An automated calibration system with distributed intelligence for the ASPERA-4 experiment of VenusExpress space mission.

1. Hardware

When configuring the automated calibration system, the environment of the existing calibration laboratory, the space at disposal, the parameters of electrical interfaces had to be considered, and also the main goal of displaying all the measured values and regulation parameters on one screen had to be reached for easy interpretation. A special requirement was the electrical isolation of measuring stations to avoid the propagation of an eventual high voltage strike or disturbance.
The measuring units had to be located near to the quantities to be measured at a relatively narrow place, and expansion of the measuring station had to be ensured for the future. PC/104 elements had been selected for this reason because in this way only three special hardware units had to be developed. Distribution of the quantities to be measured had required the building of three measuring/controlling stations. These are coupled to the central computer via a local network based on Ethernet. At first, we thought of a wireless network for ensuring galvanic isolation but owing to the numerous radio frequency noise sources and the existing wireless network, we chose fiber-optic cabling for safe communications. This fiber-optic network (Fig. 1) operates with 100 Mbps, and has substantial reserve for the data traffic required.

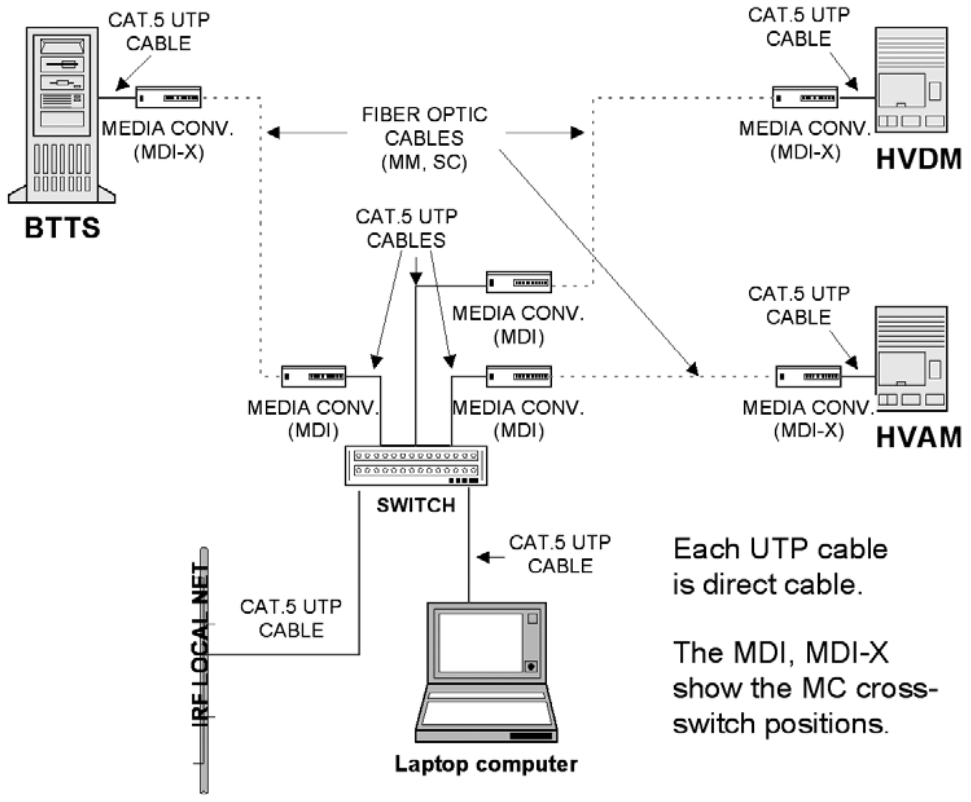

Figure 1. The 4-computer calibration system

In addition to the „isolated communications", power supplies should be isolated as well to avoid even the minimal influence of individual measuring stations to each other (Fig.2). If the emergency shutdown of a measuring unit were needed it could be done quickly by a radio frequency network switch far away from the endangered place.

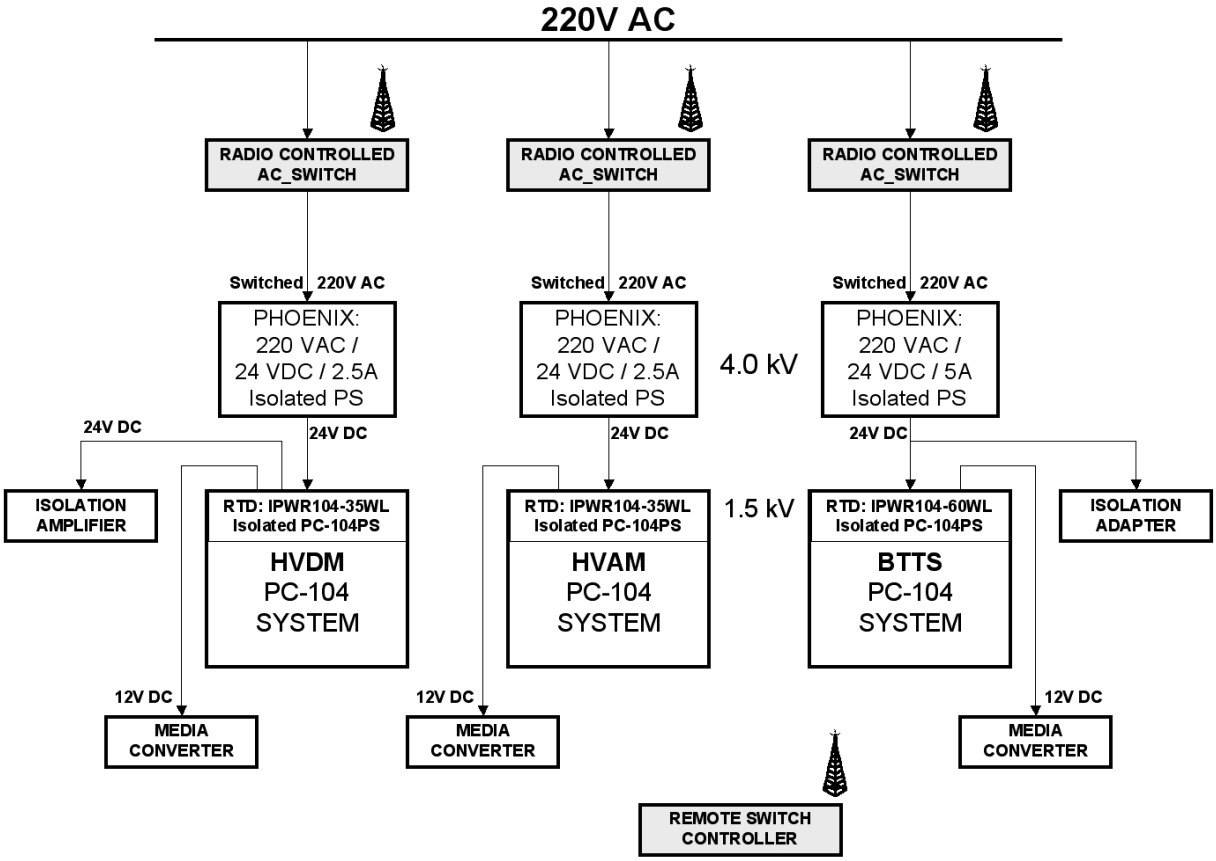

Figure 2. Galvanic isolation of power supplies

Main functions of the measuring stations:

1. HVDM (High Voltages Direct Measurement)
   This station measures high voltages directly at inputs of the ion generator, needed for the calibration, with high voltage probes and isolated amplifiers developed by us. Several safety elements have been built into the divider chain to eliminate the danger of high voltage access to the sensible low voltage measuring inputs.

2. HVAM (High Voltages on Analog Monitor)
   This measures the other part of high voltages needed by the ion source. It receives the signals ready to be processed at the output of the Analog Monitor.
3. BTTS (Beam, Turn-Table and Sensor system)
   Its main functions are:

- to move the 10 cm diameter parallel ion ray and ASPERA-4 put onto the road of neutral components of the ray in the 1.5 m diameter vacuum chamber by a turntable which can be moved and turned in every direction
- to transfer data measured by the sensors to the computer
- to record and transfer measurement parameters (coordinates, pressure, etc.)

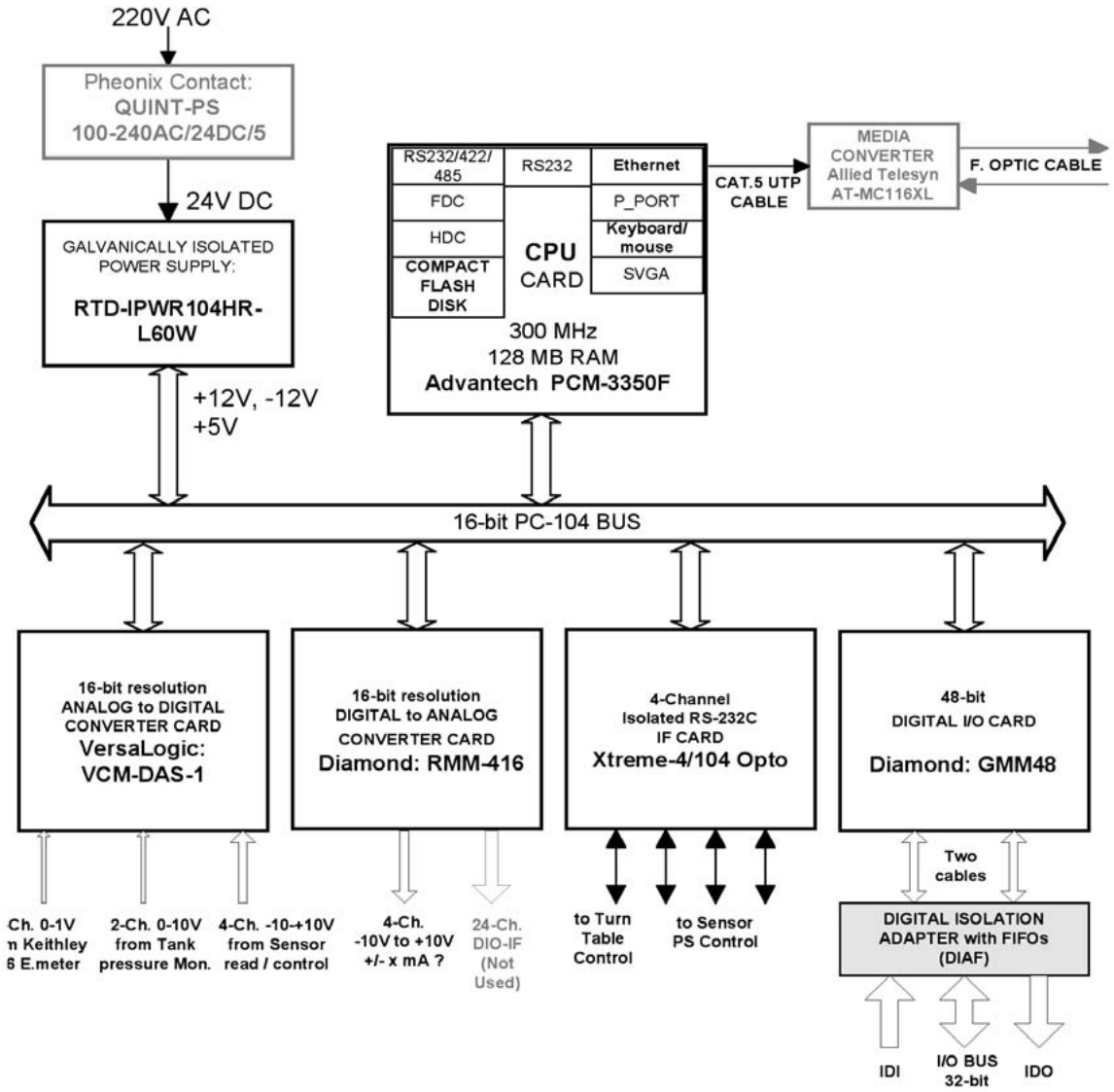

Figure 3. Block diagram of BTTS

The block diagram of BTTS can be seen in Fig. 3. For gathering sensor data from the vacuum chamber, the Digital Isolation Adapter had to be developed which has a 32-bit, 4 Mbyte/s bidirectional, multiplexed or non-multiplexed (selectable) data bus and a 4 kbyte buffer, and ensures the 3.5 kV isolation between devices in the vacuum chamber and the outer world. Owing to its special construction, it provides a 32-bit Isolated Digital Input port (IDI) and a 32-bit Isolated Digital Output port (IDO) too for the PC/104 processor. This property greatly helps in extending BTTS functions in the future which ease the calibration of the next members of ASPERA series or similar scientific measuring devices in a vacuum chamber.

2. Software of the embedded processors

Having studied user requirements, it became clear that such a multi-tasking operating system is needed which ensures even the real-time handling of connected signals but material resources required minimization of costs. Open source code programs are available free of charge for developers but their use needs deep knowledge. Support is provided only by the international programming community available on the Internet. Buying a program, expectable user manuals and manufacturer's help are not available but most likely the problems can be found on the Internet.
Linux operating system is ideally suited for solving multiple tasks but its time-sharing, task-tacting does not guarantee real-time operation. There is an open code RT-Linux which is a real-time kernel providing basic but enough service where the usual Linux environment runs as a lowest priority task. Real-time tasks can be loaded and even removed as kernel modules. It enables directly handling of interfaces, making interrupt handling routines and the use of high resolution timing. RT-Linux is the product of FSMLabs but its support has lately been taken over by Valencia Technical University. Special thanks for Nicholas McGuire alias Herr Hofrat of his unselfish help.
The use of a real-time operating system was especially required because four serial devices should be serviced. It is well known that the response of a reading command issued on a serial line can come earlier than the receiving command could be executed. Of course, not in every case, only when the program is suspended between the writing and reading system call but the chance for this is rather high when multiple programs run on a machine connected to a network. The consequence is the break of communications and the program.
Using a real-time operating system, the handling of serial lines can be entirely controlled by the programmer. Instead of using the standard Linux driver, the data traffic can be optimized by a real-time kernel module. Free area can always be assured for receiving data and the user program should transfer only whole sentences according to the applied protocol. The watchdog timing of communications interrupt can be easier handled as well.
Reading in large volumes of data from the I/O page with precise timing cannot be solved either without the use of a real-time operating system. Several hundred kilobytes should be read out from the interface to the system to be calibrated, and transferred via the Ethernet without data loss. Even the 1 kHz cycle of the analog-digital conversion cannot be realized without the use of a real-time clock.

All the time critical tasks can be done by kernel modules of RT-Linux. The kernel modules view the area of the kernel memory, of course; therefore the addressing range is limited. High speed data can be transferred to the Linux programs through a real-time FIFO interface where they can use with virtual addressing a big buffering area from where they can be transferred via Ethernet with TCP/IP protocol routines. This part of the task is not time critical but RT-Linux is not a wonder, the impossible cannot be realized either. If more data have been gathered than can be stored within a certain time, some data will be lost. Therefore, already at the beginning of the project, it was important to define how much data should be handled and which way.
These limits must be considered when choosing the hardware which fulfils the requirements and also the proper operating system and programming tools for the critical parameters. The most critical point of the ASPERA-4 calibration system was reading-in the high data volume and transferring them through the Ethernet. Between the two processes, a substantial buffer

should be inserted because timing relations of data transfer through the Ethernet are not deterministic. The two-level solution of the tasks ensures that all the data of the calibration are read in with proper sampling while their storage and transfer take place bit delayed in time, balancing load fluctuation.

3. User interface

Calibration process of the sensors is a series of long lasting measurements because the influence of parameters should be defined upon many measurement results. As a characteristic case of automated measuring systems, this is a repeated process in which several parameters are systematically changed for new and new measurement cycles. In the first version of the automated calibration system of ASPERA-4 sensor, its 4-axis position should be controlled while the other parameters, like vacuum value and high voltages for the ion source, had only to be measured and recorded in a diary. The position controlling electronics is a device manufactured in the US, and it uses ASCII characters on an RS-232 interface without any absolute position read-out.
The operator's interface is a C language program under Windows XP. For faster development of the graphic user interface, National Instrument LabWindows/CVI developer's environment was used which has many library functions for this purpose. Communications with the embedded processors is realized via TCP/IP protocol on Ethernet. Positioning is controlled by the nearest embedded processor which passes the PC's ASCII character series without any conversion.
Automated control of positioning, and also of some other setting elements, is done by processing an external file which can be easily read out and edited in XML language. The XML syntax made easy creating and running of files. The position control file can be created not only with a text or XML editor but complete operating sequences can be written into the file as well. Of course, instead of waiting for the end of measurement time periods, data acquisition time can be defined by writing in the proper time values for the given state.

User interface of the ASPERA-4 instrument's automated calibration system is displayed in Fig. 4. Similarly to other data acquisition systems developed by us, the program has two threads. The first thread contains the user interface and graphic display of data while the other contains receiving and storing of data, typically coming in random or burst mode. This 2-thread operation prevents data loss when waiting for a random event in the sensor's data flow. Archiving takes place together with receiving an event in one thread while its display and user interactions are handled in the other one.

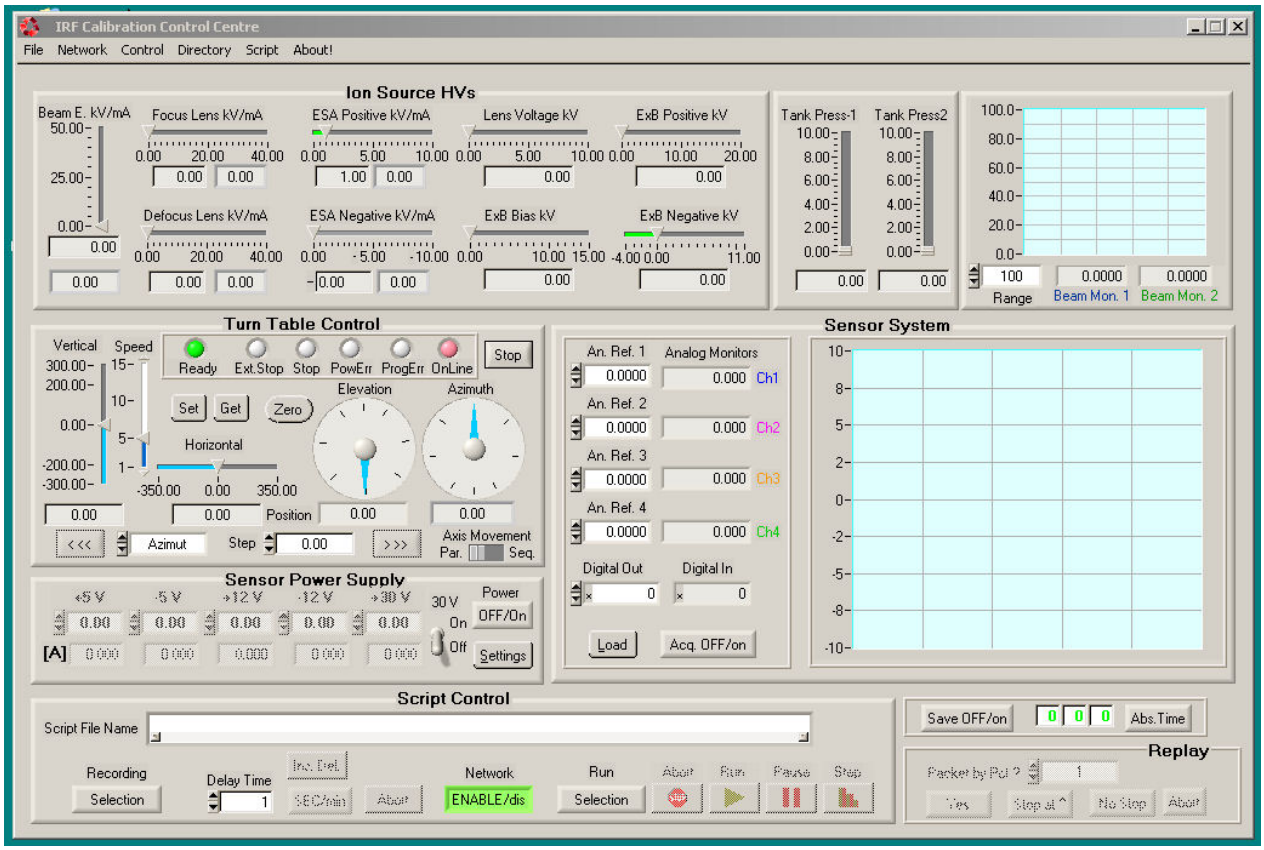

Figure 4. User interface of the Windows program