Automation and Robotics in Construction XII
E.Budny, A.McCrea, K.Szymanski (Editors)

On-board computer system for an excavator

Henryk Dobrowolski

Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, 00-66 Warsaw, POLAND

Abstract

The stucture of experimental heavy machine operator's assistant system is described. Main part of the system was built using factory made VME modules and programmed in C under OS-9. First version of the system was implemented on an excavator while advanced versions are under development. The paper describes the architecture of two versions of the system, starting from the single processor system to multiprocessor (two SBCs) using common VME-bus. For each hardware sketched out the software i.e. description of processes and interprocess communication. Given the outline of the distributed version of the system with fieldbus networking.

1. INTRODUCTION

In Institute of Heavy Machinery Engineering at Warsaw University of Technology a team led by Prof. Jan Szlagowski is engaged in some projects related to heavy machine's automation. A project of the heavy machine operator's assistant system was set within these problems. The foundations of design were of two kinds:

- investigating and testing several concepts of system's architecture (i.e. hardware and software solutions) in order to choose the right one being suitable for industrial application,
- constructing a tool to measure and collect dynamical data of time variant parameters of a heavy machine for research purpose.

The main goals of a project were:

- diagnostics of a machine, including monitoring of a relatively wide set of parameters and preventing damages using warnings and alarms,
- introducing an intelligent user interface for machine's operator, enabling the choice among several options while reducing the number of data being displayed simultaneously and aiding the operator in precise or blind operation,
- collecting of data measured by the system for a free chosen set of signals, frequency and interval of recording,

- providing a framework for implementation of various control algorithms starting from movement primitives up to sophisticated sequences optimized for several criteria.

The whole project was divided into three phases, each of them considering one type of architecture and with increasing degree of complexity. Up till now only the first of them was fully realized while the second is under development. As far as possible the system was built using factory made modules to reduce the cost of hardware design and testing.

2. SINGLE PROCESSOR CENTRALIZED SYSTEM WITHOUT MOVEMENT CONTROL

The system uses a 68HC000 processor at 16MHz clock linked with external modules using single (3U,P1) VMEbus **[1]**. The outline of a hardware is shown in Fig.1. As a mass storage for data recording there is a battery backed SRAM-disk within system memory or using a PCMCIA card as a solid state diskette.

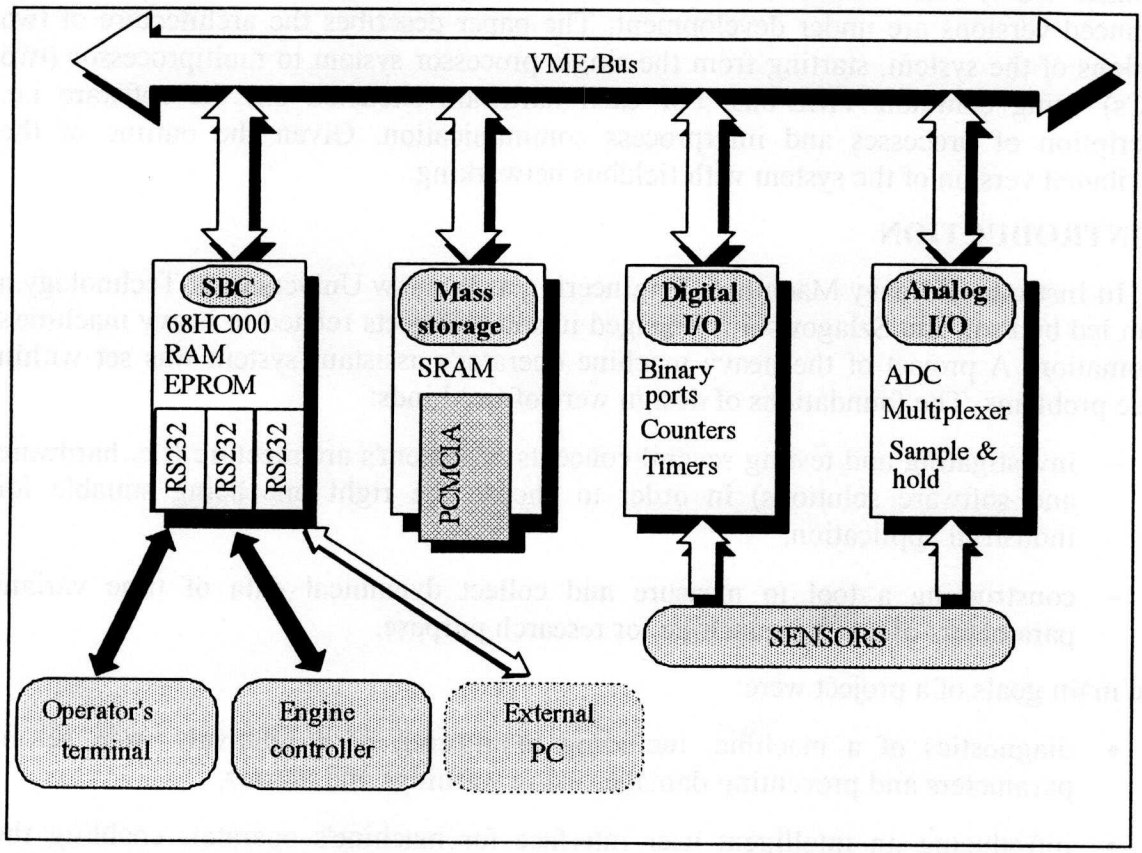

Fig.1 **Hardware of the simple centralized system**

The set of sensors includes binary, pulse (e.g. angle incremental encoders) and analog sensors. To convert analog signals there is a single AD converter employed. It is

equipped with several signal conditioning circuits multiplexed on ADC input. This solution is relatively cheap and enables conversion speed fast enough for data acquisition (the software defined burst readout period of 20ms for 20 sensors can be significantly reduced by single conversion time of 25μs). The collected data have to be processed off-line using external (immobile) computer system. In the first case the data can be sent to an external computer for subsequent processing using serial port with KERMIT protocol. As a user interface equipment there is a small programmable terminal with a high contrast LCD display employed. The terminal has a limited graphical functions and can store up to 250 separate screens preprogrammed, significantly simplifying the communications within user interface. The user interface was described in **[2]**.

Although the system is discussed as a centralized, an engine's velocity controller is built as a separate module and connected with the main computer using serial port. The engine controller unit based on Intel's 8051 microprocessor can also work in an autonomous way manually adjusted by the operator.

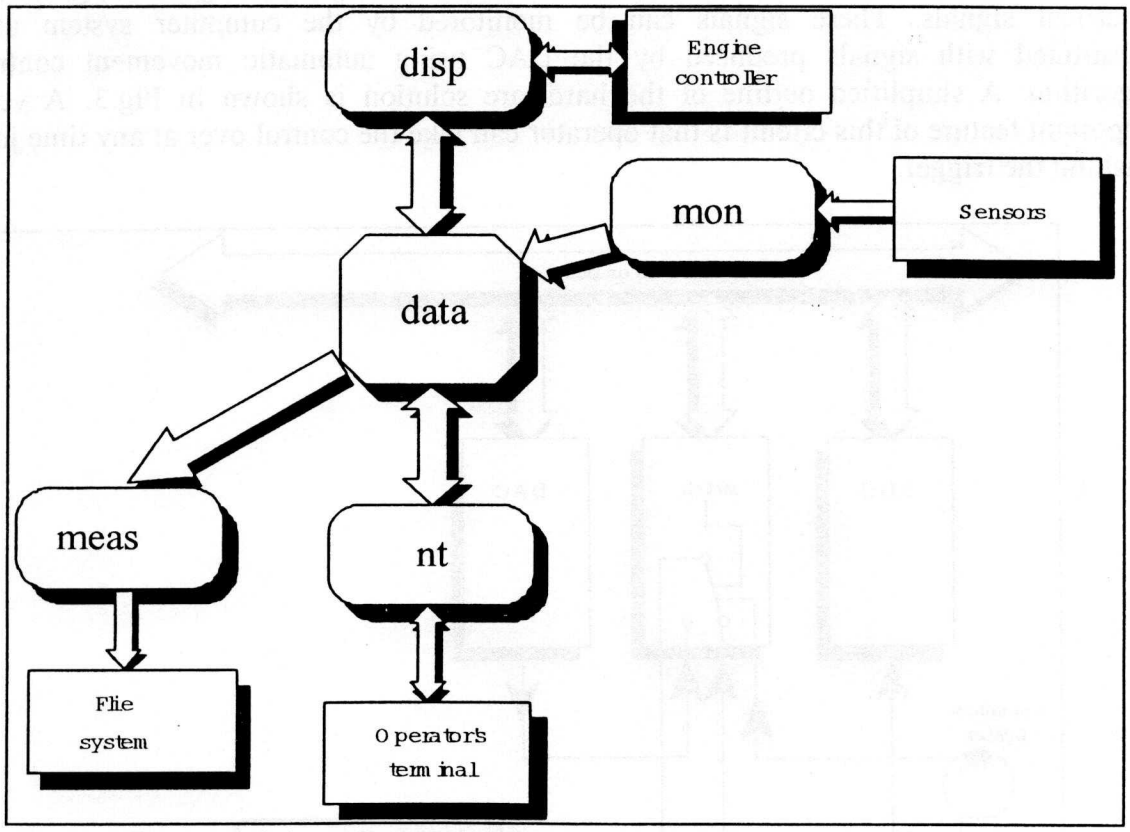

Fig.2 Outline of the software structure (without movement control)

The system works under control of OS-9 **[3]** operating system. The whole software, including operating system, is written into ROM. The software consists of several programs being executed parallel as individual processes communicating with each other using common data module, signals and an event. The outline of a software structure is shown in Fig.2. In order to isolate the low level hardware dependent IO functions there

is a separate, interrupt driven process *mon* implemented, which periodically reads the sensor states/values and writes them into cyclic buffers located in a data module *data*. The second process *nt* can read the data and is responsible for the machine diagnostics and operator interface. The *meas* process realizes the data averaging and recording at operator's request - otherwise this process is sleeping while waiting for that event. Furthermore there is a parent process *disp* of all above mentioned processes which initiates the whole system and controls the communications with the engine controller unit.

The system described above was developed within KBN (Committee of Scientifical Research) project no. 706749101, implemented to a backhoe excavator and is under test at the time of preparing this paper.

3. MULTIPROCESSOR CENTRALIZED SYSTEM

The system described above does not include the machine movement control. The hydraulic system of a machine is controlled manually with joysticks generating analog electrical signals. These signals can be monitored by the computer system and substituted with signals produced by the DAC using automatic movement control algorithm. A simplified outline of the hardware solution is shown in Fig.3. A very important feature of this circuit is that operator can take the control over at any time just pushing the trigger.

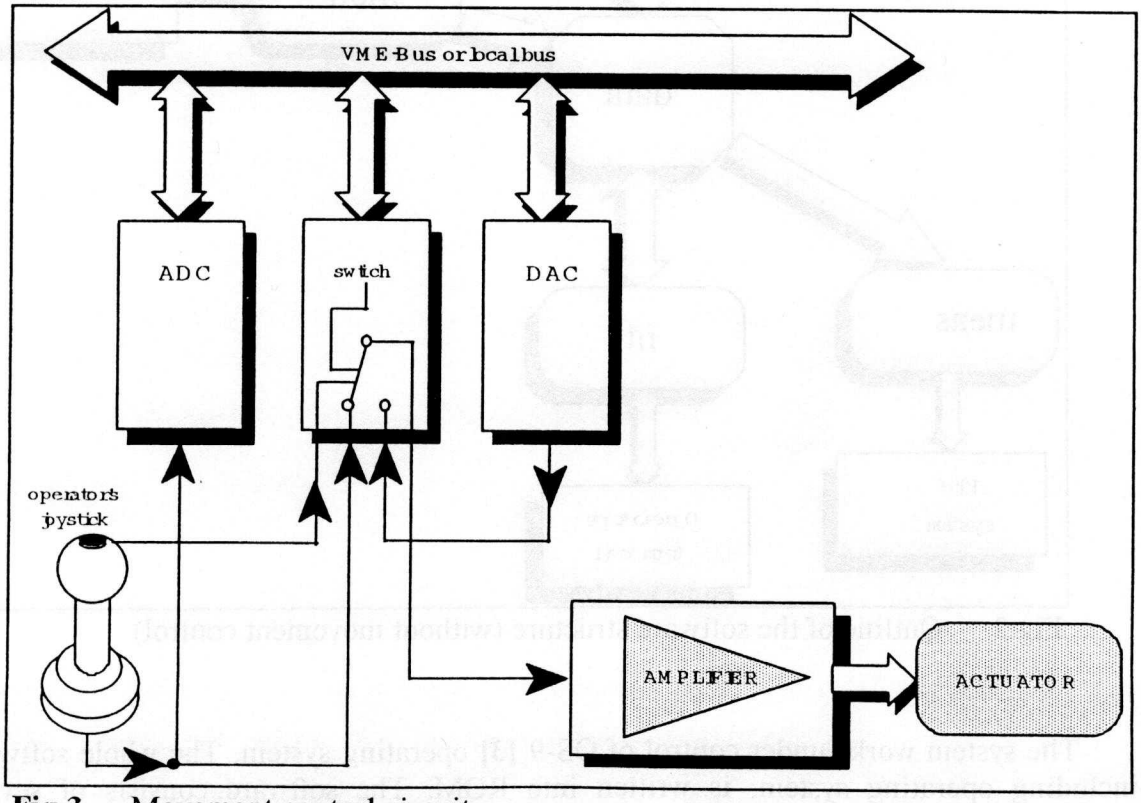

Fig.3 Movement control circuit

There is another new task which should be accomplished by the system - remote communication to enable precise machine control from outside the operator's compartment and interaction with the remote immobile system for monitoring and high level task planning.

Because of the relative low performance of the system's architecture described in a previous chapter to realize these both tasks the second processor board has been added. This board is provided with 68EC030 processor at 25/40MHz supported with FPCP and the multiprotocol communication processor 68302. The last processor is devoted to the communication task which includes a radio link simple network for three nodes: on-board system, remote machine control unit and an immobile system. The levels of this network are not discussed in this paper.

System software includes a new interrupt driven process called *mvmt* executing the movement control algorithms and the *ncom* communications process (replacing *disp*). Processes *nt* (operator's interface and diagnostics) and *meas* (data recording) are executed as before within the first processor board - the software outline is shown in Fig.4. The module *data* is no more a regular relocatable OS-9 data module but is a set of data structures located within the common memory address space on the new processor board which can be reached through system bus by *nt* and *meas* processes. This location enables faster data access for movement control process.

Although the program structure for *mvmt* process is defined, it is only a framework for experiments with various control algorithms.

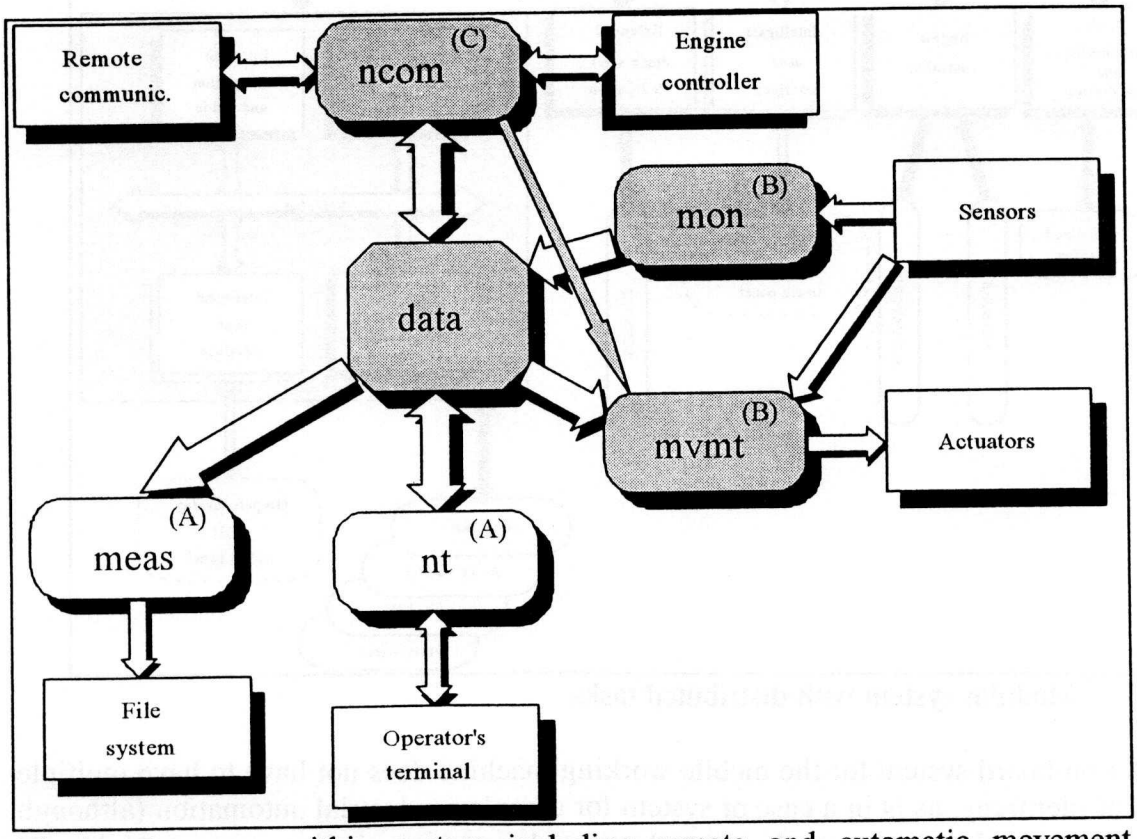

Fig.4 Processes within system including remote and automatic movement control

(A) - 68000 processes

(B) - 68EC030 processes

As mentioned before the above introduced system is being under development. The experiments with it should lead to precise description of modules for open distributed system, assumptions of it are described in the next chapter.

4. OPEN DISTRIBUTED SYSTEM

The system described in previous chapter is not suitable for serial manufacturing for several reasons:

- it can not be easily configured for individual customer or machine, i.e. reduced (e.g. only basic diagnostics and engine/hydraulic pump control) or extended (e.g. with vision subsystem supporting the "blind" operation),
- it has a common user interface equipment, which can be excessive for reduced configuration and too weak for extensions,
- any damage can cause the whole system is halted, i.e. all its functions are not available (they can not be used autonomously)

These problems are solved in almost all industrial systems in a way that modular and hierarchical open architecture is employed. The same solution can be chosen for a mobile system. Some systems using this approach are manufactured.

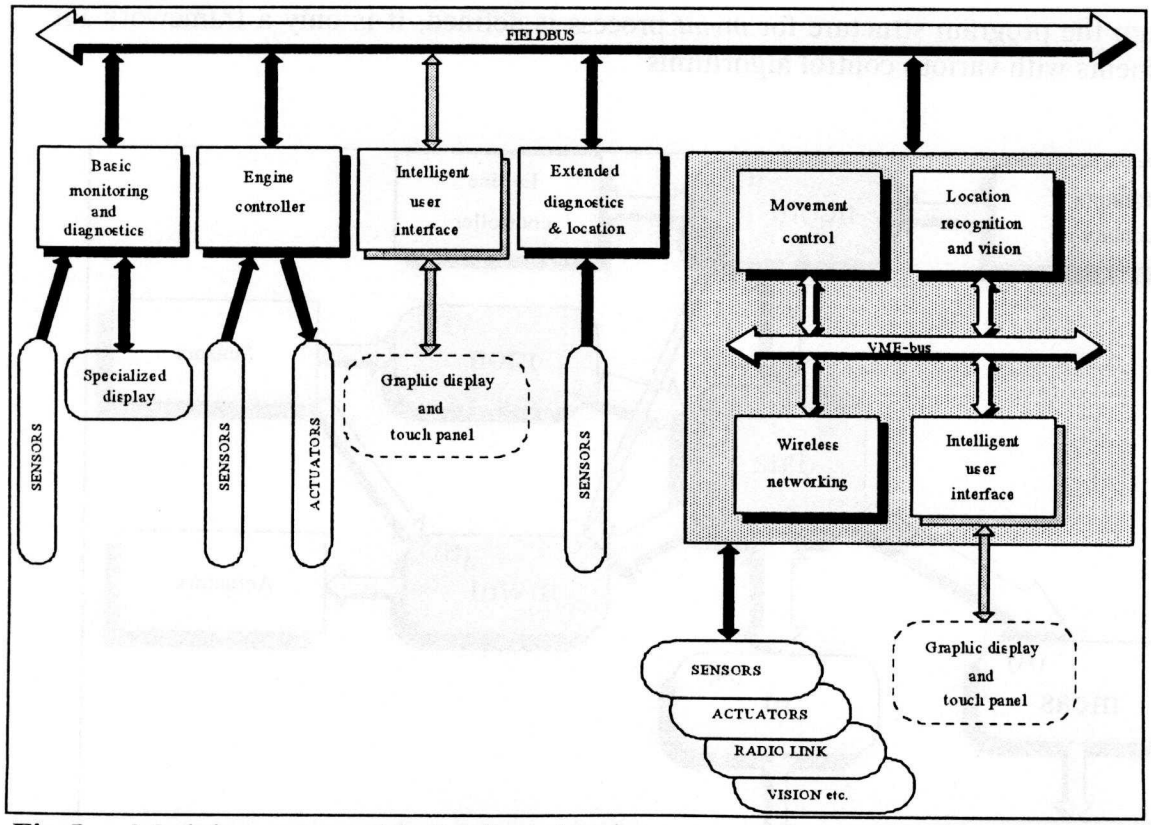

Fig.5 Modular system with distributed tasks

An on-board system for the mobile working machine does not have to have multiple levels of hierarchy, as is in a case of system for complex industrial automation (although it can be considered as a node of sophisticated system including many mobile and

immobile subsystems, e.g. within a building site). The main idea of an open system outlined in Fig.5 is to compose it step by step of "bricks", enabling expansion of its functions. The smallest on-board system includes only the *Basic monitoring and diagnostics* module, which substitutes a traditional operator's console. The only difference is that the module can communicate along *FIELDBUS* with any other module, if only one exists. In this way the simple tasks are moved into separate simple modules and at the same time all modules can share the set of sensors reading the needed values through the fieldbus.

FIELDBUS is a serial bus especially designed for industrial application. There are two standards which could be employed within the system. First of them, *CAN* (Controller Area Network, **[4]**) is intended for control applications, especially in vehicles. Second, a bit younger, *PROFIBUS* (Process Field Bus, **[5]**) differs from the previous one mainly in the maximum frame length.

The most sophisticated tasks like movement control, wireless networking for remote manual or automatic movement control or vision/scene recognition are interdependent and require a powerful computer (multiprocessor or RISC based). This computer is based on VME-bus and, in a case of applying vision/scene analyze subsystem, extended with *AutoBahn* **[6]** very fast data bus enabling e.g. real time camera picture processing.

CONCLUSION

Basing on experience with the development of systems described above we are convinced that the decision to build the hardware using factory made modules fulfilling industrial requirements and to apply the a multi-tasking real time operating system equipped with a set of software development tools, such as OS-9, is the only way to keep the cost, quality of solutions and development time within reasonable borders.

There is no doubt that the on-board systems for mobile machines such as excavators should be built using open architecture approach. This goal requires more effort as is in a case of centralized system to set up precisely the assumptions with respect to distribute tasks and communication details (application level of network).

REFERENCES

[1] **The VMEbus Specification Manual** *VITA*, 1987

[2] Dabrowski D., Dobrowolski H., Poncyliusz M., Selenta A., Szlagowski J.: **The Communication within Heavy Machine Operator Assistant System (HMOAS)**, *11th ISARC*, Brighton 1994

[3] **OS-9/68k Professional v.2.4.** *Microware Systems Corporation*

[4] **CAN für den Feldbuseinsatz**, *Markt & Technik*, 1992, no. 19

[5] Benster K.: **PROFIBUS - The Fieldbus for Industrial Automation**, *Prentice Hall*, 1993

[6] Kreidl J., Rücker G.: **Using a high-speed serial bus to move data fast**, *VMEbus Systems*, April 1992