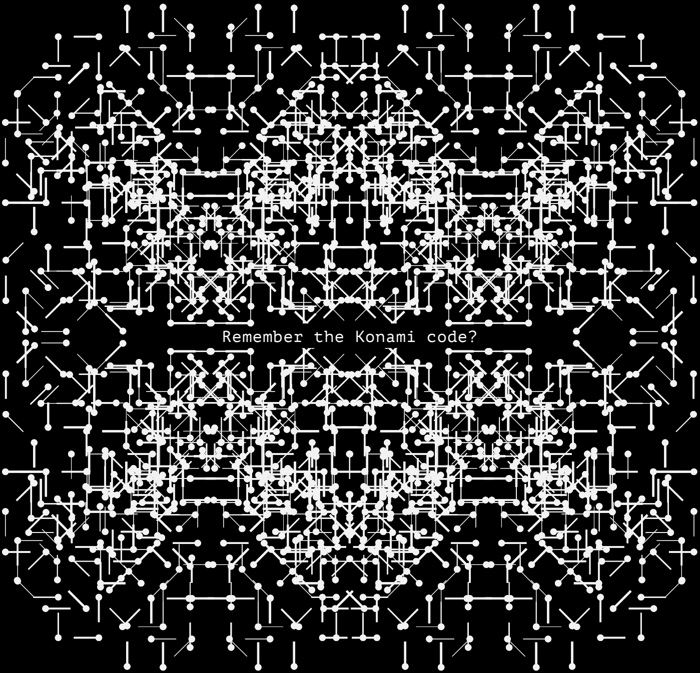


Hackaday Belgrade 2018

Badge User Manual

● Rev 1.0

The Hackaday Be BASIC retrocomp

is the work of **Voja Antonic** and **Jaromir Sukuba**. This is a computer system that hangs around your neck and it was made to be hacked. Show us the most creative BASIC program, music composition, CP/M hack, hardware add-on, or demo you can come up with and show it off during the badge hacking presentations this evening!

lgrade
puting badge

Badge Project Page
hac.io/Mz3r

Badge Firmware Repo
hac.io/yGLKZp

You can connect to this badge via serial at 19200 8N1. **(GND=GND, C13=RX, C14=TX).**

Full serial documentation is found here:
hac.io/qhN2

The Hackaday Belgrade badge includes an interpreter for the BASIC programming language. This is a simple language to pick up and rewards you for being clever (by requiring less typing)!

Itsa Me!

mario

Read badge
BASIC docs here:

hac.io/fMRF

```
Here are some
tips about
using BASIC
on the badge:
```

There are sixteen slots numbered 0 through 15; **save 0** will save your current program to slot #0

There is a “BRK” button (near power button) to break out of program execution

Don’t forget the line numbers. Space them apart by 10 so you can add lines in between later on.

Use the command **more** to page through long code (**list** for shorter programs)

Find the complete
WORD reference
for this version
of BASIC on the
badge page:

hac.io/fMRF

BASIC CLI commands

run
save X
load X
free
list
more
memclr (clears current program)

Standard words

println
print
if
goto
gosub
return

for
next
end
let
rem

Badge custom words

wait X
rnd V,X
clrscr
setxy X,Y
color X,Y
chr X
ein X
eout X,Y
edr X,Y
sload

ssave
uin X
uout X
led X,Y
tune A,B,C,D
termt X
termup
input "STRING"
peek X
poke X,VALUE

BASIC
code examples

Try typing out these example programs to get the hang of it. Remember to clear the code buffer before you start a new program (memclr).

Code examples online:
hac.io/wSdBGj

Print out all possible characters

```
10 for i=32 to 255
20 chr i
30 next i
```

Bling the LEDs randomly (d is diode [0..2], s is state [0..1])

```
10 d = rnd 2
20 s = rnd 1
30 led d,s
40 print “Press BRK to return
to BASIC prompt”
50 wait 100
60 goto 10
```

Play random Major triads

```
10 let i = rnd,77
20 if i<53 then goto 10
30 tune i,i+4,i+7,400
40 goto 10
```

Take user input (this seeks number but you might do something cool with text if you’re clever)

```
5 rem Repeats a user value
10 let a = input “Enter value: “
20 print “You entered: “
30 println a
```

Scroll some text. The space before the text erases the previous character in that position

```
10 let i=0
20 color 14,12
30 clrscr
40 setxy i,0
50 print “ Scroller”
60 i=i+1
70 wait 150
80 if i<32 then goto 40
90 color 15,0
```

Bounce ball on screen. Shows moving to different areas on screen, changing color, using delays and subroutines, and manual screen refresh (term 0 shuts off automatic scanning, termup triggers manual refresh) for a smoother animation than the scroll text example

```
5 termt 0
10 let x = 39
20 let d = 0
30 clrscr
40 color 11,0
50 setxy x,10
60 chr 32
70 if d = 1 then gosub 200
80 if d = 0 then gosub 300
90 chr 254
95 termup
100 if x = 0 then d = 1
110 if x = 39 then d = 0
120 wait 50
130 goto 50
200 x = x + 1
210 return
300 x = x - 1
310 setxy x,10
320 return
```

Your notes:

Music Syntax

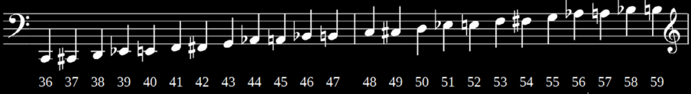
Find the complete music reference on the badge page:

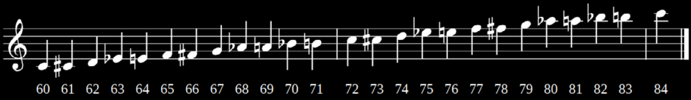
hac.io/ZGfzd

This badge has a built-in speaker and you can compose music for 3 voices using a simple scripting language. Use the BASIC interpreter to access these features with the syntax:
tune voice1,voice2,voice3,duration

Duration is the number of milliseconds this chord should be held. The voice values use Scientific Pitch Notation (**hac.io/hGf7**) where middle C is 60. Do a C major chord held for 1 second would look like this **tune 60,64,67,1000**

-- here's a cheat sheet:





As an example, try typing this into your BASIC interpreter (you may want to type memclr before your start to clear any existing code)

```
10 tune 0,0,72,131
20 tune 0,0,74,131
30 tune 0,0,77,131
40 tune 0,0,74,131
50 tune 70,74,81,262
60 tune 0,0,0,131
70 tune 70,74,81,393
80 tune 60,76,79,524
90 tune 0,0,0,131
100 tune 0,0,72,131
110 tune 0,0,74,131
120 tune 0,0,77,131
130 tune 0,0,74,131
140 tune 69,72,79,262
150 tune 0,0,0,131
160 tune 69,72,79,393
170 tune 62,69,77,393
180 tune 62,69,76,131
190 tune 62,69,74,655
```

Your notes:

Zork & CP/M

It’s a retro computer, does it play Zork? Absolutely.

For those new to it, Zork is one of the first interactive computer games. It’s really immersive and still holds up today. It has a language parser, so if you want to go North you type “Go North”.

Pro Tip: Use shift-enter as Zork is expecting a special character for user input.

Mild Spoilers: begin by opening the mailbox and reading the letter. Then look around and figure out how to get into the house.

Zork is running because your badge includes a Z80 emulator running the CP/M operating system. You can run other CP/M programs.

There are 7 drives:
A=22kB-RAM
B,C=128kB-Readonly
D,E,F=512kB FLASH disk

xmodem can be used to load any CP/M programs you want onto one of the FLASH disks

Here's how get them onto the badge:

Xmodem syntax

Type “b:” to switch to read only drive
Type “xmdm r d:FILENAME.EXT” to prepare badge for an incoming file on the d: disk
Send the file from your computer using xmodem protocol

Your notes:

Acknowledgements

Hardware Design

Voja Antonic

Software Design

Jaromir Sukuba

BASIC Interpreter based on **Adam Dunkels uBASIC**

dunkels.com/adam/ubasic/

Z80 CPU emulator core by **Udo Munk**

github.com/udo-munk/z80pack

Sponsors

Supplyframe

SevenBridges

hackaday.io/belgrade