Family of Smart Devices on the Basis of DSP for Accelerator Control.

A.S. Chepurnov, A.A. Dorokhin, K.A. Gudkov, V.E. Mnuskin, A.V. Shumakov

Institute of Nuclear Physics, Moscow State University.
119899, Moscow, Russia,
chas@rtm-cs.npi.msu.su, (095)939-56-59

Abstract.

There are many modern technologies of electronic equipment design which ensure quick and cost effective solutions for the object-dependent level of hierarchical computer control systems. On the basis of the MIL-STD-1553B fieldbus, digital signal processors (DSP), reconfigurable FPGA and RISC microcontrollers a new family of devices has been designed. Devices can be connected in different configurations via MIL-STD-1553B fieldbus, RS-485 and RS-232 interfaces. The intellectual ability and functionality of fthe amily members depend on concrete applications. Complex control algorithms, including direct digital feedback control and data processing, can be implemented by using a powerful fixed-point DSP. Distributed control and data acquisition systems could be developed based on this device set. These devices are placed in stand-alone small boxes. The main spheres of application are control systems of large experimental installations, data acquisition, "hard" real time systems and industrial control. Modules have been used for industrial accelerator control and for upgrading a CW racetrack microtron computer control system.

INTRODUCTION.

The idea to generate the family of devices described below appeared as a result of 10 years work by the authors in the field of designing different control systems for industrial and scientific applications and instrumentation.

The basic principles of construction have been chosen - concrete fields of application for the devices; general architectural principles appropriated for different control tasks; modern microelectronics component base among the dramatically large number presented today on the market; tools for component interconnection; hardware and software architecture.

During the last one and a half years this work has been implemented and the first "members of the family" have appeared. The devices form the basis for control system of an industrial accelerator [1] and they will replace parts of the racetrack microtron injector system [2,3]. Then the control system of the whole racetrack microtron will be based on these devices.

GENERAL ARCHITECTURE.

This family of smart devices is designed for the construction of distributed multilevel embedded applications according the specification of "hard" real-time systems. Therefore devices should be very reliable, have the ability to use redundancy and, at the same time, allow the implementation of complex control algorithms including direct digital dynamic stabilising. To increase the versatility of the system components scalable hardware should be realised. The Open Systems concept requires that interconnection between components should be via a standard interface: the fieldbus. The devices cover the low and intermediate levels (nearest to the control equipment) of multilayer control systems. But, it is possible to connect a low-level system directly to supervisor and operator levels based on standard software and hardware via the standard interfaces supported by our devices. In other words, in standard, well known architectural solutions, our devices in many cases, but not in all, could replace the intermediate and low levels traditionally formed with the help of different universal expensive crate-based systems (CAMAC, VME, VXI) and industrial PLCs and ensure a cost-effective technical solution. Features of the described devices should reinforce this quite questionable statement.

In figure 1 examples of possible architectural solutions and applications of smart family devices are shown. One of the basic principles taken into account during the design is that: *intellectual resources responsible for the handling of the control task in real-time should be spread as evenly as possible and feedback loops should be closed locally and digitally, where possible.*

Interfaces.

Interconnection of different parts should be as standard as possible. At the same time supervisory functions from the top level of the control system should be fully supported to ensure access to the lowest level of control algorithms. Therefore the following interfaces have been chosen for interconnection: MIL-STD-1553B, RS-435 and RS-232C.

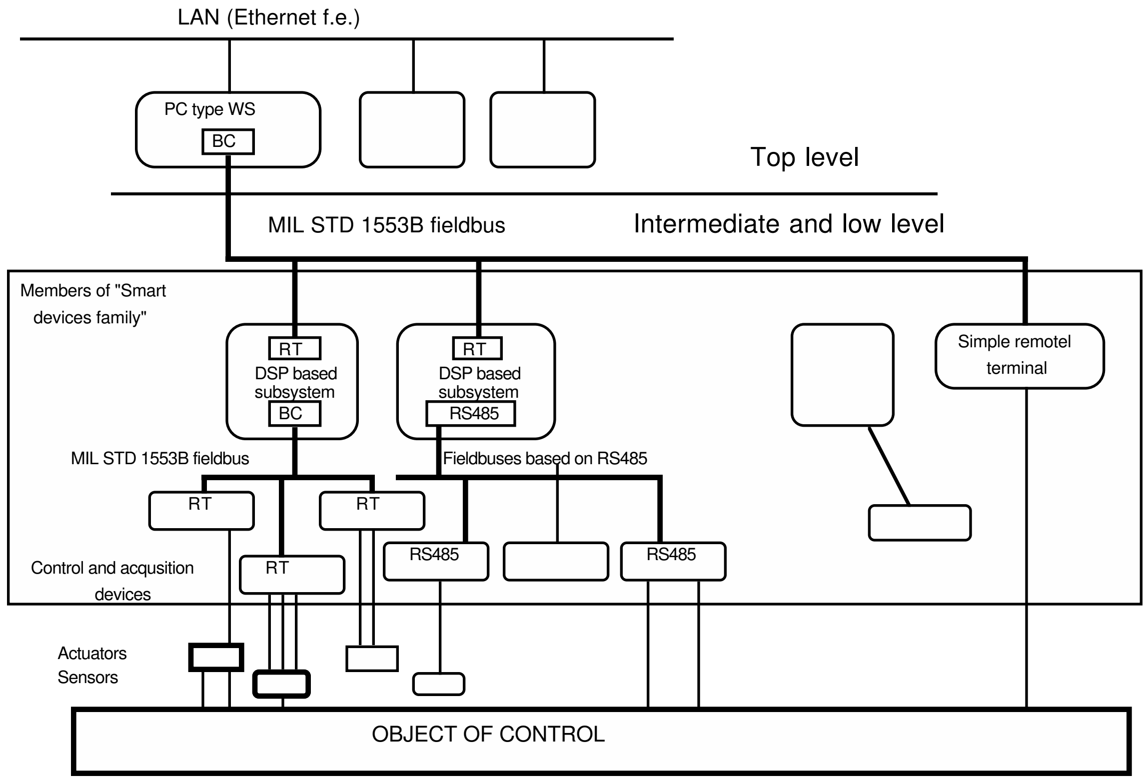

Figure 1. Location of the smart devices family in the standard control architecture.
MIL STD 1553B bus elements: RT- remote terminal, BC - bus controller.

The MIL-STD-1553B fieldbus interface is very suitable for "hard" real-time applications - aerospace onboard control, alarm interlocking, etc. Moreover it was designed for such applications and there is nothing comparable which is currently standardised. Many of the functions supporting high reliability are supported at the hardware level. This interface is very widely used in different accelerator control systems [4-7]. Another reason for our choice is that the interface is not message-passing oriented but oriented to blocks of subsystem memory exchange. There are some disadvantages to the interface - relatively high cost, low data transmission rates, limited address field in the standard message and the small number of remote terminals connected to one branch.

The RS-485 physical interface has been chosen as an additional one to ensure compatability with a great number of industrial PLCs available on the market. Features of our hardware architecture ensure the support of different logical levels of fieldbus such as BITBUS, PROFIBUS, S-BUS and others [9]. Different "smart sensors and actuators" physically placed near the control object could be connected via this interface. The appearance of very cheap RS-485 transceivers with galvanic isolation lends support to the validity of our choice. The basic principle that *data should be processed as near to the place of generation as possible and lthe ength of any wires with analog signals should be as short as possible* is implemented in this way.

The very well-known RS-232C interface could be used for connections with standard instrumentation equipment and for local testing and reconfiguration of "smart devices". In addition, an implementation of RS-232C allows the use of the device for small experiment automation, increasing the universal character of device application.

Planned for the design are special adapters between two different standard interfaces - MIL-STD-1553B and IEEE-488, for example, to allow the connection of standard powerful test equipment

SMART DEVICE HARDWARE.

It is a tragedy to any active working designer in the field of microelectronics equipment that every day fresh chips with higher and higher performance and cheaper and cheaper costappear ...

We had to choose at least three base classes of components together with appropriate instrumentation and CAD systems in order to design this family of controlling devices - base microprocessor, base microcontroller and base type of programmable logic devices. The right choice should provide a fast and effective design process.

As a base processor for general and special applications in "smart devices" the TMS320C5x generation of fixed point DSPs of Texas Instruments have been chosen. These DSPs are fabricated in accordance with static CMOS technology. The combination of an advanced Harvard architecture (separate buses for program memory and data memory), additional on-chip peripherals, more on-chip memory and a highly specialised instruction set is the basis of their operational flexibility and speed. The third generation of fixed-point 16-bit processors has a very good performance and characteristics: 35-50 ns single-cycle instruction execution time and can execute more than 28 MIPS. The C50x DSP processor can have up to 9K words of on-chip data and program RAM, up to 16K words boot ROM, JTAG scan path, one or two serial ports and a maximum address space of 64K words for data, 64K words for program and 64K words for I/O ports [10].

As a base microcontroller the PIC 16/17 product line from Microchip has been chosen. PIC 16/17 microcontrollers are RISC based on Harvard architecture.

As a base family of programmable logic devices, EPLDs and FPGAs (Electronically Programmable Logic Devices and Field Programmable Gate Arrays) from Xilinx has been chosen. Cheap EPLDs or FPGAs with small numbers of gates are very suitable to replace conventional logic chips used as glue logic in complex microprocessor devices. Quite expensive FPGAs with a large number of gates up to 20000 could be used for the silicon implementation of complex control algorithms or for interface functions such as MIL-STD-1553B interface including codes, message processing and answering blocks.

Automatic adjustment of the parameters for analog channels, such as zero shift, can be used with a special scheme of analog to digital and digital to analog channels.

The internal structure of the base processor control device is shown in figure 2. It consists of the processor core, input/output system and external interfaces.

The microprocessor core.

The microprocessor core consists of fixed-point, 57MHz TMS320C51 processors, an external boot ROM, optional fast static RAM separate for data and program, a control logic block and a crystal oscillator.

The low level software is activated after power-up or system reset and supports stand-alone operation and/or loading of the control program from ROM, a with maximum size of 8K words. Additional software can be loaded via an external interface (MIL-STD-1553B at present). Optional RAM for data and/or program can be mounted if the control algorithms require it. The maximum size of external RAM can be 32K words for data and 32K words for program.

Input/output system.

Information from the external world is received in analog and discrete digital form. Control signals can be generated in the same way. Sixteen differential or 32 non-differential analogue signals passed through input passive filters feed a multiplexer. The signal after the multiplexer is amplified and goes to the input of a 14-bit fast ADC. The instrumental amplifier on the multiplexer output has programmable 4-step gains. Up to four DACs with settling times of 1.5 µs are used for analog control.

Direct digital control algorithms could be easily implemented by digital closing of the feedback loop. The number of effective bits in all operational scales is not less then 12, which ensures an accuracy of measurement good enough for most of the analogue parameters in control systems.

The input/output subsystem is situated in the I/O region of the processor memory and is presented as register-oriented devices.

External interface subsystem.

The MIL-STD-1553-B interface is based on a special hardware scheme. It consists of parallel to serial and serial to parallel codes generating serial data (Manchester II), additional control logic and a TTL to bipolar converter hybrid scheme for feeding a special pulse transformer, connected to twisted shielded pair.

The MIL-STD-1553B interface is situated in the I/O region of processor memory and presented as a register-oriented device. A complete set of Remote Terminal functions is built into the control software.

A full duplex on-chip serial port on the DSP provides direct communication with serial devices. The serial port is fully static and thus will function at arbitrarily low clocking frequencies. In order to realise a complete RS-232C interface, simplify hardware and decrease processor loading, a PIC16C54 microcontroller is used. It is used for data buffering and transmitting and receiving additional control interface signals. An additional system function implemented by the microcontroller is a system watchdog.

A second on-chip serial port is a TDM (time-division-multiplexed) one. It allows the device to communicate serially with up to seven other TMS320C5x devices. The port can be configured in either multiprocessor or stand-alone mode. In our case the TDM serial port operates stand-alone. The RS-485 transceivers are directly connected to serial port lines. Different configurations of external connection (one, two or three line) can be used, depending on synchronisation mode and the demands of the logical part of the interface.

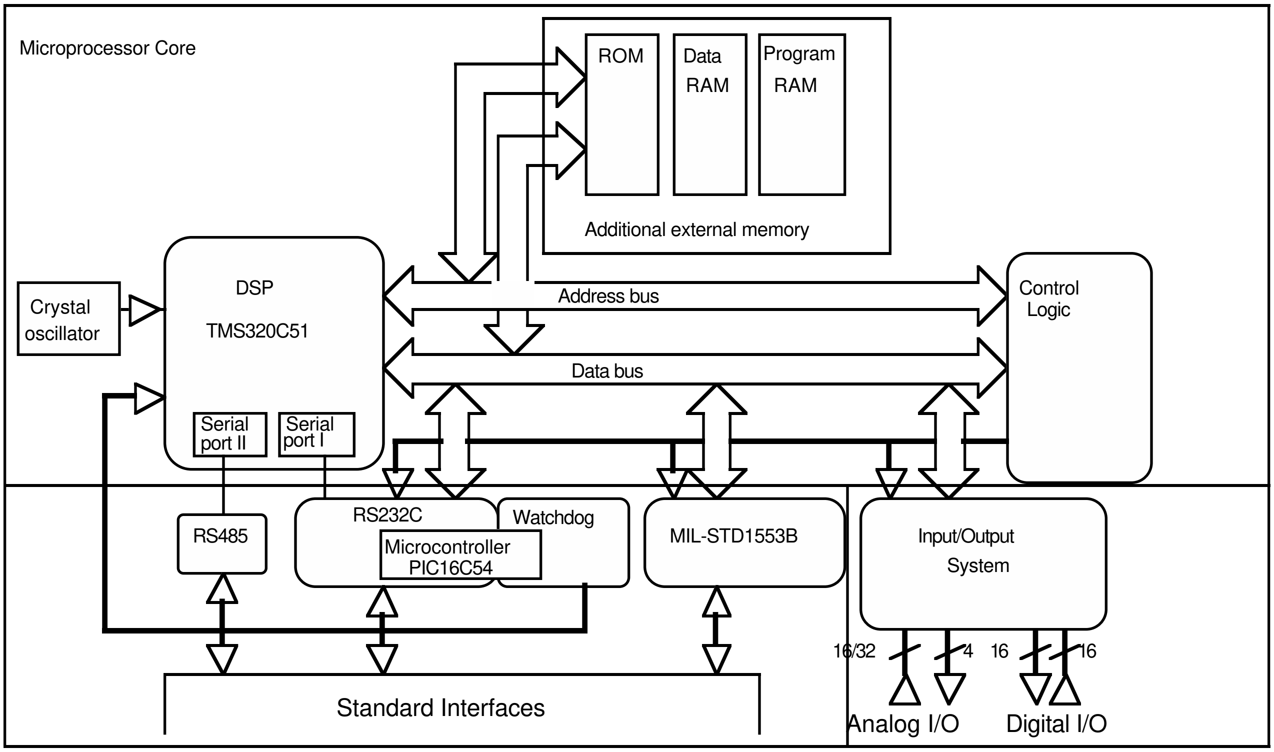

Figure 2. Internal structure of base processor controlling device.

An example of simple control devices.

An example of one possible simple control device is described below. This device is used for the simultaneous control by four stepping motors for positioning an experiment's target in space [11]. The main and practically only element of the device is the microcontroller PIC16C57 [12]. The PIC16C57 single-chip microcontroller is a low power, high speed, full static CMOS device containing ROM, RAM, I/O and a central processor unit on a single chip. It has a high-performance RISC-like 8-bit CPU with a 200 ns instruction cycle. The architecture is based on a register file concept with separate buses and memories for data and instructions (Harvard architecture). The data bus and memory are 8-bits wide while the program bus and program memory have a width of 12-bits. Peripheral features consist of 20 I/O pins with individual direction control, watchdog timer, power saving mode etc. The performance and peripheral features of the microcontroller are enough to generate correct phase signals for the stepping motors and to receive and transmit data via the RS-232C interface. The structure of the device is shown in figure 3.

SOFTWARE SUPPORT.

Software development for a control system is one of the most difficult and expensive parts. It is a peculiar feature of our times that the speed of hardware design is significantly raised by the use of powerful CAD tools. Yet software design, especially for embedded and real time applications, is still a very hard and slow process.

This smart device family has been designed to realise our intent to implement the use of the shared memory idea for control applications. We expect this approach to seriously simplify application object-specific software. The hardware will be more complex, especially for a communication system supported by a shared memory mechanism [13].

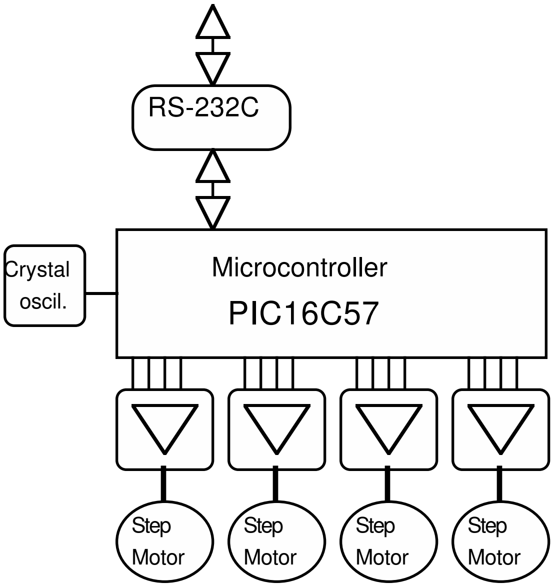

Figure 3. Structure of simple device for step motor control.

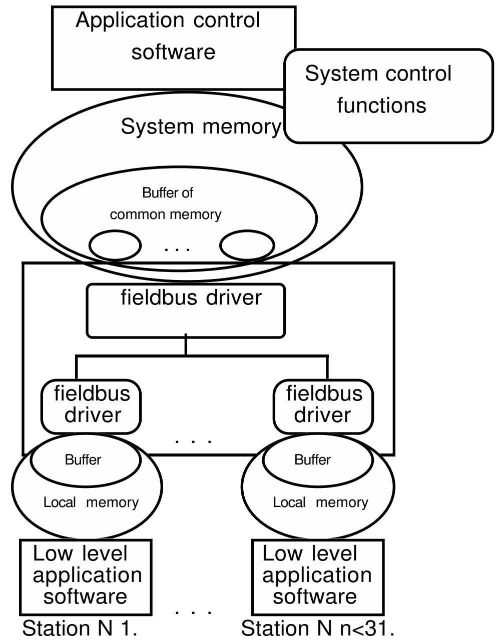

Figure 4. Structure of software system organisation with sharing memory approach.

According to this approach, high level application software knows nothing about the structure of the low level hardware. The message-passing method of data exchange between subsystems and processes is avoided. Application software is operating on a high level and directly accesses data and processes reflected in the memory of high level system. Moreover, application software doesn't check data validity; it presumes that data always correspond to reality. These data are the result of low level software operations on the powerful low level DSP-based devices and transmitted by the specially organised fieldbus, which ensures the validity of the data in the memory of the high level system. Due to the special architectural features the following idea is realised - *all the data reflecting the state of a control object is processed and transmitted for storing and supervising at the rate of its physical generation*. And vice versa, if the control system were to control the object, it could generate a reply, write it to memory and forget it, because it could be sure that the signal would influence the right object at the right time. It is necessary to know only the real physical nature of the received and transmitted (controlling) signals.

The general structure of the software is shown in the figure 4.

High level software support.

An IBM PC compatible computer has been used for studying and testing high-level software, supporting the shared memory approach. An ISA-bus-based MIL-STD-1553B interface card has been used.

Application software consists of different control, supervisory and data base algorithms and depends on the controlled object and control algorithms. The application software is independent of the structure of the controlling hardware, the type and number of actuators and the data acquisition channels. It operates only with the reflection of the external world in correct blocks of shared memory. This approach continues the tradition of the software that has been designed and is operating today in the control system of the racetrack microtron injector [2]. Man-machine interfaces and data base functions can be implemented in this part of software.

High level software can be written without special knowledge of real-time programming technology and the hardware structure of the control system. Programmers should know the physical properties of the controlled objects and algorithms and the rules of correct object operation. The one thing that should be done is the correct initialization of the driver supporting the memory reflecting mechanism.

The bus controller functions of the MIL-STD-1553B standard are supported in hardware. In order to implement the memory block reflection, a special card-oriented driver has been written. It ensures software support of the shared memory approach and compensates for the disadvantages of using a new interface card. It is necessary to point out that MIL-STD-1553B is very suitable for the implementation of the memory-shared mode of operation. Data block exchange instead of message-passing is one of the most important features of the interface.

A special interface card that supports in silicon many of the interface and driver functions is under design now based on FPGA technology.

System control and configuration is the most difficult part of the high level software. It can be written only by a person knowing the controlled object as well as the system structure. This software handles alarm interrupts and critical loops. With the shared memory approach, it is possible not only to distribute data but to dynamically configure low level software, to control the real-time processing of the low level systems.

Low level software.

The structure of the low level software is shown in figure 5.

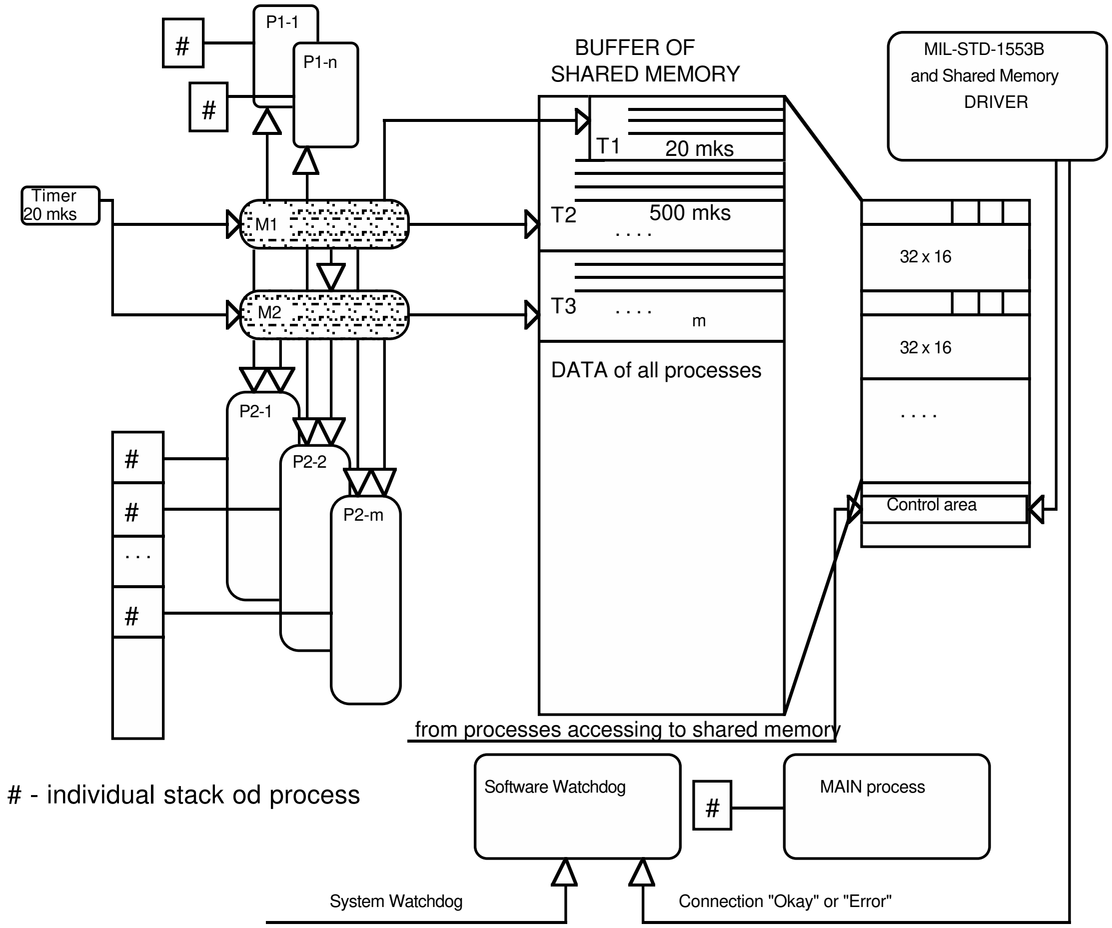

Figure 5. Structure of low level software.

The low level software system is based on monitors and processes. Memory is divided into a few blocks:

- shared memory, accessible from the top level;
- system data memory, not accessible from the top level;
- program individual process stack, accessible only for the process which is the owner of the stack.

According to the MIL-STD-1553B interface specification, 32-word length blocks could be exchanged during one cycle of network operation. Every 32-word memory block has an additional first word reserved for establishing the correct access to shared memory and is processed by the network driver.

Five different process types are used. The following three types of process exist in one copy:

- a network and shared memory support driver;
- a "software watchdog" to ensure overall control and check time synchronisation and process validity;
- an asynchronous background process for system control

They should be configured during program compilation and couldn't be dynamically changed during the system operation.

The network and shared memory support driver is the most critical part of this software because it must ensure the correct fieldbus time cycle operation during message exchange and ensure the correctness of data reflection in the common shared memory. Maximum loading of the processor by the fieldbus driver is less than 25% when the fieldbus loading is at a maximum. It should be pointed out that the remote terminal functions of the interface are implemented in software. Actual loading during normal operation is less than 15%.

Two types of synchronous processes exist. The first type (P1) is a single cycle "fast" process. The second type (P2) is a multi-cycle "slow" process. This process operates under the control of two monitors. The monitors are synchronised by a 20 µs cycle timer. A rate monotone scheduling algorithm is used [14]. The first monitor (M1) only initiates processes. The second monitor (M2) switches processes and stops them.

There are three description tables supporting monitors, operation and the scheduling algorithms. The first table (T1) is a description table for the first type of process. The "time step" of table scanning by the M1 monitor is 20 µs for P1 processes. The length of the T1 table is limited in today's realisation to 500 µs. The total sum of P1 type process times should be less than 500 µs. The number of processes is currently 25. T1 is a first "special" element of the description table T2. It is scanned by the M1 monitor element-by-element with a time step 500 µs. According to this table P2 processes are initiated. The length of the table is limited by the maximum number of implemented processes. This table can be dynamically changed. The T1 table can only be changed after stopping all P1 type processes. Due to this mechanism, P2 type processes could be loaded and initiated via the fieldbus and stopped and unloaded without stopping system operation. Access to this table via shared memory ensures dynamic control of the initiated process.

Information about all initiated P2 type processes is stored in the third description table - T3. This table is scanned by the M2 monitor with a 500 µs step and stopa a process if its time is over.

According to a special time cycle (currently 1s), a "software watchdog" checks the state of the software system and the number and state of all initiated processes. If there are uncontrolled or excessively time-consuming operating processes, they are destroyed. The "software watchdog" handles interrupts from a hardware watchdog and the network interface (for connection failure) for correct process ending and switching off of the system.

Examples of P1 type processes are: reading data from one of the input ADC channels, internal clock, one step of a dynamic stabilizing algorithm and processing of digital inputs. Examples of P2 type processes are the control of motor movement and current scanning in a steering element.

All low level software is written in C50 assembler.

IMPLEMENTATION EXAMPLE.

The systems described above can be used in the following applications:

- distributed control systems for complex experimental installations;
- distributed data acquisition and processing systems with low and middle data rates;
- scientific experiments automation;
- industrial automation;
- stand-alone industrial controllers.

On the basis of the described hardware and software approaches the control system for the industrial accelerator CWELL 0.6/6.0 have been designed and implemented [1] . This accelerator is used for cacao powder processing with an electron beam. The output energy of the continuos electron beam is 0.6 MeV and the output beam power is 6.0 kW. The structure of control system is shown in figure 6.

One "smart DSP-based" device enough for the implementation of all the "hard" real-time algorithms. Moreover, direct digital feedback loops for high voltage and RF power stabilization are implemented. Based on cheap and well known hardware, the PC-type computer is used for man-machine interface support and storing of operational data.

This technology will be used soon for upgrading the control system for the Moscow racetrack microtron injector [14].

CONCLUSIONS.

The described "family of smart devices" is one more set of tools for the full scale automation of different types of objects. All the features and advantages of the set become evident during hard real-time system implementation. The shared memory approach simplifies high-level application software whereas complexity of the interconnection interface is increasing.

ACKNOWLEDGEMENTS.

The authors wish to thank ICALEPCS'95 Chairman Dr. Peter Lucas and the Organising Committee of the conference for the financial support for conference participation which initiated the appearance of this article.

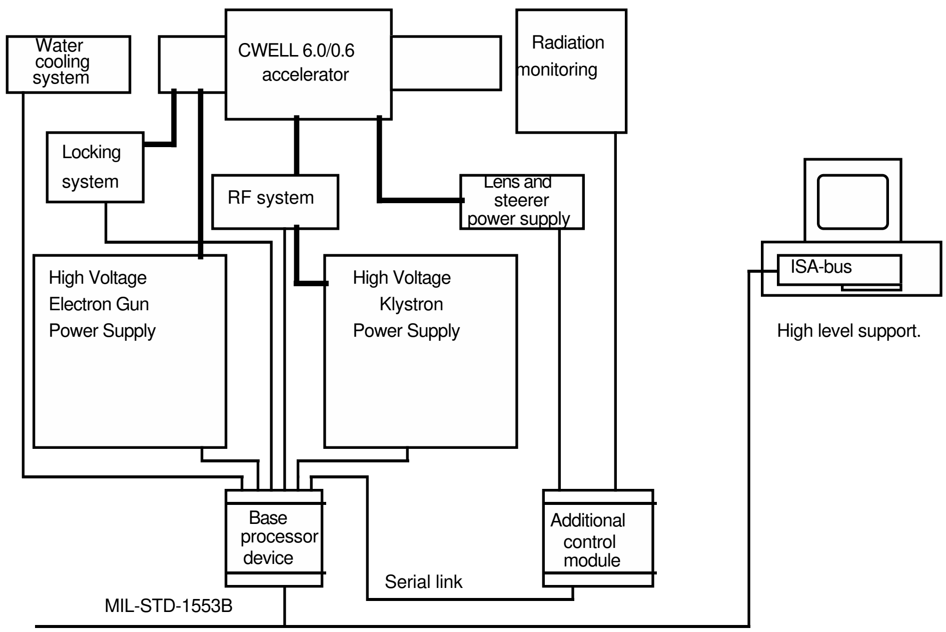

Fig.6. Structure of industrial accelerator control system.

REFERENCES.

[1] A.S. Alimov, K.A. Gudkov et al. Experimental Study of a Prototype High-Current CW Linear Electron Accelerator // Instruments and Experimental Techniques, Vol.37, No.5, Part 1, 1994

[2] A.S. Chepurnov, I.V. Gribov, et al., Moscow University Race-Track Microtron Control System: Ideas and Developments //Proc. Int. Conf. on Accelerator and Large Experimental Physics Control Systems (Tsukuba, Japan, KEK, 11-15 Nov.1991) Tsukuba, KEK,1993 pp.140-142.

[3] A.S. Chepurnov, I.V. Gribov, S.Yu. Morozov, et al. Systems for Local Control of Race Track Microtron Accelerating Section. //Proc. Int. Conf. on Accelerator and Large Experimental Physics Control Systems (Tsukuba, Japan, KEK, 11-15 Nov.1991.) Tsukuba, KEK,1993 pp.424-426.

[4] P.G.I nnocenti. The LEP Control System: Architecture, Features and Performance., Particle Accelerators, 1990, Gordon & Breach, Science Publ. Vol.2,pp.183-190

[5] M.V. Tyrrell. The LEP Alarm System. //Proc. Int. Conf. on Accelerator and Large Experimental Physics Control Systems (Tsukuba, Japan, KEK, 11-15 Nov.1991.) Tsukuba, KEK, 1993 pp.254-260

[6] M.R. Shea, R.W. Goodwin, M.J. Kucera and S. Stirbu. ARCNET as a Field Bus in the Fermilab Linac Control System./Proc. Int. Conf. on Accelerator and Large Experimental Physics Control Systems (Tsukuba, Japan, KEK, 11-15 Nov.1991.) Tsukuba,KEK,1993, pp.291-294.

[7] D. Bulfone, P. Michelini et al. The ELETTRA Field Highway System./Proc. Int. Conf. on Accelerator and Large Experimental Physics Control Systems (Tsukuba, Japan,KEK,11-15 Nov.1991.) Tsukuba,KEK,1993 pp.313-317

[8] R. Rausch Tutorial on Standart Fieldbuses Application in Physics Laboratories, Invited Tutorial Paper ICALEPCS'93, Berlin,1993.

[9] TMS320C50 User's Guide //2547301-9721 revision D, January 1993, Texas Instruments Incorporated.

[10] A.S. Alimov, V.K. Grishin, et. al. Studying of coherent mechanisms of X-ray generation on the electron accelerator NPI MSU. Preprint No 95 - 23/387. Moscow 1995. - 24 p.

[11] Microchip Data Book 1994 Edition// April 1994, Microchip Technology DS00018G.

[12] V.I. Vinogradov,. A.S. Chepurnov, K.A. Gudkov, A.V. Shumakov. Modern Architectural Solutions on Automation and Computer Control Systems for Accelerator and Other Objects (These Proceedings).

[13] A.D. Ferreri. Real-Time Scheduling Algorithms.// Dr.Dobb's Journal, Vol.19, Iss.15, December 1994, Miller Freeman Inc.

[14] A.S. Chepurnov, K.A. Gudkov, A.V. Shumakov. Digital control of RTM linac RF system with DSP.//Proc. of the IV European Particle Accelerator Conference, London, 1994. vol.3, pp. 1845-1847.