Kazimierz KRZYWICKI, Grzegorz ANDRZEJEWSKI
UNIVERSITY OF ZIELONA GÓRA, INSTITUTE OF COMPUTER ENGINEERING AND ELECTRONICS
ul. Licealna 9, 65-417 Zielona Góra, Poland

Simple distributed system hardware platform for basic research

Abstract

This paper presents the simple distributed system hardware platform for basic research. It allows to study the different variants and aspects of the data exchange or synchronization methods in distributed systems. Moreover, the platform has the ability to implement distributed embedded systems. The modularity of a system allows for fast reconfiguration of the platform, such as the exchange of end modules. Therefore, it is possible to make quick changes and verify the system operation.

Keywords: distributed systems, embedded systems, simple hardware platform, microcontrollers.

1. Introduction

The dynamic development of microcontroller production technology has resulted in lower manufacturing costs and significant increase in performance. This has allowed to develop very large and complex distributed systems, where multiple independent devices are connected in some logical way [1, 2]. These devices are typically located in some distance from each other, such that the data exchange is made by using dedicated communication protocols.

A simple test platform based on ATmega8 was developed to investigate the implementation of control algorithms in hybrid hardware-software structures. The authors are conducting research in the area of the automatic code generation for the distributed embedded systems. It is important to make verification of the generated code directly in the hardware distributed embedded system. Depending on the configuration, such a system can be built with a few dozen of end modules (units), so using the commercial development boards/kits would be expensive and economically unjustified. Designed platform has the ability to execute series of tests and algorithms. Therefore, it can be used to test different variants and aspects of the data exchange in communication protocols or synchronization in distributed systems. Furthermore, it can work as a dedicated embedded system.

2. System structure

The general design assumptions for distributed system platforms are:

- flexible configuration,
- unified structure,
- easy to extend,
- simple data exchange between devices,
- low cost,
- as far as possible it should meet requirements for real control object (system).

The above assumptions can be realized in many different ways. In this paper, authors describe their own design idea, which is based on the repetitive microcontroller modules equipped with necessary I/O (input/output) ports. Such modules provide simple implementation, communication with user and other modules. Moreover, each module is able to control real objects, such as embedded system.

The designed platform is shown in Fig. 1. It is composed of four microcontroller modules, USB communication microcontroller module and a personal computer.

Modules can be divided into:

- master module,
- end module,
- communication module.

A personal computer communicates with the master module through the communication module. Depending on implemented synchronization method (central module control vs. distributed control system) [6], the master module can operate as a data exchanger or as a master unit for all end-modules. The end modules are responsible for the direct control of specific fragments of real controlled objects.

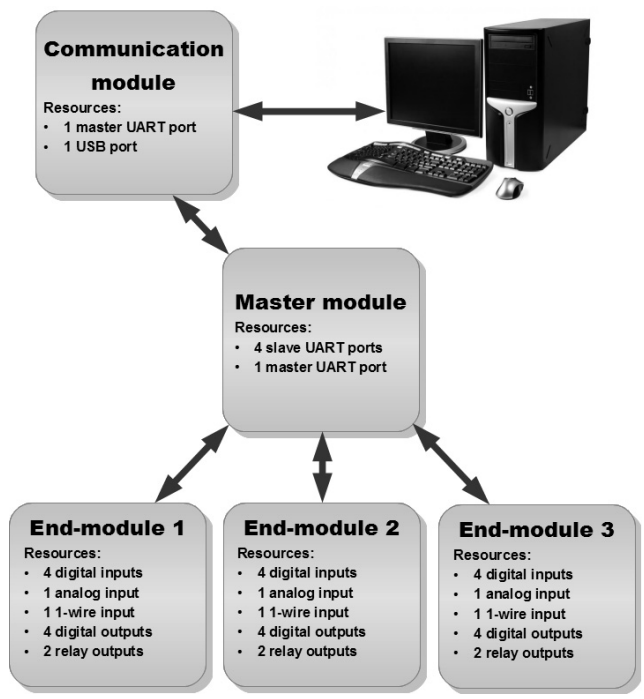

Fig. 1. General block diagram of the distributed system platform

To provide the platform flexibility, all modules have universal, unified structure, Fig. 2., which can be divided into the microcontroller and I/O blocks.

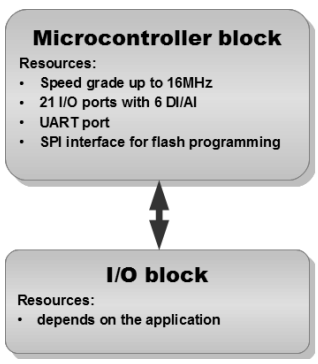

Fig. 2. General block diagram of the end-module

Microcontroller block

The main part of each module is the microcontroller block. In our platforms we used ATmega8, which is one of the most

popular microcontrollers used in the industry [3]. It consists of a Reduced Instruction Set Computing (RISC) core with a clock speed up to 16 MHz.

The basic features of the ATmega8 [5] microcontroller used in the platform are shown in Tab.1.

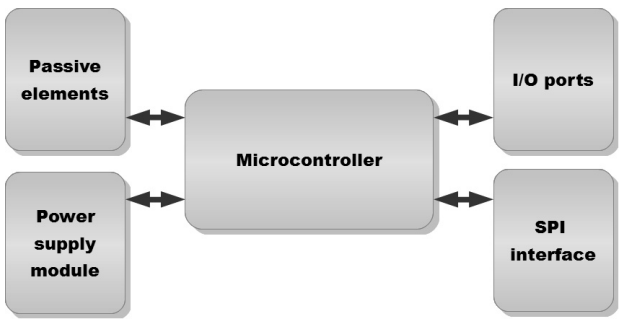

Fig. 3. Block diagram of the microcontroller block

Tab. 1. Basic features of ATmega8 microcontroller

No.	Parameter	Value
1.	FLASH memory	8 KB
2.	EEPROM	512 B
3.	Internal SRAM	1 KB
4.	Timer/Counters with separate prescaler	2 x 8 B, 1 x 16 B
5.	Real Time Counter with separate oscillator	1
6.	ADC	1 x 6 channels (10-bit)
7.	Programmable Serial USART port	1
8.	Master/Slave SPI interface	1
9.	On-chip analog comparator	1
10.	Watchdog	1
11.	I/O	23
12.	Power consumption at 4MHz (active/idle/ power-down mode)	3.6 mA/1.0 mA/0.5 μA

An important advantage of the described microcontroller is its low price and the free development platform: compiler – WinAVR, debugger – AVRStudio, programmer – avrdude.

The microcontroller block is shown in Fig.3.

I/O block

The system is designed and implemented with two types of I/O blocks, Fig. 4. The first, known as the control block, is used for direct control of the object or communicate with a user. The second, known as the communication block, is commonly used to communicate to the personal computer.

Tab. 2. Selected features of I/O block.

No.	Parameter	Qty.
1	Digital input	4
2	Analog input	1
3	1-wire input	1
4	Digital output	4
5	Relay output	2

The control block was designed as a universal device to meet the requirements of simple real control objects (systems). Frequently in practice, there are on/off drive-types (digital output or relay) with several feedback signals, for example: the position of the actuator (1 analog or two digital limit switches inputs) or relay executive state. Also control block must have the ability that is used very often in real object control – state and temperature monitoring.

After requirement analysis the following resources were implemented in a control block.

Digital inputs have been equipped with SPST (Single-Pole, Single-Throw) micro switches which can be used to simulate limit switches or as a user interaction interface. Analog input has been equipped with a suitable potentiometer for simulating changes of the measured value (voltage). The value of the potentiometer has been selected to maintain possibly good relationship disruption to current consumption. Temperature measurement has been realized with 1-wire interface using integrated circuit Dallas Semiconductor DS18B20. It is also possible to connect in this place, the analog sensor, but configuration of the specified input type of ATmega8's microcontroller must be also changed. Digital outputs equipped with LED diodes that can be used to indicate the status of selected system informations. The relay outputs can be used for direct connection of DC powered devices up to 30 V or AC to 230 V with power consumption up to 5 A. This restriction is determined by the parameters of used relay – Fujitsu JV-5S-KT.

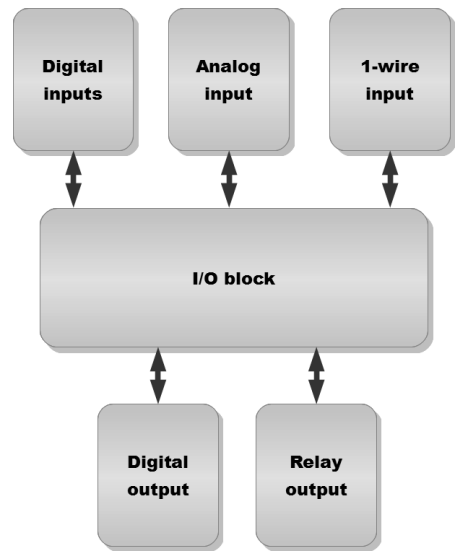

Fig. 4. General block diagram of the I/O block

Communication block

Communication block is responsible for two-way data exchange between distributed system and the personal computer. Communication block features:

- monitoring the status of the system,
- system management,
- fault detection.

USB interface was chosen as the most popular connection. Implementation was made through UART to USB translation microcontroller – FT232 – manufactured by FTDI Company. Communication block also has been equipped with EEPROM memory, which gives ability to create own individual VID (Vendor Id) and PID (Product Id) numbers for USB interface. Additional input allows to specify the reference voltage for the FT232, which determines the level of voltage detection of logic states. This allows to connect microcontrollers powered at 5 V and 3.3 V.

Fig. 5. presents general block diagram of communication block.

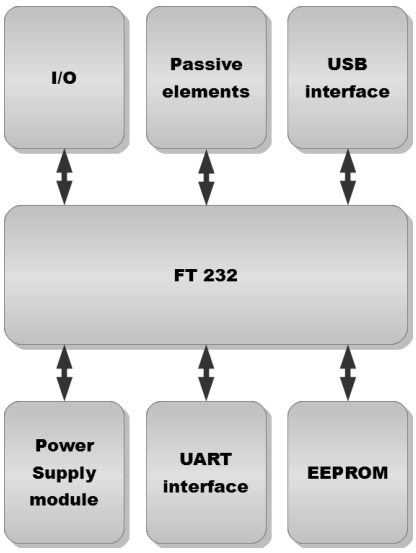

Fig. 5. General block diagram of the communication block

Wired connection type between modules

Depending on the application, the platform allows to use two types of connections between end modules: wired or wireless. If it requires greater stability and resistance to interference – wired connection between end modules is used. When the priority is the scalability and the distance between the end modules – wireless connection is used.

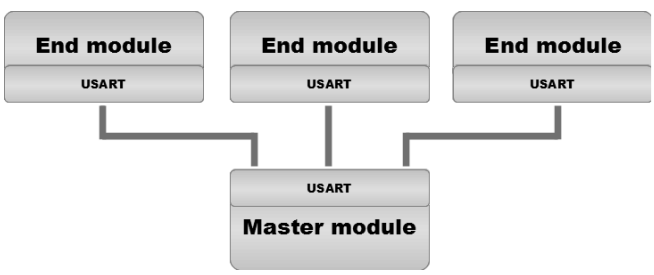

Fig. 6. Basic wired connection type between end modules

Basic wired connection type between end modules is shown in Fig.6. In this case, each end module is connected to the master module (which is communicating with a PC).

All of the end modules are using the built-in hardware USART port. The master module (which is supporting connection between end modules and the PC) requires additional (software) implementation of the USART port for each connection with the end module. The number of software USART ports is limited by the number of available I/O pins in the microcontroller. All physical connections between modules are made using copper wires.

Wireless communication method between modules

An alternative to a wired connection is wireless RF communication, which allows for a less CPU load of the master module, and a less usage of the hardware resources (there is no need to use I/O pins of the master module for USART communication with the end modules).

Fig. 7. presents diagram of the wireless connection type between end modules. RF transmission is realized by RFM12 chip manufactured by HOPE Microelectronics [7]. The chip is a typical SPI-RF transceiver that allows to work with the following frequencies: 315, 433, 868 and 915 MHz. The main advantages of the RFM12 is small size, built-in FIFO, a wide supply voltage range (from 2.2 to 5.4 V) and simple operation via the serial interface (SPI). This RFM12 circuit is connected via copper wires to the SPI port of the microcontroller.

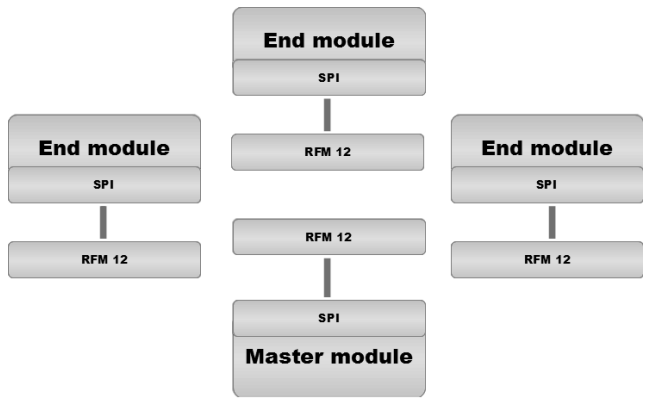

Fig. 7. Wireless connection type between end modules

Using the RFM12 for each end module allows to: avoid a large number of connections made by wires, extend the whole system at a greater distance and gives the ability to support a much larger number of end modules, because there is no need of physical connection of each end module to the master module. Furthermore, using RFM12 gives the ability to connect in a simple distributed system a different types of the end modules, e.g. DSP (Digital Signal Processing), FPGA (Field Programmable Gate Array) devices or various types and architectures of the microcontrollers without using dedicated voltage level translator to interface the data transfer between low voltage and higher voltage systems.

The main disadvantage of this solution is the low communication resistance to interferences and the necessity of handling conflicts between end modules, which in the same unit of time are trying to perform various operations.

3. Application

Last research performed on the simple distributed system hardware platform were made by implementing the embedded system described by Petri net, Fig. 7. This implementation was used to run control tasks decomposed into individual processes.

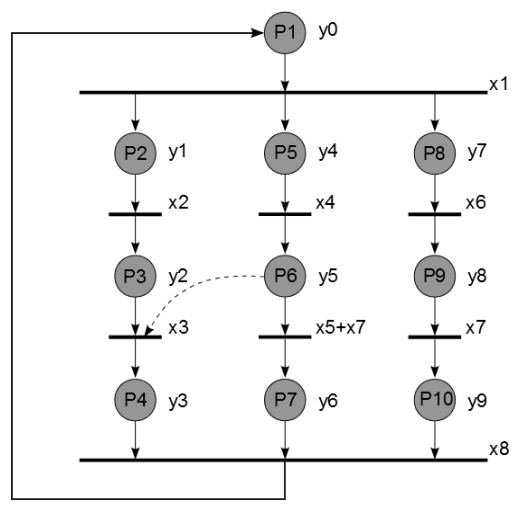

Fig. 8. Embedded system described by Petri net

Fig. 9. presents concurrent process from Fig. 8. which is divided into individual sub-processes and implemented by the three end modules (MODULE 1, MODULE 2 and MODULE 3).

P1..P10 represent the places (states), $x1..x8$ – inputs and $y0..y9$ – outputs. Transitions between states were described using the logic equations, for example: $x3$*P6M2 which means $x3$ input AND place P6 from MODULE 2. Furthermore, it was necessary to add resting points (RP1) for the MODULE 1 and MODULE 2 to maintain proper functioning of the system and to ensure proper synchronization between all of the end modules. The $y0$ output and P1 state were assigned to the MODULE 2.

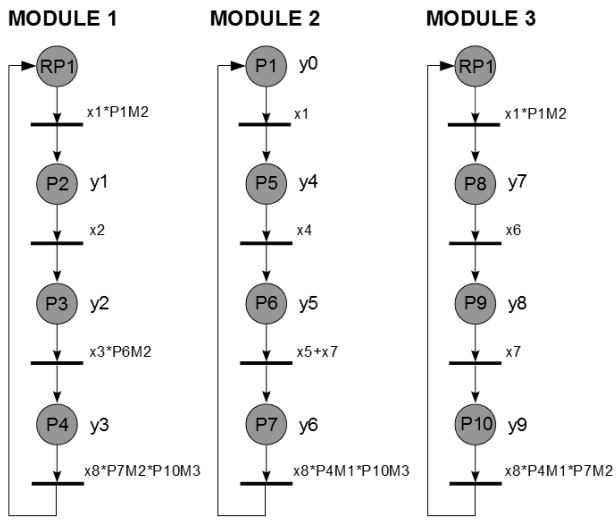

Fig. 9. Control tasks decomposed into individual processes

The simple distributed system hardware platform was also used to analyze and compare two methods of data exchange: with centralized and decentralized control. Furthermore, we had to design and implement two variants of communication protocols. Research results shown, that under some conditions the amount of necessary data to transfer is over 100 times smaller for decentralized control systems than for central control [6]. Further research in the data exchange area, has led to develop novel protocol called CloudBus [8]. It provides a control mechanism for a number of processing units distributed in a network.

Current research are focused on the automatic direct synthesis from Petri net model to the simple distributed system hardware platform (distributed embedded system).

4. Conclusions

This paper presented simple distributed system hardware platform for basic research. Platform provides embedded system implementation and allows to study the different variants and aspects of the data exchange in distributed systems e.g. synchronization methods [6, 8]. Distributed system hardware platform can use different microcontrollers and other hardware devices depending on requirements. Fast reconfiguration is possible due to the modularity of a platform, so it allows to make quick changes and verify of the distributed system operation. Using the wireless communication between end modules allows to build complex embedded distributed systems which consists of a large number of executive modules. Furthermore, simple structure, low cost and ability of implementing full redundancy of executive modules opens up new perspectives in large and complex control systems design. Moreover simple distributed system hardware platform can be a cheap and efficient alternative to the PLC systems.

Currently, the platform is used for implementing research in area of automatic code generation and synthesis of concurrent processes for the distributed embedded systems.

5. References

[1] Attiya H. and Welch J.: Distributed Computing: Fundamentals, Simulations and Advanced Topics. J. Wiley Interscience, 2004.
[2] Garg V. K.: Elements of Distributed Computing. Wiley&Sons, 2002.
[3] Baranowski R.: Mikrokontrolery AVR ATmega w praktyce. Wydawnictwo BTC, Warszawa 2005.
[4] Adamski M. A., Karatkevich A. and Wegrzyn M.: Design of Embedded Control Systems. Springer, 2005.
[5] Atmel ATmega8 Datasheet, Rev.2486AA–AVR–02/2013.
[6] Krzywicki K., Andrzejewski G.: Concurrent process synchronization in distributed systems. XV International PHD Workshop – OWD 2013.
[7] HOPE Microelectronics RFM12 Universal ISM Band FSK Transceiver datasheet.
[8] Krzywicki K., Andrzejewski G.: Data exchange interfaces in distributed systems. MAM. vol. 60, no 7, pp. 495-497, 2014.

Received: 07.11.2014 *Paper reviewed* *Accepted: 05.01.2015*

M.Sc. Kazimierz KRZYWICKI

Graduate at the University of Zielona Góra, Faculty of Electrical Engineering, Computer Science and Telecommunications. He is a member of PTI (Polish Information Processing Society). He is a PhD student at the University of Zielona Góra. His research interests include design and implementation of distributed embedded systems and hardware synthesis for reprogrammable devices.

e-mail: k.krzywicki@weit.uz.zgora.pl

Ph.D. Grzegorz ANDRZEJEWSKI

Grzegorz Andrzejewski was born in 1970. He graduated in 1995 from Technical University in Poznań. He received a Ph. D. degree in the field of computer science from the Technical University of Szczecin in 2002. His research is focused on modelling and synthesis of digital control systems.

e-mail: g.andrzejewski@iie.uz.zgora.pl