GBTx Communication Document

 $University\ of\ Maryland\ LHCb\ group$

April 18, 2018

Rev: 0.2 (84f91ee)

Contents

L	Har	ardware setup							
	1.1	Overview	2						
	1.2	Configure GBTx to use external $\mathrm{I}^2\mathrm{C}$ adapter	2						
	1.3	Configure slave GBTx to use master SCA channel	2						
	1.4	Configure GBTx to use GBT-IC channel	3						
	1.5	Reset GBTx	4						
2	Soft	Software setup							
	2.1	Program master GBTx with external $\mathrm{I}^2\mathrm{C}$ adapter	4						
	2.2	Check communication between master GBTx and MiniDAQ .	5						

1 Hardware setup

1.1 Overview

Our current setup consists of one master and one slave GBTx board. The master is connected to the MiniDAQ GBTx channel 3 (fiber 8), and the slave can be connected to either GBT channel 0 (fiber 6) or channel 6 (fiber 11). The master synchronizes its on-board clock to the signal from the MiniDAQ, and propagates its clock signal to the slave. The slave does not have an on-board clock, and is configured to obtain clock signal externally.

The master I²C port is connected to an external USB device. The slave I²C port is connected to the master. Both are set to be programmed by the I²C channel, rather than GBT-IC channel.

The current setup is capable of:

- 1. Program the slave GBTx board with MiniDAQ directly.
- Read/Write the register value of the master GBTx board with GBT-IC specification on the MiniDAQ.
- 3. Do PBPS tests from MiniDAQ to the slave, then back to the MiniDAQ. The master is also required as the slave can only obtain its reference clock from the master.

1.2 Configure GBTx to use external I²C adapter

This setup is required to program a GBTx board using an external I²C adapter. Follow Figure 1 to connect an external I²C adapter.

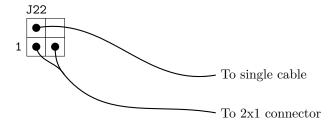


Figure 1: Schematic for external I²C adapter setup.

The I^2C adapter used in our lab is made in-house. For the 2x1 connector, make sure the side that has *no* metal contact is facing up.

1.3 Configure slave GBTx to use master SCA channel

This setup is required to program a slave via a master SCA channel. In a typical scenario, the master is connected to a MiniDAQ so that programming the slave using the MiniDAQ directly² is possible. Follow Figure 2 to connect a slave GBTx to the SCA channel of a master GBTx board.

¹ The master GBTx is also connected to the MiniDAQ with a different channel, to provide reference clock to the slave.

² MiniDAQ \rightarrow master GBTx \rightarrow slave GBTx.

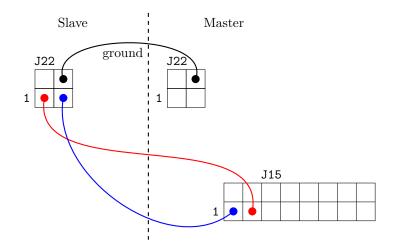


Figure 2: Schematic for slave to master SCA setup.

The black ground cable can be connected to any of the ground pin on the master GBTx.

There is a 2x1 to 2x1 cross-type cable made in-house to replace the redblue cables. To use that cable, make sure the two 2x1 connectors have the same orientation (e.g. the sides *without* metal contact are both facing up).

1.4 Configure GBTx to use GBT-IC channel

It might be useful to read/write individual registers from/to a GBTx board. In this case, follow the Figure 3 to flip the configSelect switch.

Flip the configSelect switch will render the external I²C adapter ineffective. None of the GBTx register value is fused onto the board, so a GBTx board in our lab must always be programmed externally via I²C before flipping the switch.

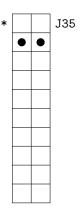


Figure 3: Schematic for flipping the configSelect switch. A jumper should be used to connect the two pins marked above.

1.5 Reset GBTx

Sometimes GBTx boards will not be properly reset by reprogramming. In such case, a hard reset is needed. Follow the Figure 4 to reset GBTx boards.



Figure 4: Schematic for resetting GBTx boards. A jumper should be used to connect the two pins marked above.

2 Software setup

2.1 Program master GBTx with external I²C adapter

Before proceed, follow the instruction on subsection 1.2 to configure the hardware first. As said in the previous section, the master GBTx board must be programmed via an external I^2C adapter first. A Windows 7 computer on the rack is used. The **GBTX programmer** is located at:

DT_Rack\GBTx_programmer\GBTxProgrammer.jar

SB dongle open	w v1.c	Register Select		#Register	Value (hex)	Name			
1 s			0	0x00	ckCtr0				
1 sent to VTarget LDO GBTX found on address 1					1	0x00	ckCtr1		
			2	0x00	ckCtr2				
0-1			3	0x00	ckCtr3				
Selected GBTX address 1						4	0x00	ttcCtr0[7:0]	
			5	0x00	ttcCtr1[7:0]				
Configuration loaded from txt file						6	0x00	ttcCtr2[7:0]	
				7	0x00 0x00	ttcCtr3[7:0] ttcCtr4[7:0]			
ritten and read	I. Programming								
was successful!						9	0x00	ttcCtr5[7:0]	
						10	0x00	ttcCtr6[7:0]	
-						11 12 13	0x00 0x00 0x00	ttcCtr7[7:0]	
								ttcCtr8[7:0]	
					ttcCtr9[7:0]				
					_	14	0x00	ttcCtr10[7:0]	
						15	0x00	ttcCtr11[7:0]	-
	V	-	16 17	0x03 0x03	ttcCtr12[7:0]	-			
Load GBTX configuration			V	-	18	0x03	ttcCtr13[7:0]	-	
			V	-	18	0x03	ttcCtr14[7:0] ttcCtr15[7:0]	-	
Write ALL to the GBTX				V		20	0x03	ttcCtr16[7:0]	-
				V		21	0x03	ttcCtr17[7:0]	-
	V		22	0x03	ttcCtr18[7:0]	-			
100-14-	V		23	0x03 0xff	ttcCtr19[7:0] ttcCtr20[7:0]				
Write selected to the GBTX Read all registers							V		
				V			25	0x03	ttcCtr21[7:0]
				V		26	0x7f	ttcCtr22[7:0]	\neg
			V		27	0x28	serCtr0[7:0]		
Reset 1V	Reset		set 2V5			28	0x00	txCtr0	
Reset 1V3		Ke	361 2 V 3	V		29	0x15	txCtr1	
BTX on 1			1	V		30	0x15	txCtr2	
	Read	state:	24 (dec)	V		31	0x15	txCtr3	
				V		32	0x66	txCtr4	
		dle (norm	ai status			33	0x00	txCtr5	
	when ru	when running)				34	0x0d	desCtr0	
	Power GBTX trough I2			V		35	0x42	rxCtr0	
	Power	GBTX troug	jh I2C adapter			36	0x00	rxCtr1	
	Enable expert mode					37	0x0f	rxCtr2	
	EII	able exp	ertilloue	~		38	0x04	rxCtr3	
		V		39	0x08	rxCtr4			
Fuse	Select all registers				DEselect all registers				

Figure 5: A typical UI for GBTx programmer.

Launch the programmer, a typical UI is shown in Figure 5, Click Load GBTX configuration and load a configuration file, which is located at:

```
{\tt DT\_Rack\backslash GBTx\_programmer\backslash GBTx\_TRx\_v12\_test\_withWatchDog.txt}
```

Then click Write ALL to the GBTX. Check the returned message to make sure everything works (supposedly). Now click Read state. If the master GBTx is configured correctly and is connected to a working MiniDAQ, the return value should be:

```
24 (dec): Idle (normal status when running)
```

2.2 Check communication between master GBTx and MiniDAQ

After program the master GBTx board and verify the return value, we can check the communication between the GBTx and the MiniDAQ with **GBT** Client.

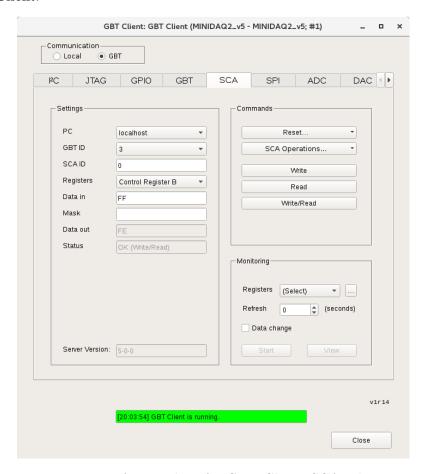


Figure 6: A typical UI for GBT Client, SCA tab.

Here we need to use the Linux server on the rack. To launch that program, locate the **gedi** panel. Under **JCOP framework**, click the **GBT Client**. Choose **GBT** option under the **Communication** tab. Navigate to **SCA** tab.

To check whether the link between GBTx and MiniDAQ is successfully established, configure the parameters *exactly* as shown in Figure 6. Now click **Write/Read**. The **Data out** field should have a value of "FE", and the **Status** should be "OK (Write/Read)".

GBT ID corresponds to the physical optical link that is connected to the master GBTx board. Recall that in our setup, the master is connected to optical fiber 8, which is mapped to GBT channel 3.

There are only 4 **Registers**. All 4 registers work, the "Control Register B" is chosen for no apparent reason. What matters is the return value should be consistent with our expectation.