DBBC.2 Backend System: Status Report

March 23, 2009

G. Tuccari[1], W. Alef[2], A. Bertarini[3], S. Buttaccio[1], D. Graham[2], A. Neidhardt[4], G. Nicotra[1], A. Roy[2], M. Wunderlich[2], R. Zeitlhoefler[4]

1 Istituto di Radioastronomia - INAF, Noto, Italy
2 Max Planck Institute für Radioastronomie, Bonn, Germany
3 Institut für Geodäsie und Geoinformation,Bonn, Germany
4 FESG – TU, München, Country

Summary

- Team
- Architecture
- Single Element Performance
- System Performance
- Field System
- Network Interface
- Project Status
- Deployment
- DBBC3

DBBC TEAM

Collaborators in different tasks:
Project development, Organization, FS Integration, Testing, etc.

- G. Tuccari, S. Buttaccio, G. Nicotra - IRA Noto
- W. Alef, A. Bertarini, D. Graham, M. Wunderlich - MPI Bonn
- A. Neidhardt , R. Zeitlhoefler - TUM Wettzell

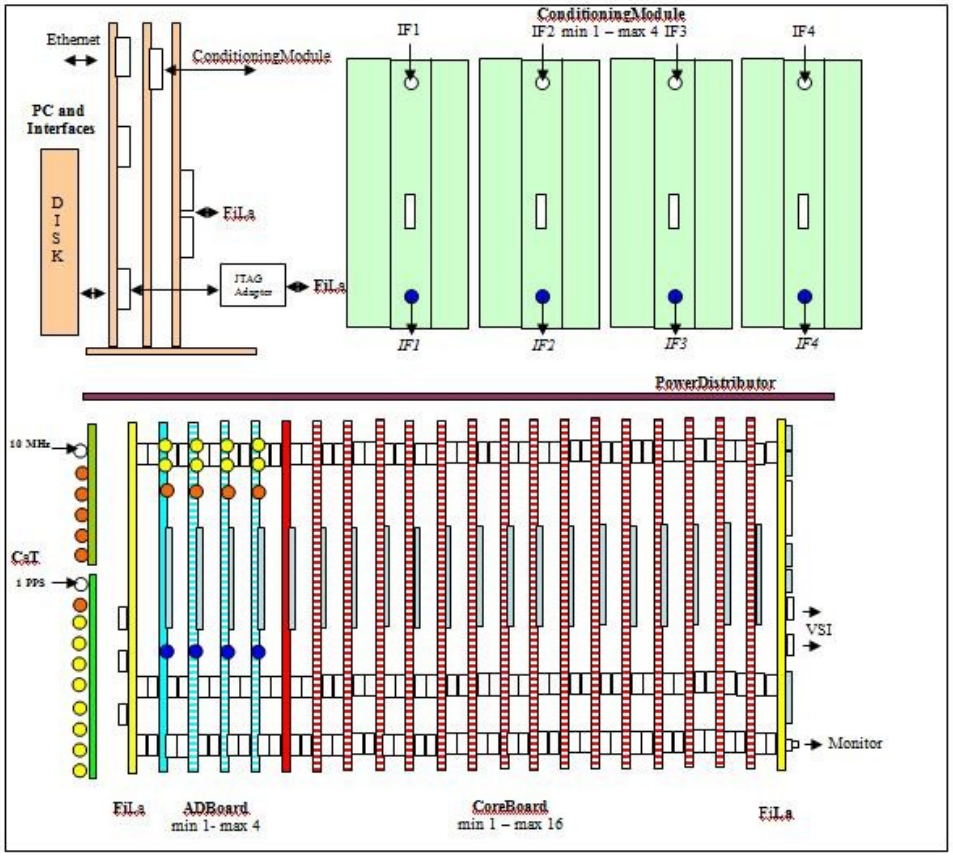
Related projects:

- A. Roy, K. Das - MPI Bonn
- G. Comoretto - AO Florence

In addition for the FiLa10G:

- Y. Xiang - SHAO
- J. Wagner - Metsahovi

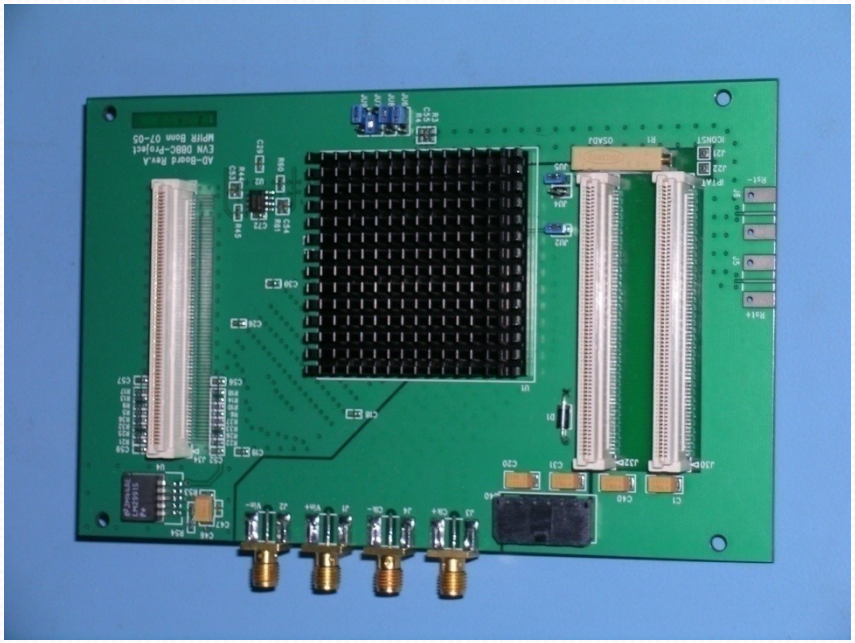
Architecture



DBBC top view

ADB 1

Analog to Digital Converter



Analog input: 0 - 2.2 GHz

Max Sampling clock single board: 1.5 GHz

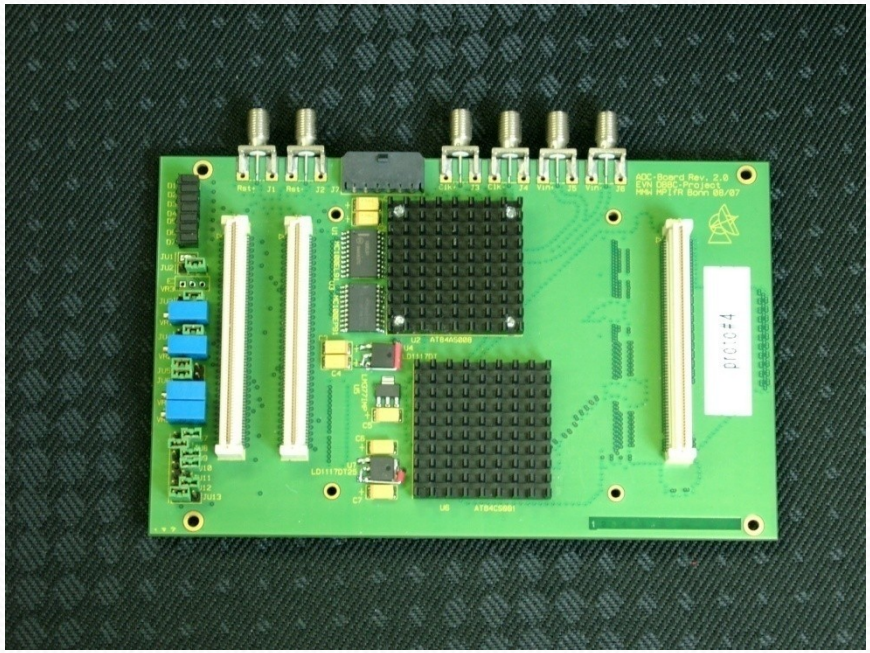
Max Istantaneous Bandwidth in Real Mode: 750 MHz

Max Istantaneous Bandwidth in Complex Mode: 1.5 GHz

Output Data: 2 x 8-bit @ ¼ SClk DDR

ADB 2

Analog to Digital Converter



Analog input: 0 – 3.5 GHz

Max Sampling clock single board: 2.2 GHz

Max Istantaneous Bandwidth in Real Mode: 1.1 GHz

Max Istantaneous Bandwidth in Complex Mode: 2.2 GHz

Output Data: 2 x 8-bit @ ¼ SCIk DDR
4 x 8-bit @ 1/8 SCIk DDR

Piggy-back module support for 10-bit output and connection with FiLa10G board.

Core 2

Basic processing unit

Input Rate:
(4 IFs x 2 bus x 8 bit x SCIk/4 DDR) b/s
(2 IFs x 4 bus x 8 bit x SCIk/8 DDR) b/s
More…

Typical Output Rate:
(64 ch x 32-64-128-256) Mb/s

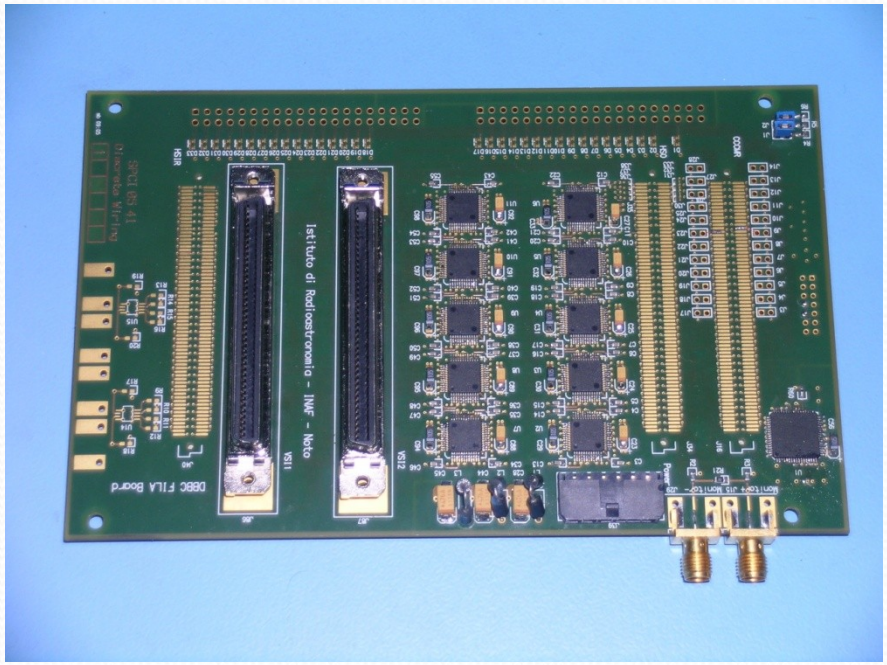
Programmable architecture

Es. Digital Down Converter:
1 CoreBoard2 = 4 BBC

Max Input/Output Data Rate 32.768 Gbps

FiLa

Connection and Service



First and Last board in the stack

First:
- Communication Interface
- JTAG Programming Channel
- 1PPS Input

Last:
- 2 VSI Interfaces
- DA Converter

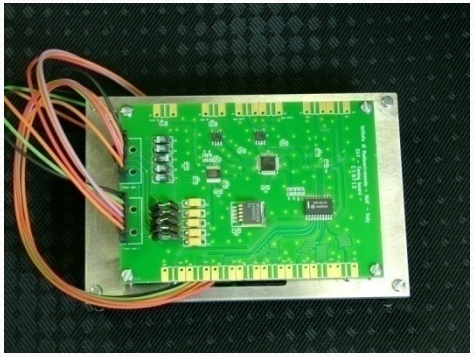
1PPS Monitor Out

80Hz Continuous Cal Out

CaT

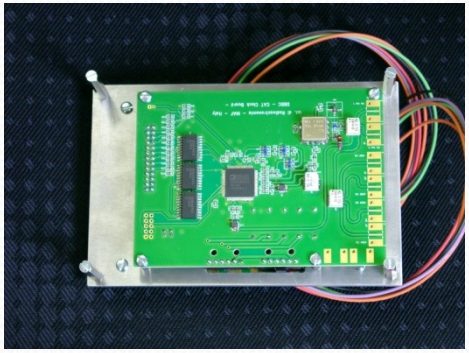
Clock and Timing

Timing Board



Timing Synchronization:
High Resolution UT1PPS Generation

Clock Board



System Clock Generation:
Input Reference
Programmable
(es. 1024 MHz, 2^30 Hz,
2048 MHz, 2^31 Hz)

ConditioningModule

Pre-AD Conversion Analog Signal Conditioning

Pre-AD Conversion Band Definition

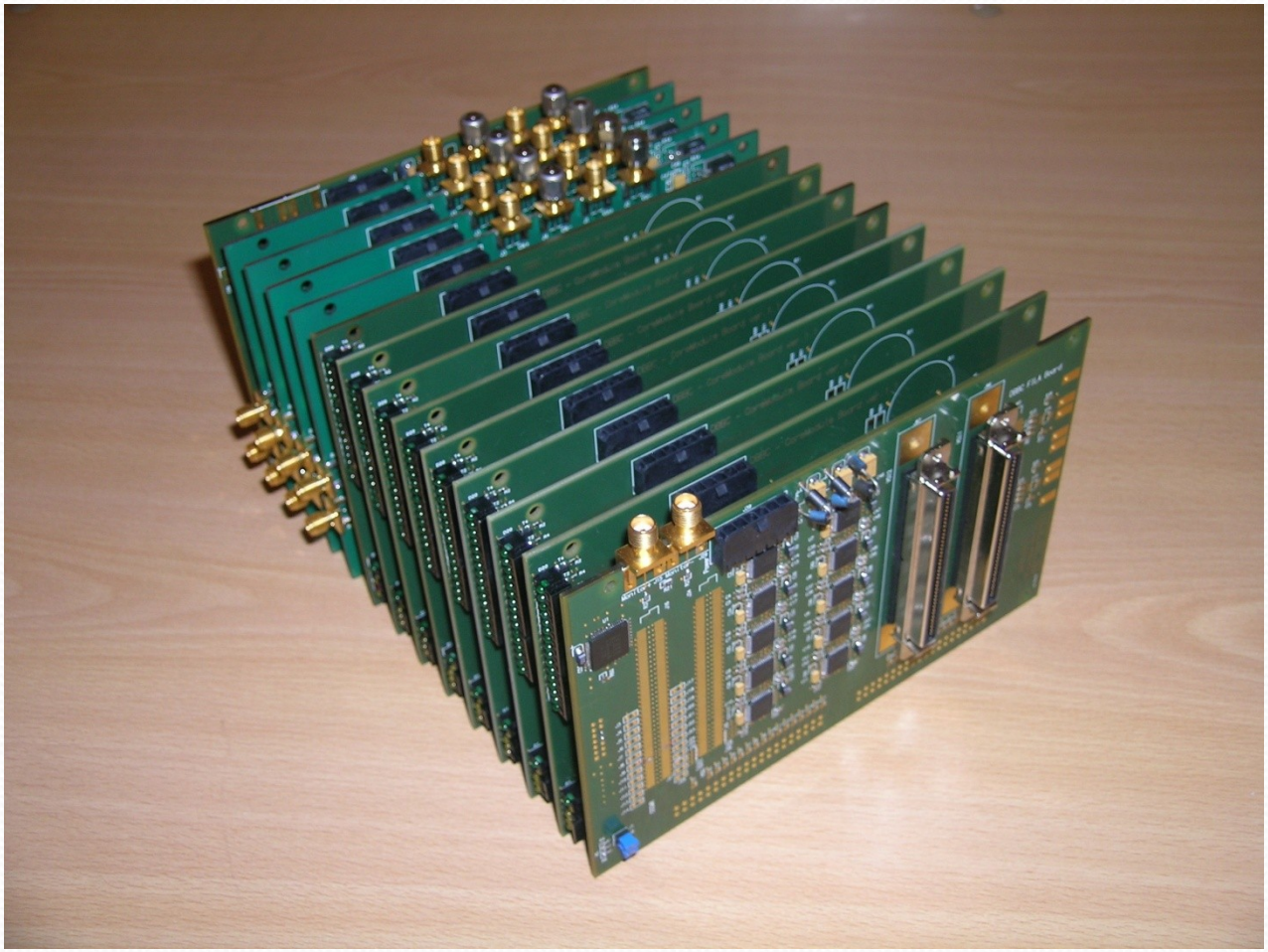
4 IFs Selectable Input

Total power measurement

RF Gain Control

Amplitute equalization

4 ADBoard + 8 Core Stack



PCSet

FPGA device configuration through USB – JTAG interface

**Communication with 32-bit bus for CoreBoards register setting,
total power measurement,
statistics of the state,
single channel automatic gain control, etc.**

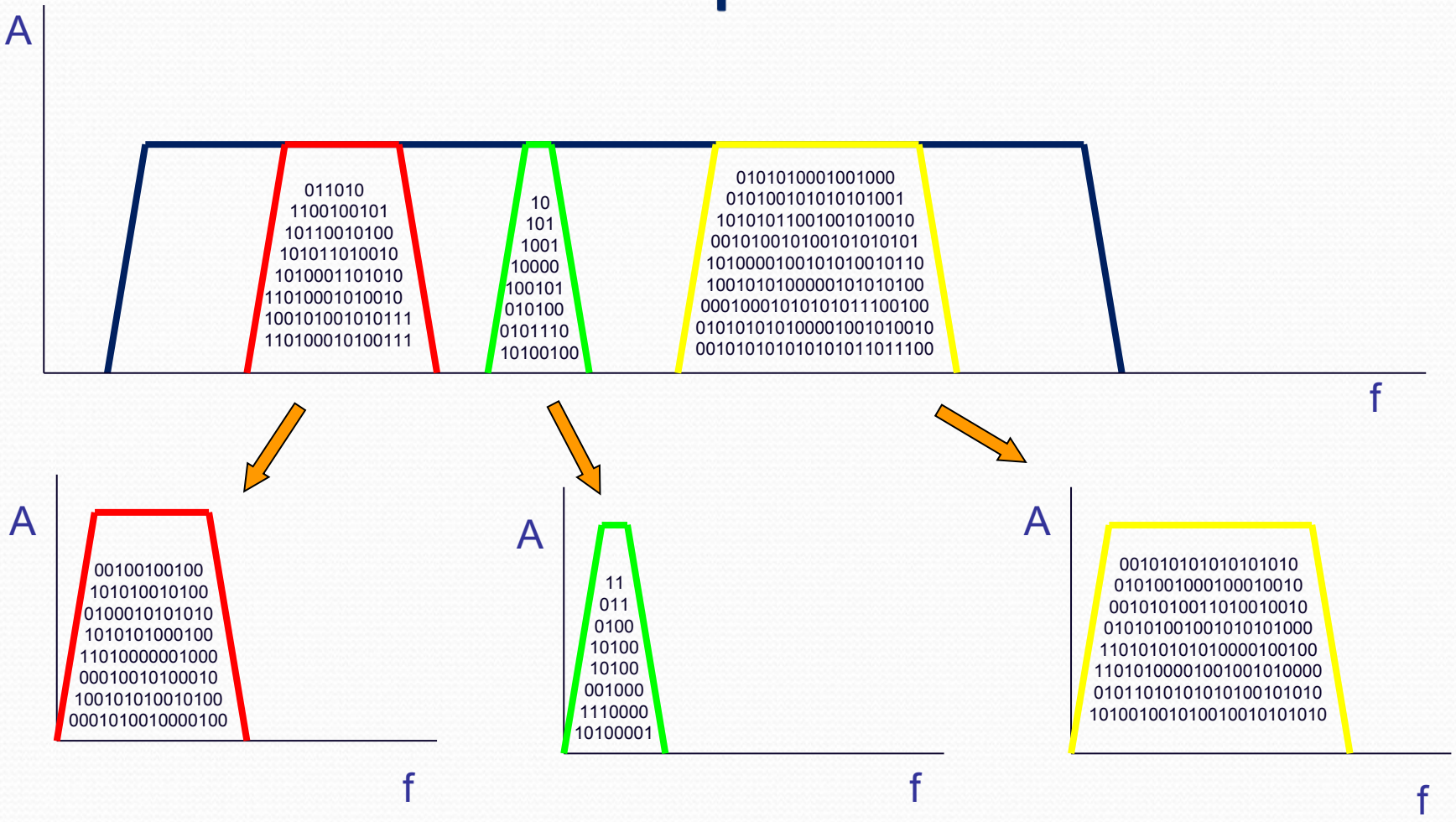
Communication with Conditioning Modules for IF total power measure, automatic gain control, registers control

Field System interface through a network connection

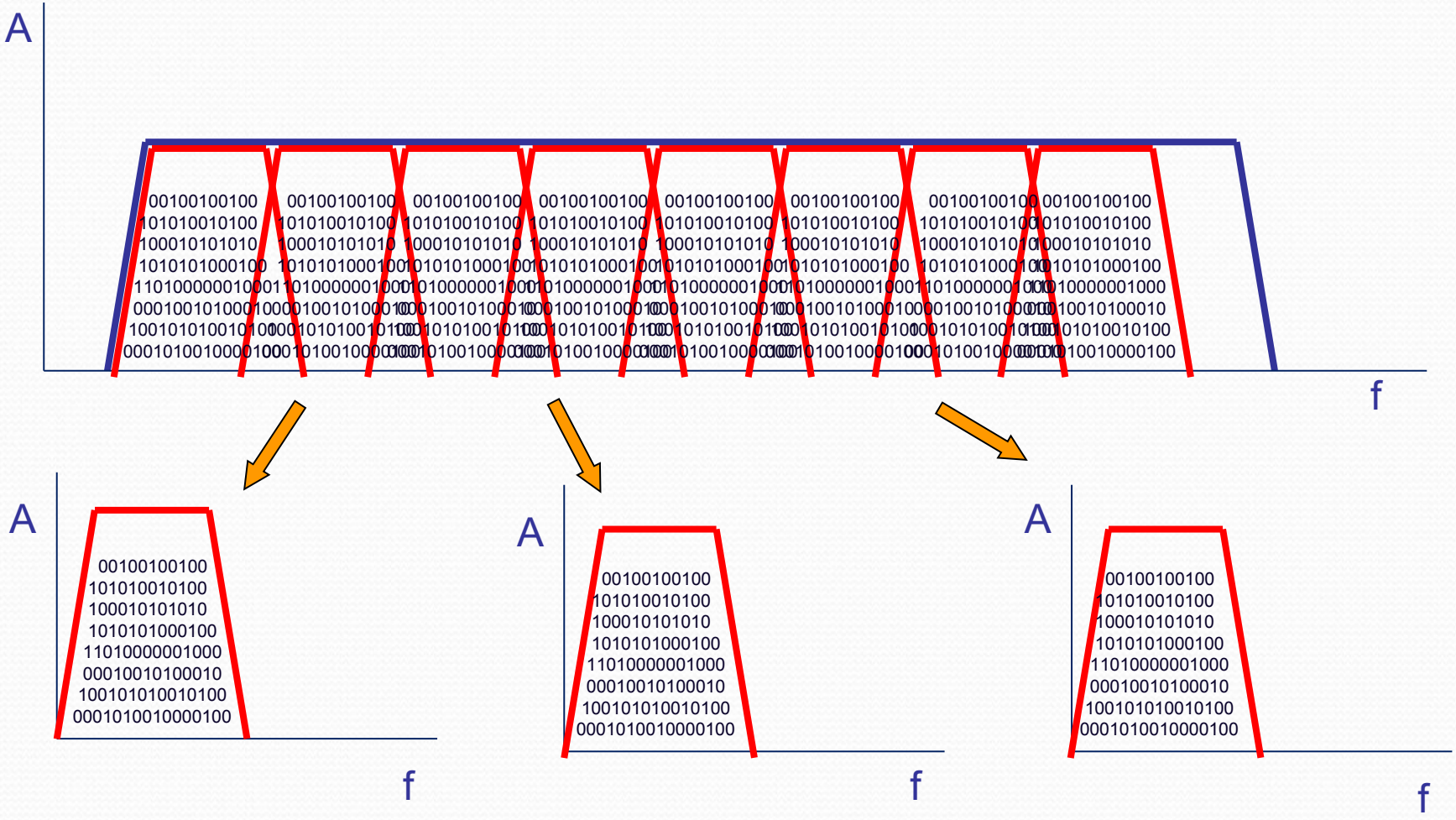
DBBC Backend General Features

- 4 RF/IF with 4 Input each in a range up to 2.200 GHz
- Four polarizations or bands available for a single group of 64 output data channel selection (2 VSI output connectors with max total of 16 Gb/s)
- 1024 MHz sampling clock frequency
- Channel bandwidth ranging between 500 KHz and 32 MHz, U&L
- Wider channel bandwidth: 4 x 512, 4 x 1024 MHz
- A maximum of 64 BBC are possible in one system
- Tuning step subHz, usable geo frequencies xxx.99
- 80 Hz continuous cal support
- Multiple architecture using fully re-configurable FPGA
- Modular realization for cascaded stack processing

Digital Down Conversion to Base Band of Independent Channels



Multi Equispaced Channel Conversion to Base Band

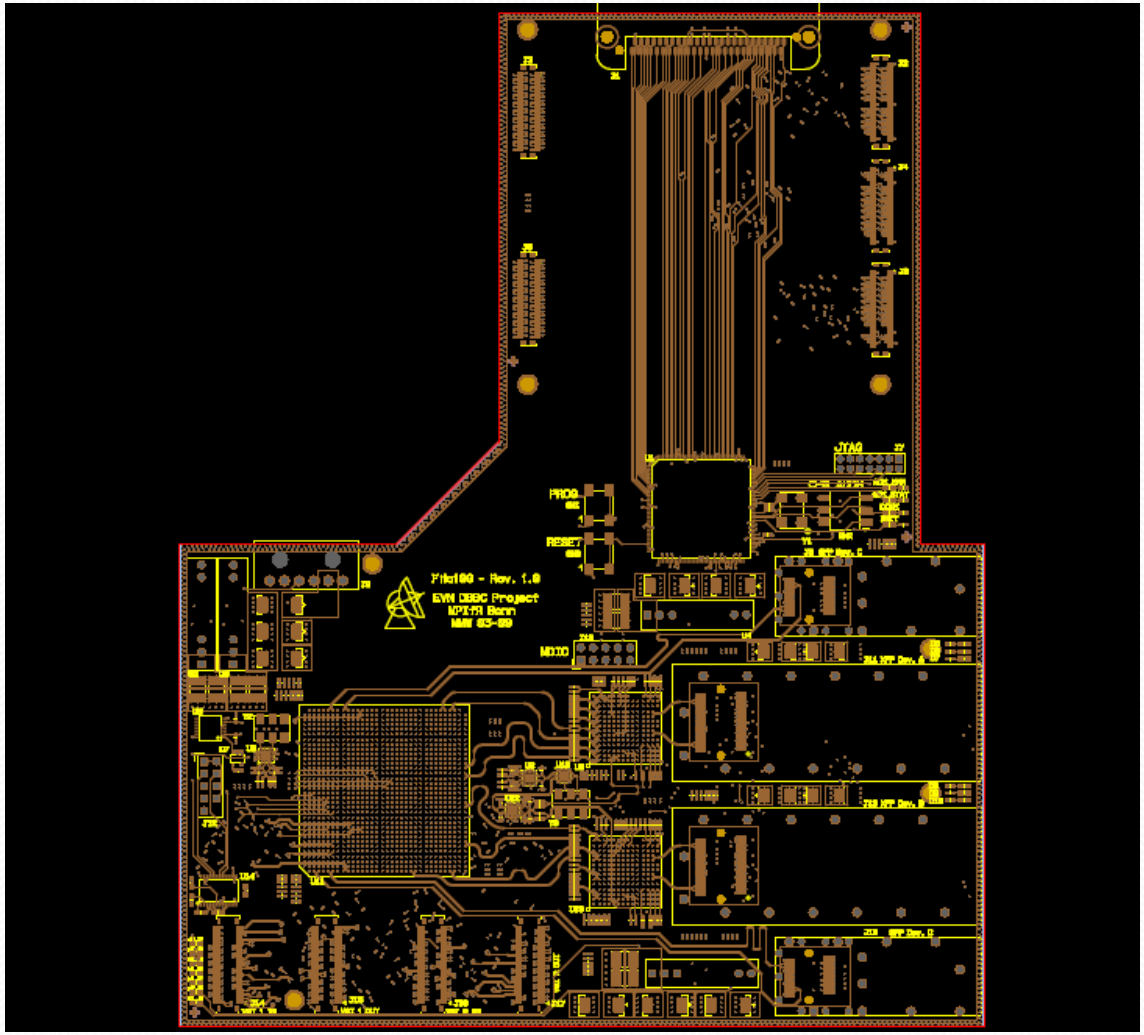


Network Board

FILA10G

- Development Team IRA-MPI-Metsahovi-SHAO
- FILA10G the interface between the DBBC (or any VSI device) to 10G network (including MK5C) is under development
- The board will be interface for the MK5C or as direct connection to the network at 1–2–4–10–20 Gbps
- Can be used as standalone between VSI and network
- Can be used as standalone with ADB2
- VDIF compliant

FiLa10G


JTAG
PROG
RESET
MDIO
FiLa10G - Rev. 1.0
EVN DBBC Project
MPIfR Bonn
MAR 03-09

News on Hardware - Firmware - Software - Testing

- A new Conditioning Module based on a single board (integrates also filters) is available
- The Core2 has on board Virtex 5 LX220 FPGA, but can also be populated with the bigger 330 device (expensive but possible)
- The firmware in its present version can provide 4 BBCs (U+L) functionality on one FPGA.
- A fixed filter-bank firmware with real output is available too, one Core2 makes all the job (could 4 for times)
- More additional feature under testing to be inserted in the std package: autocorrelation, cross-correlation between channels, phase-cal detection
- Wettzell is working on the integration in the FS (see next slides)
- Conversion to Linux is underway as all the drivers are available
- Testing is underway with the units in Wz, soon with Ef too
- Xxx.99 MHz frequency problem solved, still to be improved sensitivity to IF levels

FS Integration

Reinhard Zeitlhoefler
Forschungseinrichtung Satellitengeodaesie
Technische Universitaet Muenchen

Field System Integration of the Digital Base Band Converter (DBBC) at Wettzell

Abstract

A command set for the DBBC controlling is defined in the IRA-INAF Technical Report DBBC Management Command Set.
This command set is implemented as Field System Snap Commands in the station programms (user2/st) at Wettzell.
The purpose is, to make first experiences with connections from Field System to DBBC for tests and developing.

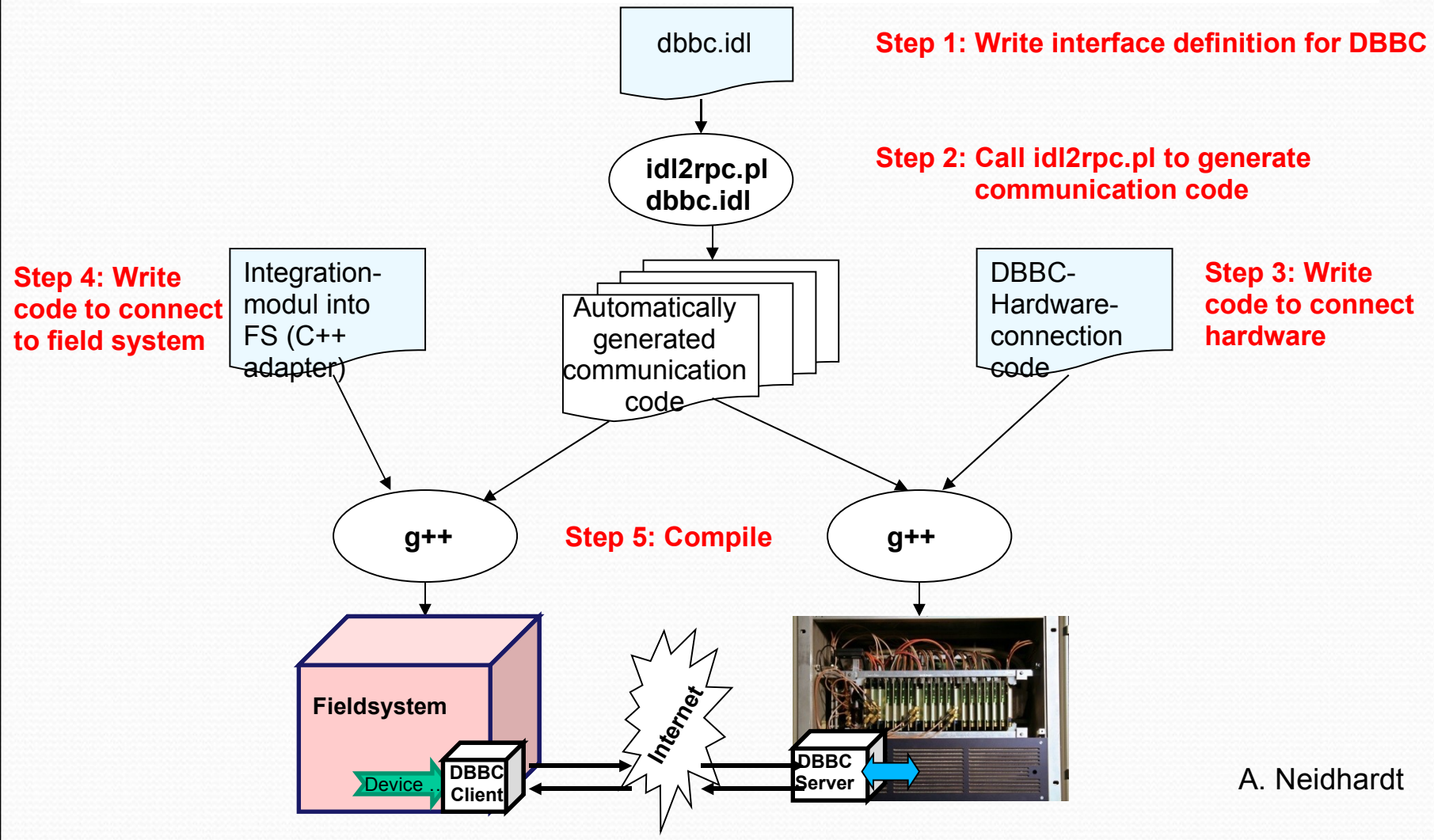
DBBC Command Implementation in the Field System Software

According to the description of the command set in the Technical Report
DBBC Management Command Set, the commands are defined in the control file user2/stcmd.ctl to be known to the Field System as Snap Commands.

The program user2/stqkr/stqkr.c calls the corresponding functions
for parsing, and if inputs are accepted, for sending to the server (DBBC)
using TCP,IP protocol.
The server is simulated by a program running also at FS-PC.
While working on the dbbc command implementation, this experience showed,
that minor changes in the command set could be usefull and a data hanshake
(in ASCII characters) should be defined.

Concept suggestion for realization of DBBC communication with autonomous process cells

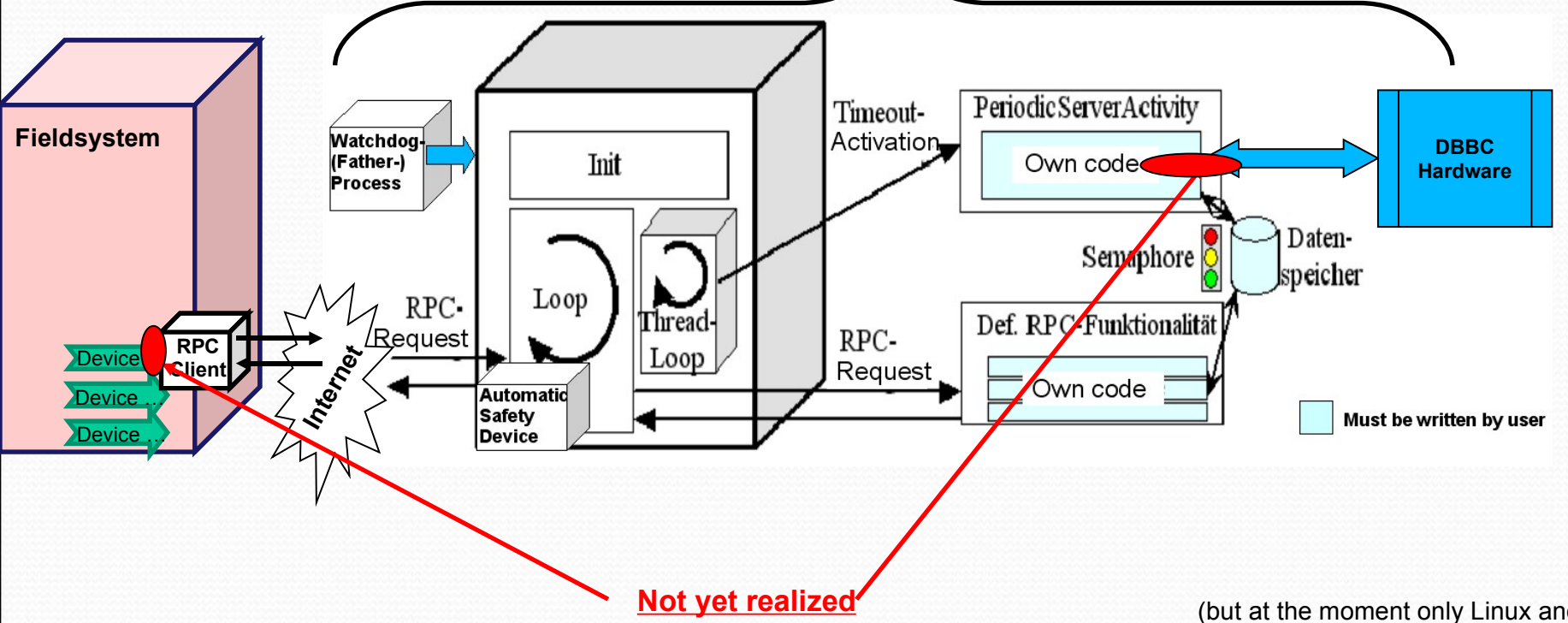
Standardized communication on the basis of ONC Remote Procedure Calls (RPC) using idl2rpc.pl



A. Neidhardt

Concept suggestion for realization of DBBC communication with autonomous process cells

Standardized communication on the basis of ONC Remote Procedure Calls (RPC)

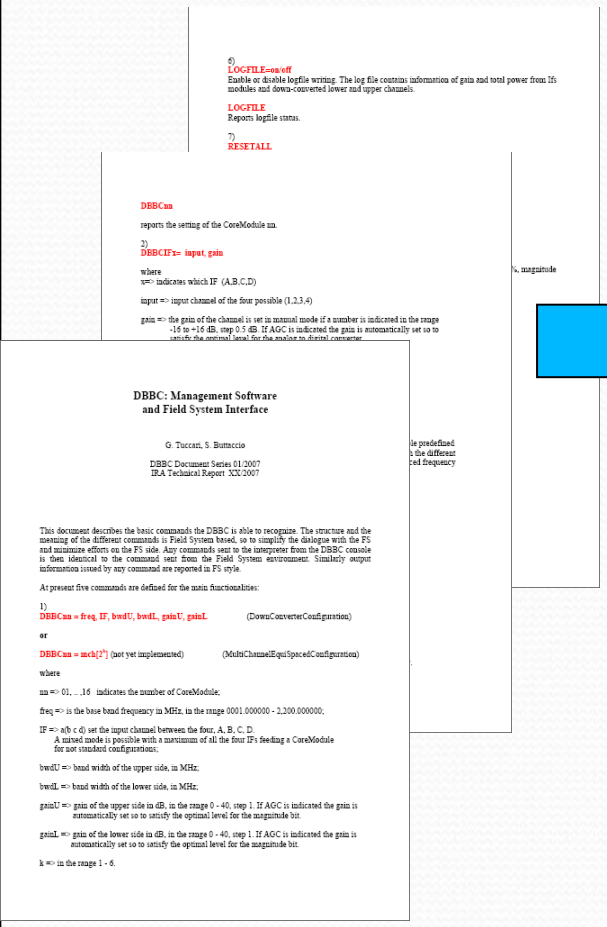


Not yet realized

(but at the moment only Linux and on field system side C++ is supported)

Concept suggestion for realization of DBBC communication with autonomous process cells

A first RPC test interface definition is realized but not yet included to hardware and field system



6).
LOGFILE=on/off
Enable or disable logfile writing. The log file contains information of gain and total power from Ifs modules and down-converted lower and upper channels.

LOGFILE
Reports logfile status.

7).
RESETALL

DBBCnn

reports the setting of the CoreModule nn.

2).
DBBCIFx= input, gain

where
x= indicates which IF (A,B,C,D)

input => input channel of the four possible (1,2,3,4)

gain => the gain of the channel is set in manual mode if a number is indicated in the range -16 to +16 dB, step 0.5 dB. If AGC is indicated the gain is automatically set so to satisfy the optimal level for the scaler to digital converter.

%, magnitude

le predefined
h the different
ed frequency

DBBC: Management Software and Field System Interface

G. Tuccari, S. Buttaccio

DBBC Document Series 01/2007
IRA Technical Report XX/2007

This document describes the basic commands the DBBC is able to recognize. The structure and the meaning of the different commands is Field System based, so to simplify the dialogue with the FS and minimize efforts on the FS side. Any commands sent to the interpreter from the DBBC console is then identical to the command sent from the Field System environment. Similarly output information issued by any command are reported in FS style.

At present five commands are defined for the main functionalities:

1)
DBBCnn = freq, IF, bwdU, bwdL, gainU, gainL (DownConverterConfiguration)

or

DBBCnn = mch{2^} (not yet implemented) (MultiChannelEquiSpacedConfiguration)

where

nn => 01, . .16 indicates the number of CoreModule;

freq => is the base band frequency in MHz, in the range 0001.000000 - 2,200.000000;

IF => a/b c d) set the input channel between the four, A, B, C, D.
A mixed mode is possible with a maximum of all the four IFs feeding a CoreModule for not standard configurations;

bwdU => band width of the upper side, in MHz;

bwdL => band width of the lower side, in MHz;

gainU => gain of the upper side in dB, in the range 0 - 40, step 1. If AGC is indicated the gain is automatically set so to satisfy the optimal level for the magnitude bit.

gainL => gain of the lower side in dB, in the range 0 - 40, step 1. If AGC is indicated the gain is automatically set so to satisfy the optimal level for the magnitude bit.

k => in the range 1 - 6.

```
    double dTotalPower);
    double dGain;
} UnitReportType;

interface dbbc
{
    // ================================================================================
    // 1) "DBBCnn=freq,IF,bwdU,bwdL,gainU,gainL,tpint" and "DBBCnn" - commands equivalent methods
    // ================================================================================
    unsigned short usSetDownConverterConfiguration (in unsigned int uiNumberOfCoreModules,
                                                    in double dFrequency,
                                                    in char cInputChannel,
                                                    in double dBandwidthOfUpperSideBand,
                                                    in double dBandwidthOfLowerSideBand,
                                                    in unsigned short usGainOfUpperSide,
                                                    in unsigned short usGainOfLowerSide,
                                                    in double dTotalPowerIntegrationTime);
    unsigned short usGetDownConverterConfiguration (in unsigned int uiNumberOfCoreModules,
                                                    out double dFrequency,
                                                    out char cInputChannel,
                                                    out double dBandwidthOfUpperSideBand,
                                                    out double dBandwidthOfLowerSideBand,
                                                    out unsigned short usGainOfUpperSide,
                                                    out unsigned short usGainOfLowerSide,
                                                    out double dTotalPowerIntegrationTime);

    // ================================================================================
    // 2) "DBBCIF(a,B,C,D)=input_ch,gain,filter" and "DBBCIF" - commands equivalent methods
    // ================================================================================
    unsigned short usSetIFModules (in char cInputChannel,
                                   in double dGain,
                                   in unsigned short usFilter);
    unsigned short usGetIFModules (in char cInputChannel,
                                   out double dGain,
                                   out unsigned short usFilter);

    // ================================================================================
    // 3) "DBBCFORM=VSI1mode,VSI2mode" and "DBBCFORM" - commands equivalent methods
    // ================================================================================
    unsigned short usSetVSIForm (in string strVSIMode1,
                                 in string strVSIMode2);
    unsigned short usGetVSIForm (out string strVSIMode1,
                                 out string strVSIMode2);

    // ================================================================================
    // 4) "DBBCMON=bnn[u/l]" and "DBBCMON" - commands equivalent methods
    // ================================================================================
    unsigned short usSetDigitalToAnalogChannel (in unsigned short usNumberOfBand,
                                                in char cSideband);
    unsigned short usGetDigitalToAnalogChannel (in unsigned short usNumberOfBand,
                                                out char cSideband);
```

Deployment

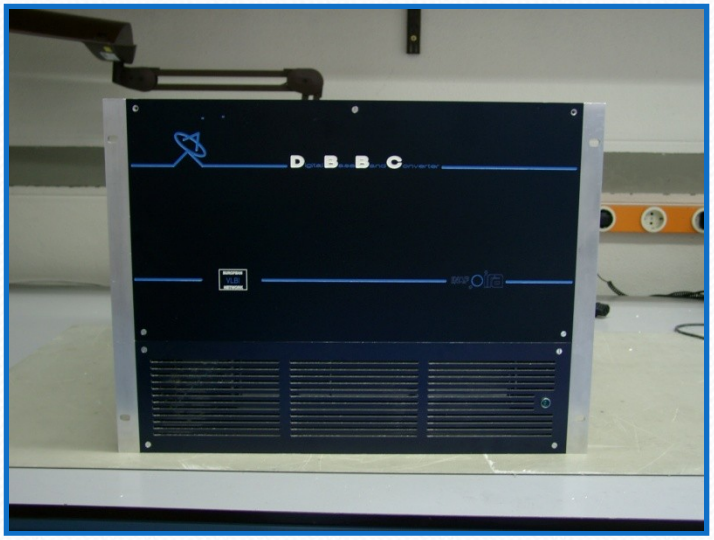
- Two DBBC2 systems in Wettzell. A third system in Wettzell will be upgraded from ver.1 to ver.2
- One system in Effelsberg, still to be integrated in the station
- Yebes, Noto, AuScope: are in completion phase, delivery in 1-2 months.
- Two systems delivered to Arcetri and Irbene, need to be upgraded to the ver.2. to be operative with the standard observing requirements, as they behave only Core1 boards.
- Other units in pending waiting for a INAF spin-off company:

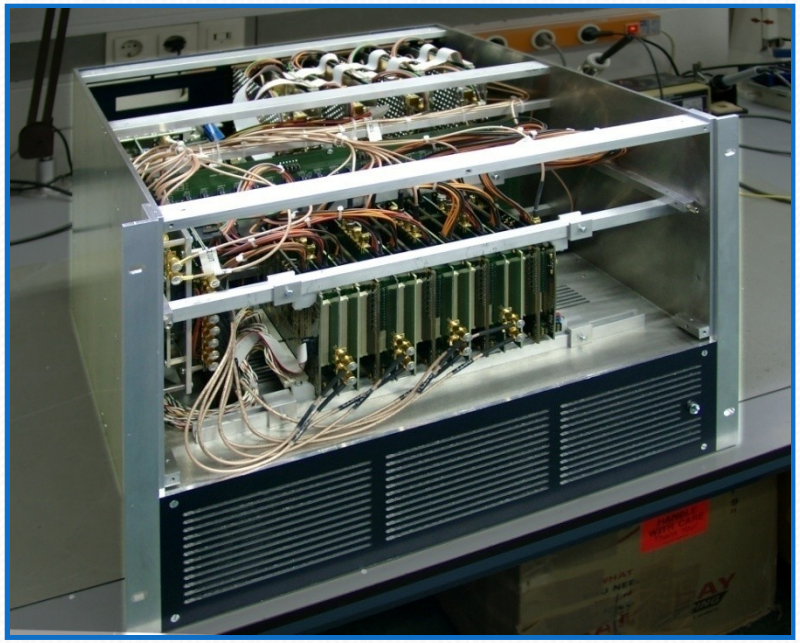
  Metsahovi, Sardinia, Medicina, Onsala, Evpatoria, etc.

Spin-off HAT-Lab

- The backend will be produced by a spin-off company named HAT-Lab which will start operation as soon as the numerous bureaucratic procedures will be completed
- At the formal set a message to EVNtech mailing list indicating also the name of the person who will be interface between users and the spin-off company.
- Delivery time for a batch of production is 3-4 months. One batch can handle the construction of 4 units.

Some pictures





DBBC 3

- Development started for a new set of boards to increase maximum bandwidth and data rate
- Compatible with precedent versions (replacement of boards in the stack, or mixed operation)
- New CoMo: 6 GHz bwd
- New CaT: integrates Clock and Timing boards and allow interleaved operations
- New Core3