I2C Master Core with Interface to GTB Bus

for

FOPI Experiment: TACQUILA Board

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master Core

The task: To design an *I2C master* core with Interface to *GTB Bus*.

The Application: *FOPI* experiment, *Tacquila Board*.

The *I2C core* with Interface to *GTB Bus* will be used for a controlling of temperature sensors, located inside of the *FOPI* detector and measuring the detector's internal temperature.

The *I2C core* (in VHDL) will be implemented in *Tacquila's FPGA* (Altera, ACEX 1K).

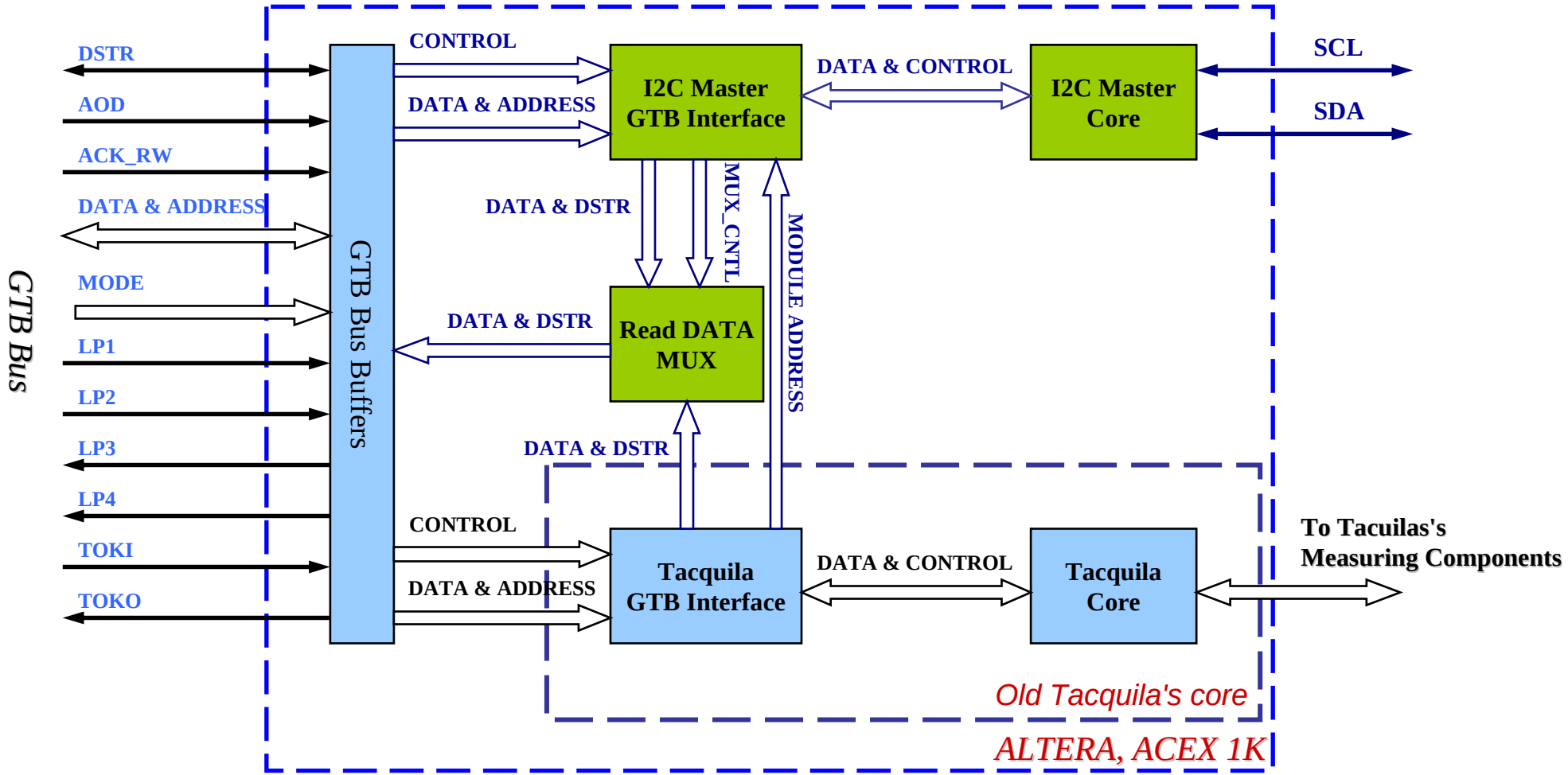
The Start: Jun, 2008.

The End: December, 2008.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

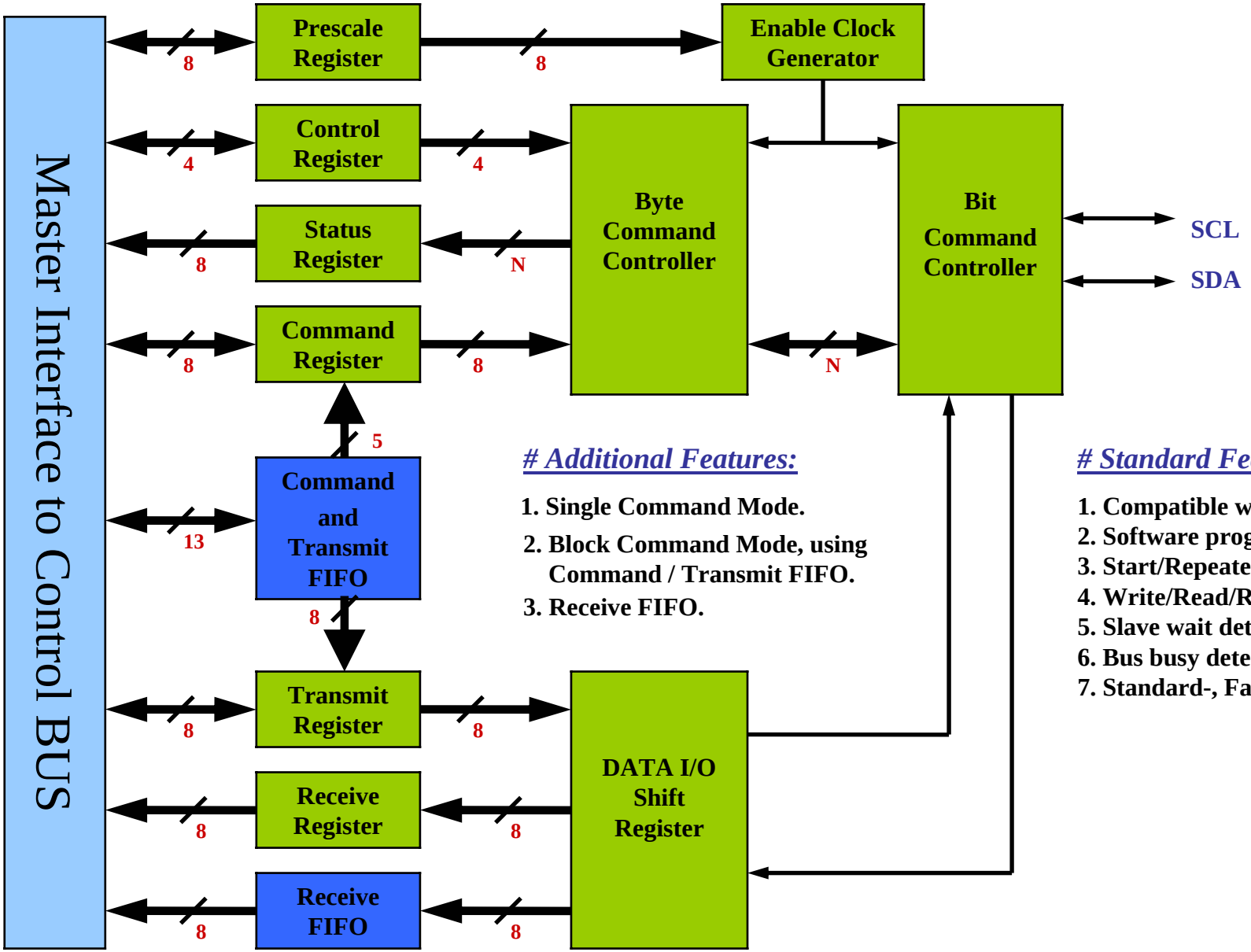
I2C Master Core with Interface to GTB Bus

(Block Diagram)



Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: Block Diagram



Additional Features:

1. Single Command Mode.
2. Block Command Mode, using Command / Transmit FIFO.
3. Receive FIFO.

Standard Features:

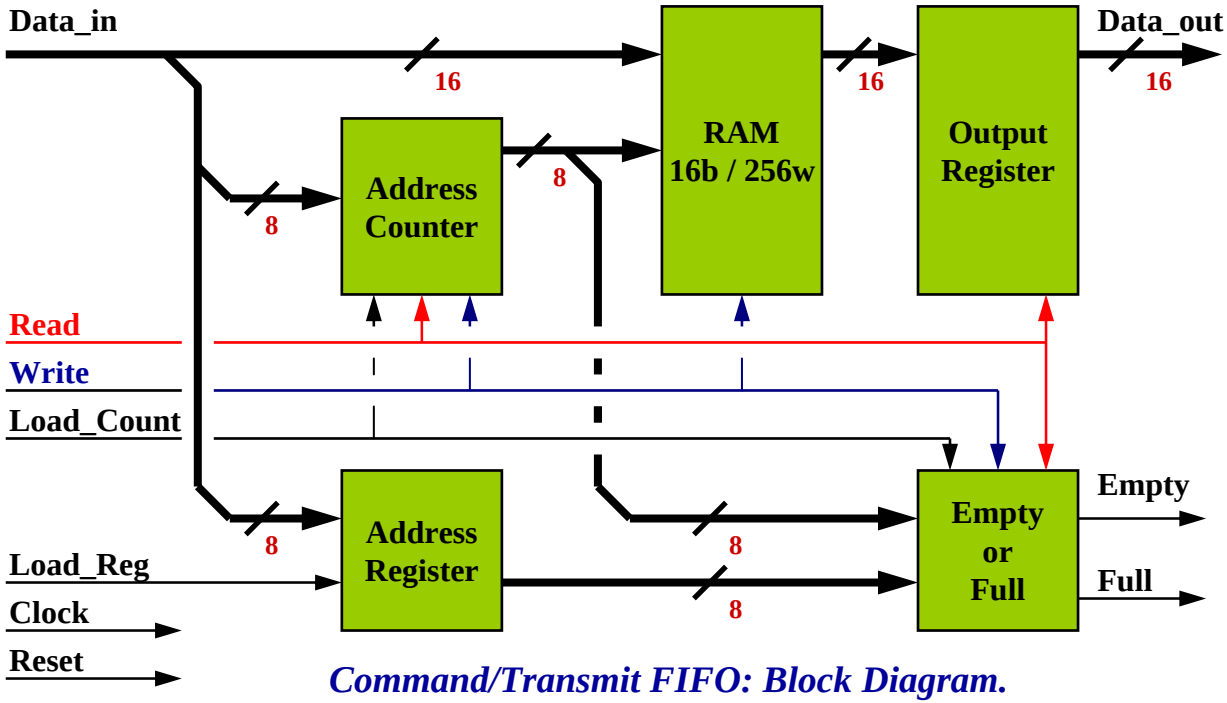
1. Compatible with Philips I2C standard.
2. Software programmable clock frequency.
3. Start/Repeated and Stop generation.
4. Write/Read/Repeated generation.
5. Slave wait detection.
6. Bus busy detection.
7. Standard-, Fast- and High-Speed Modes.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: Command / Transmit FIFO

1. The ***Receive FIFO*** is standard ***FIFO memory*** (8b x 512w).

2. The ***Command / Transmit FIFO***: This is a ***RAM***, working as a ***FIFO*** (16b x 256w).
   (Each ***Block of Commands*** could be executed a lot of times, no limit.)



Command/Transmit FIFO: Block Diagram.

Mapping of "Block of Commands"
(Command / Transmit FIFO)

Address *255*	Address Register	ST0
Block 3	Count: From: Value of Counter To: Value of Register	
Address *N +1*	Address Counter	STA
Address *N*	Address Register	ST0
Block 2	Count: From: Value of Counter To: Value of Register	
Address *M +1*	Address Counter	STA
Address *M*	Address Register	ST0
Block 1	Count: From: Value of Counter To: Value of Register	
Address *0*	Address Counter	STA

***Important*: When is executed command *"Load Address Counter"* both flags "*Empty* and *Full*" are settled in '*0*'.**

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: Example for "Block of Commands"

(for TMP101)

ACK bit	Command Register	Transmit Register	Action
---------- General call ----------			
b"1"	b"1000"	b"00000000"	-- General Call *(STA, FIFO's Address Counter)*.
b"1"	b"0001"	b"00000110"	-- Latch-of-address and reset *(All Slaves)*.
---------- Slave's Configuration Register ----------			
b"1"	b"1000"	b"10010100"	-- Slave's address and R/W = '0' *(Write)*.
b"1"	b"0001"	b"00000001"	-- Pointer Register's address: *Control Register*.
b"1"	b"0001"	b"01100100"	-- One byte written into slave's *Control Register*.
---------- Slave's T_Low Register ----------			
b"1"	b"1000"	b"10010100"	-- Slave's address and R/W = '0' *(Write)*.
b"1"	b"0001"	b "00000010"	-- Pointer Register's address: *T_Low Register*.
b"1"	b"0001"	b"00000000"	-- The first byte written into slave's *T_Low Register*.
b"1"	b"0001"	b"00000000"	-- The second byte written into slave's *T_Low Register*.
---------- Slave's T_High Register ----------			
b"1"	b"1000"	b"10010100"	-- Slave's address and R/W = '0' *(Write)*.
b"1"	b"0001"	b"00000011"	-- Pointer Register's address: *T_High Register*.
b"1"	b"0001"	b"01100100"	-- The first byte written into slave's *T_High Register*.
b"1"	b"0001"	b"00000000"	-- The second byte written into slave's *T_High Register*.
---------- Set the pointer register of Slave ----------			
b"1"	b"1000"	b"10010100"	-- Slave's address and R/W= '0' *(Write)*.
b"1"	b"0001"	b"00000000"	-- Pointer Register's address: *Temperature Register*.
---------- Read the temperature register of Slave (Point of Interrupting) ----------			
b"1"	b"1000"	b"10010101"	-- Slave's address and R/W = '1' *(Read)*.
b"0"	b"0010"	b"00000000"	-- The first byte read from slave's *Temperature Register*.
b"0"	b"0010"	b"00000000"	-- The second byte read from slave's *Temperature Register*.
---------- I2C Stop command ----------			
b"1"	b"0100"	b"00000000"	-- The Stop Command for I2C Core *(STO, FIFO's Address Register)*.

Block of Commands

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: Standard Registers

1. Control Register (8 bit word):

Bit (7) = **Enable of I2C core, active '1'.**
Bit (6) = **Command Interrupt, active '1'.**
Bit (5) = **Command Cancel, active '1' (new).**
Bit (4) = **Block Command Mode, active '1' (new).**
Bit (3 downto 0) = Reserve.

2. Command Register (8 bit word):

Bit (7) = **Start Command, active '1'.**
Bit (6) = **Stop Command, active '1'.**
Bit (5) = **Read Command, active '1'.**
Bit (4) = **Write Command, active '1'.**
Bit (3) = **ACK_Read, active '0' (transmited).**
Bit (2 downto 0) = Reserve.

3. Transmit Register (8 bit word):

Bit (7) = Adrress (MSB) or **DATA (MSB).**
Bit (6) = Adrress or **DATA.**
Bit (5) = Adrress or **DATA.**
Bit (4) = Adrress or **DATA.**
Bit (3) = Adrress or **DATA.**
Bit (2) = Adrress or **DATA.**
Bit (1) = Adrress (LSB) or **DATA.**
Bit (0) = R/W Command or **DATA (LSB).**

4. Prescale Register (8 bit word):

Bit (7 downto 0) = N (code).

I2C serial clock:

SCL = [CLK_CORE/(5*(N+1))].

5. Receive Register (8 bit word):

Bit (7) = DATA (MSB).
Bit (6) = DATA.
Bit (5) = DATA.
Bit (4) = DATA.
Bit (3) = DATA.
Bit (2) = DATA.
Bit (1) = DATA.
Bit (0) = DATA (LSB).

6. Status Register (8 bit word):

Bit (7) = ACK_Write, active '0' (received, by STA or W).
Bit (6) = I2C Bus is busy, active '1'.
Bit (5) = Block Command Transfer is done, active '1'.
Bit (4) = Single command in progress, active '1'.
Bit (3) = I2C interrupt in progress, active '1'.
Bit (2) = I2C Slave Wait, active '1'.
Bit (1) = Transmit FIFO (RAM) is empty, active '1'.
Bit (0) = Receive FIFO is full, active '1'.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: The single Command Operation

The sequential steps for the executing of the single command:

The Step 1: Writing of the code *"N"* into **Prescale Register** (fixing I2C core's speed).

after reset **N = 24**, where, by the core clock of **50 MHz**, the I2C serial clock has a value of **SCL = 400 KHz** (fast mode).

The Step 2: Writing of DATA into **Command and Transmit Registers** (Command and DATA).

The Step 3: Writing of DATA into **Control Register** – only **ENA_CORE** [Bit (7)] has to be settled in '1'.

The Step 4: Reading of DATA from **Status Register** to check that the current command is done in proper way:

Bit (4), **Single Command in progress**, has to be '0'.

After the commands **Start** and **Write** Bit (7), **ACK_WRITE**, has to be '0'.

If the executed command was **Start**, **Stop** or **Write** then go in Step 2. New command could be executed.

The Step 5: When the executed command was **Read** then read DATA from Receive Register, and go in Step 2. New command could be executed.

Important: In this mode, after executing of each command, all bits of the **Command Register** are settled in '0'. The I2C core **is waiting** for new **Command** and **DATA** for the next transmission.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: The Block Command Operation

The sequential steps for the executing a block of commands:

The Step 1: Writing of the code *"N"* into **Prescale Register** (fixing I2C core's speed, as Single Command Mode).

The Step 2: Writing of DATA into **Command / Transmit FIFO** (Commands and DATA).

The Step 3: Writing of DATA into **Address Register** (end of Block of Commands).

The Step 4: Writing of DATA into **Address Counter** (start of Block of Commands).

The Step 5: Writing of DATA into **Control Register** :

Ena_Core [Bit (7)] has to be settled in '1'.

Block_Mode [Bit (4)] has to be settled in '1'.

The Step 6: Reading of DATA from **Status Register** to check that the block of commands is done in proper way:

Bit (5), **Block of Commands is done**, has to be '1'.

Bit (1), **Transmit FIFO (RAM) is empty**, has to be '1'.

If the executed commands ware **Start**, **Stop** or **Write**, then go in Step 3. New block of commands could be executed.

The Step 7: When the executed commands ware **Start**, **Stop** and **Read**, then read DATA from Receive FIFO, and go in Step 3. New block of commands could be executed.

Important: After executing of each bock of commands, all bits of the **Command Register** are settled in '0'.
The I2C core **is waiting** for new block **Commands** and **DATA** for the next transmission.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: Interface to GTB Bus

1. The ***Tacquila's Interface to GTB*** Bus is divided on two parts:

The first part was designed and it supports all needed ***GTB Bus modes***: reset, initialization, 16-bit read/write, token read.

The second – Interface to ***GTB Bus for I2C Master Core*** was designed to support only "***GTB16 address mode read/write***", which is enough to have access to all registers of I2C Master.

In additional – the second part of the ***GTB*** interface controls the multiplexing of ***I2C*** or ***Tacquila*** output data.

2. Interface to ***GTB Bus for I2C Master Core***: The Register Addresses.

I2C's *Registers*	***GTB Bus: ADDR (10 downto 0)***											**Access**
	10	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
Command/Transmit Register	1	0	0	0	0	0	0	0	0	0	1	Read/Write
Command/Transmit FIFO	1	0	0	0	0	0	0	0	0	1	0	Read/Write
Receive Register	1	0	0	0	0	0	0	0	1	0	0	Read only
Receive FIFO	1	0	0	0	0	0	0	1	0	0	0	Read only
Control Register	1	0	0	0	0	0	1	0	0	0	0	Read/Write
Prescale Register	1	0	0	0	0	1	0	0	0	0	0	Read/Write
Address Counter/Register	1	0	0	0	1	0	0	0	0	0	0	Read/Write
Tacquila DATA – FIFO	1	0	0	1	0	0	0	0	0	0	0	Read only
Status Register	***No address***. It is read together with Control Register (Read only).											

3. The GTB Module address: ***ADDR(15 downto 11)***, 5 bits.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master: Register DATA on GTB Bus

1. The Register DATA of ***I2C Master Core*** on ***GTB Bus***:

I2C's *Registers*	GTB Bus: DATA (15 downto 0)															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Command/Transmit Register	*	*	*	Ack¹	STA	STO	R	W	D7	D6	D5	D4	D3	D2	D1	D0
or	- or -			*Command Register/FIFO*					*Transmit Register/FIFO*							
Command/Transmit FIFO	*	*	*	Ack¹	STA	STO	R	W	A7	A6	A5	A4	A3	A2	A1	R/W
Receive Register	*	*	*	*	*	*	*	*	Rd7	Rd6	Rd5	Rd4	Rd3	Rd2	Rd1	Rd0
Receive FIFO	*	*	*	*	*	*	*	*	Rd7	Rd6	Rd5	Rd4	Rd3	Rd2	Rd1	Rd0
Control Register and Status Register	Ct7	Ct6	Ct5	Ct4	*	*	*	*	Sr7	Sr6	Sr5	Sr4	Sr3	Sr2	Sr1	Sr0
Prescale Register	*	*	*	*	*	*	*	*	Pr7	Pr6	Pr5	Pr4	Pr3	Pr2	Pr1	Pr0
Address Counter/Register	Ar7	Ar6	Ar5	Ar4	Ar3	Ar2	Ar1	Ar0	Qr7	Qr6	Qr5	Qr4	Qr3	Qr2	Qr1	Qr0
Tacquila DATA – FIFO	Tq15	Tq14	Tq13	Tq12	Tq11	Tq10	Tq9	Tq8	Tq7	Tq6	Tq5	Tq4	Tq3	Tq2	Tq1	Tq0

 # Ack¹ signal, transmitted from master to salve. STA – start, STO – stop, R – read, W – write (commands).

 # Rd (7 downto 0) – data, received by the master (Receive Register/FIFO).

 # Ct (7 downto 4) – data of Control Register.

 # Sr (7 downto 0) – data of Status Register.

 # Pr (7 downto 0) – data of Prescale Register.

 # Ar (7 downto 0) – data of Address Register. The end address of Transmit FIFO.

 # Qr (7 downto 0) – data of Address Counter. The start address of Transmit FIFO.

 # Tq (15 downto 0) – Tacquila's data.

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt

I2C Master Core: Summary

1. ***The I2C Master core***:

The design languages: ***VHDL*** (< 250 CLBs).

It is ready, implemented and fully tested with "Vulom 3". For the visualization of read temperature is used Vulom 3's LCD display.
(For the testing was designed "I2C Commands and DATA" generator).

2. ***The I2C Master Core's interface to GTB Bus***:

The design languages: ***VHDL***.

It is ready, implemented and fully tested with "Vulom 3". For the visualization of read DATA is used Vulom 3's LCD display.
(For the testing was designed "GTB/I2C Commands and DATA" generator).

3. ***The next steps (setup is not ready)***:

Implementation in ***Tacquila's FPGA*** (Altera ACEX 1K) and full test of the ***I2C Master Core with Interface to GTB Bus***. (only new part of the design).

Implementation in ***Tacquila Board*** and full test of the ***I2C Master Core, Full Interface to GTB Bus and Tacquila's Control Block*** (the full design).

Thank you!

Dr. Ivan Rusanov
Experimental-Electronics
GSI, Darmstadt