ANALYSIS AND SUBSTANTIATION OF SELECTION OF MICROCONTROLLERS FOR CNC MACHINES CONTROL

[1]Urinov Nasillo Fayzilloevich, [2]Zayniyev Xudoyberdi Muhiddinovich, [3]Narzullaev Mukhammad Azamatovich
Candidate of technical sciences, associate professor[1], Probationer teacher[2], Undergraduate[3] Bukhara Engineering-Technological Institute

---

ANNOTATION

The article covers the analysis of microcontrollers for CNC machines controlling and the substantiation of its selection.

Keywords: Arduino Nan, Arduino program, interface, microcircuit, reboot of Nano platform, microcontroller, register, industrial computer.

INTRODUCTION

In most industries, an industrial computer (PC) is used as a control system for the operation of CNC machines, which performs the role of reading the processing parameters and trajectories of the movement of the working mechanism from a program written in the G-code and transferring them to the machine. However, this PC is quite expensive and takes up a decent place, if the program being executed does not imply frequent changes and editing of parameters and trajectories, then you can use a fairly cheap and compact control panel on a microcontroller that will read the program from memory and convert it into machine instructions to make the final products. To select the right microcontroller that would meet all the requirements, we will be acquainted with the reason for its appearance, the history of creation and the prayer line that is now on the market.

Fig. 1. Microcontrollers

One of the main reasons stimulating the emergence of microcontrollers is the introduction of computer technology in all spheres of human activity, including industrial. This led to a miniaturization and reduction in the cost of products. Advances in microcircuit technology contributed greatly to this in 1967. Texas Instruments released the first calculator on an integrated circuit, which launched the process of miniaturizing computer technology. However, microcircuits for calculators were created in accordance with the requirements of each specific customer. These calculator processors were low power and did not meet the requirements for computing power.

MAIN PART

First, let us cover the very concept of "microcontroller". A microcontroller can be defined as a miniature computer based on a single chip, including, in addition to the processor, a number of auxiliary elements, such as RAM, EPROM, timer, etc. The microcontroller is designed to perform any predefined tasks.

The easiest way is to compare a microcontroller with a PC. Like a PC, a microcontroller has a processor, RAM, and read-only memory. However, unlike a personal computer, all these elements are located on a single chip.

- MegaAVR is the most popular line that has a sufficient amount of internal memory (up to 256 KB), many additional peripherals and serves for tasks of medium to high complexity.

- XmegaAVR is used for advanced business applications that require a lot of memory and high performance.

Table 1. Analysis of technical characteristics of microcontrollers

Series name	Number of contacts	Flash memory capacity	Feature
TinyAVR	6-32	0.5 – 8 KB	Small size
MegaAVR	28-100	4-256 KB	Peripherals
XmegaAVR	44-100	16-384 KB	Interrupt system, DMA support

1) Family features

First, microcontrollers in this series are high-speed. The microcontroller processor executes many instructions in one cycle. AVR microcontrollers are about 4 times faster than PICs. In addition, they consume significantly less energy and can operate in several energy saving modes.

Many AVR controllers are 8-bit, although there is also a 32-bit variant of AVR32 controllers. In addition, as mentioned above, AVRs belong to the type of RISC microcontrollers. The RISC architecture (Reduced Instruction Set Computers) means that the set of instructions that the device processor can execute is truncated, but at the same time, this architecture provides a speed advantage. The analogue of the RISC architecture is the CISC [1] (Complex Instruction Set Computers) architecture.

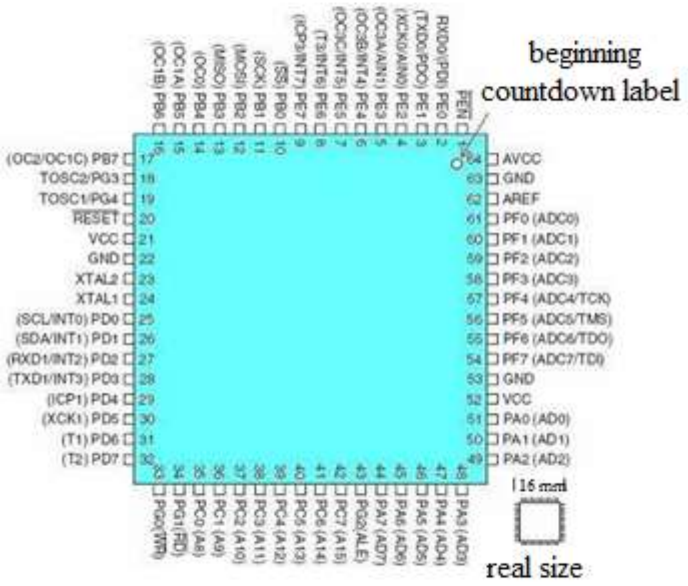


Fig. 2. Bit version of AVR32 controllers.

An 8-bit controller means it is capable of transmitting and receiving 8-bit data. The provided I / O registers are also 8-bit.

The controller architecture is based on registers [2]. This means that the controller uses the registers /

The controller processor receives data from two input registers, performs a logical operation, and stores the result in an output register. All this takes one executable cycle.

2) Architecture of the controller

The AVR controller has 32 general purpose 8-bit registers in total. During a cycle, the processor takes data from two registers and uses it in an arithmetic logic unit (ALU), which performs computation on the data and places it in a random register. ALU can perform both arithmetic and logical operations on registers. ALU can also perform actions with one register. In this case, the controller does not have an accumulator operand, unlike the controllers of the 8051 family - any operands can be used for operations, and the result of the operation can be placed in any operand. The microcontroller conforms to the Harvard Computing Architecture, whereby a computer has independent memory for programs and data. Therefore, while one operation is being performed, the next operation is pre-fetched from memory. The controller is capable of performing one operation per cycle. It follows from this that if the clock frequency of the microcontroller is 1 MHz, then its performance will be 1 million operations per second. The higher the clock speed of the controller, the higher its performance will be. However, when choosing the clock frequency of the controller, a trade-off should be observed between its speed and power consumption. In addition to the flash memory and the processor, the controller [3] has such peripherals as input-output ports, analog-to-digital converter, timers, communication interfaces - I2C, SPI and serial UART port. All of these peripherals can be controlled at the program level.

3) Programs for the microcontrollers

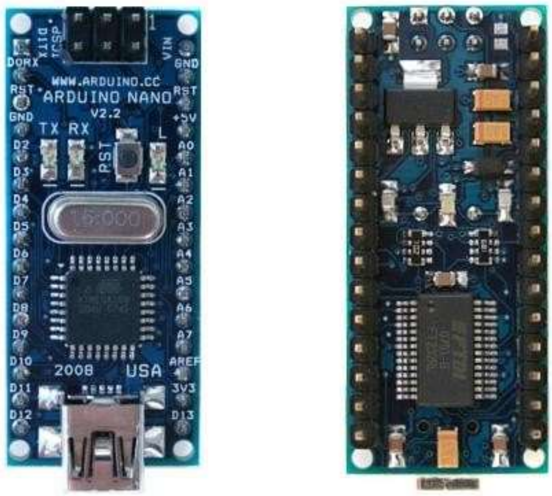
As mentioned above, a microcontroller is like a PC, and it follows from this that, like a PC, the AVR can also execute any program, albeit only one at a time. The microcontroller program can be stored in the internal memory of the controller and is a series of elementary commands that select data and perform certain operations with them. In some cases, this means reading the input data, checking its status, and outputting the appropriate output values. Sometimes it may be necessary to change data and perform operations with it, as well as transfer the data to some external peripheral device, such as an indicator, or a serial port.

For such simple tasks, sets of elementary commands are used, each of which has an analogue in a language more accessible to human perception. Therefore, the most common way to write controller programs is to write them in the language of machine instructions.

The advantage of machine instructions is very fast, compact and efficient code, but the creation of such programs at the same time requires extensive knowledge of the microcontroller processor, manual memory management and control of the program structure. Therefore, other high-level languages such as Basic, C, and Java are often used to write programs. In this case, the task of controlling the program structure and memory management is taken over by the compiler that creates the firmware. In addition, frequently used functions can be placed in special libraries and extracted from them as needed.

Microcontrollers of the AVR family are widely used today in computers to automate the control of electronic equipment, various electrical appliances and mechanisms used in industrial, commercial, and household needs. Low cost, large assortment and wide capabilities of this series of microcontrollers contributed to their great popularity.

4) Selection of the microcontroller.



5)

Fig. 3. Arduino Nano

Nano platform based on the ATmega328 microcontroller is small and can be used in various projects. It has a structure similar to the Arduino Duemilanove, but differs in appearance. The difference lies in the lack of a DC power connector and operation via a MiniUSB cable.

Arduino Nano is powered via a MiniUSB cable, either from an unregulated 6-20V (pin 30), or a stabilized 5V (pin 27) external power supply. The power supply with the highest voltage is automatically selected.

FTDI FT232RL is powered only if the platform itself is powered from USB. Thus, when operating from an external source, 3.3 V voltage generated by the FTDI microcircuit will not be there, while the RX and TX LEDs flash only when there is a signal at pins 0 and 1.

Inputs and Outputs

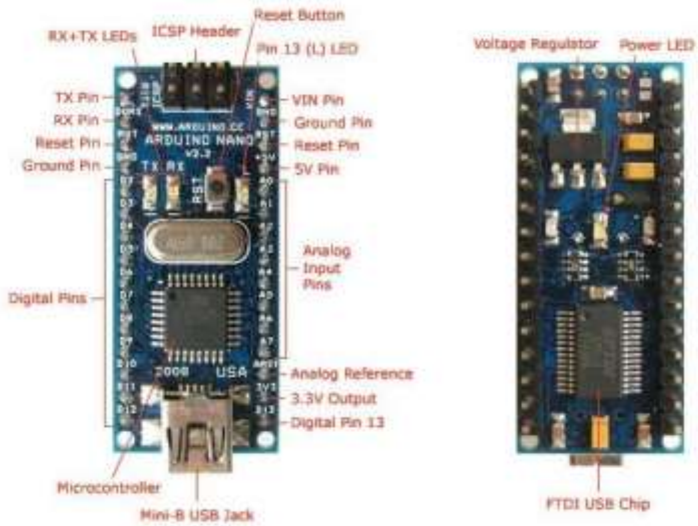


Fig. 4. Microcontroller outputs

Each of the 14 digital outputs can be configured as input or output using the pinMode(), digitalWrite(), and digitalRead() functions. The pins work only at 5 V. Each pin has a resistor (disabled by default) 10-40 kOhm and can pass up to 30 mA. Some pins have specific functions:

- **Serial bus: 0 (RX) and 1 (TX).** Pins are used to receive and transmit data.
- **External interrupt: 2 and 3.** These pins can be used to trigger an interrupt on the lowest value or on the

rising, falling edge, or when the value changes.

- **PDM: 3, 5, 6, 9, 10, and 11.** Each of the pins provides PDM with a resolution of up to 8 bits using the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** SPI communication is carried out through these pins.

Nano platform has 8 analog inputs, each with a resolution of 10 bits. As standard, the pins have a measurement range of up to 5 V relative to zero, although it is possible to change the upper value using the analogReference() function. Some pins have specific functions:

- **I2C: A4 (SDA) and A5 (SCL).** Through the pins, I2C communication (TWI) can be carried out.

Additional pair of platform pins:

- **AREF**. Required for analog inputs. Used with analogReference().
- **Reset**. To reboot the microcontroller. It is usually used to connect a reset button on expansion cards, which prevents access to the button on the card itself.

Arduino Nano platform contains many devices for communicating with a computer, other Arduino devices, or microcontrollers. ATmega328 have a TTL (5V) UART serial interface via the RX and TX pins. A microcircuit installed on the board directs this interface through USB, and the drivers provide a virtual COM port to the program on the computer. The Serial Monitor of the Arduino software allows sending and receiving text data when connected. The RX and TX LEDs on the platform will indicate TX and RX communication.

ATmega328 supports I2C (TWI) and SPI interfaces. Arduino has a Wire library for easy use of the I2C bus.

The platform is programmed using Arduino 1.6.5 software. From the Tools> Board menu, select "Arduino Diecimila, Duemilanove, Nano". ATmega328 microcontroller comes with a pre-recorded bootloader to make it easier to write programs without using external programmers. Communication is carried out using the STK500 protocol. It is possible not to use the bootloader and program the microcontroller via the in-circuit protocol with the ICSP block pins. The Nano is designed in such a way that before writing a new program code, the reboot was carried out by the program itself, and not by pressing a button on the board. One of the FT232RL data flow-controlled pins is connected to the reset pin of the ATmega328 via a 100nF capacitor. Activation of this line reboots the microcontroller. Arduino program, using this functionality, uploads the code with one click of the Upload button in the development environment itself. The low-level signaling on the DTR line is coordinated with the start of the program recording, which reduces the bootloader time.

The function applies one more purpose. The Nano reboots every time it is connected to the Arduino program on the computer. The bootloader works for the next few seconds after reboot. During programming, several bytes of code time out to prevent the platform from receiving incorrect data. If performing a one-time debugging of the firmware written to the platform, or enter any other data at each launch, it should make sure that the program on the computer waits for the first second before transferring data.

CONCLUSION

We made the conclusion that our machine can be controlled both using personal computer and using autonomous control panel carried out based on Arduino Nan universal platform.

REFERENCES

1. Baikov V.D. Solution of trajectory problems in CNC microprocessor systems. Mechanical Engineering, 1986
2. Monakhov G.A. Machines with programmed control. Handbook. Mechanical Engineering, 1975

3. Kenio T. Stepping motors and their microprocessor control systems, 1987