A new PROFIBUS interface for vacuum sector gate valve controllers

G. Pigny, R. Ferreira, P. Gomes, L. Gyori and M. Roda

CERN, Geneva, Switzerland
Karlsruhe Institute of Technology, Karlsruhe, Germany

E-mail: gregory.pigny@cern.ch

ABSTRACT: The vacuum control systems of the accelerators complex at CERN are based on PLCs, which communicate with controllers either with direct inputs or outputs, or via PROFIBUS. In order to improve the efficiency of the sector valve controller communication, a low cost PROFIBUS interface card has been designed.

This paper presents the developed hardware and firmware, together with the corresponding assessment tests. It flags the improvements of this new interface, in comparison with the former system. Furthermore, this paper can be helpful for any custom design that needs a PROFIBUS interface.

KEYWORDS: Accelerator Applications; Hardware and accelerator control systems
1 Introduction

The vacuum chambers where circulates the beam of CERNs’ accelerators must be maintained under Ultra High Vacuum (UHV) to minimize the beam interactions with residual molecules. UHV is crucial to increase the beam lifetime, reduce the background noise to experiments, and to avoid electrical sparking inside high-voltage accelerator sub-systems (like kicker magnets and radiofrequency cavities).

The beam pipes are divided in several vacuum sectors that can be isolated by vacuum sector valves, to avoid the propagation of leaks over a large volume. Upon the detection of a pressure rise above a predefined level, these valves close within 1 to 3 s, depending on the model. In general, a sector valve is interlocked by several neighbouring pressure measurements, either to make it close or to disable its aperture. The pressure thresholds can be set individually for each gauge. A valve
closed by a pressure interlock will also propagate it to its neighbouring valves, to further isolate the faulty sector.

As a closed valve would be in the beam path, and thus damaged, the beam must be stopped before this may happen. The information of a pressure interlock triggering a valve closure or of any valve losing the open state, is forwarded to the Beam Interlock System (BIS); this way, the beam can be extracted from the accelerator before the valve has time to close [1].

The vacuum control system is based on PLCs (Siemens S7-400 and 300), which access the field equipment (valves, gauges and pumps) through their specific controllers. Modern controllers are often intelligent, as they have an embedded microprocessor or other programmable devices. When equipped with the corresponding interface, they can communicate with the PLC via PROFIBUS; this minimises the complexity and price of cabling and allows a wider exchange of information and configuration parameters.

On the other hand, older controllers are directly connected to individual IO channels on the PLC, like for the current version of the sector valve controller, or connected to remote-IO stations, like for the Sputter Ion Pump Controller [2, 3].

2 Sector valve controller architecture

2.1 Present architecture and communication protocol

Each sector valve is controlled by a valve card within a 3U Europa crate; a maximum of 8 valve cards can be installed in one crate. They are plugged together on the same backplane, which carries a custom parallel bus. Up to 4 crates can be chained together in order to control a maximum of 32 sector valves.

The PLC is connected to the first crate through a 26-wire shielded cable, carrying 24 V digital I/O signals. The interface between the PLC and the valve cards is assured by a multiplexor (MUX) card; this one is also used to forward the signals to the following crates, through a 38-wires flat ribbon cable, carrying 5 V digital I/O signals.

In reference to figure 1, the communication protocol is as follows:

Every valve card stores its status in two Bytes. These can be read on the 8-bit Input bus, when selected by the respective Byte-select bit, which is carried by the 6-bit Output bus.

The MUX card decodes the 6 bit Address from the PLC, and selects a valve card with the corresponding Select bit. The valve card then acknowledges with the FeedbackSelect bit, and puts the appropriate status Byte on the Input bus.

In case of writing, the protocol is the same, and in addition, the MUX card puts the write data received from the PLC on the 6-bit Output bus.

2.2 New architecture

In addition to the need of a large number of wires to connect the PLC to the crate, this way of communication is rather slow; it uses PLC Digital Input (DI) or Digital Output (DO) channels, which are not efficient as a parallel communication bus. Moreover, each set of 4 crates needs one DI and one DO modules. This solution is costly both in PLC-backplane space and in price, comparing to a single field bus module, who could easily connect tens of crates. Finally, this architecture
appears as an exception in the vacuum control systems; the use of PROFIBUS-DP would allow a homogenization of the controls architecture.

Thus, a new architecture is proposed in order to connect the PLC to the crate via PROFIBUS-DP. In this new architecture, depicted in figure 2, each MUX card is connected to the fieldbus and the number of crates is only limited by the number of available PROFIBUS addresses. For a dedicated PROFIBUS network, up to 126 crates could be connected together.

3 Hardware

The new MUX card must now communicate with the PLC via PROFIBUS (as a PROFIBUS-DP Slave device). It has to decode incoming commands, select valve card and write the appropriate command through the parallel bus. It has to repeatedly read the status Bytes of all valve cards, by selecting them one by one, reading the first and the second Bytes of each, and finally sending back the complete status Word to the PLC.

The complexity of the PROFIBUS-DP protocol justifies the usage of a commercial “PROFIBUS-DP-sllave ASIC controller”, the VPC3+S from Profichip. This ASIC must be associated to a microcontroller, with enough digital I/Os to access the valve cards parallel bus, and with SPI bus connectivity to communicate with the ASIC. The block diagram of the new MUX card is shown in the figure 3.

3.1 PROFIBUS-DP slave ASIC

To connect to the fieldbus, an RS485 transceiver is needed; the ADM2486 from Analog Devices has been chosen for this purpose. Both sides of the transceiver are galvanically isolated; an isolated supply is therefore needed to provide power for the PROFIBUS side.
Figure 2. New architecture with the new MUX card.

Figure 3. Block diagram of the new MUX card.

The PROFIBUS communication is managed by the ASIC, who handles the message and address identification, the data security sequences and the protocol processing for PROFIBUS-DP. In addition, it supports the acyclic communication and alarm messages, described in DP-V1 extension.

Furthermore, it also provides the slave-to-slave communication Data eXchange Broadcast (DXB) and the Isochronous Bus Mode (IsoM), described in DP-V2 extension. For high-precision synchronized motion-control applications, the chip is equipped with an HW-PLL for IsoM.

The ASIC provides an automatic recognition of baud-rates up to 12 Mbit/s, integrates the complete PROFIBUS-DP (V0, V1 and V2) protocol, and includes a 4 kByte communication RAM, and a configurable processor interface. The device is operated with 3.3 V single supply, and all inputs are 5 V tolerant [5].
The ASIC can be interfaced through an 8-bit parallel bus, or the serial bus SPI or I²C. Since a parallel bus would use too much I/Os of the microcontroller, the SPI is preferred in this application. The microcontroller is always master while the ASIC is always slave.

3.2 Microcontroller

The core of the new MUX card is a PIC18F6527 microcontroller from Microchip. It needs a 5 V supply voltage, has 54 digital I/Os and integrated SPI capability [4]. For debugging, the tracking of the communication between the microcontroller and the ASIC is possible through an additional connector, accessing the SPI signals and supply voltages.

The software used to develop the application for the microcontroller is the MPLAB X Integrated Development Environment (IDE). To program and debug the microcontroller, the Microchip MPLAB ICD 3 In-Circuit-Debugger tool has been used and connected to an on-board 6-pin RJ12 socket.

3.3 Clocking

The ASIC needs a 48 MHz external oscillator; as it also integrates a clock divider, it can be source of 24 or 12 MHz. This allows the use of a slower microcontroller (which is our case) without additional expenditures.

The maximum SPI clock frequency for the ASIC is 6 MHz. As the microcontroller is able to clock its SPI line with maximum one quarter of its clock frequency, this one must be at maximum 24 MHz. It can be directly taken from the ASIC clock divider output.

3.4 Power supplies

The MUX card is supplied by 5 V through the backplane connector. The line drivers use the same 5 V since the parallel bus towards the valve cards is only 5 V compatible. The microcontroller operates on 5 V; the ASIC on 3.3 V, but the pins are 5 V tolerant, making it possible to connect directly to the microcontroller without any level shifter. The oscillator for the ASIC needs 3.3 V supply. Additionally, an isolated 5 V supply is needed for the PROFIBUS side of the RS-485 transceiver.

3.5 Power supplies supervisor and reset

For the safe and well-defined reset behaviour of the microcontroller and the ASIC, in case of a brownout reset, it is advantageous to use voltage supervisor circuits. Such circuits monitors the supply voltages, and in case of a voltage drop generate a long-enough reset pulse, to assure a complete restart of the ASIC and microcontroller.

3.6 User interface

Figure 4 shows the prototype for the new MUX card. Four status LEDs are placed on the front panel to give basic information about the card, like: 5 V and 3.3 V supplies presence, data being exchanged over PROFIBUS, and reset in progress. There is an additional Debug LED in the middle of the PCB (not on the front panel), for various debugging options.
Figure 4. MUX card prototype.

There is a reset button accessible only through a pinhole, in order to avoid accidental resets. It resets only the microcontroller; the ASIC must be reset by the microcontroller software. The PROFIBUS address can be set by two hexadecimal rotary switches.

4 Firmware

4.1 PROFIBUS-DP

According to the PROFIBUS-DP communication standard, after a power-on several conditions must be checked prior to start exchanging data. This implemented in the ASIC by a state machine [6, 7]. As the information about the current state is used by the microcontroller, we will next summarise these ASIC states.

Power ON state. Following a power-up, the slave will receive a telegram from a master asking for its address. The slave will be held in this state if it does not have a valid. After completion of its power-on initialization routine, and if the address is valid, the slave will proceed to the Wait for Parameterization state.

Parametrization state. In this state, the DP slave awaits the parameterization telegram from the master which identifies the slave’s master and the mode the slave is to operate in. A slave in this state will reject all other telegrams except a request telegram for diagnostics or configuration. After its parameters have been set, the slave will proceed to the I/O Configuration State.

I/O configuration state. In this state, the slave awaits a configuration telegram that specifies the number of input and output bytes that are to be exchanged in each data telegram cycle with the slave. The configuration telegram also causes the slave to check the configuration which was sent against the stored configuration. A slave in this state will accept a request telegram for diagnostics or configuration, or a set parameters telegram.
Data exchange. After parameterization and configuration have been accomplished, the slave cyclically exchanges I/O data with the master. This is a cyclic transfer of I/O data and possible diagnostic information.

4.2 Core functionality

Once the ASIC is in *Data Exchange* State, the main loop is executed. A simplified version of the communication cycle executed in the main loop is depicted on the flowchart in the figure 5.

Each iteration starts by checking the current PROFIBUS state. In case of Error, the execution stops. In the *Data Exchange* state, the input data is gathered. Otherwise, the execution skips to handling the output data, to where the other branch also proceeds after sending the inputs to the ASIC. The output is only handled if there is new data present.

For gathering the input data that is reading the statuses of the valve cards, the following sequence is played for all 8 cards, one-by-one after each other. First, the MUX card waits with a timeout for the selection acknowledge (*FeedBackSelect*) signal to become non-active, which indicates that no
card is falsely connected to the read data bus. Status Byte 0 is then selected by the corresponding bit, and only then the card itself, in order to avoid any hazards caused by timing or racing conditions.

After the selection, the MUX card again waits (with a timeout) for \textit{FeedBackSelect} to become active, indicating the valve card is present and has put the data on the read \textit{Input} bus, thus it can be read by the MUX card. The deactivation of the \textit{Select} signals of all cards follows, after which the whole sequence starts over for reading status Byte 1. Then, the same is repeated for the other valve cards.

\section*{4.3 GSD file and protocol for PLC master}

The “General Station Description” file defines the PROFIBUS slave (in this case the MUX card) for the master (the PLC). It is an electronic device data sheet or device data base file that identifies the PROFIBUS device. All PROFIBUS devices have their own GSD files.

For the purposes of this project, it is necessary to define two input status Bytes and one output Byte (i.e. 6 bit output + write signal +1 leftover) for each valve card. In one crate (i.e. one PROFIBUS address), up to 8 valve cards can be present. In total, 16 input bytes and 8 output bytes are necessary for one crate or MUX card. The input and output data length must be defined in the GSD file.

\section*{5 Integration and test}

To test the produced hardware and firmware, a set of debugging levels has been programmed. The tests have been designed to start with the bare minimum and then incrementally testing bigger parts of the hardware, of the code, and of the whole functionality. The more significant tests are described in the following sub-chapters.

\subsection*{5.1 SPI communication tests}

\textbf{SPI functionality test.} A constantly increased counter is periodically written to the SPI. An SPI analyser tool (Total Phase Beagle) is connected to the provided header on the card, making it possible to see, log and analyse the SPI communication on the PC. This test verifies that the SPI functionality of the board (hardware) and of the microcontroller (firmware) is set up and working properly.

\textbf{SPI communication test between the microcontroller and the ASIC through its memory.} The communication between the PROFIBUS ASIC and the microcontroller is tested here, with the built-in memory test function of the ASIC. Data is written to the whole memory area of the ASIC, then read back and verified. This test verifies, that the communication protocol of the ASIC is implemented correctly in the microcontroller firmware, and that the complete SPI communication with the ASIC is working flawlessly.

\textbf{SPI communication and signal integrity test for all available frequencies.} In order to test all the different supported SPI frequencies, the previous test is run repeatedly in an infinite loop to allow continuous measurement with an oscilloscope. The test is to be executed separately with the different SPI clock settings. On the microcontroller, the three options are: clock/4, clock/16 and clock/64, resulting in 6 MHz, 1.5 MHz and 375 kHz SPI frequencies respectively, if clocked by 24 MHz as it is the case in this setup.
5.2 PROFIBUS communication tests

Unidirectional PROFIBUS communication test (MUX card → PLC). A much larger part of the provided code-frame is used in this test. The ASIC builds up the PROFIBUS connection, and in place of the real input data, it sends a constantly increased counter. The sent data are verified on the PLC with the Siemens Step7 software, after setting up the hardware configuration by using the Monitor/Modify function of the inputs and outputs of the DP slave, or using a Variable Table (VAT). This test verifies, that the communication channel is working from the microcontroller all the way to the PLC.

Bidirectional PROFIBUS communication test (PLC ↔ MUX card) at all baud rates. The input data sent to the PLC in this test is the modification (multiplication by two) of the output data received from the PLC. Thus, it is to verify on the PLC that the sent output and the received input values correspond. As this test is checking the communication in both directions, the testing of all available baud rates has been performed here. The desired speed is to set up in the Step7 software in the properties of the DP interface connection. This test verifies that the complete write-read cycle between the PLC and the microcontroller is working properly at all baud rates.

5.3 Complete system tests

Commands translation and signals generation tests (PLC → MUX → valve card). In the first test using the final code, without any valve cards connected, commands are sent from the PLC through the output Monitor/Modify. The bus signals generated by the MUX card are monitored with the oscilloscope. This test verifies, that the MUX card generates the correct signal protocol on the bus towards the valve cards for writing.

Command and status reading tests (PLC ↔ MUX ↔ valve card). With the valve cards connected, the writing functionality is tested by sending commands from the PLC through the output Monitor/Modify, and checking the actual behaviour of the cards/valves. The different states and changes of all status bits for each card are also checked. This test verifies, that the writing functionality is working properly from the PLC to the valve cards and that the statuses appearing on the PLC represent the real statuses of the cards.

Benchmarks. Several benchmarks have been implemented to characterize the behaviour of the new MUX card:

The first one was aimed to determine how long time the waiting function of the MUX card must wait in reading and writing mode. It turned out that no additional timer has to be set due to the very fast response of the valve cards. The time needed to call the waiting function (~10 us) is sufficient.

Then, a benchmark to measure the time needed for the MUX card to read all 8 valve cards has been implemented. The time to read the statuses of all cards takes ~261 us.

Another benchmark to measure the time needed for the MUX card to make a complete read and write cycle in different configurations according to the number of cards has been implemented. The result is that a complete read and write cycle takes less than 1 ms (typically between 700 us and 1 ms).
6 Conclusion

This new design of the MUX card allows to connect the valve controller crates directly to PROFIBUS, replacing the complicated and slow data exchange through several dedicated DI/DO channels. This provides: a reduction of the number of cables and PLC cards, and an increased data exchange speed, while simplifying and homogenising the architecture.

Regarding the performance, the present version of the MUX card needs 30 to 50 ms to read or write one single status Byte or command into one single valve card. With the new version of the MUX card, the cycle period is less than 1 ms. In addition, all valve cards in one crate can be read and written within a single cycle and it takes only 1 ms instead of 880 ms. Faster sampling period is important particularly for data logging of interlocks and beam dump request.

References

[1] G. Pigny et al., Measurements, Alarms and Interlocks in the Vacuum Control System of the LHC, at International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS15), Melbourne Australia (2015).

[2] P. Gomes et al., The Control System of CERN Accelerators Vacuum, at International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS11), Grenoble France (2011).

[3] M. Roda et al., A New Profibus-DP Interface for CERN’s Sputter Ion Pump Controllers, at 10th IEEE International Symposium on Industrial Embedded Systems (SIES 2015), Siegen Germany (2015).

[4] Microchip Technology Inc., PIC18F8722 Family Data Sheet, U.S.A. (2008).

[5] Profichip GmbH, VPC3+5 User Manual, Revision 1.06 (2012).

[6] Acromag Inc., Introduction to Profibus DP, U.S.A. (2002).

[7] Profichip GmbH, VPC3+ Software Manual, Revision 6.00 (2012).