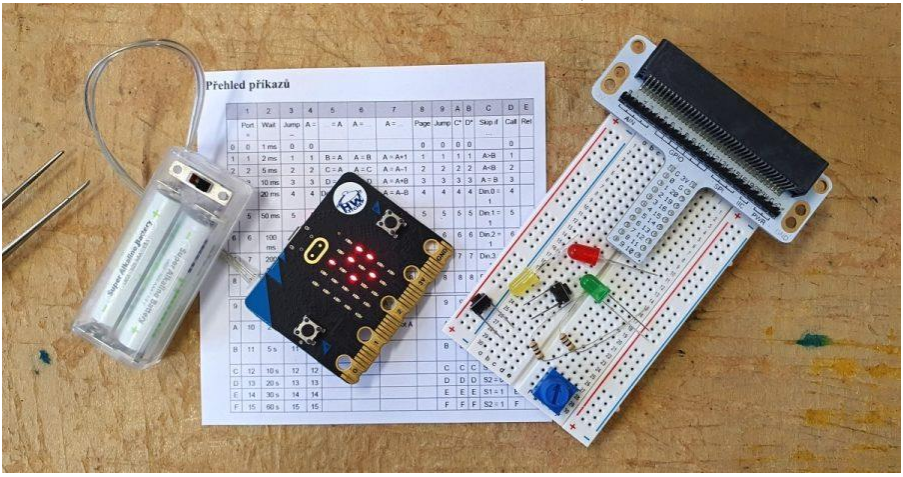
MINICOMPUTER PROGRAMMABLE WITH THREE BUTTONS - MYCO

3/1/2021 HONZA HORÁČEK ARTICLES , MICRO: BIT



Can you imagine a minicomputer that you can program with just three buttons? In this article, we will introduce one of these and you may be surprised at what can be created with such a minimalist solution. A universal kitchen timer, a simple AD converter or a compass are a matter of a few minutes.

Article content:

How did I get into MyCo programming?

A few years ago, some time after I started getting to know the amazing world of electronics, I came across the Mikrokontroler kit - an electronics tutorial set from the age of 14. I assembled the kit, put the microcontroller, buttons and diodes into the solderless field according to the diagram, connected the wires and proceeded according to the instructions. By a special combination of **pressing three buttons, I had to achieve the desired effect** .

The original MyCo *kit*

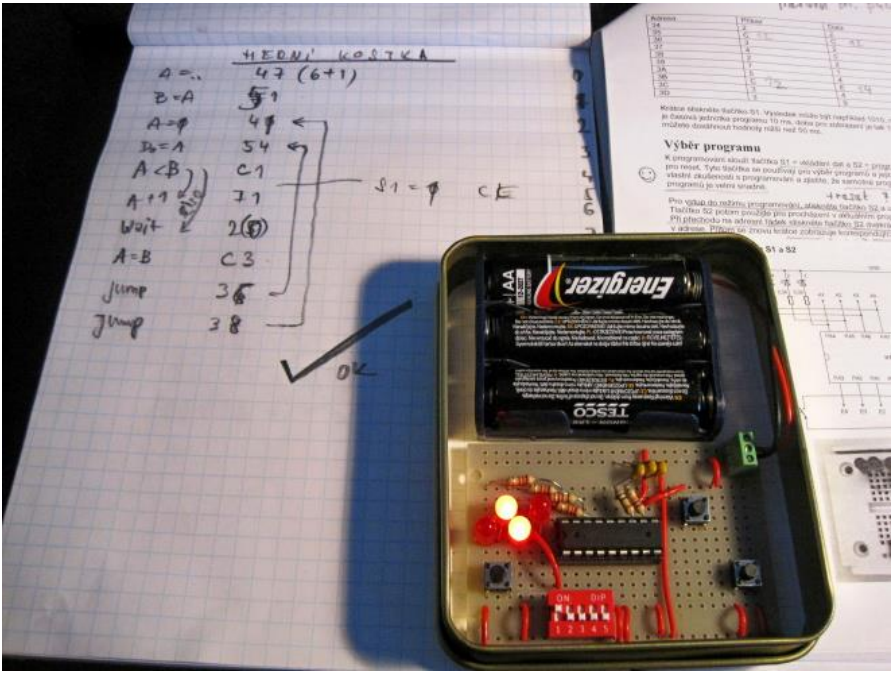
At first it seemed unattainable, but the German thorough instructions did not leave me in the lurch - it didn't take long and a series of four colored LEDs flashed in the order they were! In a short time, I had mastered the basic programs and all I had to do was solder the whole kit into a more permanent form and try my own programs on it.

What exactly is MyCo?

At the beginning was the idea of a small computer with a simple interface (Tastenprogrammierbare Steurung, abbreviated TPS, in the English variant **MyCo, or My Computer**). The author of the idea was the creator of electronic kits, B. Kainka.

The idea was that microcomputers or microcontrollers need a connection to a computer or mobile phone for their programming. The creators have come up with a complex

environment that **will allow programming, so to speak, by hand** , with a minimalist graphic output, specifically four LEDs.



Programming in MyCo

The above is a building block of the whole system - four diodes can display a maximum value of 15 (F) in hexadecimal code, the set of programming commands and values is limited to the same value, ie four bits.

On the one hand, this is limiting, but thanks to this, the whole system is also simple, clear and compact. And the limitations are significantly reduced by other features of the system: four digital inputs, four digital outputs, two analog inputs and one PWM output.

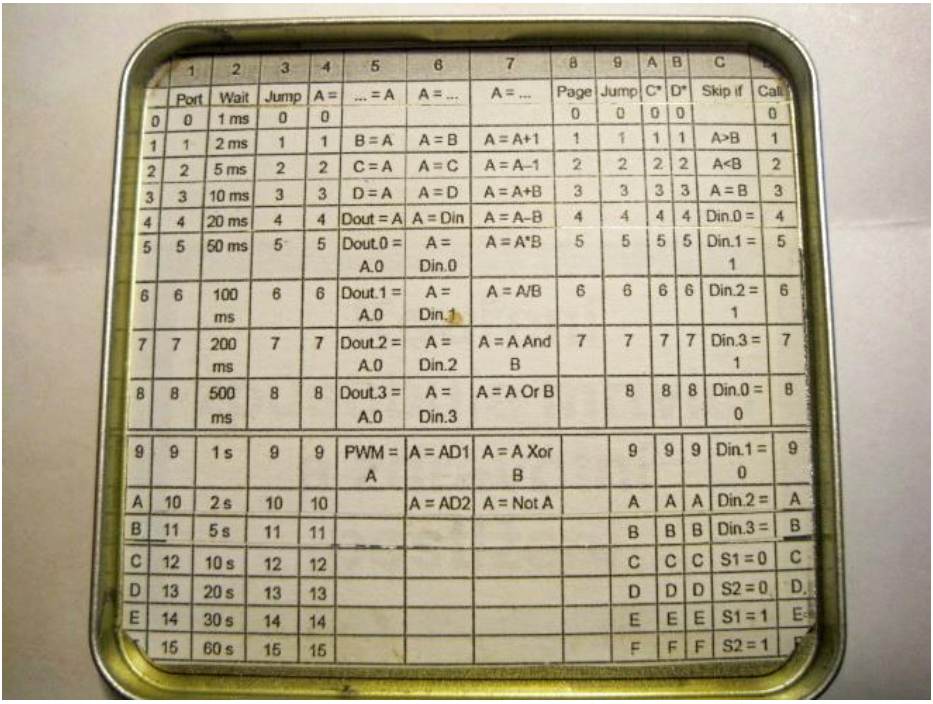
What can be created with such equipment? After a beginner wins with all sorts of flashing output diodes, timing, variables, two buttons or conditions, it is not a problem as with any

open system to connect to inputs and outputs such as a seven-segment display, 4 × 4 keyboard, audio output or a sensor. Thanks to analog inputs, MyCo can serve as a simple AD converter, for example.

With handy inputs (for example with a micro: bit, I will explain below) I managed to **encode a universal kitchen timer** , a simple compass or a VUmeter **in a few minutes** .

How is programming in MyCo?

I do not want to describe the complete procedure for programming in Myco, everything is well documented in the links. However, I'll try at least a simple outline to get an idea of how it works. We have three buttons and four LEDs. Not great, but not terrible!



	1	2	3	4	5	6	7	8	9	A	B	C	
	Port	Wait	Jump	A =	... = A	A = ...	A = ...	Page	Jump	C*	D*	Skip if	Call
0	0	1 ms	0	0				0	0	0	0		0
1	1	2 ms	1	1	B = A	A = B	A = A+1	1	1	1	1	A>B	1
2	2	5 ms	2	2	C = A	A = C	A = A–1	2	2	2	2	A<B	2
3	3	10 ms	3	3	D = A	A = D	A = A+B	3	3	3	3	A = B	3
4	4	20 ms	4	4	Dout = A	A = Din	A = A–B	4	4	4	4	Din.0 =	4
5	5	50 ms	5	5	Dout.0 = A.0	A = Din.0	A = A*B	5	5	5	5	Din.1 = 1	5
6	6	100 ms	6	6	Dout.1 = A.0	A = Din.1	A = A/B	6	6	6	6	Din.2 = 1	6
7	7	200 ms	7	7	Dout.2 = A.0	A = Din.2	A = A And B	7	7	7	7	Din.3 = 1	7
8	8	500 ms	8	8	Dout.3 = A.0	A = Din.3	A = A Or B		8	8	8	Din.0 = 0	8
9	9	1 s	9	9	PWM = A	A = AD1	A = A Xor B		9	9	9	Din.1 = 0	9
A	10	2 s	10	10		A = AD2	A = Not A		A	A	A	Din.2 =	A
B	11	5 s	11	11					B	B	B	Din.3 =	B
C	12	10 s	12	12					C	C	C	S1 = 0	C
D	13	20 s	13	13					D	D	D	S2 = 0	D
E	14	30 s	14	14					E	E	E	S1 = 1	E
F	15	60 s	15	15					F	F	F	S2 = 1	F

List of commands for MyCo. The version of other microcontrollers

is extended by other internal inputs (magnetometer, accelerometer, etc.)

Working with numbers

Due to the limited inputs and outputs, all that remains is to enter and read data in binary or hexadecimal form. For a person unfamiliar with the problem, it is good to find a converter from decimal to hexadecimal, and try to figure it out. Or count button clicks. Subtraction of the "output" must then be able to be transferred back to the decimal system. It's not complicated, **in a short time the three-button programmer converts from the head** , similarly to users of binary watches.

Address, command, value

Each line of the program consists of three numbers. Writing one line looks like this: 0 1 1

The first number indicates the address or line number on which it is located. The original system has a navigation for clear reading of the address - before the first command lights up, the line number flashes.
The second number is the command, MyCo has, surprisingly, 15. For an idea, I will show three commands (in the order command, name, hex code of the command and its description):
1 Port 0001 The command lights the first diode (counts from the right side)

2 Wait 0010 The program waits for the specified time. For example, 1 ms begins, and a value of 9 is, for example, 1 second

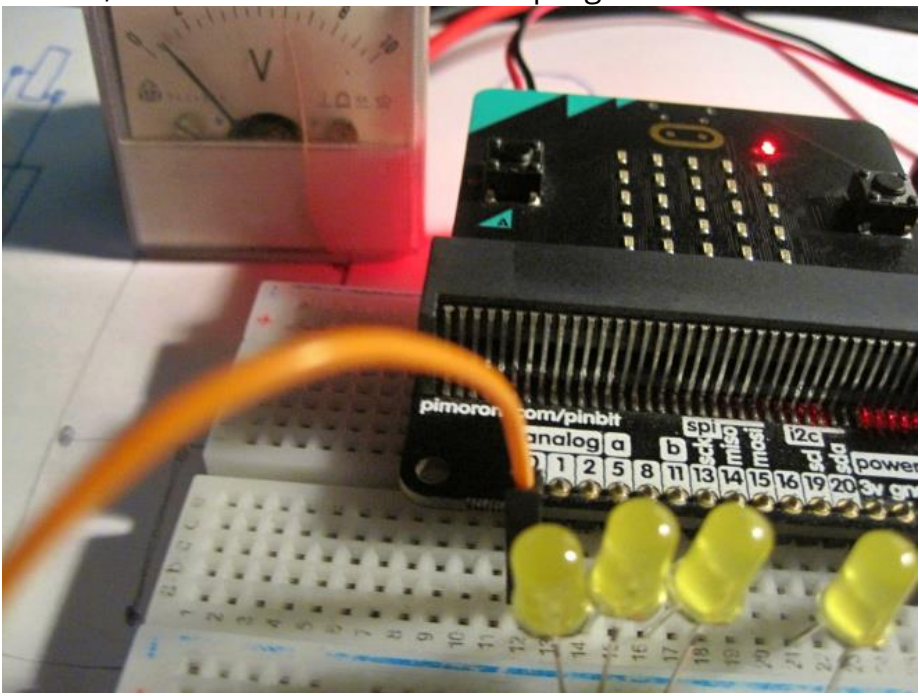
3 Jump 0011 Jump back by the specified number of lines

The third number is the value of the command, in the case of the Port command it is the output diode, in the case of the Wait time command, in the case of the Jump command it is the number of lines by which the program returns.
If I wanted to use the above directly, the program would look like this: 0 1 1, 1 2 9, 2 3 2. The result is a lit right LED.

I will try to elaborate a specific example with an explanation in the following.

Write to MyCo and run the program

Probably the most interesting chapter, this **is where the real thumb gymnastics begins** . The program works in two modes, in the edit mode and in the program run mode.



Digital and analog (PWM) outputs from Mycobit

To edit, just press the reset button and the right button at the same time. The right button jumps first line (command,

value), then to the next line. The left button enters commands and their values.

If I want to write the line "light up the first diode", the inch choreography will be left, right, left, right (1, 1 - light the first diode). And done, just confirm the program (usually both buttons) and reset (third button) to start the first lit LED!

Program repair

The code can be freely entered and edited. All you have to do is combine the reset and the right button to jump to edit mode, find the line that needs to be corrected, enter a new value and confirm.

Starting the program

Just confirm the program and restart. And it's running! And that's actually it.

You should already be able to create a simple flasher with the above-mentioned commands. For immediate testing, you can download the simulation - SPS Emu

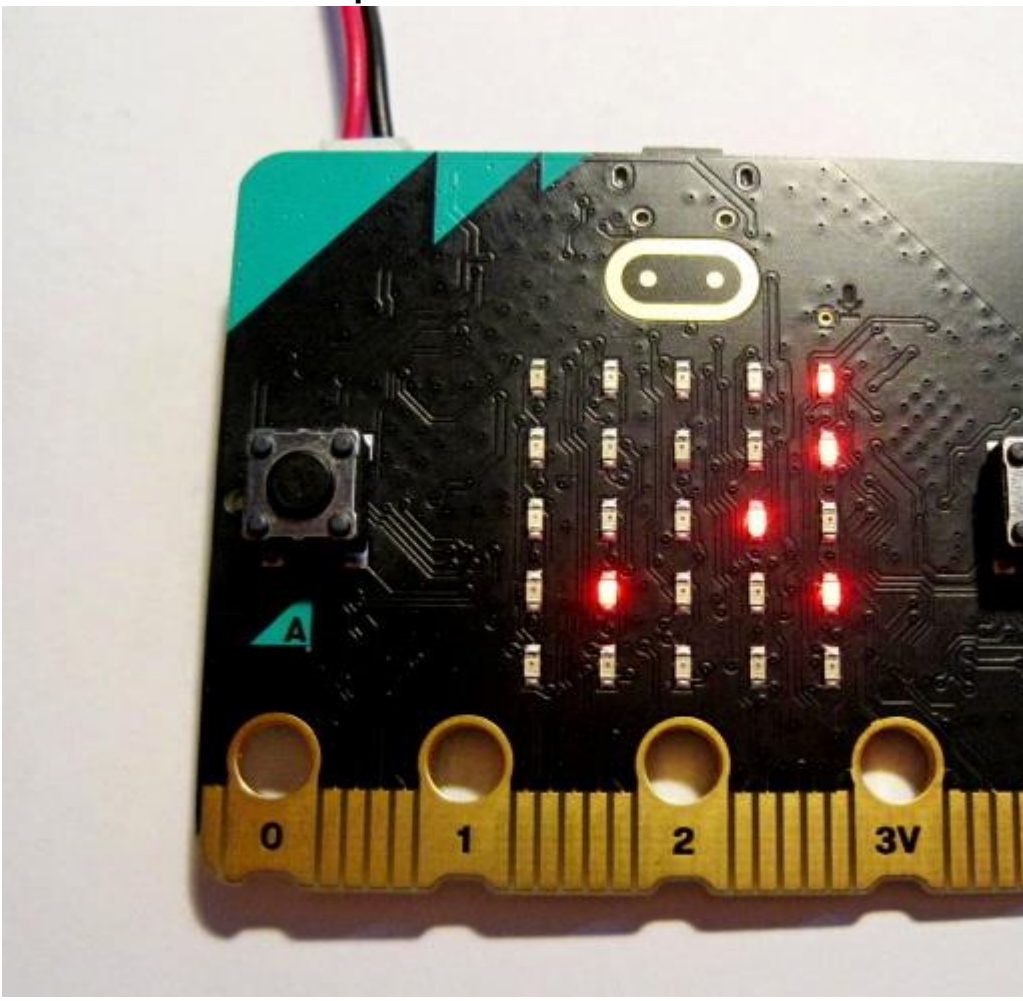
Other platforms

The original MyCo was loaded in the HT46F47 microcontroller, other platforms are available on the creator's website , such as ATtiny44 or ATmega8. The main condition is that the microcontroller has enough legs for inputs and outputs. So the sluggish craftsman could reach into the drawer for the right microcontroller and try it all out on his own in a short time.

Myco: a bit and more!

Recently, Neil Avery has also been involved in the project and prepared an implementation of MyCo for micro: bit-based boards. The program also runs on old microbits (1.3B),

however it is starting to be interesting with the micro: bit V2 . He managed to expand the program table with internal microbit inputs, so there is **an audio input and output, accelerometer or compass** .



Mycobit for microbit

All you have to do is open the author's website , download the hex file for the correct version of the board and flash it into the microbit. The link contains basic information, links, and a few sample examples. The **table of commands** is also important , here for micro: bit V2 .

In Myco: bit the reading of the program is clearer, on each "side" (4 × 4 matrix) of the edited program you can see two lines of the program, in one line the command, in the next its value. The navigation in the program is displayed in the left column of the matrix (line number) and the bottom line (page).

Example of microbit programming:

The latest implementation that Neil is currently working on is the new Raspberry Pico.

People around MyCo

The father of the programming language and the whole system is Burkhard Kainka .
Juergen Pintaske is a popularizer and author of books not only about MyCo .
And the comet that transferred this interesting system to the currently hot Neil Avery platforms - source code for mycobit / microbit and other micro-bit-like platforms.

Conclusion

Mycom was intended primarily as an educational tool for programming for children from the age of 14, but it is a fun toy for adults, and it is certainly possible for him **to create even more serious, but in any case fun app** - see page Mr. Kainky. Source codes are also available, which gives me a lot of interesting options.
But its biggest advantage is that it can be **programmed by hand** . In my opinion, this is a unique matter and especially a simple educational system with well-processed documentation suitable for children, beginners in programming, but a curious geek will certainly enjoy it.

By expanding to other platforms, MyCo gains another dimension, it can create a fairly versatile tool from mostly single-purpose programmed microcontrollers.

For the design of more complex programs, it is advisable to take paper and a pencil on hand, which will also not hurt in today's over-technological times. One works well with imagination and logical thinking in programming. And one more important thing, **MyCo programming is just addictive!** :)
In the next sequel, **we will show and try a specific program** . So you can look forward!

Links and forums for MyCo

- TPS / MyCo Facebook group
- Programming in MyCo
- Original kit
- Kits in the finished board
- Explanation of DEC / HEX transmissions
- About

- Latest Posts

Honza Horáček
Amateur in electronics, micro: bit user and programmer of low-threshold languages - Makecode, TPS, Gamemaker.