PiKoder/SPE

User Manual

Version 1.0

dated 07/15/23

Gregor Schlechtriem

webmaster@pikoder.com

www.pikoder.com

Content

Overview 3

Features 5

Pin Description and Packaging 7

Description of pins....8

Standard application 10

LED status indication (in USB mode)....10

The PiKoder Control Center G2 13

Getting started....13
Real-time control....14
Watchdog....14
UART baudrate....14
PPM Settings....14
Startup....15
Save Parameters....15

Serial Interface 17

ASCII Command Interface....17
Mini SSC Protocol....19

Connect the SPE to a Raspberry Pi 21

Connect the SPE to an Arduino 23

1

Overview

The PiKoder/SPE (Serial PPM Encoder) is a single chip solution using a modern PIC controller to generate a PPM-Stream with 2 – 8 channels controlled from serial port (UART) of an SBC (Single Board Computer) such as a Raspberry Pi or an Arduino.

As a second generation PiKoder the SPE features a USB interface for customizing the controller to the requirements of your application in addition to the serial port.

This User Manual covers the features, the programming, and the command interface of the PiKoder/SPE as well as common applications.

In section 2 you will find a brief description of key features, the overall function and an overview of the interfaces supported. It is recommended to carefully read this section to get a good basic understanding of the PiKoder/SPE.

The next two sections 3 and 4 deal with the controller hardware. Section 3 provides the pinning and section 4 describes the reference schematic for the PiKoder/SPE. **Please note that the applications and interfaces described in the following sections assume that the PiKoder/SPE is setup in line with this reference schematic.**

Your next step is most likely to commission and test your PiKoder/SPE. Rather than using the "bits and bytes"-serial interfaces directly, which are laid out in section 6, you may consider using the more sophisticated PCC G2 (PiKoder Control Center Generation 2) Windows 10 software with a graphical user interface described in section 5.

Section 7 demonstrates how you would interface your PiKoder/SPE to a Raspberry Pi and Section 8 focusses on connecting the PiKoder/SPE to an Arduino.

This User Manual is based on the most recent hard- and firmware version 4.0 available for the PiKoder/SPE and the related PCC programming software.

Please check for updated information and new software releases on www.pikoder.com.

Hyperlinks were integrated into the text for convenience. You would also find all downloads referenced on the PiKoder/SPE webpage.

Please share with me any comments, improvement ideas or errors you will find or encounter in working with your PiKoder/SPE. I can be reached at webmaster@pikoder.com. Thank you very much!

2

Features

This section will familiarize you with the feature set and provide a high-level overview of the intended use of the PiKoder/SPE allowing you to customize the controller to your specific needs and requirements.

The PiKoder/SPE generates a PPM stream which is controlled by a command protocol via a UART port. More specific, the PiKoder/SPE features a two-way ASCII-Protocol designed to be used in combination with standard terminal programs such as (but not limited to) Tera Term and TTY.

In addition, various application specific parameters such as startup position after powering up, failsafe value per channel, PPM polarity, the number of channels and the baud rate of the UART can be adjusted to your specific needs and the settings are maintained in a non-volatile memory.

You can connect an SBC (Single Board Computer) such as your Arduino or Raspberry Pi without any additional interface hardware directly to the PiKoder's UART (please refer to sections 7 and 8 for more details). In this capacity the PiKoder would free up the Arduino or Raspberry Pi of responding to real-time events such as generating pulses within "real-time" and thus free up resources such as internal timers and PWM generators ('Set and Forget'-function).

You would program the PiKoder/SPE by connecting it with a standard USB cable to your computer. A free graphical and intuitive configuration and control program, the "PCC G2 (PiKoder Control Center Generation 2)" is available for Windows 10, making it simple to test and program the controller over USB (please refer to section 5 for more details). All settings are stored in a non-volatile memory.

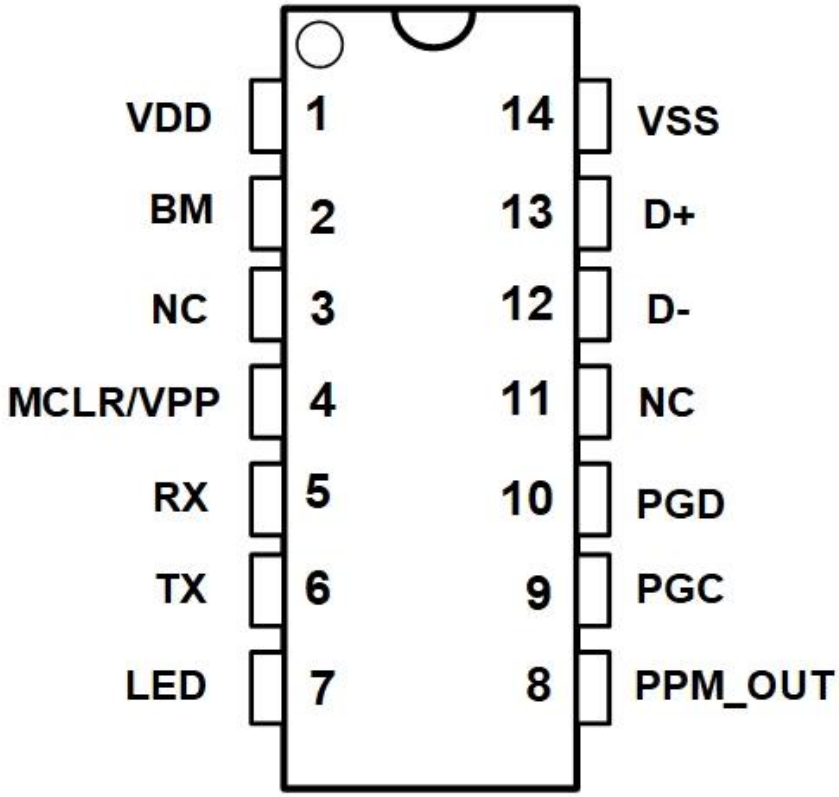
The PiKoder/SPE does support a failsafe position for the use in autonomous and RC applications. You can activate a watchdog-timer for 0.1 s to 99.9 s to

monitor the communication. If no message is received within this preset time frame, then the PiKoder would set all channel values to the failsafe position.

3

Pin Description and Packaging

The PiKoder/SPE comes in a 14 pin DIP package (see below). The device operates from 3.3 – 5 Volts. Please refer to the PIC 16F1455 data sheet from Microchip (www.microchip.com) for complete electrical and physical specifications.



A complete description of the pins is given on the following page. If a different package would be required for your application, then please contact sales@pikoder.com for more information.

Description of pins

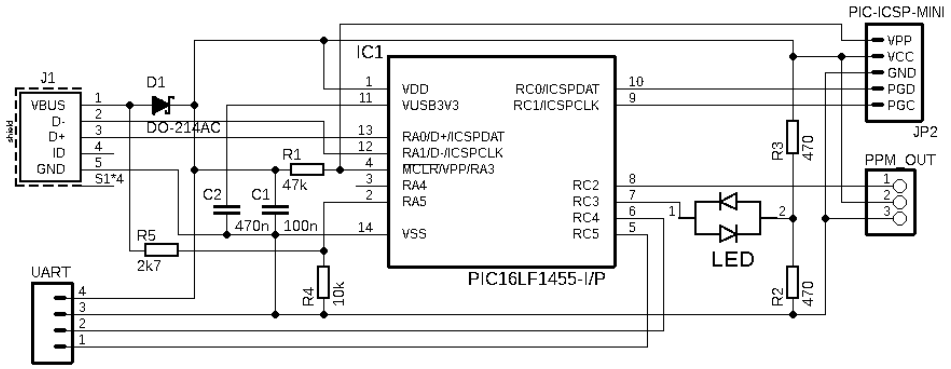
Pin	Symbol	Description
1	VDD	Supply voltage. Connect to 3,3 – 5 V DC
2	BM	Boot Mode. If this pin is high while booting, the PiKoder will start in USB mode (activating the USB interface), otherwise UART would be active
3	NC	Not connected (reserved for later use)
4	MCLR/VPP	Reset pin, active low. Connects directly to Vss for automatic reset at power up.
5	RX	UART receive. Active only when BM = 0 when starting up
6	TX	UART transmit. Active only when BM = 0 when starting up

7	LED	LED indicator output (please refer to Appendix B for more information)
8	PPM_OUT	Output pin with PPM signal
9	PGC	Clock pin for In-Circuit-Serial-Programming
10	PGD	Data pin for In-Circuit-Serial-Programming
11	VUSB3V3	Positive supply for USB transceiver; 470n capacitor required
12	D-	USB data line
13	D+	USD data line
14	VSS	Ground connection

4

Standard application

The following schematic shows the standard application of the PiKoder/SPE. Please note that the bi-color LED indicates the USB status and is optional and will remain red in UART mode.



LED status indication (in USB mode)

This following table shows the status of the USB connection ("DEVICE STATE") based on the LED signaling state ("LED RESPONSE").

DEVICE STATE	LED RESPONSE
DETACHED	OFF
ATTACHED	RED
POWERED	RED
DEFAULT	RED & GREEN
ADRESS	RED & GREEN
CONFIGURED	GREEN

Given the simplicity of the schematic, the PiKoder/SPE can easily be evaluated on a prototype board. If you are looking for a more permanent prototype then please consider the evaluation board which is available as a kit on **www.pikoder.com.**

5

The PiKoder Control Center G2

The USB2PPM's USB interface provides access to configuration options as well as support for real time control. The *PCCpro PiKoder Control Center* is a Windows 10 based graphical tool that makes it easy for you to use this interface. For almost any project you will start by using the *PCCpro PiKoder Contol Center* to set up and test your PiKoder. This section explains the features of the *PCCpro PiKoder Control Center*.

Getting started

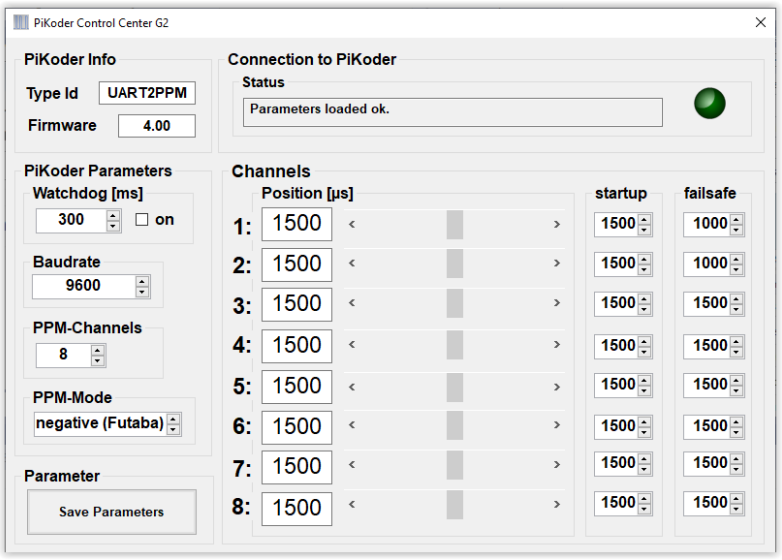
The hardware setup is simple and straight forward with a PiKoder/SPE engineering board or your own prototype. You must connect your PiKoder/SPE with the USB port of your Windows PC using a suitable cable. This cable will provide also for the power supply of the board.

When you connect your PiKoder/SPE for the first time, Windows will automatically install the drivers needed. On the engineering board the LED would be blinking red in fast mode during the installation process. Please refer to section 4 for more information regarding the LED indicator.

The PCC G2 app is available free of charge in the Windows app store. It is highly recommended that you install the latest version of the program to enjoy the complete feature set of your PiKoder/SPE.

Please make sure to connect your PiKoder to a USB port before starting the PCC as the PCC will scan your computer's USB ports to automatically connect to the PiKoder.

The connection is made and the actual parameters of the PiKoder are displayed.



You would now have full control of your PiKoder/SPE: either for real-time control by the sliders or for changing the settings with respect to PPM polarity, number of channels, startup values, etc.

Real-time control

The sliders are used for controlling the PiKoder's outputs and the respective numerical fields monitor the status in real time displaying the current channel value in µs within a range of 1000 – 2000 µs. A separate row of controls is displayed for each of the active channels.

Watchdog

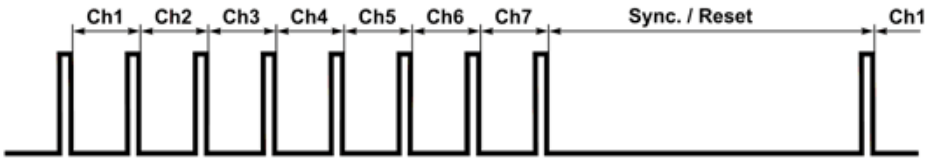
The watchdog would be used to make sure that the failsafe values for all channels are set when the connection to the host would be interrupted. Activating the watchdog is comprised of setting the value (default is 300 ms) and activating the watchdog itself by checking the respective box.

UART baudrate

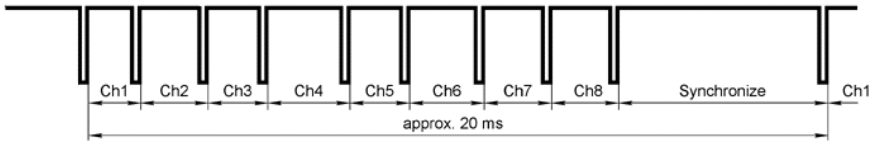
The PiKoder/SPE offers baudrates of 9600, 19200, 38400, 57600 and 115200. The setting would become effective after the next reset of the PiKoder.

PPM Settings

The PiKoder/SPE features either a negative or positive PPM stream. The polarity is related to the output of the PPM_OUT pin. The following picture presents the signals. Also, you can select the number of channels encoded in the stream. A range from 1 to 8 channels is supported; more than 8 channels can technically not be encoded in a standard 50 Hz PPM frame.



Above: "positive" PPM Frame at PPM_OUT (7 channels selected)



Futaba R/C transmitter trainer interface signal (inverted PPM modulation)
Ch1..8 - variable pulse width from 1 to 2 ms (center 1,52 ms)
Synchronize - usually around 5 ms

Above: "negative" PPM Frame at PPM_OUT (8 channels selected)

Startup

In many applications the pulse width for the neutral value would be 1.5 ms. However, some ESCs (Electronic Speed Controllers) for drones might need 1.0 ms as a start-up value to make sure that they are not spinning immediately after turning the PiKoder/SPE on. To address these applications, you would set the initial value and safe the parameters by hitting the respective button on the lower right of the screen.

Save Parameters

Changes made while using the *PCC G2* will not be permanent unless you select to save the parameters. Hitting this button transfers all settings into the non-volatile memory of the PiKoder/SPE to be retrieved when started up the next time.

Please note that saving new values may take some time and requires disabling some internal interrupt logic. This may result in erroneous servo behavior.

6

Serial Interface

The PiKoder/SPE supports a two-way ASCII-Protocol named Ascii Command Interface (ACI) designed to support controlling PiKoders with standard terminal programs such as (but not limited to) Tera Term.

ASCII Command Interface

The ASCII Command Interface (ACI) is probably the most versatile way to program any PiKoder without any specific host software such as the *PCC G2 PiKoder Control Center*. All commands are simple ASCII and are sent using a Windows based terminal program such as Tera Term. The commands can be typed in right away and the response of the controller is readable without referring to any specific code tables. Please note that neither 'CR' nor 'LF' is needed to send the command to the controller.

There are two basic types of commands: commands for querying parameters and for setting parameters.

If a parameter is read the PiKoder will provide for proper formatting by sending a “CRLF” prior to sending the parameter value and support readability by sending another “CRLF” after the parameter value.

If a parameter is set the PiKoder will acknowledge the proper execution by sending an “!” framed by “CRLF”.

If a command could not be interpreted at all then a question mark '?' framed by 'CR' 'LF' would be echoed. Please note that protocol syntax checking is extremely limited now.

The following ACI commands are available:

- '?': query the PiKoder type information; PiKoder/SPE will respond in a format 'UART2PPM' framed by 'CR' 'LF'
- '0': query the firmware version; PiKoders will respond in a format 'n.nn' framed by 'CR' 'LF'
- 'i?': query the current pulse width for channel i (i = 1..8); PiKoders will respond 'CR' 'LF' 'xxxx' 'CR' 'LF' with xxxx representing the pulse width in µs
- 'i=xxxx': set the pulse width for channel i to xxxx µs (xxxx in decimal format, i = 1..8); PiKoders will acknowledge execution of the program by sending an 'CR' 'LF' '!' 'CR' 'LF'
- 'B?': query the current UART baud rate. PiKoder/SPE will respond a number from 0..4 meaning 0=9600, 1=19200, 2=38400, 3=57600, and 4=115200 - the command is not case sensitive
- 'B=k': set the UART baud rate with k = 0..4 (see encoding above) - the command is not case sensitive and the PiKoder will acknowledge execution with a '!' framed by 'CR' 'LF'.
- 'C?': query the current number of PPM channels in the frame. PiKoder/SPE will respond a number from 1..8 - the command is not case sensitive
- 'C=k': set the number of channels in the PPM frame with k = 1..8 - the command is not case sensitive and the PiKoder will acknowledge execution with a '!' framed by 'CR' 'LF'.
- 'Nk?': query the startup value for channel k. PiKoders will respond 'xxxx' with xxxx representing the pulse width in µs (xxxx in decimal format, k = 1..8) - the command is not case sensitive
- 'Nk=xxxx': set the startup value for channel k to xxxx µs (xxxx in decimal format, k = 1..8); - the command is not case sensitive and the PiKoders will acknowledge execution with a '!' framed by 'CR' 'LF'.
- 'P?': query the current PPM polarity. PiKoder/SPE will respond 'N' with 'P' for positive and 'N' for negative - the command is not case sensitive
- 'P=x': set the PPM parameters with x representing the polarity of the PPM signal ('P' or 'N'). The command is not case sensitive and the USB2PPM will acknowledge execution with a '!' framed by 'CR' 'LF'
- SU]U], sU]U]: will save the current parameters to the controller's non-volatile memory making the current servo positions the start up positions after powering up; returns a '!' upon successful completion framed by 'CR' 'LF'
- 'T?': query the current watchdog value. PiKoders will respond 'xxx' with xxx representing the time in increments of 20 ms (frames missed) - the command is not case sensitive.

- 'T=xxx': set the timeout value in increments of 20 ms; - the command is not case sensitive and the PiKoder/SPE will acknowledge execution with a '!' framed by 'CR' 'LF'.
- 'W?': query the current watchdog status. The PiKoder/SPE will respond either with '1' indicating an active watchdog or '0' for watchdog not activated. - the command is not case sensitive.
- 'W=b': activate / deactivate watchdog. For b=1 the watchdog would be activated, for b=0 the watchdog would be turned off; - the command is not case sensitive and the PiKoder/SPE will acknowledge execution with a '!' framed by 'CR' 'LF'.

Mini SSC Protocol

The PiKoder/SPE supports the miniSSC protocol for the first eight channels. The protocol is simple, and it only takes three serial bytes to set the target position of one servo.

The miniSSC protocol is to transmit 0xFF as the first (command) byte, followed by a servo number byte, and then the 8-bit servo target byte for the servo position. This means that the miniSSC protocol allows for 255 positions (0xFF is not available as it indicates a command start) resulting in about 4 µs as a maximum resolution compared to 1 µs in the Ascii command mode.

A servo target byte of 127 (0x7F) will always indicate the neutral position maintained as a PiKoder/SSC parameter.

The servo position will be calculated by

servo position = (target byte – 0x7F) * 4 µs

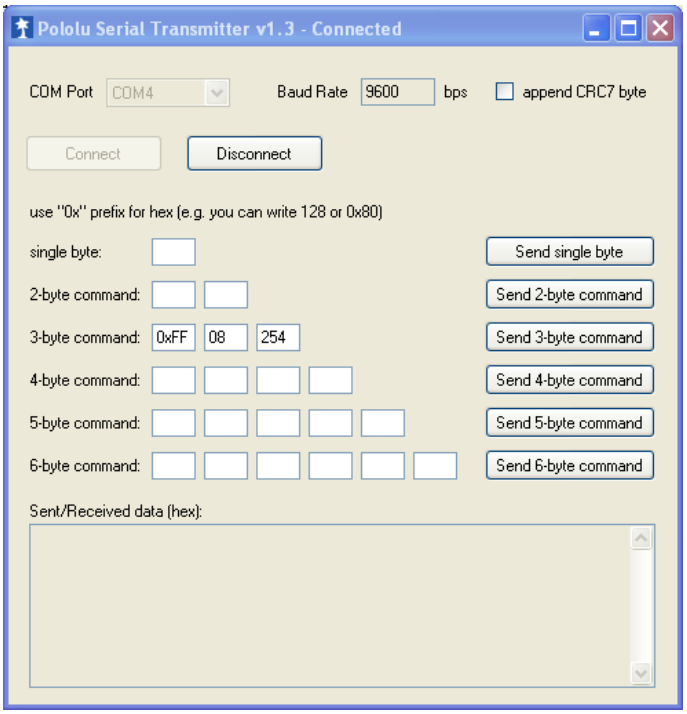
for a target byte bigger than 0x7F and by

servo position = target byte * 4 µs

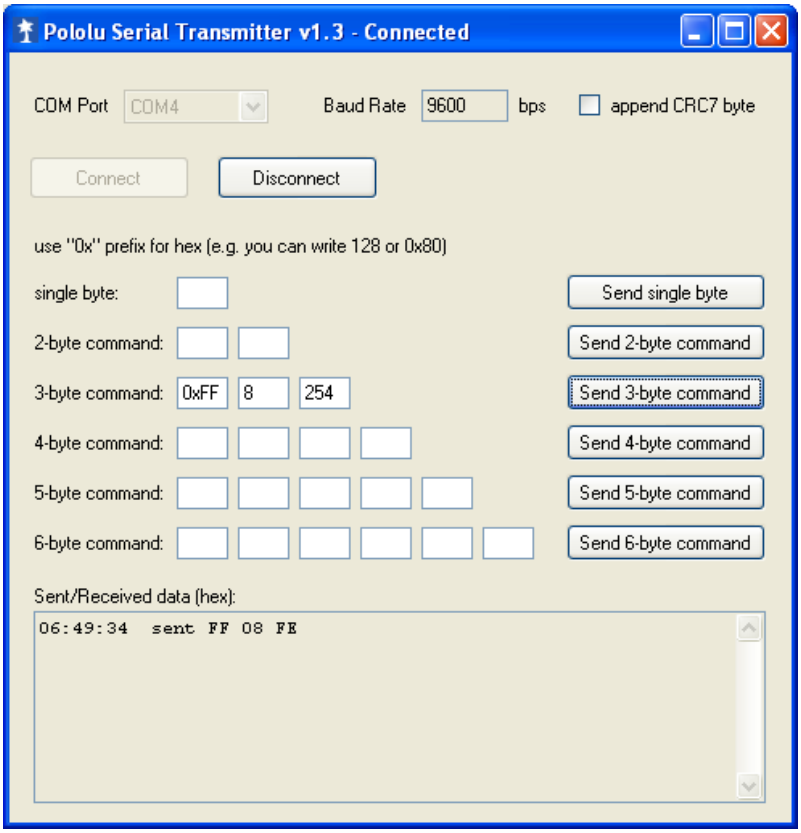
for a target byte smaller than 0x7F.

For testing the miniSSC protocol you may want to use a byte oriented tool such as the "Pololu Serial Transmitter" utility for Windows (see https://www.pololu.com/docs/0J23 for more details).

After installing and starting the software you would have to connect to the PiKoder/SPE by selecting the COM port and pushing the connect button. The miniSSC-protocol is a three-byte command and should be entered in the respective column as shown below.



Once you hit the "Send 3-byte command"-button the bytes are sent to the PiKoder (see below); as per protocol definition there is no response by the SSC.



7

Connect the SPE to a Raspberry Pi

The PiKoder/SPE releases your Raspberry Pi from generating real-time pulses for the PPM frame ('Set and forget'-function).

This is advantageous when using operating system such as LINUX, because their real-time capabilities are limited due to the number of concurrent tasks resulting into a limited precision of the signals generated.

The PiKoder/SPE connects to your Raspberry's UART. It operates with 3.3 Volts, which is supplied by your Raspberry Pi also.

The schematic is shown on the next page.

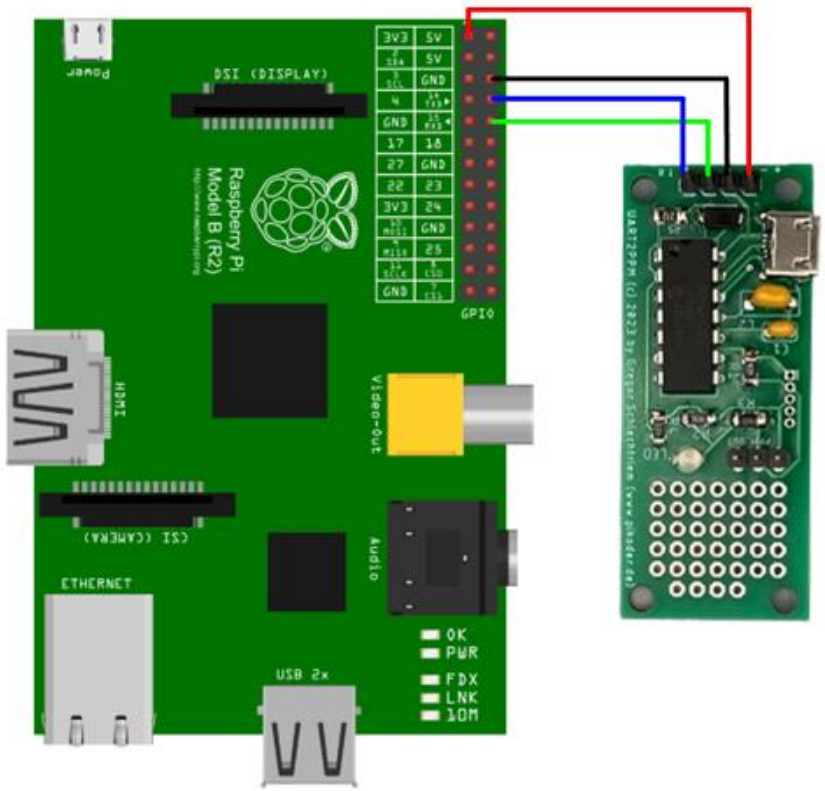
Software configuration and hardware setup

The software configuration and the hardware setup require three simple steps:

- Turning of the UART as a serial console
- Wiring
- Installing terminal software such as Minicom

Turning of the UART as a serial console

In your Raspberry Pi's default configuration, the UART is used as a serial console. This function must be deactivated to use this port to control the PiKoder/SPE with a terminal software. The process of turning of the serial console is described in various blogs and may vary slightly depending on your Raspberry version and your Linux distribution.



Wiring scheme for connecting your PiKoder/SPE to a Raspberry Pi

Wiring

The wiring is shown in the schematic at the top of this page. Please turn of all components to avoid damage by unintended shorts.

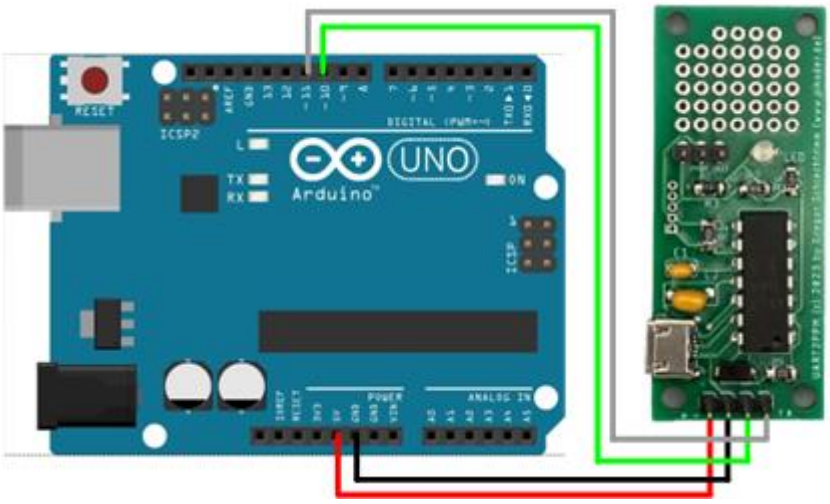
Installing terminal software such as Minicom

You will also find more information about the installation of Minicom in various Raspberry Pi related blogs. When starting the software please make sure to set the baud rate of the UART to the baud rate setting required by the PiKoder. For your convenience it is recommended that you activate the local echo (command "E"). After setting these parameters you can control your PiKoder/SPE using the ASCII commands listed in section 6 of this User Manual.

8

Connect the SPE to an Arduino

The schematic for connecting a PiKoder/SPE to an Arduino is presented below. In this application the PiKoder would be powered with 5 Volt from the Arduino.



ASCII-based interface sketch

The following sketch would allow you to test the PiKoder/SPE by entering commands (please refer to section 6 for a full interface description). The Sketch Interface Test (.ino-Datei) can be downloaded here. This sketch is Open Source and is released under a GNU General Public License Version 3.