

# Virtual TPS

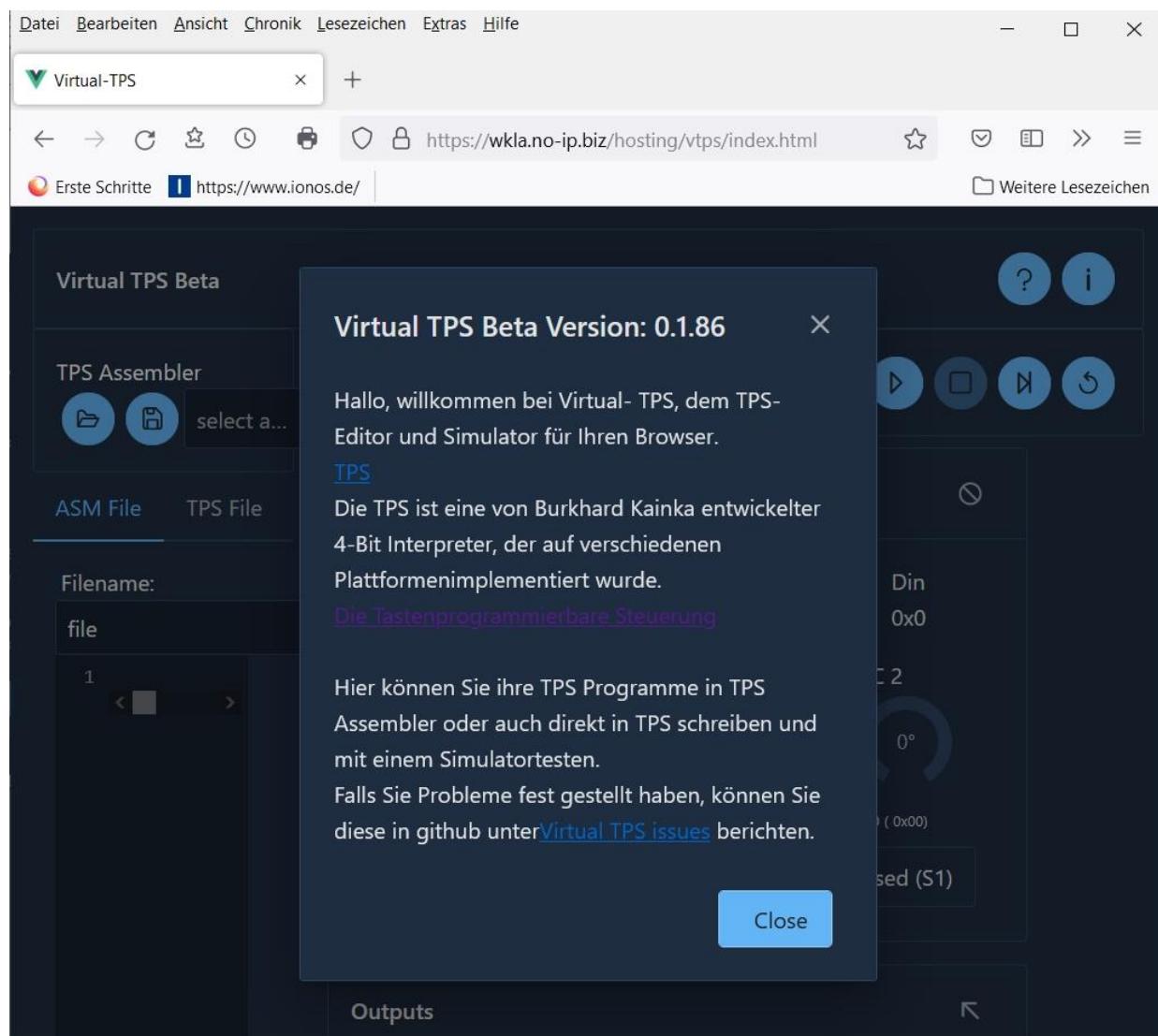
<https://www.elektronik-labor.de/Lernpakete/TPS/TPS47.html>

[Elektronik-Labor](#) [Literatur](#) [Projekte](#) [Lernpakete](#) [TPS](#)

## Virtual TPS Review by Burkhard Kainka

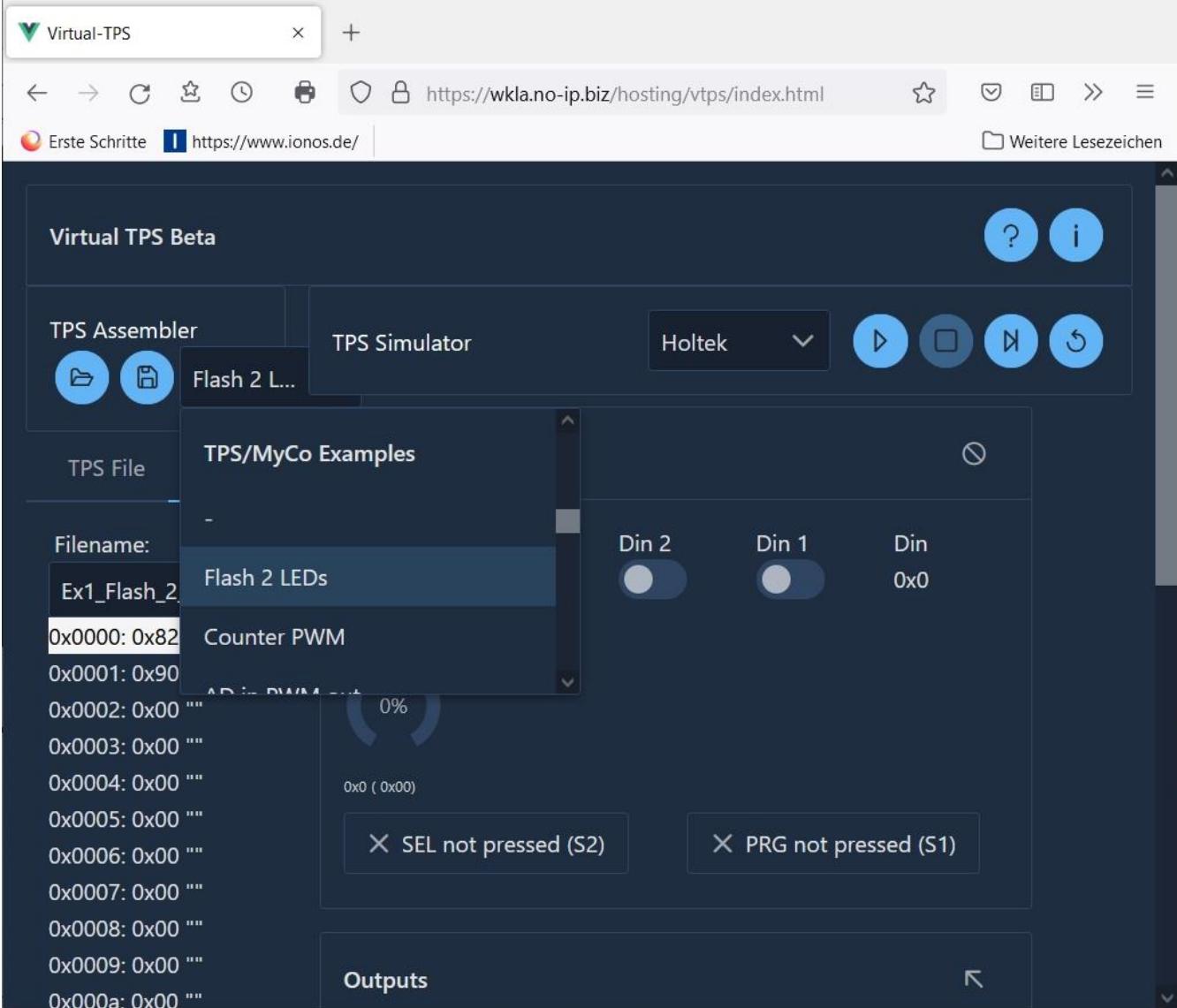
TPS / MyCo is a project, where you can program a small 4 Bit Processor via just 3 push buttons. There is a project to download an Emulator to run TPS on your PC, and then to program the relevant hardware.

Wilfried Klaas now did the next step: implementation to run online – no download necessary. A quick translation of Burkhard's review for the English speaking community.



Virtual TPS Beta: <https://wkla.no-ip.biz/hosting/vtps/index.html>

Virtual TPS by Wilfried Klaas is an online simulator that anyone can use to get familiar with the system, even without the any hardware. In addition to the original TPS version with the Holtek controller, this software also supports all other systems developed to date, including **Arduino, MicroBit, Rpi Pico, and ESP32**.



For a first test I selected the Holtek version and then loaded the first program "Flash 2 LEDs" under TPS/MyCo Examples.

What??? -- only two instructions?

I had to get my memories back from then.

There it says: **Go to Page 2, Jump to Address 0.**

Now I remembered: the first examples call up the pre-programmed code from page 2 onwards. So, now let's scroll down.. .

The screenshot shows the Virtual-TPS software interface. On the left, a list of memory addresses and their values is displayed:

- 0x001e: 0x00 ""
- 0x001f: 0x00 ""
- 0x0020: 0x11 ""
- 0x0021: 0x28 ""
- 0x0022: 0x18 ""
- 0x0023: 0x28 ""**
- 0x0024: 0x34 ""
- 0x0025: 0x71 ""
- 0x0026: 0x54 ""
- 0x0027: 0x59 ""
- 0x0028: 0x26 ""
- 0x0029: 0x34 ""
- 0x002a: 0x00 ""

In the center, the **Outputs** section shows five digital outputs (Dout 4, Dout 3, Dout 2, Dout 1) and one PWM output (PWM 1). Dout 4 is active (blue), while others are off. The PWM 1 value is set to 0%.

The **Internal** section displays various parameters:

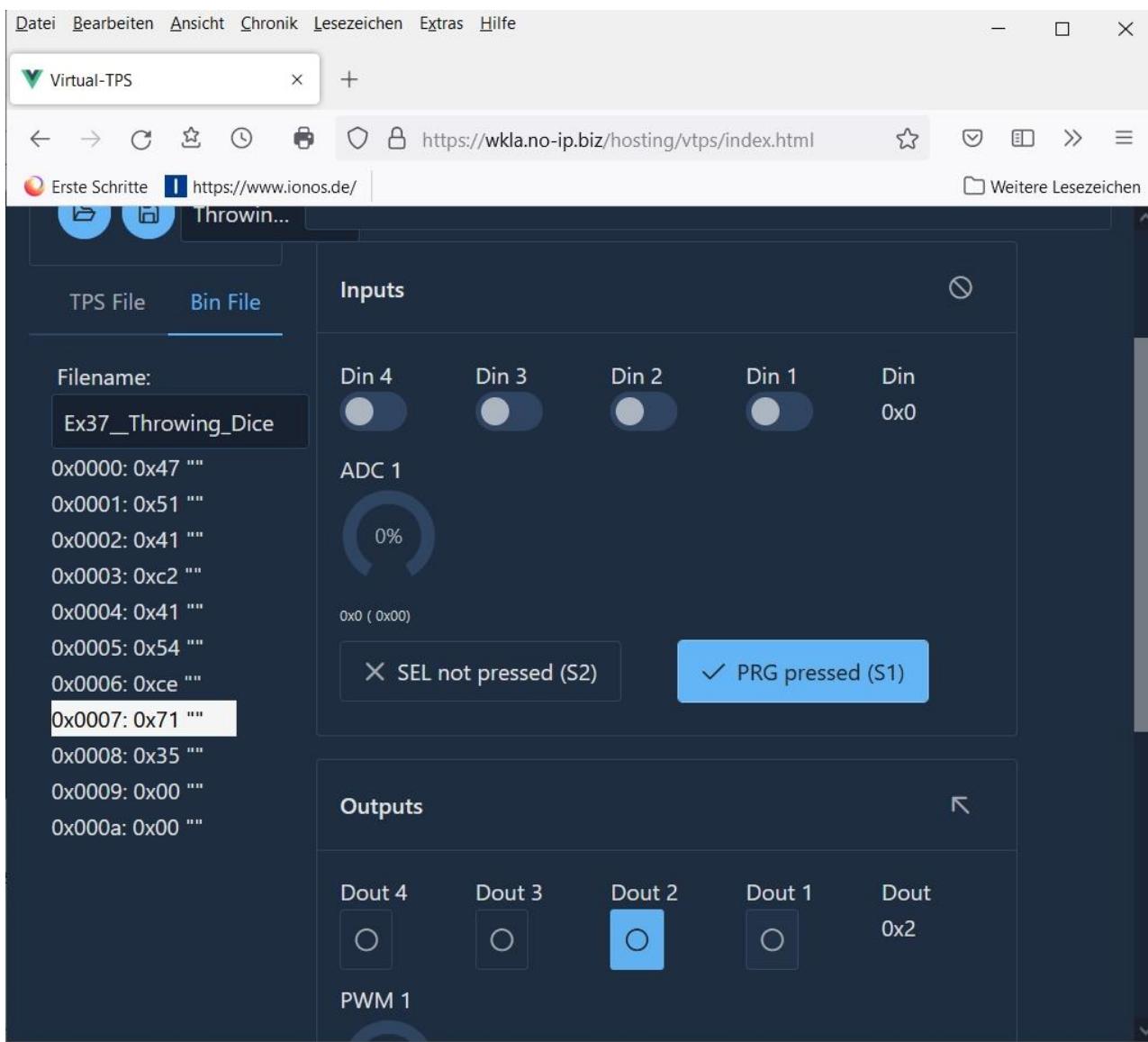
A	0x00	B	0x00	Page	0x02
C	0x00	D	0x00	RAddr	0x0000
			Addr	0x0023	
cmd	0x1	data	0x8	delay	500

On the left, there is a button labeled **export as** and a dropdown menu set to **TPS**.

When the program is executing, you can always see exactly which line is currently being processed.

Currently here waiting for 500 ms.

Before that, the output Dout4 was switched on with 0x18, (8 = 1000), the left LED lights up.  
After this comes the jump back by four addresses: 0x34.



The games included in Franzis' current TPS release "Maker Kit Controller Board" are also included among the examples. Here I tested the **Dice Example**. As long as you press S1, numbers from 1 to 6 scroll through. If you let go of S1, the last number remains. I also typed the same program into my real TPS. There, the numbers run through so quickly when the button is pressed, that all three LEDs showing 1,2,3,4,5,6 seem to light up permanently. Only when you let go S1, does the binary number on the dice stop. Everything runs slower in the simulation so that you can follow the counting up directly. The program can be exported in TPS format as a text file **Ex37\_Throwing\_Dice.tps** and then be modified in an editor.

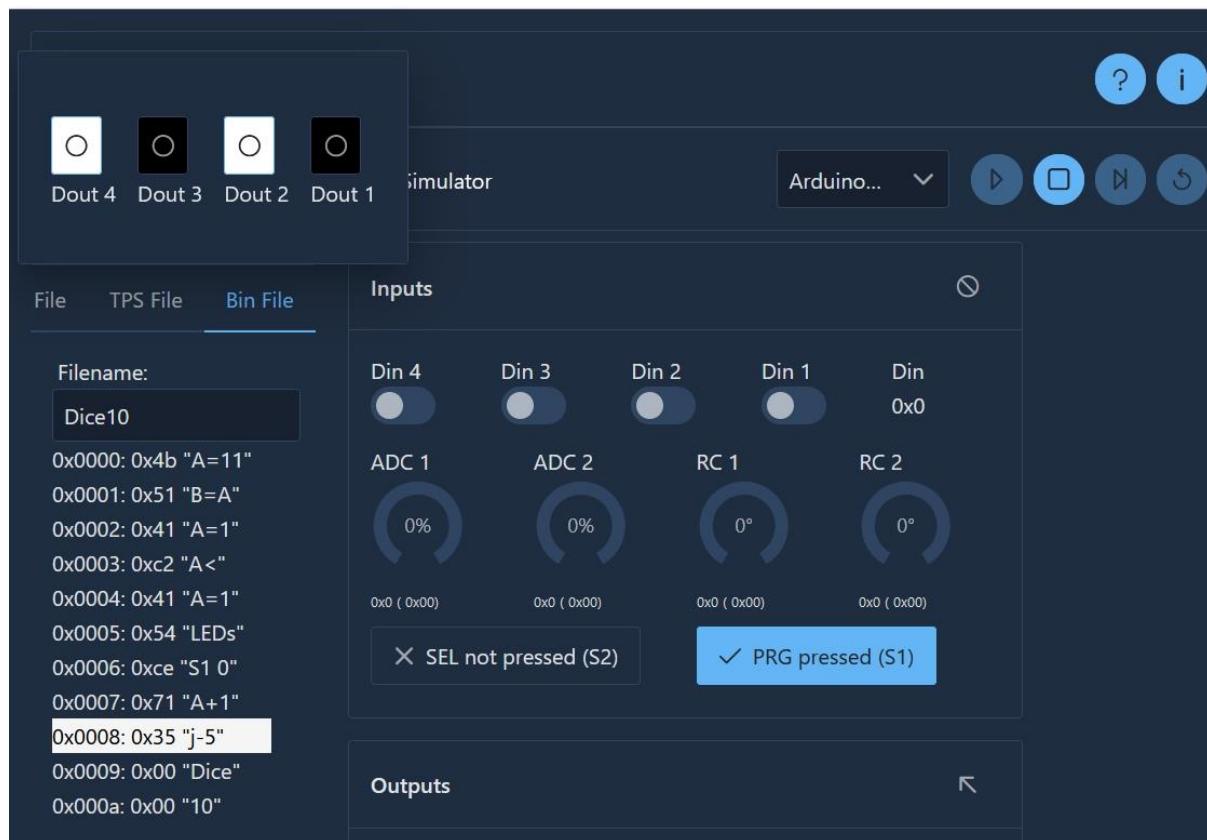
```
#TPS:TPS Holtek
0x00,4,7,""
0x01,5,1,""
0x02,4,1,""
0x03,C,2,""
0x04,4,1,""
0x05,5,4,""
0x06,C,E,""
0x07,7,1,""
0x08,3,5,""
0x09,0,0,""
0x0A,0,0,""
```

I wanted to comment this program in a similar way as in the manual.

And at the same time I wanted to test a change.

The dice should now roll up to 10. To do this, the value 11 must be brought into B. The new program is now called Dice10.tps.

```
#TPS:TPS Holtek
0x00,4,B,"A=11"
0x01,5,1,"B=A"
0x02,4,1,"A=1"
0x03,C,2,"A<""
0x04,4,1,"A=1"
0x05,5,4,"LEDs"
0x06,C,E,"S1 0"
0x07,7,1,"A+1"
0x08,3,5,"j-5"
0x09,0,0,"Dice"
0x0A,0,0,"10"
```



And indeed, the program works. As long as S1 is pressed, the binary numbers run from 1 to A.

### And one further explanation:

Top left you see in this screen print the additional display window of the 4 OUPUT LEDs, opened via the arrow left in the Output LED area.

This Window has been added to prepare for

**Reading the 4 outputs on the display monitor via 4 LDRs ( Light Dependant Resistors )**  
plus a series resistor each. The resulting voltage drives a CD4093 chip with 4 LEDs on your desk. **I tried it and it works – real LEDs show the display of 4 Virtual Outputs.**