CALOCUBE T+ROC1 USER MANUAL

G. Martínez, J. Marín, J. Casaus. CIEMAT.

October, 2019. Firmware version v1806.

1. INTRODUCTION

This document contains a description of Calocube T+ROC1 firmware and it is intend to be use as a user manual.

2. FIRMWARE ARCHITECTURE BLOCK DIAGRAM

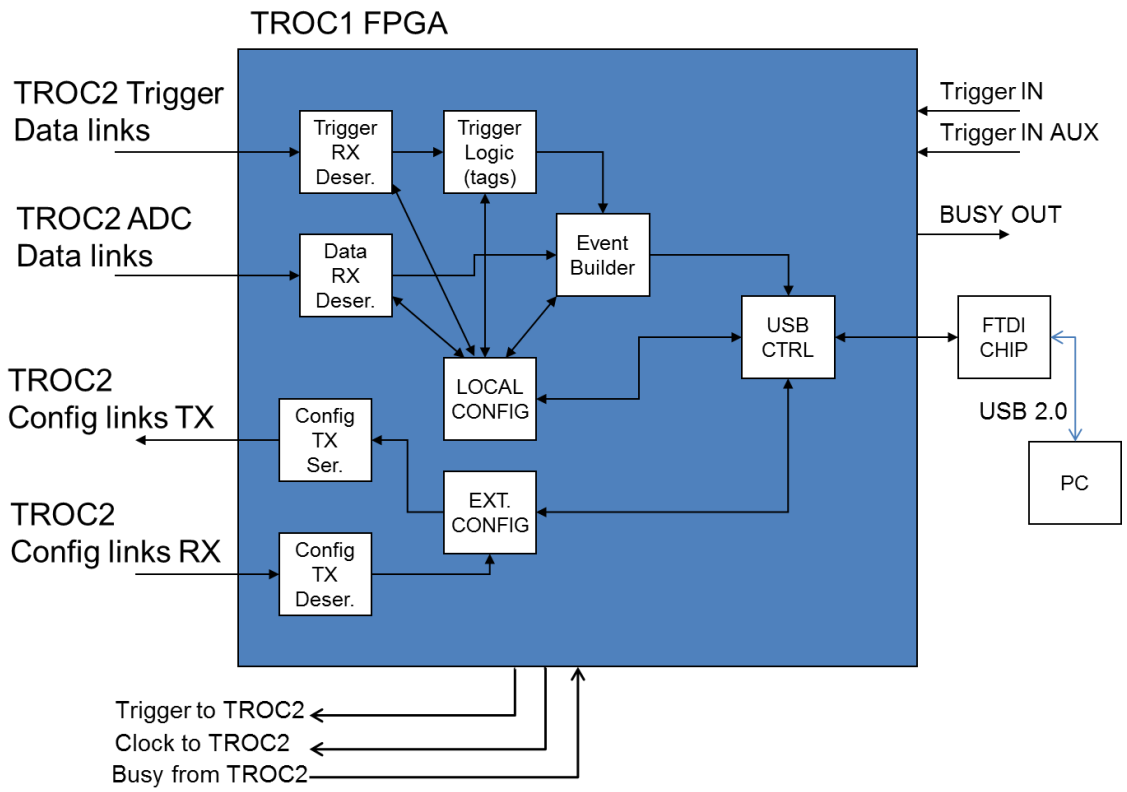


Figure 1 T+ROC1 firmware block diagram.

3. USB PROTOCOL

TROC1 uses FTDI chip FT2232H for USB Communication with a computer. This chip implements a bridge between the USB side and the FPGA side. The communication between the chip and the FPGA is base in an 8-bit wide parallel bus running at 60MHz. The software in the computer uses high-level functions in order to make USB accesses.

The FPGA implements a protocol in order to access properly to internal registers and memory. This protocol is implement in USB CTRL module (Fig. 1). The protocol is base in a header of 4 bytes in every write access; these bytes contain the base address, the number of bytes to be write or read and the Read/write bit. The format is show in table 1.

Base address(7..0) Less Significant Byte	
RD(1)/WR(0)	Base address(14..8) Most Significant Bits
Number of bytes to be RD/WR(7..0)	
Number of Bytes to be RD/WR(15..8)	

Table 1 Communication protocol command format

In order to make a write access (write from the PC into the FPGA), a FT Write function is executed containing the 4 control bytes followed by the bytes that should be written. An example is show in table 2:

0x00	
b0	0x01
0x04	
0x00	
Byte 0	
Byte 1	
Byte 2	
Byte 3	

Table 2 Writing 4 bytes in base address 0x0100

In order to make a read access (read bytes from the FPGA), a FT Write function is executed containing the 4 control bytes. Then, a FT Read function is execute in order to get the bytes requested. An example is show in table 3.

0x00	
b1	0x41
0x04	
0x00	

Table 3 Reading 4 bytes from address 0x4100

4. REGISTER AND MEMORY MAP

Address	RD/WR	Type	Description
0x00	**RD/WR**	**REG**	**Data taking enable and test trigger control**
0x01	**RD/WR**	**REG**	**Configuration TX enable**
0x02	**RD/WR**	**REG**	**Reset (FIFOs, counters)**
0x03	**RD/WR**	**REG**	**Test trigger period**
0x04	**RD/WR**	**REG**	**Test trigger period**
0x05	**RD/WR**	**REG**	**Hidra mask**
0x06	**RD/WR**	**REG**	**Hidra mask**
0x07	**RD/WR**	**REG**	**Hidra mask**
0x08	**RD/WR**	**REG**	**Hidra mask**
0x09	**RD/WR**	**REG**	**Random generator control**
0x0A	**RD/WR**	**REG**	**Random generator control**

0x0B	**RD/WR**	**REG**	**Maximum number of events in input buffer**
0x0C	**RD/WR**	**REG**	**Maximum number of events written in output buffer, lower byte (Burst number)**
0x0D	**RD/WR**	**REG**	**Maximum number of events written in output buffer, higher byte (Burst number)**
0x0E	**WR**	**REG**	**Reset of max. events in output buffer (Burst reset)**
0x20	**RD**	**REG**	**Firmware version LSB**
0x21	**RD**	**REG**	**Firmware version MSB**
0x22	**RD**	**REG**	**Configuration TX FIFO FULL**
0x23	**RD**	**REG**	**Configuration TX FIFO EMPTY**
0x24	**RD**	**REG**	**Configuration RX FIFO FULL**
0x25	**RD**	**REG**	**Configuration RX FIFO EMPTY**
0x26	**RD**	**REG**	**TX Data count Bits 7..0, WR CLK**
0x27	**RD**	**REG**	**TX Data count Bits 15..8, WR CLK**
0x28	**RD**	**REG**	**TX Data count Bits 7..0, RD CLK**
0x29	**RD**	**REG**	**TX Data count Bits 15..8, RD CLK**
0x2A	**RD**	**REG**	**RX Link 0 Data count Bits 7..0, WR CLK**
0x2B	**RD**	**REG**	**RX Link 0 Data count Bits 15..8, WR CLK**
0x2C	**RD**	**REG**	**RX Link 0 Data count Bits 7..0, RD CLK**
0x2D	**RD**	**REG**	**RX Link 0 Data count Bits 15..8, RD CLK**
0x2E	**RD**	**REG**	**RX Link 1 Data count Bits 7..0, WR CLK**
0x2F	**RD**	**REG**	**RX Link 1 Data count Bits 15..8, WR CLK**
0x30	**RD**	**REG**	**RX Link 1 Data count Bits 7..0, RD CLK**
0x31	**RD**	**REG**	**RX Link 1 Data count Bits 15..8, RD CLK**
0x32	**RD**	**REG**	**RX Link 2 Data count Bits 7..0, WR CLK**
0x33	**RD**	**REG**	**RX Link 2 Data count Bits 15..8, WR CLK**
0x34	**RD**	**REG**	**RX Link 2 Data count Bits 7..0, RD CLK**
0x35	**RD**	**REG**	**RX Link 2 Data count Bits 15..8, RD CLK**
0x36	**RD**	**REG**	**RX Link 3 Data count Bits 7..0, WR CLK**
0x37	**RD**	**REG**	**RX Link 3 Data count Bits 15..8, WR CLK**
0x38	**RD**	**REG**	**RX Link 3 Data count Bits 7..0, RD CLK**
0x39	**RD**	**REG**	**RX Link 3 Data count Bits 15..8, RD CLK**
0x3A	**RD**	**REG**	**RX Link 4 Data count Bits 7..0, WR CLK**
0x3B	**RD**	**REG**	**RX Link 4 Data count Bits 15..8, WR CLK**
0x3C	**RD**	**REG**	**RX Link 4 Data count Bits 7..0, RD CLK**
0x3D	**RD**	**REG**	**RX Link 4 Data count Bits 15..8, RD CLK**
0x3E	**RD**	**REG**	**RX Link 5 Data count Bits 7..0, WR CLK**
0x3F	**RD**	**REG**	**RX Link 5 Data count Bits 15..8, WR CLK**
0x40	**RD**	**REG**	**RX Link 5 Data count Bits 7..0, RD CLK**
0x41	**RD**	**REG**	**RX Link 5 Data count Bits 15..8, RD CLK**
0x42	**RD**	**REG**	**RX Link 6 Data count Bits 7..0, WR CLK**
0x43	**RD**	**REG**	**RX Link 6 Data count Bits 15..8, WR CLK**
0x44	**RD**	**REG**	**RX Link 6 Data count Bits 7..0, RD CLK**
0x45	**RD**	**REG**	**RX Link 6 Data count Bits 15..8, RD CLK**
0x46	**RD**	**REG**	**RX Link 7 Data count Bits 7..0, WR CLK**
0x47	**RD**	**REG**	**RX Link 7 Data count Bits 15..8, WR CLK**
0x48	**RD**	**REG**	**RX Link 7 Data count Bits 7..0, RD CLK**
0x49	**RD**	**REG**	**RX Link 7 Data count Bits 15..8, RD CLK**
0x4A	**RD**	**REG**	**Occupancy**
0x4B	**RD**	**REG**	**Event buffer data count**
0x4C	**RD**	**REG**	**Event buffer data count**

0xFF	**RD/WR**	**REG**	**Configuration TX links enables**
0x100	**WR**	**MEM**	**Configuration TX FIFO**
0x4100	**RD**	**MEM**	**Configuration RX Link 0 FIFO**
0x4900	**RD**	**MEM**	**Configuration RX Link 1 FIFO**
0x5100	**RD**	**MEM**	**Configuration RX Link 2 FIFO**
0x5900	**RD**	**MEM**	**Configuration RX Link 3 FIFO**
0x6100	**RD**	**MEM**	**Configuration RX Link 4 FIFO**
0x6900	**RD**	**MEM**	**Configuration RX Link 5 FIFO**
0x7100	**RD**	**MEM**	**Configuration RX Link 6 FIFO**
0x7900	**RD**	**MEM**	**Configuration RX Link 7 FIFO**

Table 1 – Register and memory map

5. REGISTER DESCRIPTION

Base address: x0

This register contains different control bit for data taking and trigger managing.

Bit 0. Data taking enable. When set to ‘1’ event buffer data are written in FTDI chip buffer. When set to ‘0’ RX Configuration link data are written in FTDI chip buffer.

Bit 1. When set to ‘1’ a constant frequency trigger pulse is generate. Period is control with registers 0x3 and 0x4.

Bit 2. When a transition from ‘0’ to ‘1’ is detect, a single shot test trigger is generate.

Bit 3. Data test multiplexer control. When set to ‘0’ input data from T+ROC1 are ADC data and Trigger data. When set to ‘1’, test data generated in the FPGA are use, instead of ADC data and Trigger data.

Bit 4. Random trigger enable. When set to ‘1’ a pseudorandom trigger is generate.

Bit 5. External trigger enable. When set to ‘1’ external trigger input is enabled.

Bit 6 External trigger auxiliary enable. When set to ‘1’ external Trigger auxiliary is enable

Bit 7. Unused.

Base address: x1

This register is use to enable transmission in Configuration links:

Bit 0. Global enable Configuration transmission. When set to ‘1’ data in TX buffer is sent by parallel to serial transmitter. When set to ‘0’ data is kept in the buffer. Individual data line enable for each link can be control with register 0xFF.

Bit 1 to 7. Unused.

Base address: x2

This register is use for reset different elements of internal logic;

Bit 0: FIFO reset, active high. This bit resets all FIFOs in the FPGA: Configuration, ADC data, Trigger data, Time tags and Event buffer. User should take care of setting the bit to low in order to disable the reset.

Bit 1: Counters reset, active high. This bit resets all counters of event header: Time tag (32-bit), Input Trigger (24-bit) and Accepted Trigger (24-bit). User should take care of setting the bit to low in order to disable the reset.

Bit 2 to 7: Unused

Base address: x3

This register is use to define the period of the constant rate test Trigger generator.

Base address: x4

This register is use to define the period of the constant rate test Trigger generator.

Base address: x5

This register is use to define the mask for T+ROC2 #0 and #1. The mask is active high and uses one bit per Hidra board. That is, when a certain bit is set to '0', the corresponding Hidra is readout and included in the event. When a certain bit is set to '1' the corresponding Hidra is not readout.

For Trigger data, the mask is obtained as the AND of each hexadecimal digit in the register. So Trigger data of a given TROC2 will be readout if at least one of the 4 corresponding bits is set to '0'.This means that this TROC2 is present in the system. If all the bits are set to '1', this means that the TROC2 is not connected and the corresponding Trigger data will not be readout.

Default value of all bits in the register is '0'.

Base address: x6

This register is use to define the mask for TROC2 #2 and #3. The mask is active high and uses one bit per Hidra board. That is, when a certain bit is set to '0', the corresponding Hidra is readout and included in the event. When a certain bit is set to '1' the corresponding Hidra is not readout.

For Trigger data, the mask is obtained as the AND of each hexadecimal digit in the register. So Trigger data of a given TROC2 will be readout if at least one of the 4 corresponding bits is set to '0'.This means that this TROC2 is present in the system. If all the bits are set to '1', this means that the TROC2 is not connected and the corresponding Trigger data will not be readout.

Default value of all bits in the register is '0'.

Base address: x7

This register is use to define the mask for TROC2 #4 and #5. The mask is active high and uses one bit per Hidra board. That is, when a certain bit is set to '0', the corresponding Hidra is readout and included in the event. When a certain bit is set to '1' the corresponding Hidra is not readout.

For Trigger data, the mask is obtained as the AND of each hexadecimal digit in the register. So Trigger data of a given TROC2 will be readout if at least one of the 4 corresponding bits is set to '0'.This means that this TROC2 is present in the system. If all the bits are set to '1', this means that the TROC2 is not connected and the corresponding Trigger data will not be readout.

Default value of all bits in the register is '0'.

Base address: x8

This register is use to define the mask for TROC2 #6 and #7. The mask is active high and uses one bit per Hidra board. That is, when a certain bit is set to '0', the corresponding Hidra is readout and included in the event. When a certain bit is set to '1' the corresponding Hidra is not readout.

For Trigger data, the mask is obtained as the AND of each hexadecimal digit in the register. So Trigger data of a given TROC2 will be readout if at least one of the 4 corresponding bits is set to '0'.This means that this TROC2 is present in the system. If all the bits are set to '1', this means that the TROC2 is not connected and the corresponding Trigger data will not be readout.

Default value of all bits in the register is '0'.

Base address: x9

This register is use to define de low-level byte of the random trigger generator counter.

Base address: xA

This register is use to define de high-level byte of the random trigger generator counter.

Base address: xB

This register is use to define de maximum number of events that the system can store in the input buffers. This number is defined with bits 0 to 2, that is, is limited to 7 events by firmware. The default value is 1 event. This means that when a Trigger is receive and accepted, no new Triggers will be accept until the event is completely written in the output buffer.

Base address: xC

This register is use to define de maximum number of events that TROC1 can write in the output buffer without receiving a reset from the computer. This is so-called event burst number and it is obtained by a combination of registers 0xC (lower byte) and 0xD (higher byte), so the maximum event number in a burst is 65535. When 0 is written in these registers, TROC1 writes in the output every processed event without limitation, this is normal operation. This burst limitation is intended to avoid the problem with the computer buffer managing observed in Linux computers.

Base address: xD

This register is use to define de maximum number of events that TROC1 can write in the output buffer without receiving a reset from the computer. This is so called event burst number and it is obtained by a combination of registers 0xC (lower byte) and 0xD (higher byte), so the maximum number in a burst is 65535. When 0 is written in these registers, TROC1 writes in the output every processed event without limitation, this is normal operation. This bunch limitation is intend to avoid the problem with the computer buffer managing observed in Linux computers.

Base address: xD

This register is used to perform a burst reset, any write access to this register performs the burst reset independently of the data content.

Base address: x20

This register contains Firmware version date info:

Bits 0 to 7: Day

Base address: x21

This register contains Firmware version date info:

Bits 0 to 3: Month
Bits 4 to 7: Year - 2018

Base address: x22

Bit 0: Configuration TX FIFO Full
Bits 1 to 7: unused

Base address: x23

Bit 0: Configuration TX FIFO Empty
Bits 1 to 7: unused

Base address: x24

Bit 0: Configuration RX link 0 FIFO Full
Bit 1: Configuration RX link 1 FIFO Full
Bit 2: Configuration RX link 2 FIFO Full
Bit 3: Configuration RX link 3 FIFO Full
Bit 4: Configuration RX link 4 FIFO Full
Bit 5: Configuration RX link 5 FIFO Full
Bit 6: Configuration RX link 6 FIFO Full
Bit 7: Configuration RX link 7 FIFO Full

Base address: x25

Bit 0: Configuration RX link 0 FIFO Empty
Bit 1: Configuration RX link 1 FIFO Empty
Bit 2: Configuration RX link 2 FIFO Empty
Bit 3: Configuration RX link 3 FIFO Empty
Bit 4: Configuration RX link 4 FIFO Empty
Bit 5: Configuration RX link 5 FIFO Empty
Bit 6: Configuration RX link 6 FIFO Empty
Bit 7: Configuration RX link 7 FIFO Empty

Base address: x26 to x49

This registers contain information related with data counts of configuration links FIFOs as shown in table 1.

Base address: x4A

This register contains information about the occupancy of the system, that is, the number of Triggers that have been accept but are not already released (written in the output buffer).

Base address: x4B

This register contains information about data count in output buffer:

Bits 0 to 7: Output buffer data count (bits 0 to 7)

Base address: x4B

This register contains information about data count in output buffer:

Bits 0 to 4: Output buffer data count (bits 8 to 12)

Bits 5 to 7: unused

Base address: xFF

This register is use to enable TX Configuration links individually.

Bit 0: TX Configuration link 0 enable
Bit 1: TX Configuration link 1 enable
Bit 2: TX Configuration link 2 enable
Bit 3: TX Configuration link 3 enable
Bit 4: TX Configuration link 4 enable
Bit 5: TX Configuration link 5 enable
Bit 6: TX Configuration link 6 enable
Bit 7: TX Configuration link 7 enable

6. EVENT DATA FORMAT

Event data has the following sections:

- **Header**: contains time tags and input Trigger counters:

  - Field “Event Header” is constant with value 0xEE.
  - Field “TROC1 firmware version” contains the firmware version number”
  - Field “Time tag counter” contains the 1MHz clock cycle when the accepted Triger was registered by TROC1
  - Field “Input Trigger counter” contains the count of all enabled incoming Triggers. It also counts for internally generated test Trigger, when enabled.
  - Filed “Accepted Trigger counter” contains the count of all accepted Triggers.
  - Field “Trigger enable mask” contains the enable vector for different trigger sources that is loaded in the FPGA following the next codification:

    - Bit 0: FPGA constant frequency Trigger generator enable
    - Bit 1: FPGA software Trigger enable
    - Bit 2: FPGA pseudo random Trigger generator enable
    - Bit 3: external Trigger enable

- Bit 4: external Trigger Aux. enable

- Field “Trigger type” contains the information of the Trigger source that produced the event, following the next codification:

  - Bit 0: FPGA constant frequency Trigger generator flag
  - Bit 1: FPGA software Trigger flag
  - Bit 2: FPGA pseudo random Trigger generator flag
  - Bit 3: external Trigger flag
  - Bit 4: external Trigger Aux. flag

- Field “Occupancy” contains the ordering of the event in the input buffer when the Trigger was registered, that is , if Occupancy = 1, the buffer was empty when the Trigger corresponding to the event was accepted.

- Hidra Mask: contains Hidra mask set in Configuration registers 5,6,7 and 8.

- **Trigger data**: contains data from Self Trigger outputs from Hidra chips. There is one Trigger data link per TROC2 board. It also contains a 32-bit Trigger counter that should be the same values as TROC1 “Accepted Trigger Counter”. Trigger data from each TROC2 board is protect with a 16-bit checksum.

- **Trigger logic tags**: this is the results of applying the Trigger logic to trigger data. It is included in the event as tags to crosscheck the logic by offline software.

- **ADC data**: this data Data from each Hidra board ADC, plus information about Hidra chip Gain. It also contains a 16-bit counter with TROC2 time tag, that is, the elapsed time between last reset and input Trigger in 60 Mhz clock cycle steps. Data from each Hidra board is protect with a 16-bit checksum.

- **Global checksum**: 16 bit checksum of all previous data of the event.

A representation of all data fields are show in Table 2. The minimum event data length is 31 bytes (all Hidras masked), and the maximum is 4751 bytes (all Hidra boards enabled).

Section	Field	Data	Index
Header: constant length of 22 bytes	Event header	0xEE (constant)	0
	TROC1 Firmware version	FW version(15..8)	1
		FW version(7..0)	2
	Time tag counter: 4 bytes (1us resolution)	Time tag(31..24)	3
		Time tag(23..16)	4
		Time tag(15..8)	5
		Time tag(7..0)	6
	Input Trigger Counter: 4 bytes	Input Trigger(31..24)	7
		Input Trigger(23..16)	8
		Input Trigger(15..8)	9
		Input Trigger(7..0)	10
	Accepted Trigger Counter: 4 bytes	Accepted Trigger(31..24)	11
		Accepted Trigger(23..16)	12
		Accepted Trigger(15..8)	13
		Accepted Trigger(7..0)	14
	Trigger enable mask	x"0" & Enable mask(3..0)	15
	Trigger type	x"0" & Trigger type(3..0)	16
	Occupancy	x"0" & Occupancy(3..0)	17
	Hidra mask: 4 bytes	Mask(7..0)	18
		Mask(15..8)	19
		Mask(23..16)	20
		Mask(31..24)	21
Trigger Data: 22 bytes/TROC2 Min: 0 TROC2 Min: 0 bytes Max: 8 TROC2 Max 176 bytes	TROC2 N (22 bytes)	Hidra 0, ASIC 1, CH(7..0)	
		Hidra 0, ASIC 2, CH(7..0)	
		…	
		Hidra 3, ASIC 4, CH(7..0)	
		Trigger Counter(7..0)	
		Trigger Counter(15..8)	
		Trigger Counter(23..16)	
		Trigger Counter(31..24)	
		Trigger Checksum(7..0)	
		Trigger Checksum(15..8)	+22
	TROC2 M (22 bytes)	Hidra 0, ASIC 1, CH(7..0)	
		Hidra 0, ASIC 2, CH(7..0)	
		…	
		Hidra 3, ASIC 4, CH(7..0)	
		Trigger Counter(7..0)	
		Trigger Counter(15..8)	
		Trigger Counter(23..16)	
		Trigger Counter(31..24)	
		Trigger Checksum(7..0)	
		Trigger Checksum(15..8)	+22
Trigger Logic tags: constant length of 7 bytes	Multiplicity (2 bytes)	Multiplicity(15..8)	
		Multiplicity(7..0)	
	X projection	X projection(7..0)	
	Y projection	Y projection(7..0)	
	Z projection(3 bytes)	Z projection(23..16)	
		Z projection(15..8)	
		Z projection(7..0)	+7

ADC data: 142bytes/Hidra board Min: 0 Hidra Min: 0 bytes Max: 32 Hidra Max:4544 bytes	**Hidra Board X (142 bytes)**	**0xBB**	
		Hidra Board number	
		ASIC 1, CH0, LSB	
		ASIC 1, CH0, MSB	
		…	
		ASIC 1, CH15, LSB	
		ASIC 1, CH15, MSB	
		ASIC 2, CH0, LSB	
		ASIC 2, CH0, MSB	
		…	
		ASIC 2, CH15, LSB	
		ASIC 2, CH15, MSB	
		ASIC 3, CH0, LSB	
		ASIC 3, CH0, MSB	
		…	
		ASIC 3, CH15, LSB	
		ASIC 3, CH15, MSB	
		ASIC 4, CH0, LSB	
		ASIC 4, CH0, MSB	
		…	
		ASIC 4, CH15, LSB	
		ASIC 4, CH15, MSB	
		Gain ASIC 1, CH(7..0)	
		Gain ASIC 1, CH(15..8)	
		…	
		Gain ASIC 4, CH(7..0)	
		Gain ASIC 4, CH(15..8)	
		TROC2 time tag(7..0)	
		TROC2 time tag(15..8)	
		ADC Data Checksum(7..0)	
		ADC Data Checksum(15..8)	**+142**
	Hidra Board Y (142 bytes)	**0xBB**	
		Hidra Board number	
		ASIC 1, CH0, LSB	
		ASIC 1, CH0, MSB	
		…	
		ASIC 1, CH15, LSB	
		ASIC 1, CH15, MSB	
		ASIC 2, CH0, LSB	
		ASIC 2, CH0, MSB	
		…	
		ASIC 2, CH15, LSB	
		ASIC 2, CH15, MSB	
		ASIC 3, CH0, LSB	
		ASIC 3, CH0, MSB	
		…	
		ASIC 3, CH15, LSB	
		ASIC 3, CH15, MSB	
		ASIC 4, CH0, LSB	
		ASIC 4, CH0, MSB	
		…	
		ASIC 4, CH15, LSB	
		ASIC 4, CH15, MSB	

		Gain ASIC 1, CH(7..0)	
		Gain ASIC 1, CH(15..8)	
		…	
		Gain ASIC 4, CH(7..0)	
		Gain ASIC 4, CH(15..8)	
		TROC2 time tag(7..0)	
		TROC2 time tag(15..8)	
		ADC Data Checksum(7..0)	
		ADC Data Checksum(15..8)	**+142**
Global checksum (2 bytes)		**Global checksum(7..0)**	
		Global checksum(15..0)	**+2**

Table 2 – TROC1 data format

7. EXAMPLE OF DAQ PROGRAM INIT

This section contains an example of a typical initialization of the data acquisition program.

This is the sequence that should be implemented:

- FTDI USB session initialization
- TROC1 Reset
- TROC1 Configuration buffers initialization
- TROC2 “Hold Delay” setting
- TROC2 “Hold Gain Delay” setting
- TROC1 constant frequency internal trigger generator period setting
- TROC1 Hidra Mask setting
- TROC2 RS232 enable and Trigger enable
- TROC1 Trigger enable setting and Data Taking enable

7.1. FTDI USB session initialization

Initialize USB session using standard FTDI functions:

- FT_Create_Device_Info_list
- FT_Get_Device_Info_List
- FT_Open_Device_By_Serial_Number (From this moment it is assumed that an FThandle is available to manage FT accesses)
- FT_Set_Bit_Mode (Single Channel 245 Synchronous FIFO)
- FT_Set_Timeouts (Timeout time should be greater than maximum time between consecutive triggers in order to avoid writting zeroes between two events in output data file)
- FT_purgue (this is equivalent to make an FT_Read of 65536 bytes)

7.2. T+ROC1 Reset

This is intended to reset TROC1 FPGA internal buffers, event building process and counters. It consists of writing 0x03 in TROC1 register 0x02 in order to activate the resets, and then write 0x00 in the same register in order to disable the resets.

```
TxBuffer[0] = 0x02; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
```

```
TxBuffer[2] = 0x01;//TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; // TROC1 number of bytes(15..8)
TxBuffer[4] = 0x03; //Content of register

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);

TxBuffer[0] = 0x02; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
TxBuffer[2] = 0x01; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; // TROC1 number of bytes(15..8)
TxBuffer[4] = 0x00; //Content of register

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.3. T+ROC1 Configuration links enable

This is need in order to enable the TX of Configurationuration accesses from TROC1 to TROC2. Register 0x01 contains a global enable for all buffers. Register 0xFF contains one enable per Configurationuration link (TROC2)

```
TxBuffer[0] = 0x01; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
TxBuffer[2] = 0x01;//TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; // TROC1 number of bytes(15..8)
TxBuffer[4] = 0x01; //Content of register

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);

TxBuffer[0] = 0xFF; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
TxBuffer[2] = 0x01;//TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; // TROC1 number of bytes(15..8)
TxBuffer[4] = TROC2_Configurationuration_mask; //Content of register

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.4. TROC2 “Hold Delay” setting

Nedded for writing Hold Delay parameter in TROC2 FPGA

```
TxBuffer[0] = 0x00; //TROC1 base address(7..0)
TxBuffer[1] = 0x01; //TROC1 base address(15..8)
TxBuffer[2] = 0x06; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; //TROC1 number of bytes(15..8)
TxBuffer[4] = 0x00; //TROC2 control register address
TxBuffer[5] = 0x80; //TROC2 control register content
TxBuffer[6] = 0x26; //TROC2 data register address
TxBuffer[7] = Hold_Delay_lower_byte;// TROC2 data register content
TxBuffer[8] = 0x27; // TROC2 data register address
TxBuffer[9] = Hold_Delay_higher_byte; // TROC2 data register content


FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.5. TROC2 "Hold Gain Delay" setting

Nedded for writing Hold Gain Delay parameter in TROC2 FPGA

```
TxBuffer[0] = 0x00; //TROC1 base address(7..0)
TxBuffer[1] = 0x01; //TROC1 base address(15..8)
TxBuffer[2] = 0x06; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; //TROC1 number of bytes(15..8)
TxBuffer[4] = 0x00; //TROC2 control register address
TxBuffer[5] = 0x80; //TROC2 control register content
TxBuffer[6] = 0x24; //TROC2 data register address
TxBuffer[7] = Hold_Gain_Delay_lower_byte;// TROC2 data register content
TxBuffer[8] = 0x25; // TROC2 data register address
TxBuffer[9] = Hold_Gain_Delay_higher_byte; // TROC2 data register content


FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.6. TROC1 constant frequency internal trigger generator period setting

This is need in order to use TROC1 FPGA Trigger generator.

```
TxBuffer[0] = 0x03; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
TxBuffer[2] = 0x02; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; //TROC1 number of bytes(15..8)
TxBuffer[4] = Trigger_period_lower_byte; //TROC1 register content
TxBuffer[5] = Trigger_period_higher_byte; //TROC1 register content

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.7. TROC1 Hidra Mask setting

Writing of Hidra Mask 32-bit word to TROC1 registers. This defines the length of the event.

```
TxBuffer[0] = 0x05; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
TxBuffer[2] = 0x04; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; //TROC1 number of bytes(15..8)
TxBuffer[4] = Hidra_Mask_byte0; //TROC1 register content
TxBuffer[5] = Hidra_Mask_byte1; //TROC1 register content
TxBuffer[6] = Hidra_Mask_byte2; //TROC1 register content
TxBuffer[7] = Hidra_Mask_byte3; //TROC1 register content


FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.8. TROC2 RS232 enable and Trigger enable

This access is need in order to enable Trigger in TROC2. The full content of the register is overwrite, so it is important to provide the desired setting for the other control bits in the same register, according to the following map (see TROC2 description register):

Bit 0: 0 sequence reset / 1 normal operation.
Bit 1: GAIN2SEL.
Bit 2: 0 RS232 reset active / 1 RS232 normal operation.
Bit 3: 0 external trigger / 1 internal trigger (for test purposes)
Bit 4: 0 normal operation / 1 internal trigger counter reset
Bit 5: Used for test purposes. Should be 0 during normal operation
Bit 6: Used for test purposes. Should be 0 during normal operation
Bit 7: 1 Configuration / 0 data taking

For normal operation with RS232 scalers enabled the content of the register should be:

TROC2_Reg0=0x05;

For normal operation with RS232 enabled and GAIN1 forced, the content of the register should be:

TROC2_Reg0=0x07;

```
TxBuffer[0] = 0x00; //TROC1 base address(7..0)
TxBuffer[1] = 0x01; //TROC1 base address(15..8)
TxBuffer[2] = 0x02; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; //TROC1 number of bytes(15..8)
TxBuffer[4] = 0x00; //TROC2 control register address
TxBuffer[5] = TROC2_Reg0; //TROC2 control register content

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

7.9. TROC1 Trigger enable setting and Data Taking enable

This defines the enabled Trigger sources in TROC1 and enables data readout by writing in TROC1 register 0, with the following codification:

Reg0, bit 0: data taking enable
Reg0, bit 1, constant rate Trigger enable
Reg0, bit 2: single shot soft Trigger enable
Reg0, bit 3, test data enable
Reg0, bit 4, random trigger enable
Reg0, bit 5: external Trigger enable
Reg0, bit 6, auxiliary external Trigger enable

For normal operation with both external Triggers enabled, the content of the register should be:

TROC1_Reg0=0x61;

```
TxBuffer[0] = 0x00; //TROC1 base address(7..0)
TxBuffer[1] = 0x00; //TROC1 base address(15..8)
TxBuffer[2] = 0x01; //TROC1 number of bytes(7..0)
TxBuffer[3] = 0x00; //TROC1 number of bytes(15..8)
TxBuffer[4] = TROC1_Reg0; //TROC1 register content

FT_Write(ftHandle, TxBuffer, sizeof(TxBuffer), &BytesWritten);
```

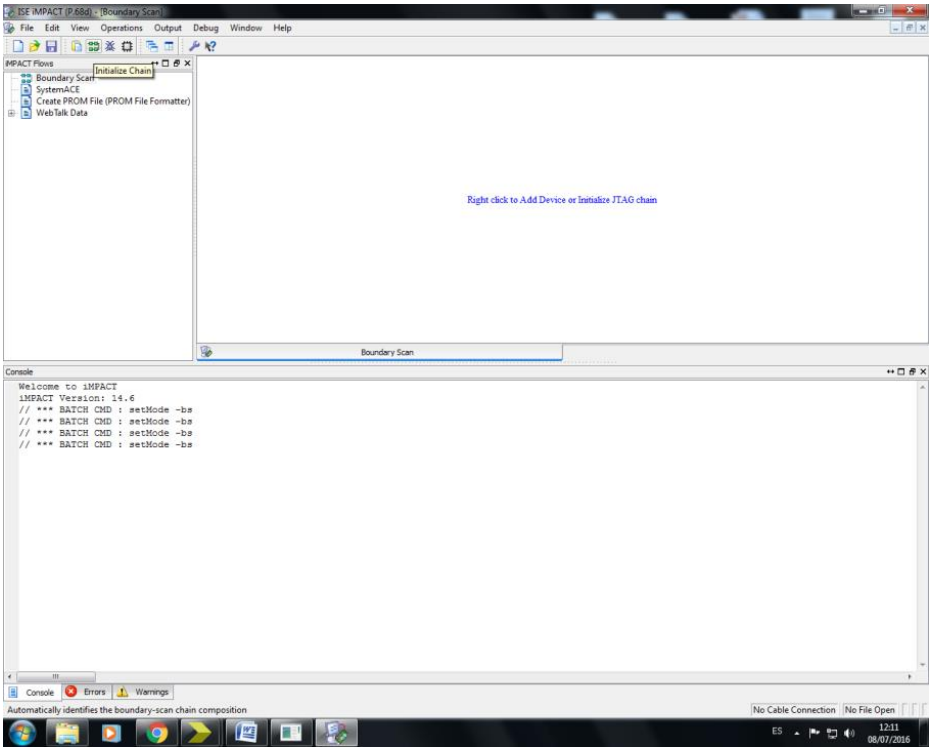
From this moment data will be accessible in USB after Trigger arrival.

8. TROC1 FIRMWARE UPDATE PROCEDURE

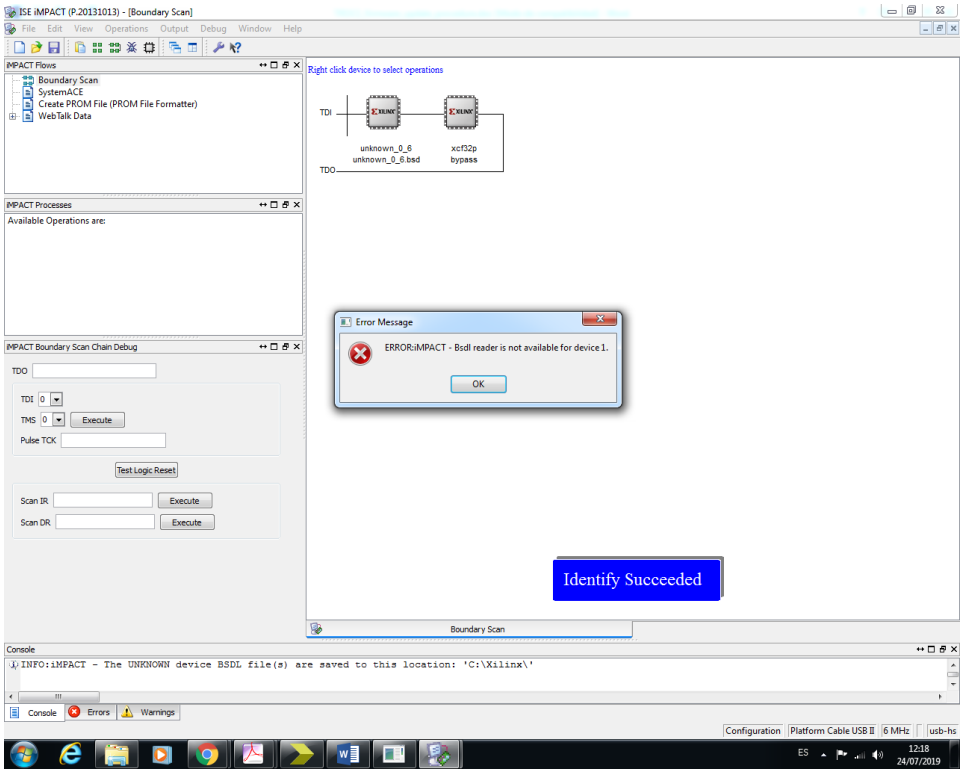
This section contains a description of the procedure for updating TROC1 non-volatile memory by using Xilinx USB programming cable and Impact software.

8.1. Connect Xilinx USB programming cable to programming connector at TROC1, J1.

8.2. Launch Impact and initiate JTAG Chain

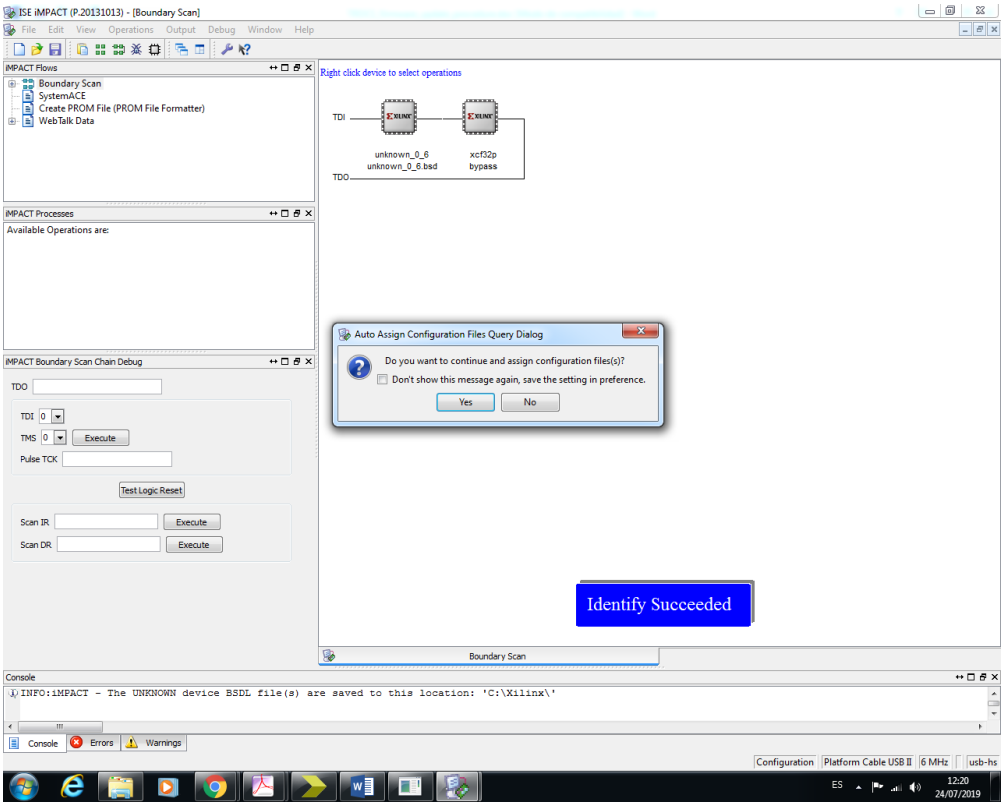


You should see the following chain:



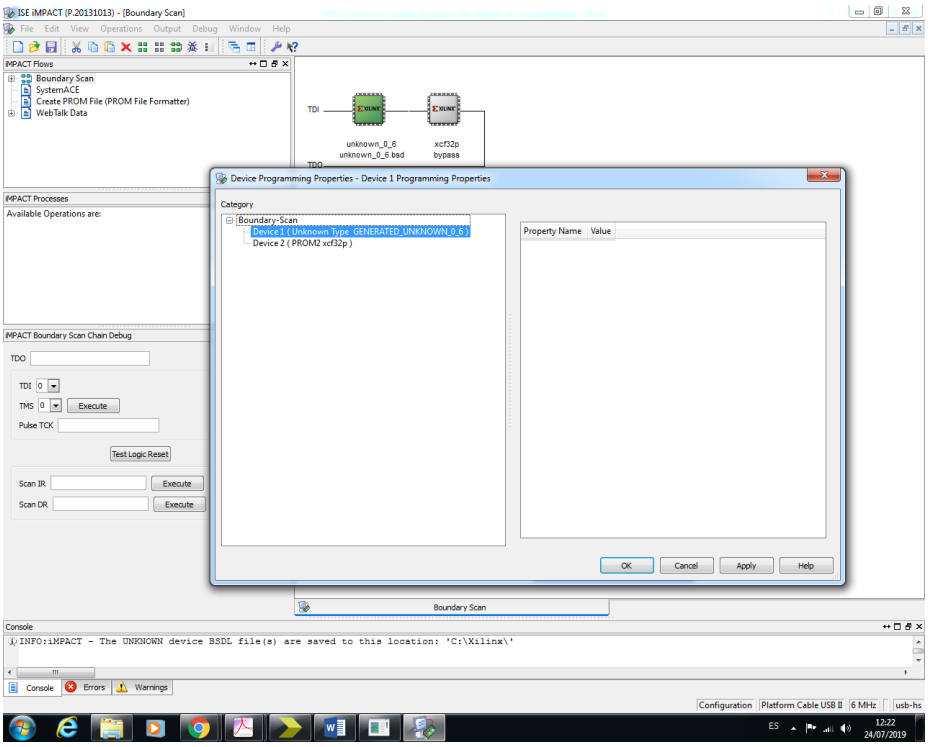
The error message is because Impact software does not have the internal JTAG description of Spartan 7 FPGA. You should provide .BSD file to avoid this issue.

Click “OK” in the error message window. You should see the following message:



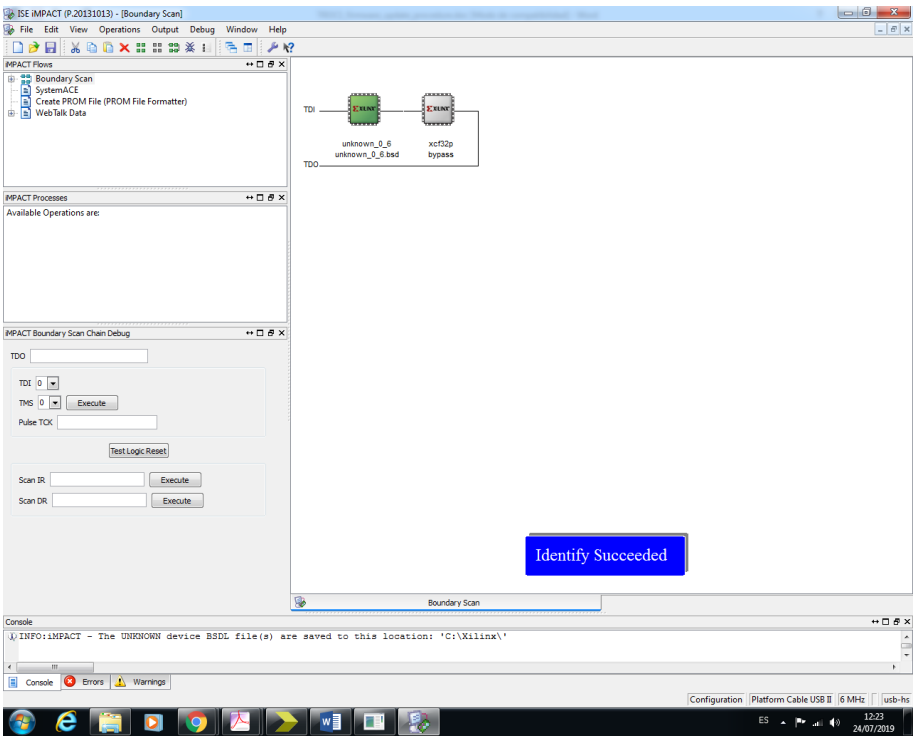
Click “No”

You should see the following window:

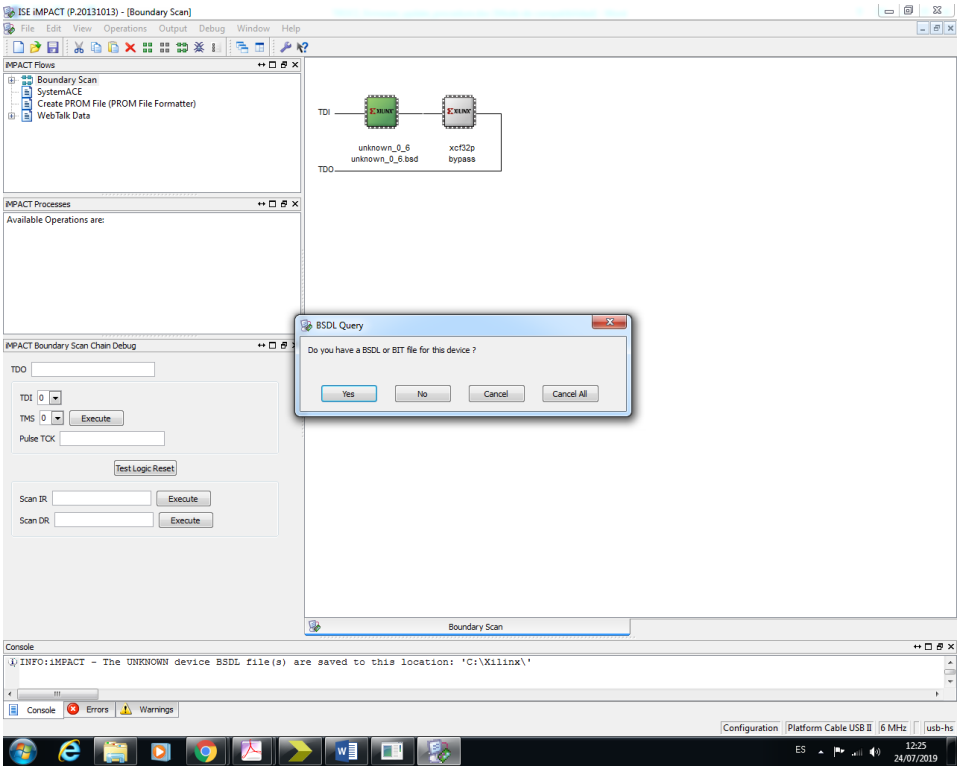


Click “Cancel”

Then you should see the JTAG chain:



8.3. Right Click on the first part in the chain XCF16P, this is the Spartan 7 FPGA. Then select "Assign new Configurationuration file", an explorer window will arise so you can select the .BSD file.

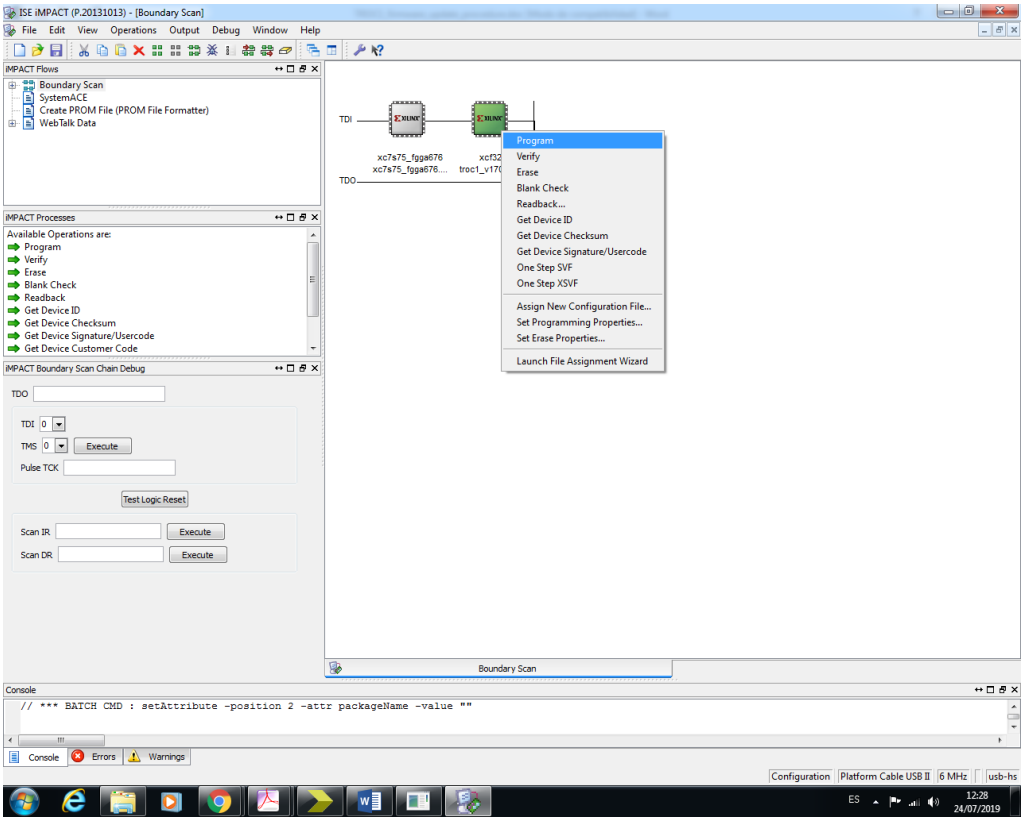


Assign the file XC7S75_FGGA676.BSD

8.4. Right Click on the second part in the chain XCF32P, this is the non volatile flash memory. Then select “Assign new Configurationuration file”, an explorer window will arise so you can select the .mcs file.

A warning message may pop up before assigning the file, alerting you about byte swapping in the memory file, this is normal, click “OK”.

8.5. Once the file is assigned, right click in the second part again and select the option “Program”



8.6. After few minutes you will get a “Programming Succeeded” message, this means everything went right. The file is write in the Flash memory and will be loaded in the FPGA in every power cycle.