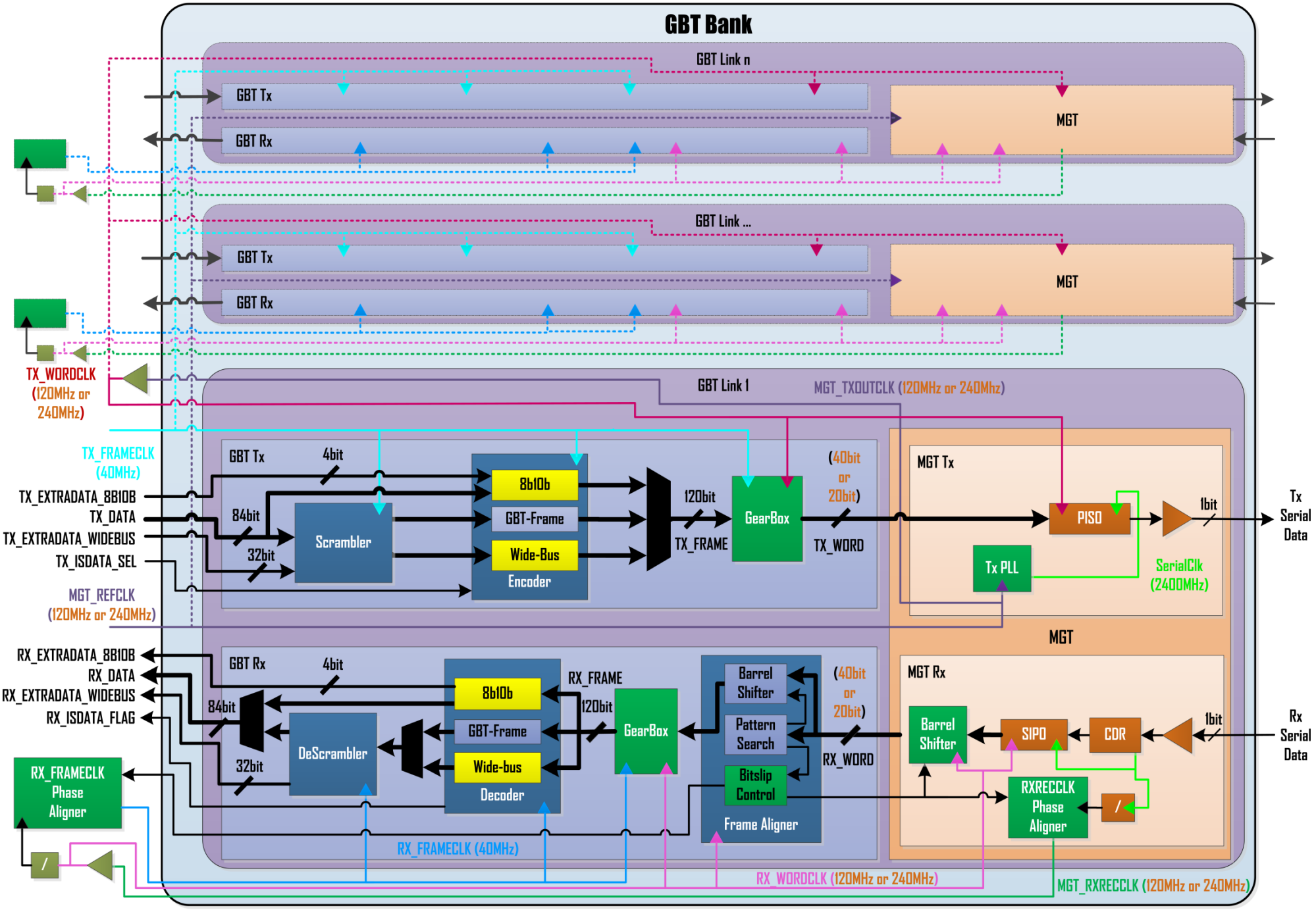
The 8b10b version is already under development and it will be available soon.

- **What type of support can I expect from the GBT-FPGA team?**

For obvious reasons, the GBT-FPGA team will not infinitely develop new firmware versions for each FPGA type on the market. The design effort (from the GBT-FPGA team) on new series is foreseen to stop during 2014, but the support will remain active on the official release available on the SVN, provided that the GBT-FPGA core itself has not been modified by the user.

The evolution of the core to follow-up the technology will depend on users' contributions. If you aim to port the GBT-FPGA core to an FPGA or a reference design which is not yet in the list of supported devices, and you are willing to share your work, you are very much welcome to contact the GBT-FPGA support team (GBT-FPGA-support@cern.ch): they will try to integrate it to further releases.

APPENDIX B: GBT Bank Block Diagram


GBT Bank
GBT Link n
GBT Tx
GBT Rx
MGT
GBT Link ...
GBT Tx
GBT Rx
MGT
TX_WORDCLK
(120MHz or
240MHz)
GBT Link 1
MGT_TXOUTCLK (120MHz or 240Mhz)
TX_FRAMECLK
(40MHz)
TX_EXTRADATA_8B10B
TX_DATA
TX_EXTRADATA_WIDEBUS
TX_ISDATA_SEL
MGT_REFCLK
(120MHz or 240MHz)
GBT Tx
4bit
84bit
32bit
Scrambler
8b10b
GBT-Frame
Wide-Bus
Encoder
120bit
TX_FRAME
GearBox
(40bit
or
20bit)
TX_WORD
MGT Tx
PISO
1bit
Tx
Serial
Data
Tx PLL
SerialClk
(2400MHz)
MGT
RX_EXTRADATA_8B10B
RX_DATA
RX_EXTRADATA_WIDEBUS
RX_ISDATA_FLAG
GBT Rx
4bit
84bit
32bit
DeScrambler
8b10b
GBT-Frame
Wide-bus
Decoder
RX_FRAME
120bit
GearBox
Barrel
Shifter
Pattern
Search
Bitslip
Control
Frame Aligner
(40bit
or
20bit)
RX_WORD
MGT Rx
Barrel
Shifter
SIPO
CDR
1bit
Rx
Serial
Data
RX_FRAMECLK
Phase
Aligner
/
RXRECCLK
Phase
Aligner
/
RX_FRAMECLK (40MHz)
RX_WORDCLK (120MHz or 240MHz)
MGT_RXRECCLK (120MHz or 240MHz)

APPENDIX C: Known Issues