Bonn TT Unit

Specification

1.6

Prepared	M. Elberich / C.Connot Name	2014/07/24 Date
Approved	N. Surname Name	yyyy/mm/dd Date
Released	N. Surname Name	yyyy/mm/dd Date

TABLE OF CONTENTS

Change Record

Issue	Date	Section/ Paragraph Affected	Reasons / Remarks	Name
1.0	24.11.2010	All	created	Elberich
1.1	07.07.2011	All		Elberich
1.1	09.06.2011	5.1.2, 6.1.1, 6.1.2		Elberich
1.2	22.09.2011	6.1.1, 6.1.2		Elberich
1.3	09.02.2012	All	revised	Connot
1.4	21.05.2012	6	revised	Connot
1.5	21.01.2013	All	minor changes	Connot
1.6	24.07.2014	p. 21	idle mode description	Connot

1 Scope

This document describes the Bonn TT Unit.

2 Applicable documents

No.	Title	Number & Issue
AD 1		
AD 2		

3 Reference documents

No.	Title	Number & Issue
RD1	ARGOS-FDR-018, Tip-Tilt Control Design	
RD2	SPCM-AQ4C data sheet	
RD3	Diamond Systems Helios User manual	Revision B3

4 Introduction

The Bonn TT Unit is used to count TTL pulses from the Perkin Elmer APD module that is used for the APD quad-cell system of the TT sensing system. For each integration interval the Bonn TT Unit computes the corrected counting values, the centroid and finally de-rotates the centroid. The output includes the corrected and de-rotated x- and y-coordinate.

The device consists of two components: The first one is a PC/104 CPU board with a x86 compatible Vortex86 microcomputer running under Linux. It keeps the ethernet connection to the ARGOS network, allows the remote control of the Bonn TT Unit and grants access for maintenance.

The second component is a FPGA board for counting and computing in real-time. It transmits the data via a serial RS422 interface to the TT Slope BCU.

Both components are connected via an ISA-bus.

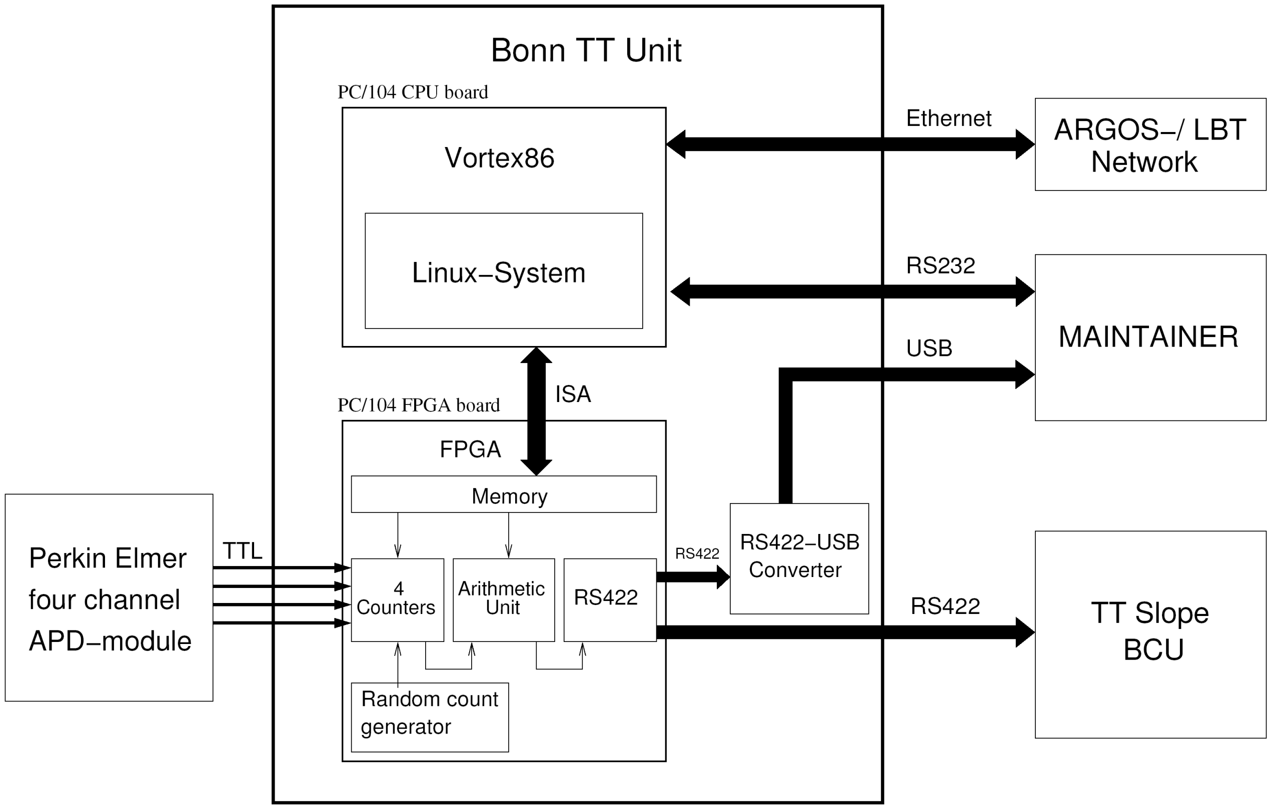


Figure 1: Bonn TT Unit block diagram

The arithmetics is done in three steps. The first step is the correction of the counted values in terms of linearity, dark count and photon detection efficiency. The formula is taken from the Perkin Elmer SPCM-AQ4C data sheet:

$$C_c = \frac{\dfrac{C}{1 - C \cdot t_d} - C_{dark}}{P_d}$$

C_c = corrected number of counts per sec
C = number of counts per sec
C_{dark} = dark counts per sec
t_d = dead time
P_d = photon detection efficiency

In the second step the centroid is computed from the corrected number of counts of all four APDs:

$$x = \frac{(C_{c2} + C_{c4}) - (C_{c1} + C_{c3})}{\sum_{n=1}^{4} C_{cn}}$$

$$y = \frac{(C_{c1} + C_{c2}) - (C_{c3} + C_{c4})}{\sum_{n=1}^{4} C_{cn}}$$

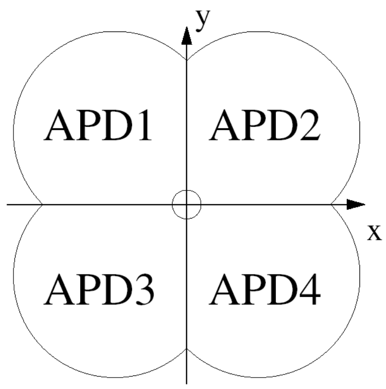


Figure 2: Adjustment of APDs (see RD1)

C_{cn} is the corrected number of counts in the n^{th} APD in the coordinate plane.

The third step is the de-rotation:

$$\begin{pmatrix} x_{rot} \\ y_{rot} \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

α is the rotation angle. Positive angles will lead to a rotation counterclockwise.

4.1 Reference values

- Maximum sampling rate: 2kHz
- Overall latency: $\leq 70\mu s$
- Rotation angle update interval: < 10s => angle error < 1° at field rotation velocity of 0.133°/s
  - Maximum jitter for each interval: < 1s
- Maximum counting value: 65535 counts/integration time

4.2 Coordinate system

x- and y-coordinates are calculated as 16 bit values. Unrotated coordinates range from -23170 to 23170. After the derotation the range is from -32767 to 32767.

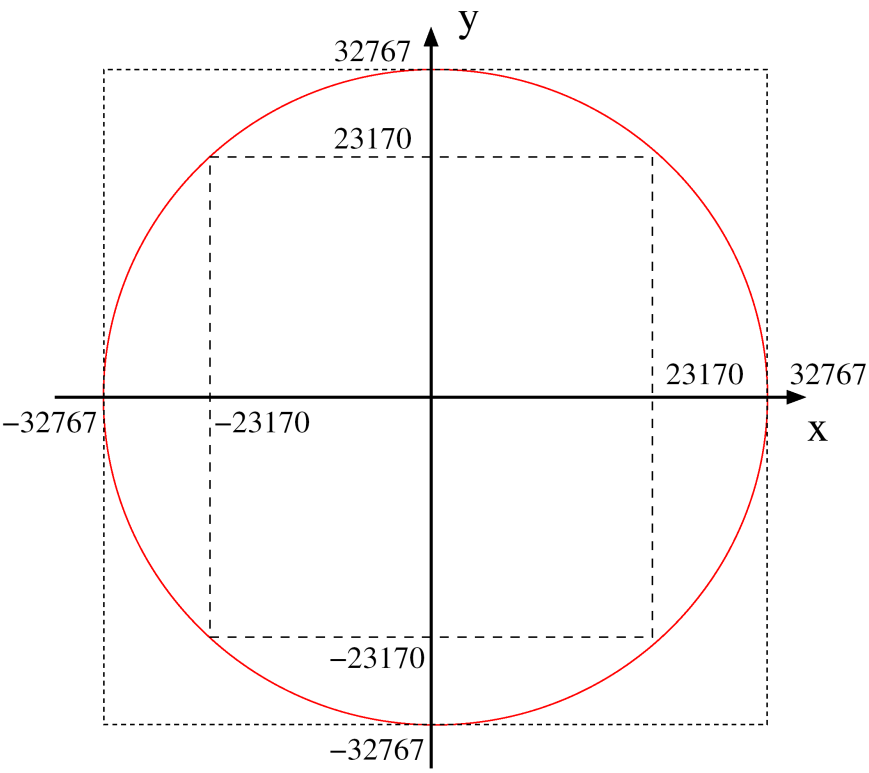


Figure 3: coordinate system

4.3 Dynamic range

Figure 4 shows the lenslet-array in front of the APDs (1 to 4) illuminated with the light beam of diameter **d** in different positions (A to F). The calculated unrotated x- and y-coordinates range from -23170 to 23170. The beam diameter maps this range to the dimension of the lenslet-array.
Case **A**: the beam is outside of the array. No valid centroid can be calculated due to no detected photons.
Case **B**: part of the beam illuminates APD2. x- and y-coordinate is clipped to 23170 because of all photons are detected on quadrant 2.
Case **C**: photons are detected by APD1 and APD2. The x-coordinate is calculated and the y-coordinate is clipped to 23170.
Case **D**: photons are detected by APD2 and APD4. The y-coordinate is calculated and the x-ccordinate is clipped to 23170.
Case **E**: all photons are detected by APD2. x- and y-ccordinate is correctly calculated as 23170. This case produces the same result as case B!
Case **F**: photons are detected by all four APDs. x- and y-coordinate are calculated without clipping.

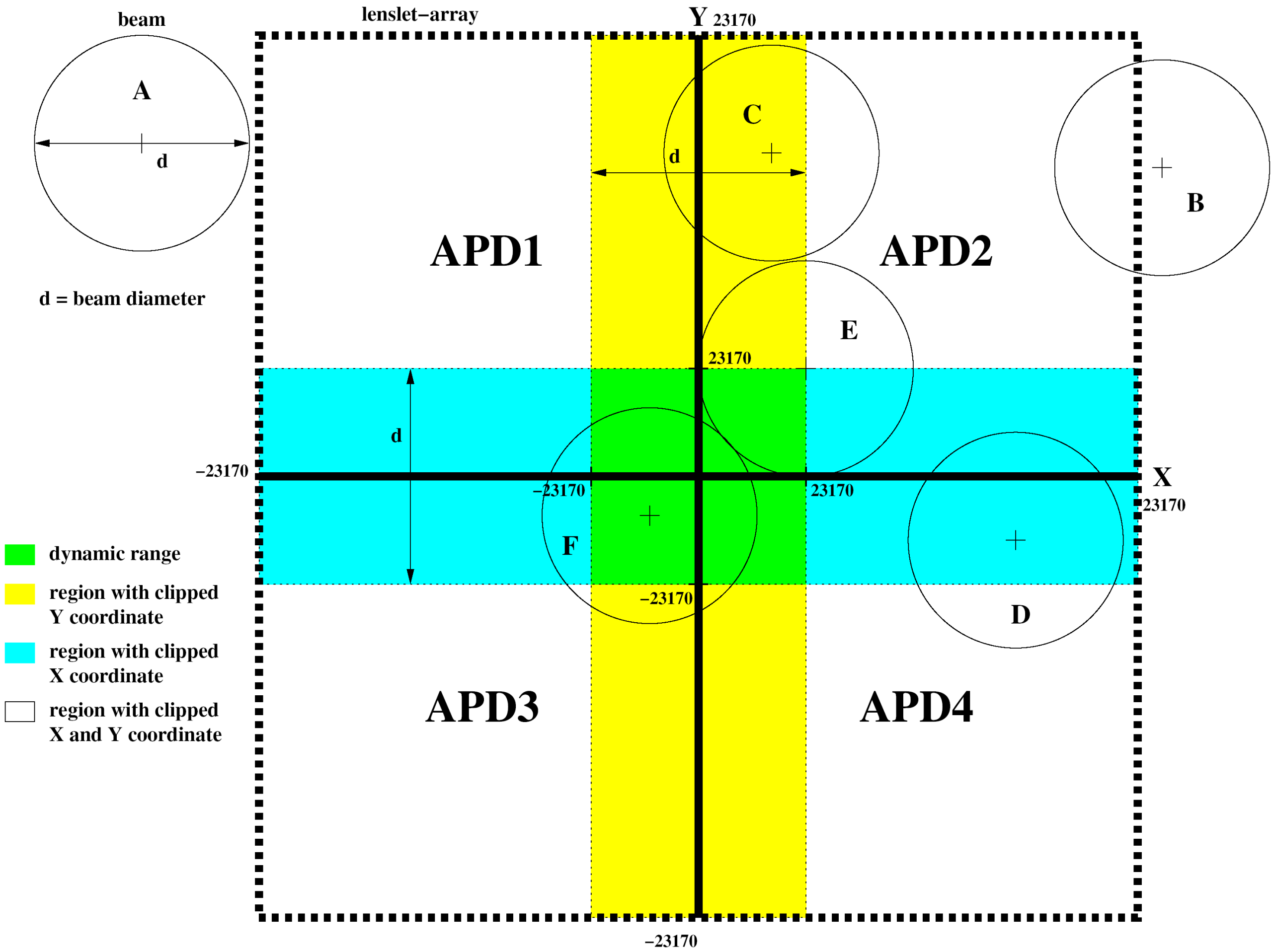


Figure 4: dynamic range (without rotation)

5 Data Input/ Output

5.1 Input

5.1.1 Number of photons

The Bonn TT Unit counts the number of pulses from the Perkin Elmer SPCM-AQ4C. Each photon is represented by a TTL pulse that is 4.5 Volt high and 25ns wide.

5.1.2 Parameters

The Bonn TT Unit requires several parameters which can be changed by the user via Ethernet. If no parameters are set the device will use a set of default parameters. For correct naming and syntax see 6.2.1 (Ethernet/ICE interface). The parameters are as follows:

- ***Integration time (t_i) [µs]*** (default 1000)

The duration of a single counting interval. It can be used to trade off the number of counts against the sampling rate. The minimum for integration time is 500µs, the maximum value is 4.000.000µs.

- ***Dark count for each APD (C_{dark}) [1/s]*** (default 500)

The number of dark counts of one APD in one second. For short integration times this parameter might be negligible but it might vary with temperature. According to the Perkin Elmer SPCM-AQ4C data sheet this value should not exceed 500/s. Apart from that the product of dark count and integration time must not exceed 4096 as the FPGA logic can not handle larger numbers. This allows a darkcount up to 1000/s at 4s integration time.

- ***Dead time for each APD (t_d) [ns]*** (default 50)

Each APD has a dead time of several nanoseconds after each detected photon. Within this time no other photon can be detected. The chance to miss photons increases with the count rate. This leads to a non-linearity. The parameter is used to correct this. The value is typically about 50ns according to the Perkin Elmer SPCM-AQ4C data sheet. The FPGA logic can handle values up to 122ns.

- ***Photon detection efficiency for each APD (P_d) [%]*** (default 50)

The APDs should be light sensitive at wavelength from 400nm to 1000nm. Within this range the detection efficiency varies from 10% to 60% according to the data sheet of the SPCM-AQ4C. The Bonn TT Unit allows setting photon detection efficiencies from 1.6% to 100%.

- ***Minimum counting value [1/s]*** (default 0)

Depending on integration time and dark count there is a minimum value of counts that must be achieved to get a reasonable result. If the sum of counts of all four APDs is below this minimum counting value the Bonn TT Unit will set the “Low-Count”-Bit in the status-nibble to “1” (see chapter 5.2.4).

- ***Rotation angle [rad]*** (default 0)

Rotation angle of the telescope. The centroid is derotated by this angle. A positiv value leads to a rotation counter clockwise.

- ***Zero angle [rad]*** (default 0)

Additional to the rotation angle of the telescope, there might be an offset angle that is caused by a misalignment of the optical apertures. It is considered in the calculation as an offset for the rotation angle α. A positiv value leads to a rotation counter clockwise.

5.1.3 Commands

The control software is based on the ICE (internet communication engine). To communicate with the Bonn TT Unit command server, you have to use the functions declared in the ICE interface file(see 6.2.1).

- ***Mode***
  run, stop or idle. For explanation see 6.3
- ***Shutdown***
  Shuts down the CPU board.
- ***Reboot***
  Reboots the CPU board.
- ***Init***
  Sets the Bonn TT Unit parameters to the software defined defaults.
- ***Reset FPGA***
  Resets the FPGA and sets the parameters to the hardware defined defaults.
- ***Reload FPGA***
  Forces the FPGA to reload its firmware from the serial PROM.
- ***Get FPGA version***
  Returns the version number of the FPGA firmware.
- ***Get APD overcurrent***
  If the APD module is overilluminated the security circuit will switch off the module until it is reset. This is indicated by the APDovercurrent bit.
- ***Set APD power***
  Turns on/off the power for the APD module.
- ***Set APD reset***
  Resets the security circuit of the APD module power supply.
  .

Additional features:

- ***Countsource***
  It is possible to feed the calculation unit of the FPGA with 3 different sources:
  APD: the pulses from the Perkin Elmer SPCM-AQ4C module
  random: hardware generated random number of counts
  dummy: user defined number of counts
- ***Powersupply voltages***
  The supply voltages of the Bonn TT Unit and Potsdam part can be measured.
  on Bonn TT Unit: +5.0V, +3.3V, +2.5V, +1.2V
  on Potsdam part: +30.0V, +2.0V, +5.0V
- ***Bitrate***
  The transmission bitrate to the slope BCU can be set in bit/s. The bitrate is corrected to the nearest possible bitrate.
- ***Get temperature***
  Measures the temperature on the FPGA board.
- ***Reset image number***
  Resets the counter for the image number.
- ***FPGA version***
  Returns the version number of the FPGA firmware.

5.2 Output

5.2.1 Parameters

The parameters mentioned in 5.1.2 that can be set by ethernet can also be requested by ethernet. They will be dumped together with the corresponding image number in "/root/tts/soft/tmp/tts_setup_dump.txt" on the CPU board's filesystem.

5.2.2 X- and Y-coordinates

The coordinates are transmitted in realtime by a serial RS422 interface to the BCU.

The coordinates are represented by a 16 bit integer value (-32768 … 32767):

$$\begin{pmatrix} x_{out} \\ y_{out} \end{pmatrix} = \frac{32767}{\sqrt{2}} \cdot \begin{pmatrix} x_{rot} \\ y_{rot} \end{pmatrix} \quad x_{rot}, y_{rot} \in [-1...1] \in \Re$$

The division by $\sqrt{2}$ is made to downscale the x- and y-coordinates to an absolute maximum (+/-) 23170, so they can be rotated in the coordinate plane.

5.2.3 Counting Values

The raw counting values are transmitted in realtime by a serial RS422 interface to the BCU.

5.2.4 Status Information

The status information nibble is transmitted in realtime by a serial RS422 interface to the BCU.

The meanining of the status bits are:

Bit 1 (MSB)	Unused
Bit 2	1=counter overflow
Bit 3	Unused
Bit 4 (LSB)	1=Low Count

The overflow bit is set to '1' if one of the four counter is feed with more than 65535 pulses per integration time. The low count bit is set to '1' if the sum of all four counter values is below the specified value for the parameter minimumcount.

6 Specifications

6.1 Layout

The hardware is designed to find place in the foreseen room of the rack for the TT System as a separate mountable module. It consists of a combination of a PC/104 board, a FPGA board and a RS-422 to USB converter for debugging purposes. All parts are mounted on a common base plate (See Figure 5).

The drawings below show overviews of the hardware and the postions of all connectors. The hardware has additional plugs for HF-, analog- and general purpose I/O signals, which may be used for future updates. It can also be populated with LEDs and switches which are not visible to keep the graphics clear.

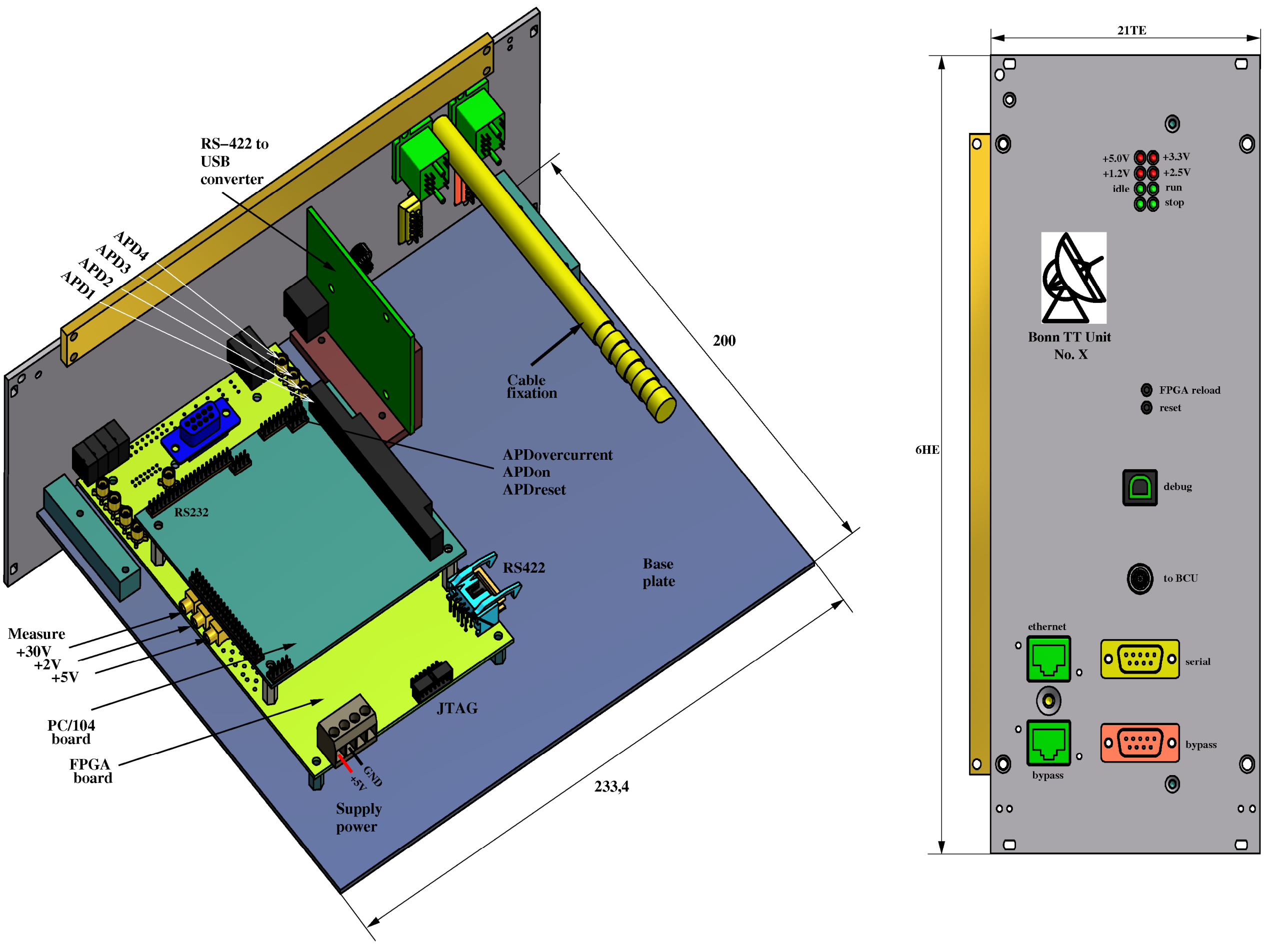


Figure 5: The Bonn TT Unit

6.2 Interface specifications

6.2.1 Ethernet/ICE interface

The Ethernet interface allows access to the Linux machine via SSH. The configuration of the network interface resides in “/etc/network/interfaces” on the CPU board’s filesystem.

In addition the Bonn TT Unit software interacts with the ARGOS software via ICE. The init-script for the ICE command server running on the CPU board “/etc/init.d/tts” can be called with start/stop/restart. The following methods are implemented:

```
idempotent void init() throws TipTiltException;

idempotent void setMode(TTMode mode) throws TipTiltException;
idempotent TTMode getMode() throws TipTiltException;

idempotent void setParameters(ApdParameters parameters) throws TipTiltException;
idempotent ApdParameters getParameters() throws TipTiltException;

idempotent void setIntegrationTime(long integrationTimeInMicrosecond) throws TipTiltException;
idempotent long getIntegrationTimeInMicrosecond() throws TipTiltException;

idempotent void setZeroAngle(double zeroAngleInRadian) throws TipTiltException;
idempotent double getZeroAngleInRadian() throws TipTiltException;
idempotent void setRotateAngle(double rotateAngleInRadian) throws TipTiltException;
idempotent double getRotateAngleInRadian() throws TipTiltException;

idempotent void setMinimumCounts(int counts) throws TipTiltException;
idempotent int getMinimumCounts() throws TipTiltException;

idempotent void setCountSource(CountSource source) throws TipTiltException;
idempotent CountSource getCountSource() throws TipTiltException;

idempotent void setDummyCounts(DummyCounts counts) throws TipTiltException;
idempotent short getDummyCounts() throws TipTiltException;

idempotent void setBitRate(float rate) throws TipTiltException;
idempotent short getBitaRate() throws TipTiltException;

idempotent void shutdown() throws TipTiltException;
idempotent void reboot() throws TipTiltException;

idempotent PowerSupplyVoltage getPowerSupplyVoltage() throws TipTiltException;
idempotent float getTemperature() throws TipTiltException;
idempotent void resetFPGA() throws TipTiltException;
idempotent void programFPGA() throws TipTiltException;
idempotent void resetImageNumber() throws TipTiltException;

idempotent string getFPGAVersion() throws TipTiltException;

idempotent int getAPDovercurrent() throws TipTiltException;
idempotent void setAPDpower(ttsStateOnoff state) throws TipTiltException;
idempotent void setAPDreset() throws TipTiltException;
```

The following data types are defined:

```
exception TipTiltException
{
    string message;
};

struct PowerSupplyVoltage
{
    float bonn1V2;
    float bonn2V5;
    float bonn3V3;
    float bonn5V0;
    float potsdam30V0;
    float potsdam2V0;
    float potsdam5V0;
};
```

```
struct ApdParameters
{
        float darkcountsPerSecondAPD1;
        float darkcountsPerSecondAPD2;
        float darkcountsPerSecondAPD3;
        float darkcountsPerSecondAPD4;
        float deadTimeInNanosecondAPD1;
        float deadTimeInNanosecondAPD2;
        float deadTimeInNanosecondAPD3;
        float deadTimeInNanosecondAPD4;
        float photoDetectorEfficiencyAPD1;
        float photoDetectorEfficiencyAPD2;
        float photoDetectorEfficiencyAPD3;
        float photoDetectorEfficiencyAPD4;
};

struct DummyCounts
{
        short counts1;
        short counts2;
        short counts3;
        short counts4;
};

enum TTMode
{
        run, idle, stop
};

enum CountSource
{
        ttsapd, ttsrandom, ttsdummy
};

enum ttsStateOnoff
{
        on,off
};
```

6.2.2 RS-422 interface

6.2.2.1 General

The Bonn TT Unit transmits its data via a serial RS-422 interface to the slope BCU.

The requirements of the Bonn TT Unit say that the delay caused by computation and transmission of the X-Y-coordinates must not exceed 50 µs. To fulfill this it uses the maximum bit rate that RS-422 is specified for, which is 10 Mbit/s.
Modification:
The transmission bitrate is set to the value of **7.1429 Mbit/s** to be compatible with the slope BCU RS-422 receiver. This leads to a delay of about **70 µs**.

6.2.2.2 Electrical Specification

According to the RS-422 standard the Bonn TT Unit uses a differential driver for transmission (TX+ and TX-). A logical '1' means TX+ is high level and TX- is low level, a logical '0' means TX+ is low level and TX- is high level. The minimum differential voltage on receiver (BCU) side is 2V.

TX+ is connected to the RX+ of the BCU on pin 6 of the 7-way Fischer 102 jack.
TX- is connected to the RX- of the BCU on pin 5 of the 7-way Fischer 102 jack.

If the transmission line is idle, it is held on a logical '1'.

The used connector is a 7-pin connector from Fischer 102 series. It is identical to the one on the BCU side. The pin layout is given in Table 1. See also APPENNDIX (4).

Pin	**Function**
1	Not connected
2	Not connected
3	Tx-
4	Tx+
5	Rx-
6	Rx+
7	GND

Table 1: pin layout of RS-422 cable

The RS-422 cable is a cross-link-cable. It is produced by Microgate and provided by MPIfR.

6.2.2.3 Protocol Specification

Coding:

The data send to the BCU is converted into ASCII characters. Therefore each byte is separated into nibbles, and instead of these nibbles an 8-bit ASCII character is transmitted, that represents the hexadecimal value of the nibble. The table shows the mapping.

Nibble	**Character**	**ASCII-Decimal-Value**	**ASCII-Binary Value**
0000	0	48	0011 0000
0001	1	49	0011 0001
0010	2	50	0011 0010
0011	3	51	0011 0011
0100	4	52	0011 0100
0101	5	53	0011 0101
0110	6	54	0011 0110
0111	7	55	0011 0111
1000	8	56	0011 1000
1001	9	57	0011 1001
1010	A	65	0100 0001
1011	B	66	0100 0010
1100	C	67	0100 0011
1101	D	68	0100 0100
1110	E	69	0100 0101
1111	F	70	0100 0110

Transmission Package:

All data is transmitted in one package. It is introduced with the ASCII character 'T' (decimal 84) as a header. Next there are 33 characters of user data, two characters for a checksum, and a “Carriage Return” (decimal 13) and a “Line Feed” (decimal 10) indicates the completion of the transmission. The table shows the whole contents of the frame.

Character No.	
0	Header character ('T')
1	Status information
2 – 9	Frame number
10 – 13	X-coordinate
14 – 17	Y-coordinate
18 – 21	APD 1 counter value
22 – 25	APD 2 counter value
26 – 29	APD 3 counter value
30 – 33	APD 4 counter value
34 – 35	Checksum
36	'CR'
37	End of package 'LF'

Table 2: contents of transmission packet

Calculation of checksum
The checksum is computed as the sum of character 0 – 33 of the related package. The lower 8 bit of the result are extracted from the result, converted into the corresponding two ASCII characters and then transmitted.

Character Transmission
Each character is transmitted independently as a stream of 10 Bits, with a start- and stop-bit, eight data bits and no parity bit. The data is transmitted with LSB first and MSB last. This is compliant to the serial RS-232 standard. Each bit is 100ns wide at a bit rate of 10 Mbit/s.

Modification:
Each bit is 140ns wide at a bit rate of 7.1429Mbit/s.

Bit No.	Description:
0	Start Bit (Logical '0')
1	Data LSB 0
2	Data Bit 1
3	Data Bit 2
4	Data Bit 3
5	Data Bit 4
6	Data Bit 5
7	Data Bit 6
8	Data MSB 7
9	Stop Bit (Logical '1')

Example

Status = 0 = $(0000)_2$
Frame number = 3600000 = $(0000\ 0000\ 0011\ 0110\ 1110\ 1110\ 1000\ 0000)_2$
x-coordinate = 5265 = $(0001\ 0100\ 1001\ 0001)_2$
y-coordinate = -10531 = $(1101\ 0110\ 1101\ 1101)_2$
Counter 1 = 1000 = $(0000\ 0011\ 1110\ 1000)_2$
Counter 2 = 1500 = $(0000\ 0101\ 1101\ 1100)_2$
Counter 3 = 2500 = $(0000\ 1001\ 1100\ 0100)_2$
Counter 4 = 4000 = $(0000\ 1111\ 1010\ 0000)_2$

Frame:

- 1 character for header ('T')
- 1 character for status information
- 8 characters for the frame-number
- 4 characters for x-coordinate
- 4 characters for y-coordinate
- 4x4 characters for counter 1-4
- 2 characters for the checksum
- 1 character carriage return
- 1 character line feed

=> 38 Characters

Transmission:

	Head-er	Stat.	Frame-Number								x-coordinate				y-coordinate			
Bi-nary		0000	0000	0000	0011	0110	1110	1110	1000	0000	0001	0100	1001	0001	1101	0110	1101	1101
Char	T	0	0	0	3	6	E	E	8	0	1	4	9	1	D	6	D	D
Ascii Dec	84	48	48	48	51	54	69	69	56	48	49	52	57	49	68	54	68	68
Ascii Bi-nary	0101 0100	0011 0000	0011 0000	0011 0000	0011 0011	0011 0110	0100 0101	0100 0101	0011 1000	0011 0000	0011 0001	0011 0100	0011 1001	0011 0001	0100 0100	0011 0110	0100 0100	0100 0100

	Counter 1				Counter2				Counter 3				Counter 4				Checksum		CR	LF
Bi-nary	0000	0011	1110	1000	0000	0101	1101	1100	0001	0011	1000	0100	0000	1111	1010	0000				
Char	0	3	E	8	0	5	D	C	0	9	C	4	0	F	A	0	A	3	CR	LF
Ascii Dec	48	51	69	56	48	53	68	67	48	57	67	52	48	70	65	48	65	51	13	10
Ascii Bi-nary	0011 0000	0011 0011	0100 0101	0011 1000	0011 0000	0011 0101	0100 0100	0100 0011	0011 0000	0011 1001	0100 0011	0011 0100	0011 0000	0100 0110	0100 0001	0011 0000	0100 0001	0011 0011	0000 1101	0000 1010

Checksum: 84+48+48+48+51+54+69+69+56+48+49+52+57+49+68+54+68+68+48+51+69+56+48+53+68+67+48+57+67+52+48+70+65+48= 1955 = 0x7A3 → Cheksum = A3

Bitstream:

The LSB of each character is transmitted first.

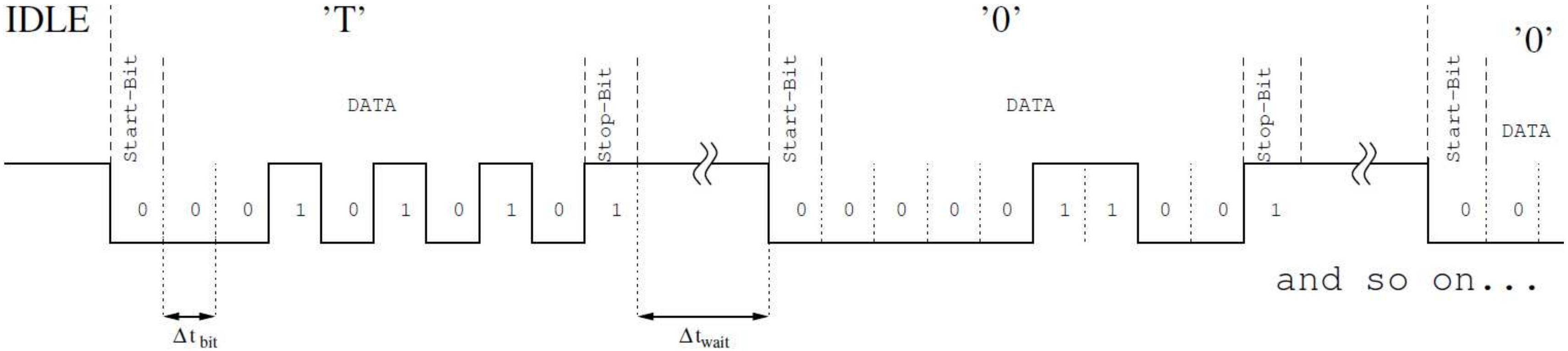


Figure 6: Low level protocol

Total package contains: 380 Bit = 38 character
Δt_{bit} = 100 ns @ 10 Mbit/s.
Δt_{wait} = 100 ns @ 10 Mbit/s (the delay is one clock cycle).
Modification:
Δt_{bit} = 140 ns @ 7.1429 Mbit/s.
Δt_{wait} = 140 ns @ 7.1429 Mbit/s (the delay is one clock cycle).

6.2.3 APD Module connectors

The APDs are connected to the Bonn TT Unit by MCX connectors. They are placed on the FPGA-board. It is only accessable if the TT System is disassembled.

6.2.4 RS-232 interface

The Bonn TT Unit provides a RS-232 interface for a login to the Linux CPU board by a terminal program (Minicom, HyperTerminal...). This interface can be used for maintenance when the unit is unreachable by network. It is placed on the front panel of the Bonn TT Unit. In normal operation the front panel connectors are jumpered/bypassed to the internal wiring going to the N-Port server inside the TT Cassette. For debugging it is possible to pull the jumper and connect directly to the serial port.
The table below shows the cabling from the CPU board (see RD3 connector J8) to the SUBD connector on the front panel. The second shows the cable going from the bypass port to the NPort server.

CPU board side J8	Colour	Front panel (SUBD-9)
3 RXD1	white/black	3
5 TXD1	brown/black	2
9 GND	Black	5

Front panel bypass	colour	RJ45 plug
3 RXD1	blue	4
2 TXD1	orange	5
5 GND	violett	3

6.2.5 Ethernet interface

The ethernet interface placed on the front panel is jumper/bypassed to the internal ethernet switch inside the TT cassette. For debugging it is possible to pull the jumper and connect directly to the ethernet port. The table below shows the cabling from the CPU board (see RD3, J11) to the RJ45 connector on the front panel.

CPU board side J11	Colour	Front panel RJ45
1 TX+	orange/white	1
2 TX-	Orange	2
4 RX-	Green	6
5 RX+	green/white	3

6.2.6 USB interface

To read out the RS-422 data packages without using the BCU, the Bonn TT Units provides an USB interface which sends the same contents. The transmission speed is 2 MBit/s. It is placed on the front panel of the Bonn TT Unit and can be connected for debugging. The receive process source code is not part of the official software but can be found in “/root/tts/soft/src/tts_receive_process_rs422.c” on the CPU board’s filesystem. Copy it from there to the receiving computer, modify your local copy if you want and compile it. Run the receive process with the USB port which is connected to the Bonn TT Unit.

The table below shows the cabling from the FPGA board connector J316 to the RS-422 to USB converter.

FPGA board side J316	colour	RS-422 to USB converter side
10 TX-	grey	SUBD-9: 4
9 TX+	grey	SUBD-9: 3
6 GND	grey	SUBD-9: 5

6.2.7 Power Supply

The Potsdam power supply, which powers the APD modules, also offers 5V/2A to the Bonn TT Unit. It will be connected by a terminal clamp which is placed on the backside of the module. From there the power supply is connected to the CPU board (see RD3, connector J4).

FPGA board side J602	colour
1 +5V	red
2 GND	black

FPGA board side J603	colour	CPU board side J4
1 +5V	red	1 +5V
2 GND	black	2 GND

6.2.8 JTAG Interface

To update the FPGA firmware version there is a JTAG interface accessible from the backside of the Bonn TT Unit.

6.2.9 Power supply measurements

The power supplies of the APD module (+30V/GND,+2V/GND,+5V/GND) are measured by a I2C ADC device on the FPGA board. The connectors on the FPGA board side are MCX. It is mandatory to connect the right voltage to the right input connector to prevent damage to the ADC inputs!

6.2.10 Additional signals

Certain of the digital I/O signals of the CPU board are used to monitor and control the power managment of the APD module. Therefore the port D is defined in the BIOS settings as output port. Port E is defined as an input port in the BIOS settings. The outputs on Port D are 3.3V at a logic '1' and 0V at a logic '0'. The input on port E is a high impedance. The table below shows the cabling of the additional signals.

CPU board side J7	Colour
1 D0, output APDpower	Green
2 D1, output APDreset	Yellow
9 E0, input APDovercurrent	Blue
17 +5V	Red
18 GND	Black

APDpower switches the powersupply of the APD module. After powering the TT Unit cassette the APD module power supply is off but the Bonn TT Unit is powered and booting. After boot the APDpower signal has to be set to "on" to switch on the APD modules power supply.
In case of overillumination of the APDs the module is switched off by a hardware circuit and resides in a safe state until reset. The APDovercurrent bit is set to '1'. Software can poll this bit at low frequency to indicate overcurrent.

When overcurrent/overillumination has occurred the illumination of the APD module has to be reduced. Then the APD can be reseted via software (see 6.2.1). After calling the reset function the APDreset pin will go to ‘0’ for 1 second.

6.3 Behavioral description

The Bonn TT Unit supports three operational modes: run, stop and idle. It does not need any information about the closed-loop or anything else of ARGOS except the current rotation angle.

6.3.1 Mode “run”

After power-on or after setting the mode to “run” the Bonn TT Unit enters the run-mode. In this state it counts, calculates and transmits the results, the raw counting values and the status infomation via RS-422 to the BCU. The device gets the de-rotation angle from the telescope in a constant interval of 10 s and includes it in the calculation.
If the sum of counts of all four APDs is below the “minimum counting value”, the “low count” bit of the status nibble is set to ‘1’. If there is an overflow of one counter the “overflow” bit of the status nibble is set to ‘1’. The x- y-coordinates are send anyway.
In run-mode the device allows the user to set and request all parameters. The latency for those parameters to take effect is below 3s. A change of the integration time takes effect on all four APDs synchronously and may take effect on a running integration interval.

If the mode is set from run to stop or idle, the device will always complete the last integration interval and send the last package before it leaves the run mode.
During run mode the user has the ability to switch between different data sources. By default the Bonn TT Unit takes the counting values of the APD counters to compute the centroid, but it can also generate random counting values for testing purposes. In addition the test counts can be set to dummy values by the user.

6.3.2 Mode “stop”

After setting the mode to “stop” the Bonn TT Unit enters the stop-mode. In this state the device does not count pulses and it does not send anything via the RS-422 interface. It allows the user to set and request all parameters. If no parameters are set the default values are used.

6.3.3 Mode “idle”

After setting the mode to “idle” the Bonn TT Unit enters the idle-mode. In this mode the device sends the corrected counts but constantly sends the coordinates [0,0] after each integration time. The device allows the user to set and request all parameters.

6.4 Arithmetic error

As seen in chapter 4 the counting value of one APD has to be corrected with this formula:

$$C_c = \frac{\dfrac{C}{1 - C \cdot t_d} - C_{dark}}{P_d}$$

As described in chapter 5.1.2 each APD has a non-linear characteristic, which is a result of the dead time t_d. The term $\frac{C}{1 - C \cdot t_d}$ is used to correct this.

This division is unfavourable to be calculated on the FPGA. The following approximation is used instead:

$$\frac{C}{1 - C \cdot t_d} \approx C \cdot (1 + C \cdot t_d) \qquad C \cdot t_d << 1$$

Because of this the Bonn TT Unit always makes a certain arithmetic error, which is affected by the number of counts and the integration time. In addition rounding errors may occure, because the Bonn TT Unit uses only integer arithmetics.

Figure 7 shows the Bonn TT Unit compensation compared to no compensation. And Figure 8 shows the corresponding arithmetic error of one single APD for an integration time of 1ms. The right side diagrams show zoomed areas of the left side diagrams.

For all plots the following parameters are used:

- Deadtime: t_d = 50 ns
- Darkcount: C_{Dark} = 500/s
- Photon detection efficiency: P_d = 100%

The error E in Figure 8 is:

$$E = \frac{C_C - C_{C,BTTU}}{C_C} \cdot 100$$

C_C = properly corrected number of counts
$C_{C,BTTU}$ = number of counts, corrected by the Bonn TT Unit

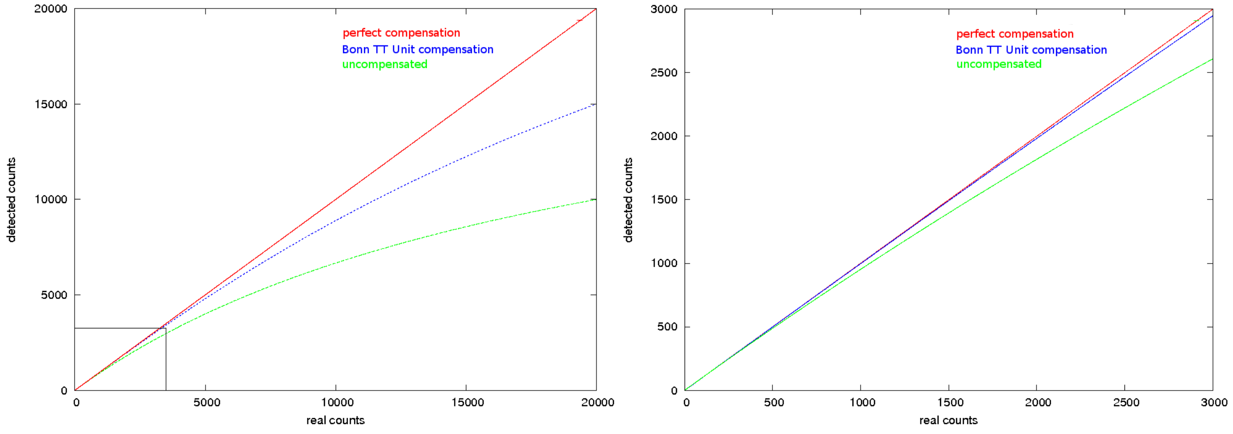


Figure 7: Compensation of a single APD at 1 ms integration time

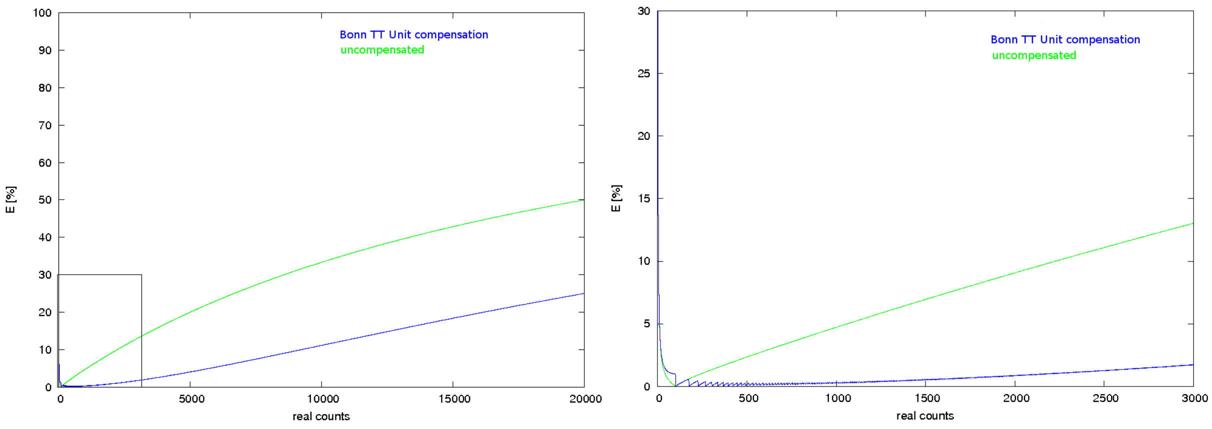


Figure 8: Arithmetic error of a single APD at 1 ms integration time

Figure 8 shows that about 10 counts are needed to keep the error below 5%.

The green uncompensated curve shows at 100 counts that the error produced by deadtime compensates the error produced by darkcount. Therefore the calculated error there is zero. Below 100 counts the error due to uncompensated dark counts dominates while above 100 counts the error due to dead time dominates.

The ripple seen in the blue curve Figure 8 (right side) results from the integer arithmetics of the Bonn TT Unit. The error rise to 100 % if counts drop to 1.

The error made by the calculation of the centroid and the rotation is neglegible.

7 Use cases sunny

7.1 Remote startup

- Power-on APDs
- Power-on Bonn TT Unit
- Bonn TT Unit will start sending frames
- Wait until getMode-function replies “run”

7.2 Remote shutdown

- The Bonn TT Unit can be rebooted with the reboot-command
- The Bonn TT Unit can be shutdown with the shutdown- command
- Power can be switched off via the Potsdam power supply if the Bonn TT Unit is unreachable. This should only be used in emergency case to prevent damage to the CPU board’s Linux system.

7.3 Set parameters

- Dark count for APD 1-4
- Dead time for APD 1-4
- Photon detection efficiency for APD 1-4
- Integration time
- Minimum count value
- Rotation angle
- Zero angle

All parameters can be set and changed at any time.

The parameters dark count, dead time, photon efficiency and integration time are necessary for operation. If they are not set, default parameters are used.

The parameters zero angle, dead time and photon efficiency should be more or less static. A change of these parameters should not be necessary under usual circumstances.

The rotation angle will be cyclicaly provided to the Bonn TT Unit.

The parameter dark count might vary with temperature.

The delay of parameters to take effect in run-mode is <1s. In between or during the running integration time interval the Bonn TT Unit uses the last valid parameters.

The parameters dark count, dead time and photon efficiency have to be set synchronously for all four APDs to prevent illegal states. Thus the setParameter-command (see6.2.1) always sets all parameters, no matter whether they were changed or not.

7.4 Get parameters

These parameters can be requested at any time by sending the related command via ICE

- Dark count for APD 1-4
- Dead time for APD 1-4
- Photon detection efficiency for APD 1-4
- Integration time
- Minimum count value
- Zero angle
- Rotation angle
- Current state

7.5 Stop

To stop the Bonn TT Unit the mode is set to “stop”. The Unit can be set to idle-mode before to ensure that the last coordinates send are [0, 0]. The related function is the setMode function.

7.6 Data dump

The Bonn TT Unit transmits the raw counting values of all four APDs to the slope BCU. At the beginning of each observation, and after each change of parameters the Bonn TT Unit stores a set of parameters together with the first framenumber from which they take effect in a local file on the Linux machine (/root/tts/soft/tmp/tts_setup_dump.txt). These data allow a complete reconstruction of all computed values of the Bonn TT Unit.

7.7 Dark count calibration

<To be clarified by Jesper Storm>

7.8 Dead time calibration

<To be clarified by Jesper Storm>

7.9 Photon detection efficiency calibration

<To be clarified by Jesper Storm>

7.10 Zero angle calibration

<To be clarified by Jesper Storm>

7.11 Acquisition

- Calibration and usage of APD parameters
- Bonn TT Unit must be in run-mode
- Adjustment of APD count rate with integration time

8 Use cases rainy

8.1 Bonn TT Unit sending bad values

- Check if APD module is powered/reseted
- Check if Low Count Bit or Overflow bit is set
- Check if parameters are right
- Switch unit from start- mode to stop-mode and back to start-mode
- Reboot

- Shutdown and Power cycle

8.2 Bonn TT Unit unreachable via network

- Connect via RS-232 and login as user “root” to fix problems on the Linux machine.
  - Check interface configuration (command: “ifconfig”)
  - Check configuration files (file: “/etc/network/interfaces”)
  - Restart ICE command server (command: “/etc/init.d/tts restart”)
- Reboot
- Shutdown and Power cycle

8.3 Bonn TT Unit gets no rotation angle via network

- Unit will use the last valid rotation angle until an actual one is set.

APPENDIX (1): List of acronyms

APD	Avalanche Photo Diode
ARGOS	Advanced Rayleigh guided Ground layer adaptive Optics System
ASCII	American Standard Code for Information Interchange
BCU	Basic Computational Unit
BIOS	Basic Input Output System
CPU	Central Processing Unit
FDR	Final Design Report
FPGA	Field Programmable Gate Array
HF	High Frequency
ICE	Internet Communication Engine
ISA	Industry Standard Architecture
LBT	Large Binnocular Telescope
LSB	Least Significant Bit
MCX	Miniature CoaX
MSB	Most Significant Bit
PROM	Programmable Read Only Memory
SSH	Secure Shell
TBD	To Be Defined
TT	Tip Tilt
TTL	Transistor-Transistor Logic
USB	Universal Serial Bus

APPENDIX (2): PCB layout and jumper settings

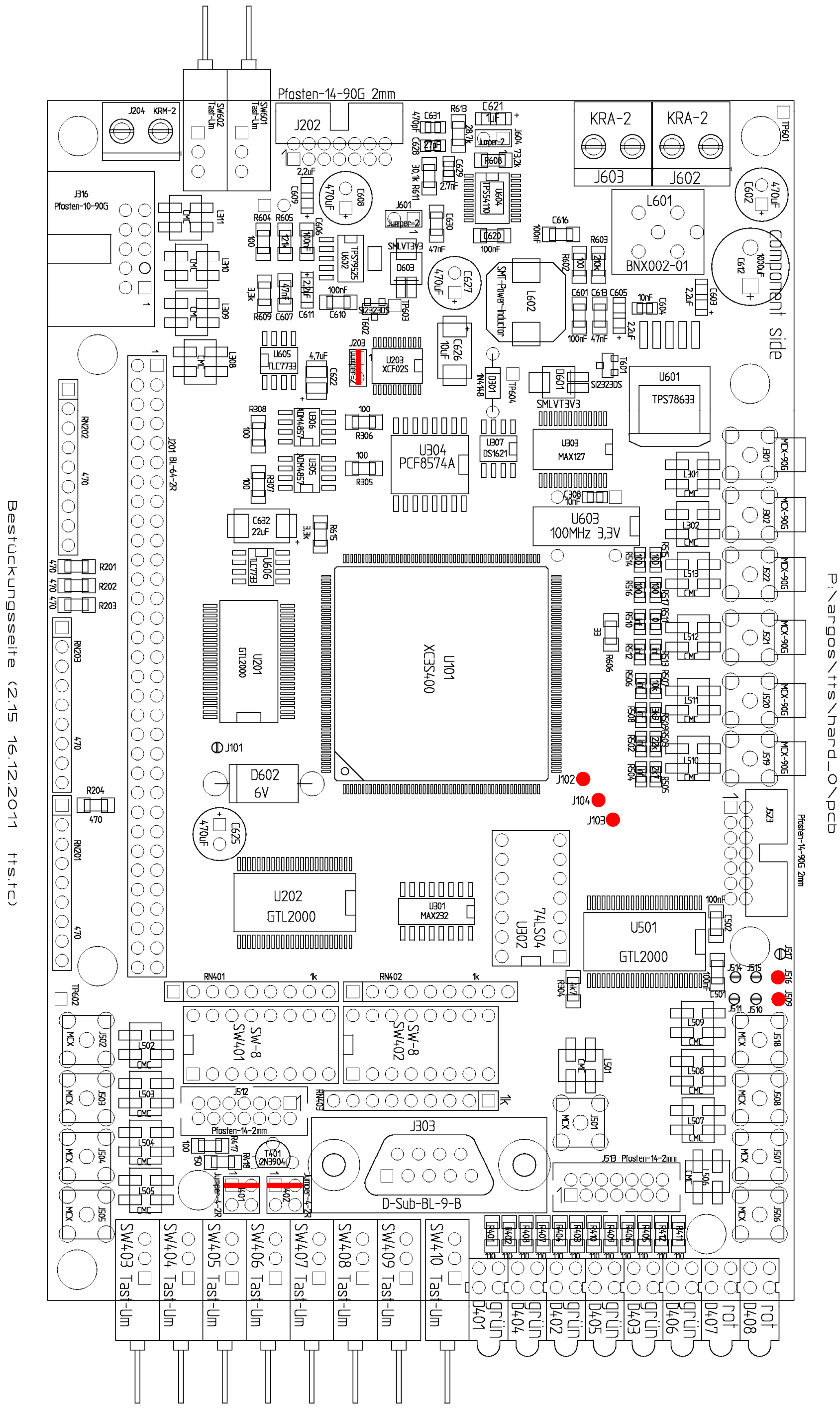


Figure 9: Assembly diagram including jumper settings (red=closed), component side

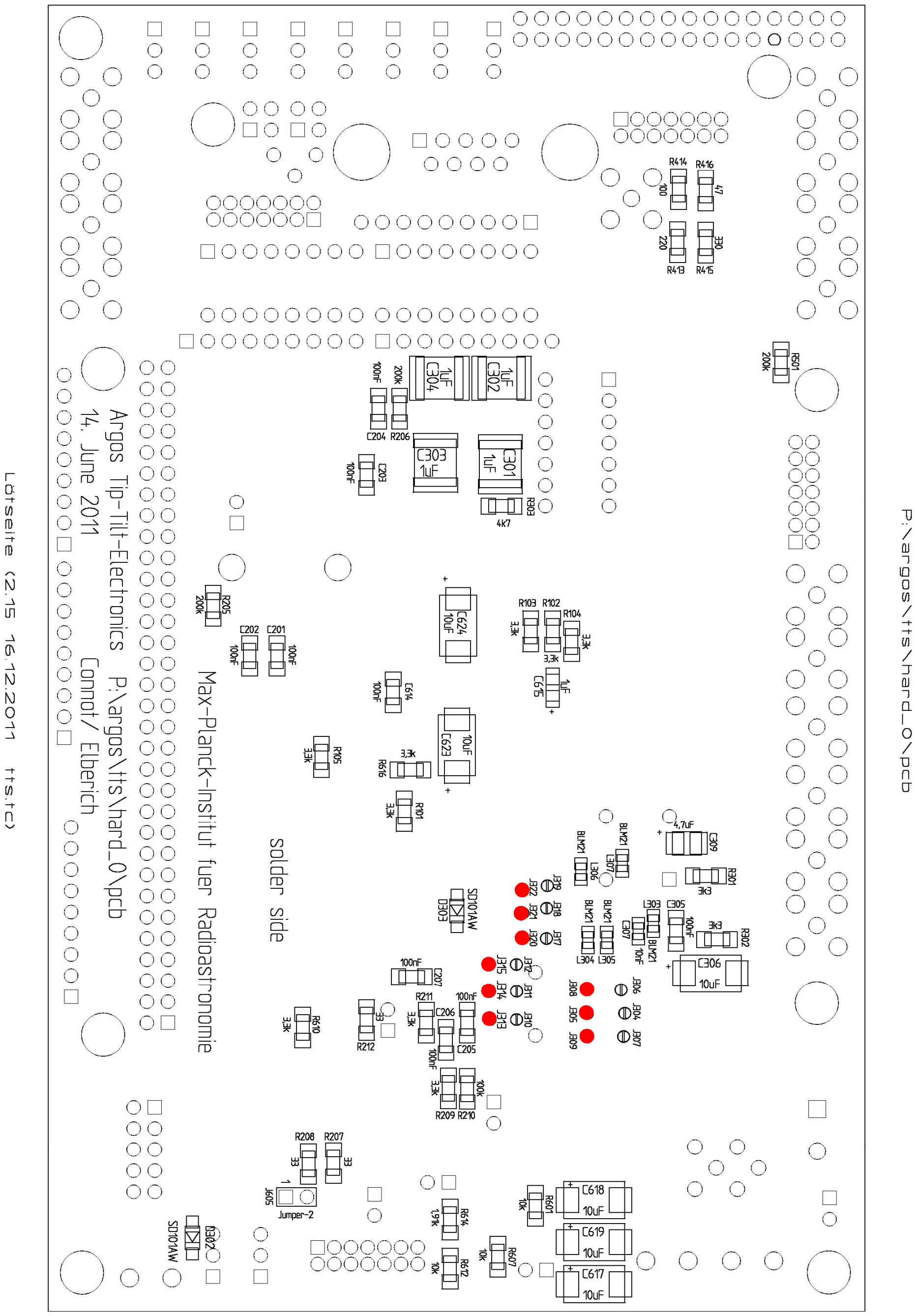
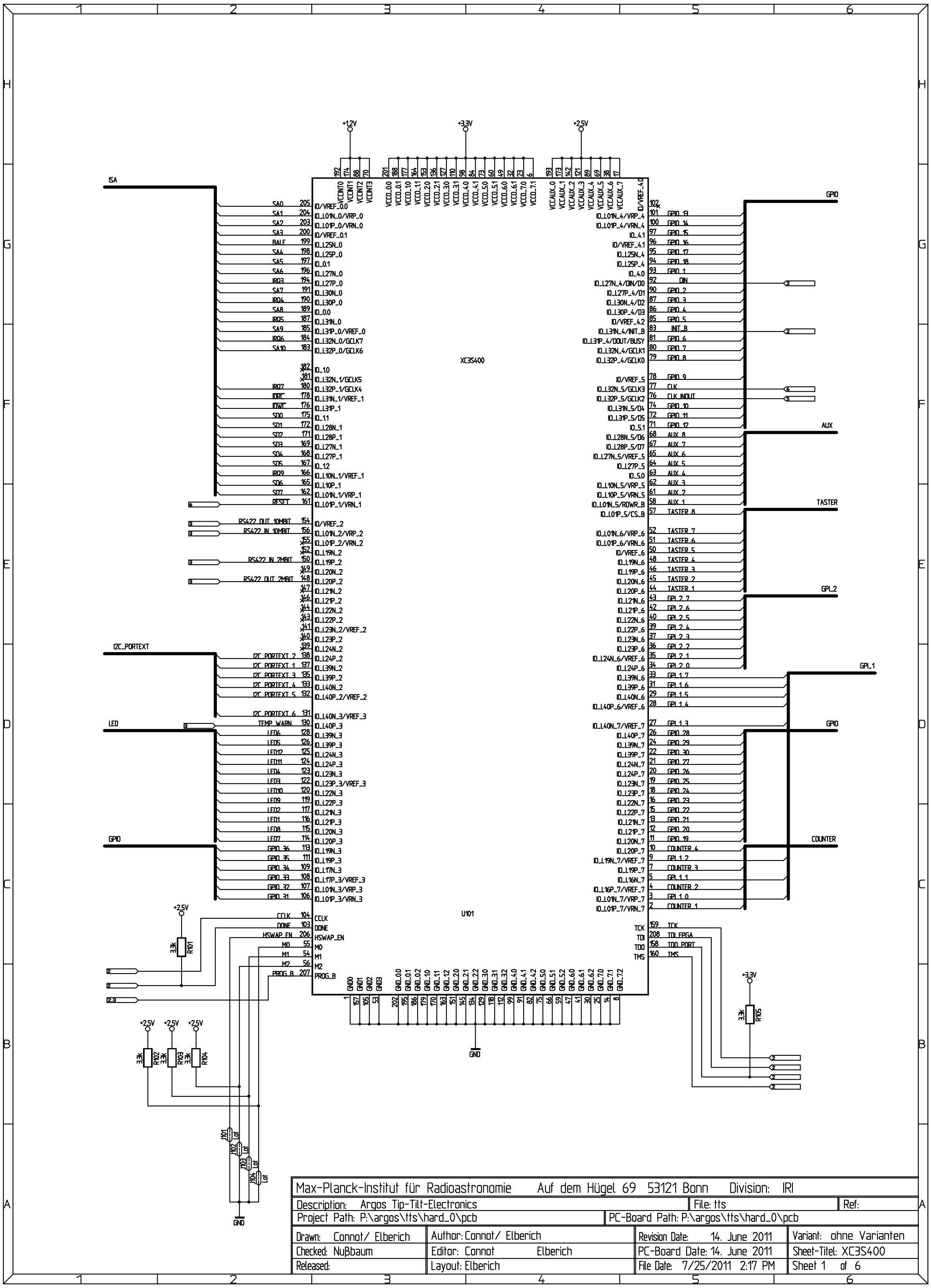
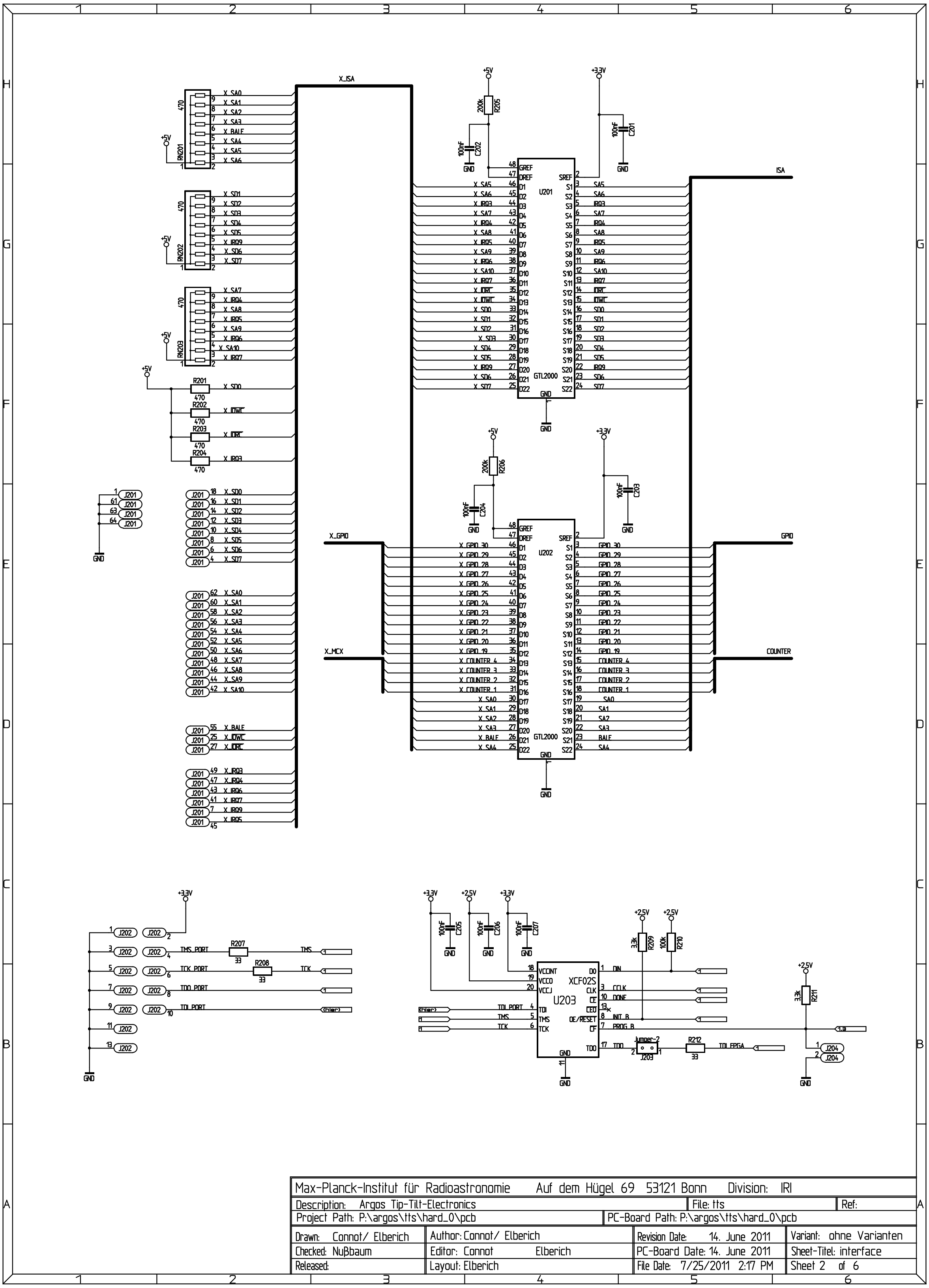
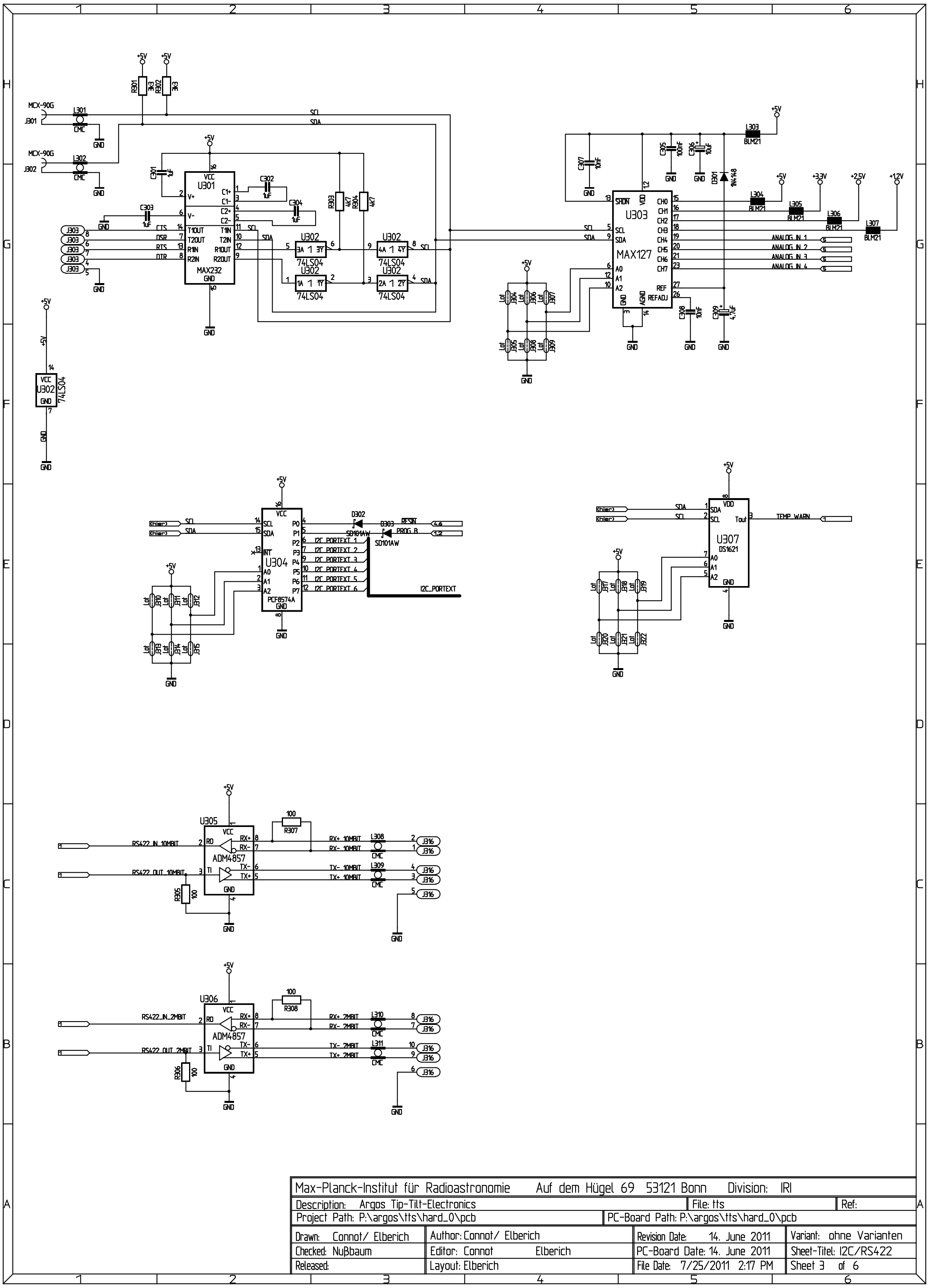


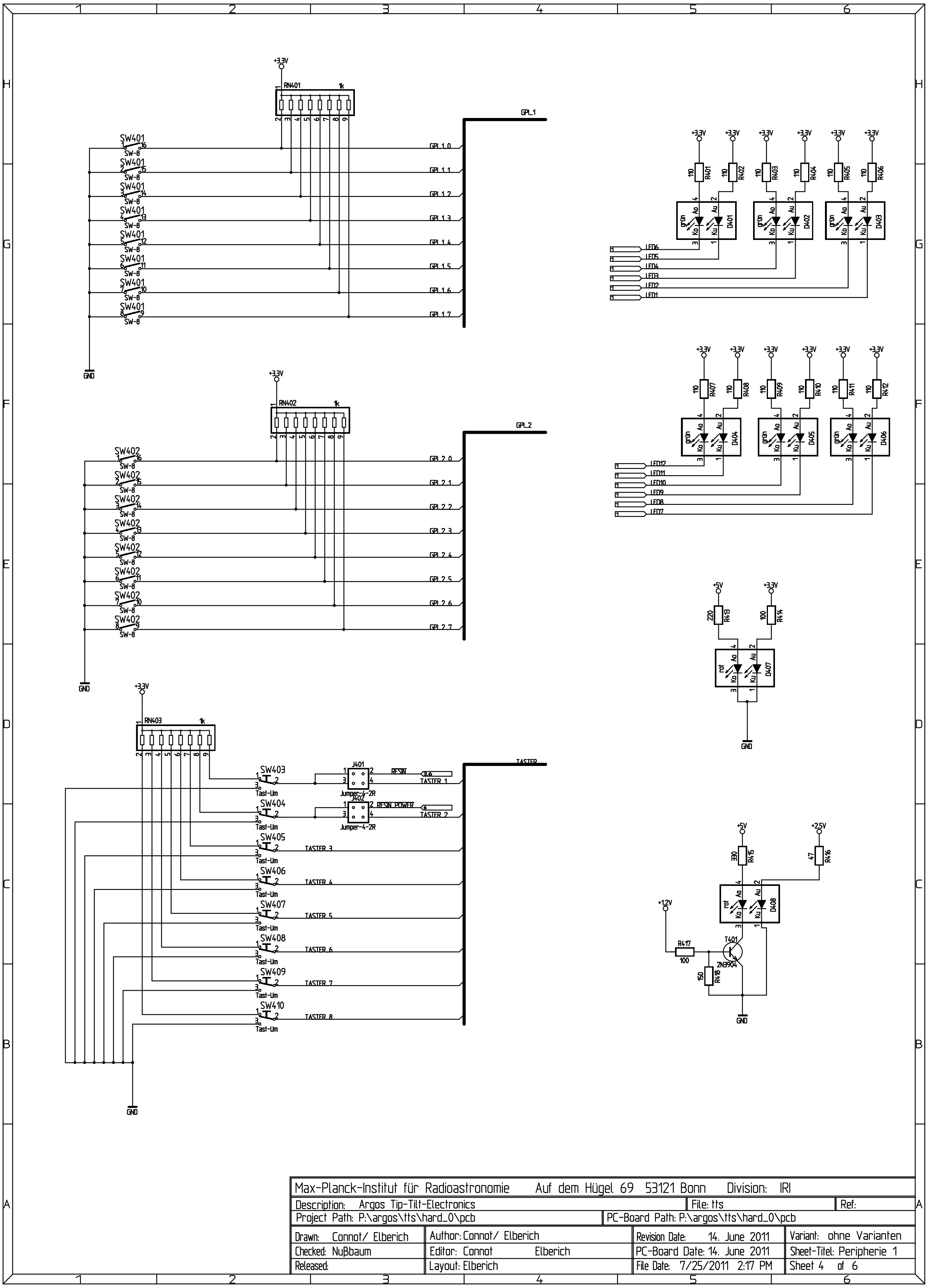
Figure 10: Assembly diagram with jumper settings (red=closed), solder side

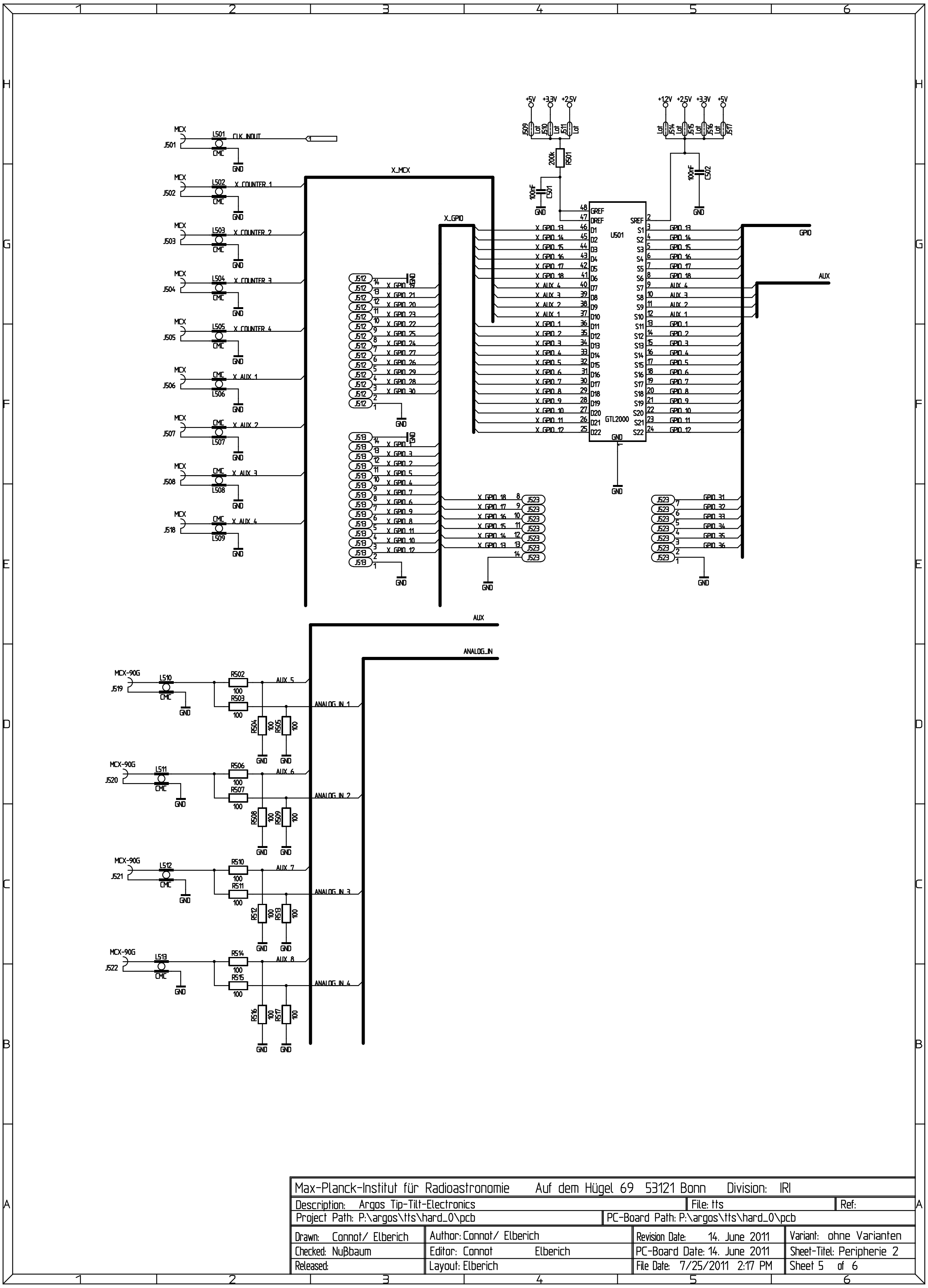
APPENDIX (3): Schematics

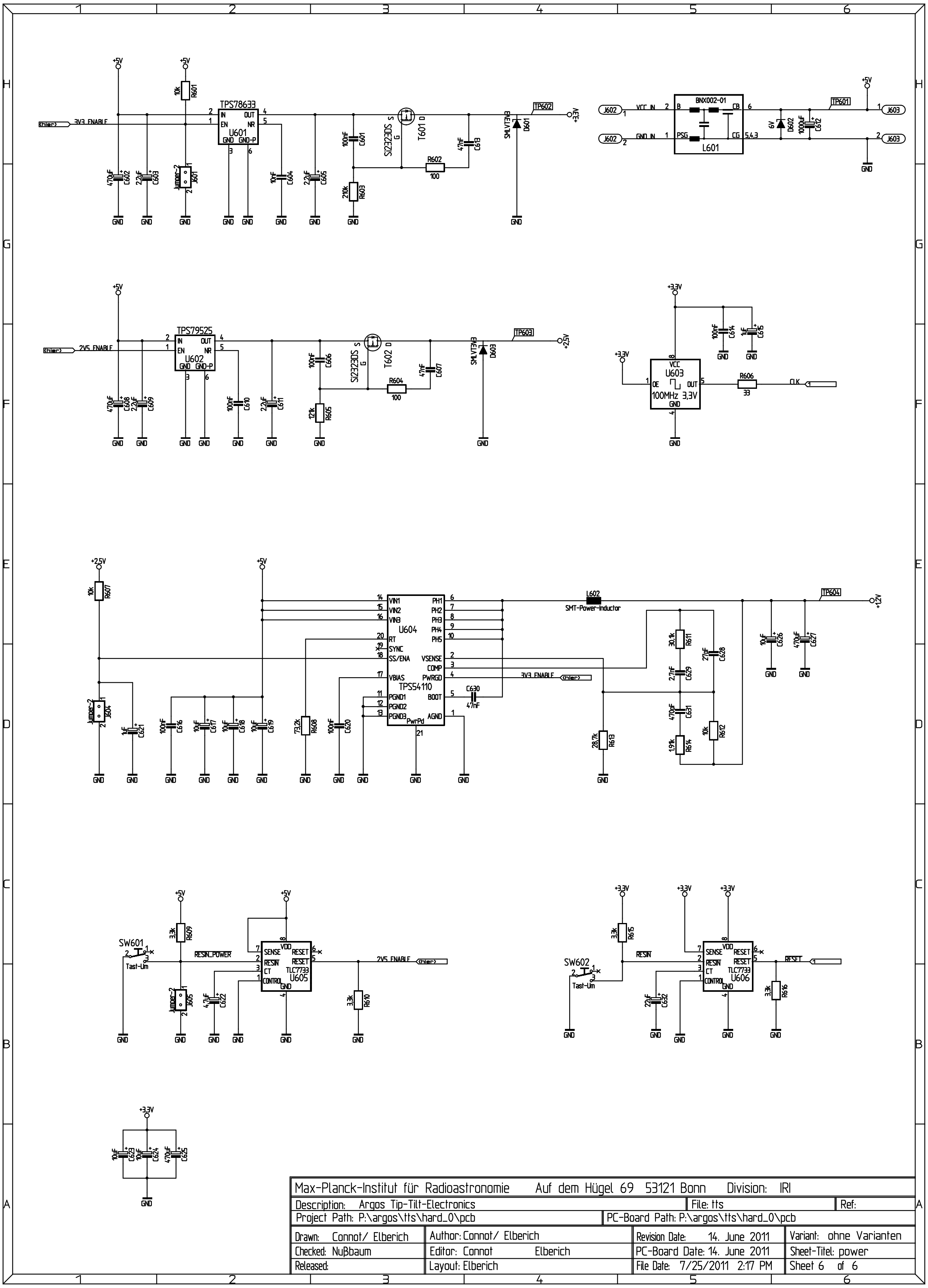



Max-Planck-Institut für Radioastronomie Auf dem Hügel 69 53121 Bonn Division: IRI
Description: Argos Tip-Tilt-Electronics
File: tts
Ref:
Project Path: P:\argos\tts\hard_0\pcb
PC-Board Path: P:\argos\tts\hard_0\pcb
Drawn: Connot/ Elberich
Checked: Nußbaum
Released:
Author: Connot/ Elberich
Editor: Connot Elberich
Layout: Elberich
Revision Date: 14. June 2011
PC-Board Date: 14. June 2011
File Date: 7/25/2011 2:17 PM
Variant: ohne Varianten
Sheet-Titel: interface
Sheet 2 of 6

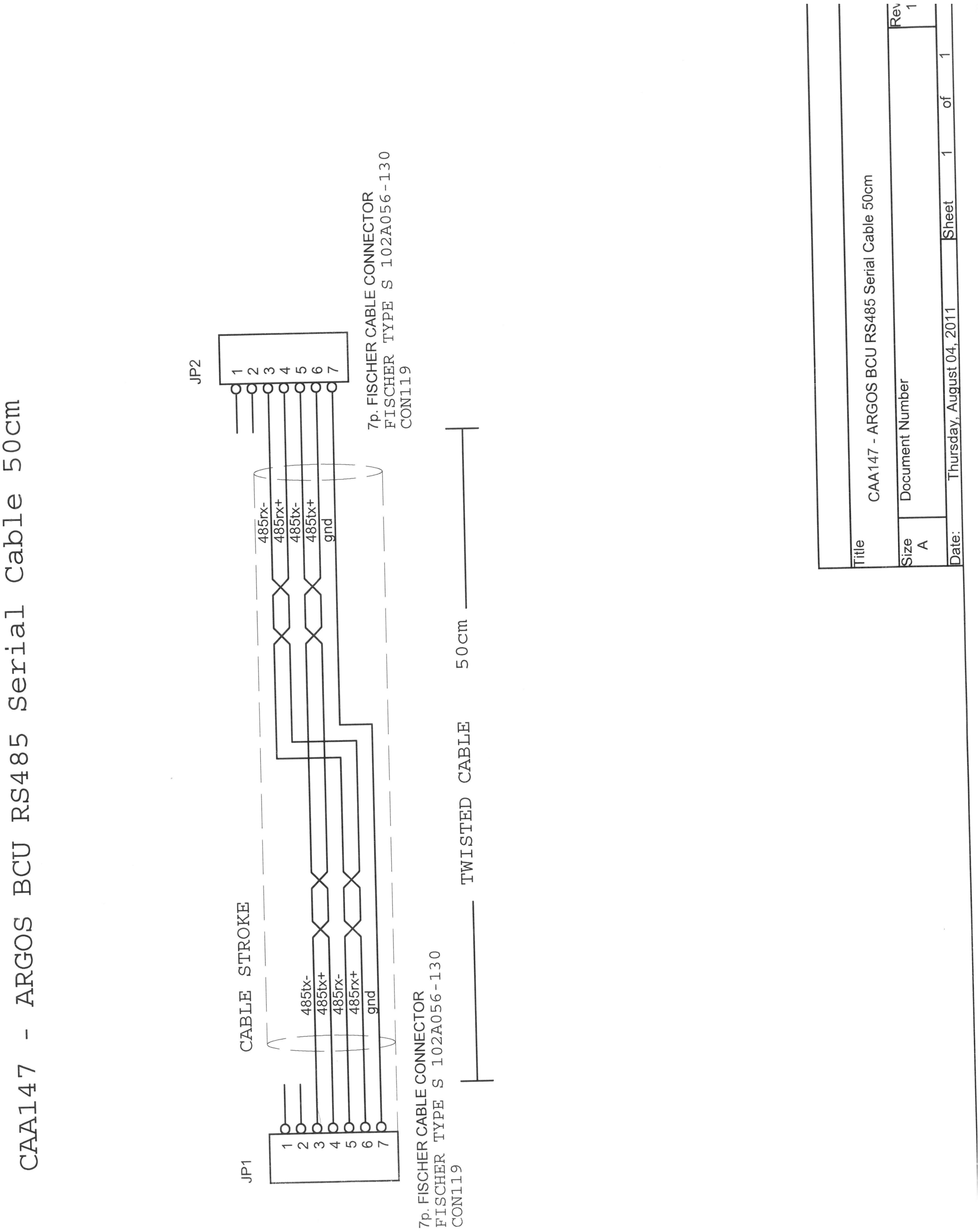

Max-Planck-Institut für Radioastronomie Auf dem Hügel 69 53121 Bonn Division: IRI
Description: Argos Tip-Tilt-Electronics
File: tts
Ref:
Project Path: P:\argos\tts\hard_0\pcb
PC-Board Path: P:\argos\tts\hard_0\pcb
Drawn: Connot/ Elberich
Author: Connot/ Elberich
Revision Date: 14. June 2011
Variant: ohne Varianten
Checked: Nußbaum
Editor: Connot Elberich
PC-Board Date: 14. June 2011
Sheet-Titel: I2C/RS422
Released:
Layout: Elberich
File Date: 7/25/2011 2:17 PM
Sheet 3 of 6
U301
MAX232
U302
74LS04
U303
MAX127
U304
PCF8574A
U305
ADM4857
U306
ADM4857
U307
DS1621
RS422_IN_10MBIT
RS422_OUT_10MBIT
RS422_IN_2MBIT
RS422_OUT_2MBIT
I2C_PORTEXT
TEMP_WARN


Max-Planck-Institut für Radioastronomie Auf dem Hügel 69 53121 Bonn Division: IRI
Description: Argos Tip-Tilt-Electronics
File: tts
Ref:
Project Path: P:\argos\tts\hard_0\pcb
PC-Board Path: P:\argos\tts\hard_0\pcb
Drawn: Connot/ Elberich
Author: Connot/ Elberich
Revision Date: 14. June 2011
Variant: ohne Varianten
Checked: Nußbaum
Editor: Connot Elberich
PC-Board Date: 14. June 2011
Sheet-Titel: Peripherie 1
Released:
Layout: Elberich
File Date: 7/25/2011 2:17 PM
Sheet 4 of 6


X_MCX
X_GPIO
GPIO
AUX
ANALOG_IN
U501
GTL2000
Max-Planck-Institut für Radioastronomie Auf dem Hügel 69 53121 Bonn Division: IRI
Description: Argos Tip-Tilt-Electronics
File: tts
Ref:
Project Path: P:\argos\tts\hard_0\pcb
PC-Board Path: P:\argos\tts\hard_0\pcb
Drawn: Connot/ Elberich
Author: Connot/ Elberich
Revision Date: 14. June 2011
Variant: ohne Varianten
Checked: Nußbaum
Editor: Connot Elberich
PC-Board Date: 14. June 2011
Sheet-Titel: Peripherie 2
Released:
Layout: Elberich
File Date: 7/25/2011 2:17 PM
Sheet 5 of 6


TPS78633
U601
T601
TPS79525
U602
T602
BNX002-01
L601
U603
100MHz 3,3V
U604
TPS54110
L602
SMT-Power-Inductor
SW601
Tast-Um
U605
TLC7733
SW602
U606
Max-Planck-Institut für Radioastronomie Auf dem Hügel 69 53121 Bonn Division: IRI
Description: Argos Tip-Tilt-Electronics
File: tts
Ref:
Project Path: P:\argos\tts\hard_0\pcb
PC-Board Path: P:\argos\tts\hard_0\pcb
Drawn: Connot/ Elberich
Author: Connot/ Elberich
Revision Date: 14. June 2011
Variant: ohne Varianten
Checked: Nußbaum
Editor: Connot Elberich
PC-Board Date: 14. June 2011
Sheet-Titel: power
Released:
Layout: Elberich
File Date: 7/25/2011 2:17 PM
Sheet 6 of 6

APPENDIX (3): Crossed Cable to BCU



APPENDIX (4): RS422 Fischer 7p connector pinout (on BCU and Bonn TT Unit the same)

11 ANNEXES

11.1 Connector pin-out for MPIfR serial interface

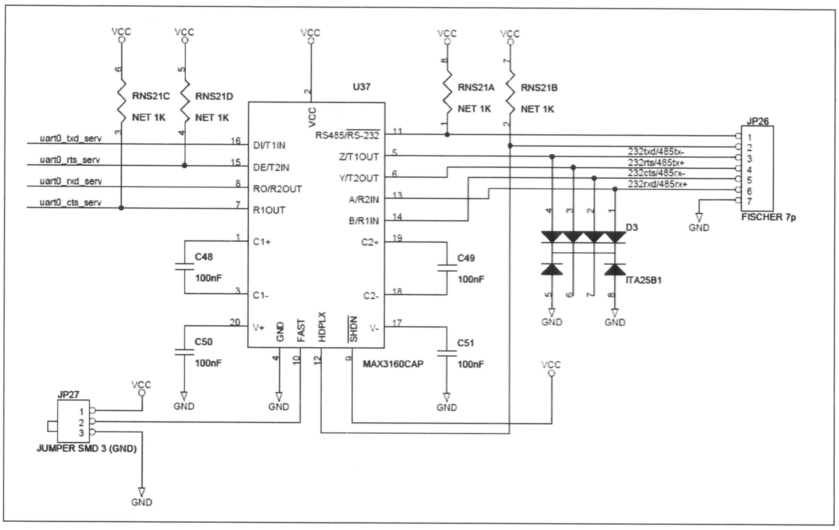
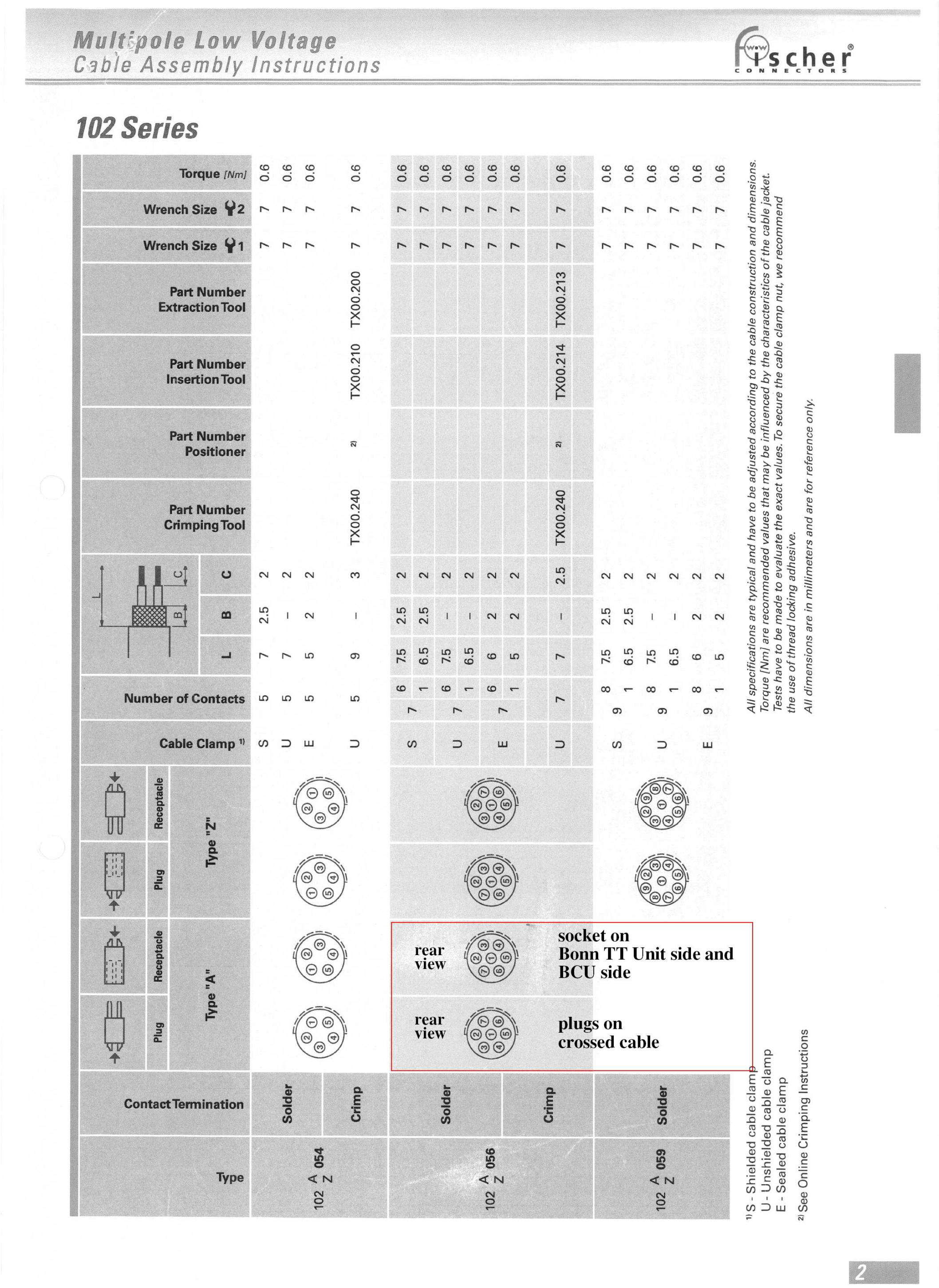


Figure 15 – RS232/485 Serial connector pin-out

APPENDIX (5): Fischer connectors



Multipole Low Voltage
Cable Assembly Instructions

fischer CONNECTORS

102 Series

Type	Contact Termination	Cable Clamp [1]	Number of Contacts	L	B	C	Part Number Crimping Tool	Part Number Positioner	Part Number Insertion Tool	Part Number Extraction Tool	Wrench Size 1	Wrench Size 2	Torque (Nm)
102 A/Z 054	Solder	S	5	7	2.5	2					7	7	0.6
		U	5	7	–	2					7	7	0.6
		E	5	5	2	2					7	7	0.6
	Crimp	U	5	9	–	3	TX00.240	[2]	TX00.210	TX00.200	7	7	0.6
102 A/Z 056	Solder	S	7: 6	7.5	2.5	2					7	7	0.6
			1	6.5	2.5	2					7	7	0.6
		U	7: 6	7.5	–	2					7	7	0.6
			1	6.5	–	2					7	7	0.6
		E	7: 6	6	2	2					7	7	0.6
			1	5	2	2					7	7	0.6
	Crimp	U	7	7	–	2.5	TX00.240	[2]	TX00.214	TX00.213	7	7	0.6
102 A/Z 059	Solder	S	9: 8	7.5	2.5	2					7	7	0.6
			1	6.5	2.5	2					7	7	0.6
		U	9: 8	7.5	–	2					7	7	0.6
			1	6.5	–	2					7	7	0.6
		E	9: 8	6	2	2					7	7	0.6
			1	5	2	2					7	7	0.6

All specifications are typical and have to be adjusted according to the cable construction and dimensions. Torque (Nm) are recommended values that may be influenced by the characteristics of the cable jacket. Tests have to be made to evaluate the exact values. To secure the cable clamp nut, we recommend the use of thread locking adhesive.
All dimensions are in millimeters and are for reference only.

[1] S - Shielded cable clamp
U - Unshielded cable clamp
E - Sealed cable clamp
[2] See Online Crimping Instructions

2

End of document