

**Description of AVR Incircuit  
Serial Programming  
Adapter for DASA3 Protocol  
Version 1.0**

# **Description of AVR Incircuit Serial Programming Adapter for DASA3 Protocol: Version 1.0**

Copyright © 2016-2017 Jost Brachert, [jost.brachert@gmx.de](mailto:jost.brachert@gmx.de)

All rights reserved. This document is licensed under a *Creative Commons Attribution-NonCommercial-ShareAlike 4.0 Unported* License <http://creativecommons.org/licenses/by-nc-sa/4.0/>

LIABILITY DISCLAIMER .....	1
INTRODUCTION .....	2
Precaution .....	2
Features .....	2
DESCRIPTION OF OPERATION .....	3
Microprocessor .....	3
Host Computer (PC) .....	3
SW To Be Programmed .....	3
Flash Program .....	3
Connections .....	4
Supply Voltage .....	5
Examples .....	5
avrdude .....	5
uisp .....	5
Tips'n Tricks .....	6
Determining The Port .....	6
Verification .....	6
Delay .....	6
PRECAUTIONARY MEASURES .....	7
APPENDIX .....	8
Circuit Diagram .....	8
Connector Pinouts .....	8
Technical Specifications .....	10
Glossary .....	10
References .....	10
Index .....	12

# LIABILITY DISCLAIMER

This is **not** a commercial or industrial device. It has been created for experimental purpose only and is not intended for professional or commercial use.

The maker of this device is not liable for any damages caused by the device or the failure of it, including any lost profits or savings or special, incidental, or consequential damages. The maker is not liable for any claim made by a third party or made by you for a third party.

This limitation of liability applies whether damages are sought, or claims are made, under this Limited Warranty or as a tort claim (including negligence and strict product liability), a contract claim, or any other claim, and cannot be rescinded or changed by anyone. This limitation of liability will be effective even if you have advised the maker of the possibility of any such damages.

# INTRODUCTION

## Precaution

**Before taking the device into operation read the section on *PRECAUTIONARY MEASURES*.**

## Features

The programming adapter is a hardware adapter to flash ATMEL AVR microprocessors ([RfAvrProg] “AVR Programmers”). The microprocessors are programmed according to [RfAVR910] “AVR Application Note AVR910: In-System Programming” and [RfAVR109] “AVR Application Note AVR109: Self Programming”.

The adapter is a simple hard-wired programming adapter to be directly connected to a standard RS-232 serial port. Its circuitry and pinout is compliant to the DASA-3 specification as described in [RfAvrProg] “AVR Programmers”. The microprocessor is programmed in serial programming mode.

Today's computers usually do not have RS-232 interfaces any more. However, it is possible to use a conventional USB RS-232 adapter to connect the adapter to a host computer.

The programming adapter should be usable with all flashing programs which can deal with so called “bit-banging” programmers. Despite the fact that it is connected to a serial port it cannot be approached with a serial protocol. Instead the control lines of the serial interface are used to form the serial protocol to program the microprocessor.

No particular driver is needed for the adapter. The flashing program is able to drive the adapter directly (this might not be true for MS Windows which usually needs at least a driver for the RS-232 adapter).

The programming adapter takes its power from the device to be programmed. However, it is possible to connect it to an external power supply which then provides power for the adapter as well as for the device to be programmed.

# DESCRIPTION OF OPERATION

To use the programming adapter you need

1. a device containing a microprocessor to be programmed,
2. a host computer (PC),
3. a SW to be programmed to the microprocessor device, e.g. an Intel HEX file or raw binary file,
4. a flash program to control the programming procedure – that program sends the SW to be programmed via the programming adapter to the microprocessor,
5. the programming adapter described here to adapt the signals from the host computer to the microprocessor serial programming interface,
6. an ISP cable to connect the microprocessor with the programming adapter, and
7. if the host computer does not have an RS-232 interface also a USB RS-232 adapter.

## Microprocessor

The programming adapter can flash all ATMEL AVR microprocessors ([RfAvrProg] “AVR Programmers”) according to [RfAVR910] “AVR Application Note AVR910: In-System Programming” and [RfAVR109] “AVR Application Note AVR109: Self Programming”.

## Host Computer (PC)

Any PC or laptop running a recent Linux as operating system should be usable. A PC running MS Windows could also be ok, but it is probably more difficult to get the required programs and drivers be installed and running.

## SW To Be Programmed

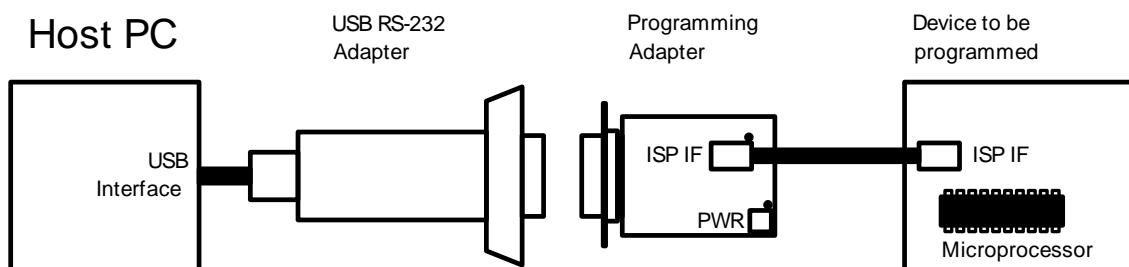
This is the program that should run on the microprocessor after successful programming. Its format – usually an Intel HEX file or raw binary file – depends on the formats that the flash program can handle, see the documentation of the flash program.

## Flash Program

The flash program is the program that controls the programming procedure. It sends the SW to be programmed via the programming adapter to the microprocessor. The flash program must be able to use DASA-3 programming adapters ([RfAvrProg] “AVR Programmers”). Examples for suitable flash programs are

- avrdude ([RfAvrdude]), and
- uisp ([RfUISP] “uisp: Micro In-System Programmer”).

## Connections



### Connections

Since recent PCs are no more equipped with RS-232 interfaces you need a USB RS-232 adapter. Any such adapter compliant to the USB communications device class (USB CDC, [RfUsbCdc] “USB communications device class”) – practically all commercially available ones – will usually do. As these adapters normally have male DE-9 connectors the programming adapter can be directly plugged into the USB RS-232 adapter. If for any reason a cable is required between RS-232 adapter and programming adapter be sure that you use a standard RS-232 cable and not a null modem cable. A null modem cable won't work.

Programming adapter and the device to be programmed must be connected with an ISP cable. The ISP connectors on the programming adapter and the device are 2 by 3 pin male header connectors. The according cable is usually a ribbon cable. The connection is one to one, pin 1 at the connector on the programming adapter is connected to pin 1 at the connector on the device, pin 2 is connected to pin 2 and so on.

Be sure that the  $V_{CC}$  pins (pin 2) are definitely connected, in particular if the supply voltage of the microprocessor to be programmed is less than 5 V, because these pins carry the reference voltage to limit the voltage on the signal pins to the appropriate value. If this pin is not connected thoroughly the microprocessor might be damaged, see section *PRECAUTIONARY MEASURES*.

In most cases it is not necessary to connect the programming adapter's power connector (to an external power supply) as it is normally powered by the device to be programmed via the ISP connector. If the device needs to be programmed while its own power supply is turned off you can connect the power connector of the programming adapter to an external power supply to power the adapter itself as well as the microprocessor to be programmed (via the ISP cable). **Please read section *PRECAUTIONARY MEASURES* carefully before doing that.** If an according power cable is not available see appendix, Table 3, “Power connector pinout” on how one could be created.

**Be sure that the voltage of the external power supply fits to the device to be programmed. If it is too high then the device will be damaged - if it is too low it will probably not be programmed correctly. Be also sure that the external power supply is connected with correct polarity (see Table 3, “Power connector pinout”). If it is connected with wrong polarity the device to be programmed and the programming adapter will both be damaged.**

## DESCRIPTION OF OPERATION

# Supply Voltage

The programming adapter can basically be used with all supply voltages specified for the ATMEL AVR microprocessors, i.e. 1.8 V to 5.5 V.

# Examples

## avrdude

Flashing the program `blink.hex` in subdirectory `build` into the flash ROM of an ATmega328p microprocessor with `avrdude` ([RfAvrdude]) via the programming adapter connected at port `/dev/ttyUSB0` could be done with the shell command:

```
avrdude -V -P/dev/ttyUSB0 -patmega328p -cdasa3 -i500 -Uflash:w:build/blink.hex:i
```

The options used here are

-V	no verification
-P	serial port
-p	part to be pogrammed
-c	programmer
-i	delay in microseconds (approximately)
-U	perform memory operation
flash:	access the flash ROM segment
w:	write
:i	input file is Intel-Hex

For a detailed description of these options and the lot of all others see the documentation of `avrdude`:

```
man 1 avrdude
```

## uisp

Reading the fuses from an ATtiny2313 microprocessor with `uisp` (“`uisp`: Micro In-System Programmer” [RfUISP]) via the programming adapter connected at port `/dev/ttyUSB0` could be done with the shell command:

```
uisp -dserial=/dev/ttyUSB0 -dprog=dasa3 -dpart=attiny2313 --rd_fuses
```

The options used here are

-dserial	serial port
-dpart	part to be pogrammed
-dprog	programmer
--rd_fuses	read the fuses and display result on stdout

Also for `uisp` exists a man page with the description of all its options:



## DESCRIPTION OF OPERATION

`man 1 uisp`

More information can be found in the “AVR Microcontrollers in Linux HOWTO” [RfAvrHowto].

## Tips'n Tricks

### Determining The Port

The flash program needs to know the serial port to which the programming adapter is connected. If you are not sure about the port you can get its name (on Linux) by issuing this command in a command shell

```
ls -dlrt /dev/tty*
```

immediately after the USB RS-232 adapter has been connected.

This command lists all serial devices in the order of their creation, the last created device as the last one in the list. If the command is issued immediately after the adapter has been connected then the RS-232 interface device is listed as the last one. The output of the command could look like:

```
...
crw--w---- 1 root tty      4, 60 May 15 08:12 /dev/tty60
crw-rw---- 1 root dialout  4, 68 May 15 08:13 /dev/ttyS4
crw--w---- 1 root tty      4, 10 May 15 09:15 /dev/tty10
crw-rw---- 1 root dialout 166,  0 May 15 14:52 /dev/ttyUSB0
```

In this example the according RS-232 device name is `/dev/ttyUSB0`. It has been created as the last one at 14:52 o'clock.

### Verification

Sometimes, when used without option `-v` – that means with verification – avrdude doesn't program correctly. So it seems to be better to use avrdude without verification (i.e. with option `-v`).

### Delay

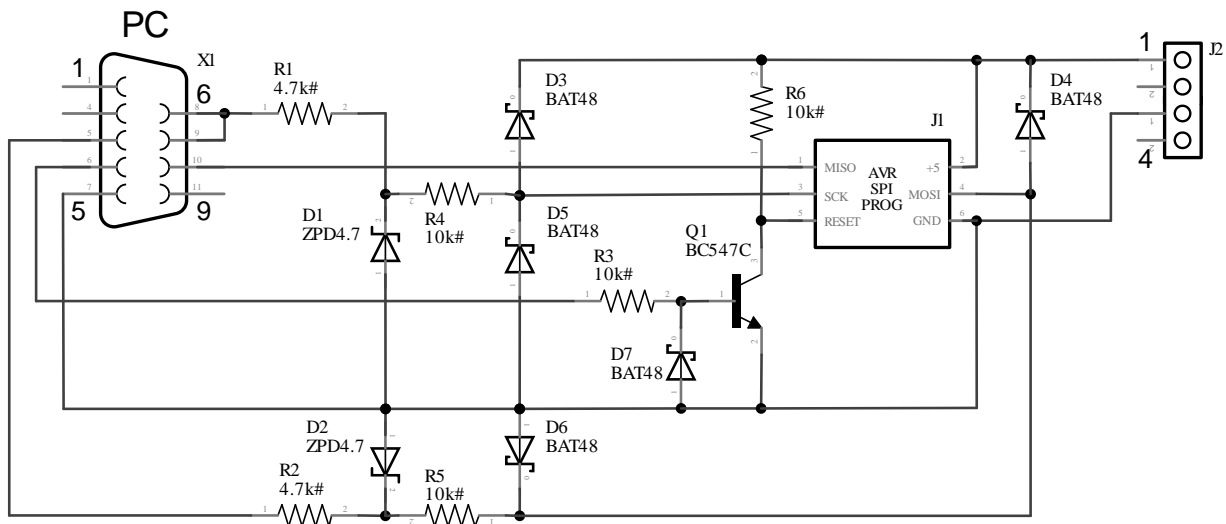
The programming adapter described here should be used with stretched signals. At avrdude that can be achieved with option `-i`. The best value should be determine with trial and error. The downside, however, is that programming time is also stretched.

# PRECAUTIONARY MEASURES

- 1. For the device to be programmed, observe the general ATMEL recommendation: No activities on SPI interface of the device to be programmed, otherwise the programming will be disturbed or interrupted.**
- 2. Don't connect an external power supply to the programming adapter if the device to be programmed is itself connected to a power supply (and turned on). Both power supplies will interfere with each other and might be damaged.**
- 3. Before connecting an external power supply to the programming adapter make sure that the connector has correct polarity, see Table 3, “ Power connector pinout ”. The microprocessor to be programmed will probably be damaged if that is not the case.**
- 4. If the programming adapter is powered by an external power supply, make sure that its voltage fits to the device to be programmed. The device to be programmed will probably be damaged if that is not the case.**
- 5. If the programming adapter is powered by an external power supply, make sure that the device to be programmed can be powered by the ISP connector. Sometimes the internal power supply of the device cannot deal with power being fed backwards.**
- 6. The  $V_{CC}$  pin in the ISP connector must always be connected to  $V_{CC}$  of the microprocessor to be programmed, even if the device to be programmed has its own power supply because the voltage on that pin is required as reference for the maximum signal voltage level.**

# APPENDIX

## Circuit Diagram



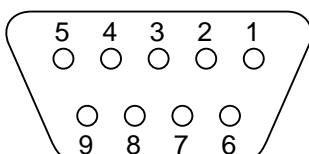
Circuit diagram of the programming adapter

## Connector Pinouts

**Table 1. PC RS-232 connector pinout**

1	DCD	Data Carrier Detect (not used)
2	Rx	Receive Data (not used)
3	Tx (output on PC side)	Transmit Data
4	DTR (output on PC side)	Data Terminal Ready
5	GND	Ground
6	DSR	Data Set Ready
7	RTS/RTR (output on PC side)	Request To Send
8	CTS	Clear To Send
9	RI	Ring Indicator (not used)

The connector is a female 9-pin DE-9 connector (Data Circuit-terminating Equipment - DCE) that can be plugged directly into the PC male socket. The pin numbering (viewed from connector side) is:

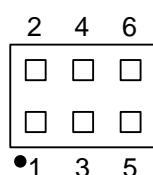


## APPENDIX

**Table 2. ISP connector pinout**

1	MISO	Master In - Slave Out
2	V <sub>CC</sub>	Target Power. If the adapter is connected to an external power supply the device to be programmed is supplied via this pin. Otherwise, if the adapter is not connected to an external power supply the adapter is supplied from the device to be programmed via this pin.
3	SCK	Serial Clock
4	MOSI	Master Out - Slave In
5	RESET	Target AVR MCU Reset
6	GND	Common Ground

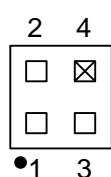
The ISP connector is a 2 by 3 male header connector. The pin numbering (viewed from connector side) is:



**Table 3. Power connector pinout**

1	V <sub>CC</sub>	Supply voltage. If the adapter is connected to an external power supply the device to be programmed is supplied via this pin. Otherwise, if the adapter is not connected to an external power supply the adapter is supplied from the device to be programmed via this pin.
2	n.c.	Not connected
3	GND	Ground
4	-	-

The power connector is a quadratic male header connector. Pin 4 is nonexistent to make a protection against reverse connection possible. The pin numbering (viewed from connector side) is:



# Technical Specifications

Current consumption:	Max. 1.5 mA, either from ISP $V_{CC}$ pin or Power Connector $V_{CC}$ pin at $V_{CC}=5V$
	Max. 1.2 mA at RS-232 DTR pin and max. 2.6 mA at RS-232 TX and RTS pins at pin voltage 12 V
$V_{CC}$ :	In principle between 1.8 V and 5.5 V either at ISP $V_{CC}$ pin or Power Connector $V_{CC}$ pin. The concrete $V_{CC}$ value to be used depends on the $V_{CC}$ of microprocessor to be programmed.
RS-232 interface pinout conforming to:	DASA-3, see [RfAvrProg] "AVR Programmers"

## Glossary

ICSP      In-circuit serial programming

ISP        In-circuit serial programming

SPI        Serial Peripheral Interface

## References

[RfPonyProg] "PonyProg Serial Device Programmer"

<http://www.lancos.com/prog.html> (retrieved 2016 Apr 5<sup>th</sup>)

[RfAVR910] "AVR Application Note AVR910: In-System Programming"

<http://www.atmel.com/Images/doc0943.pdf> (retrieved 2016 Apr 5<sup>th</sup>)

[RfAVR109] "AVR Application Note AVR109: Self Programming"

<http://www.atmel.com/Images/doc1644.pdf> (retrieved 2016 Apr 5<sup>th</sup>)

[RfAvrProg] "AVR Programmers"

<https://www.avrprogrammers.com/programmers/all-serial-port> (retrieved 2016 Apr 5<sup>th</sup>)

[RfAvrdude] "avrdude"

<http://www.nongnu.org/avrdude/> (retrieved 2016 Apr 5<sup>th</sup>)

[RfUIISP] "uisp: Micro In-System Programmer"

<http://savannah.nongnu.org/download/uisp/> (retrieved 2016 Apr 5<sup>th</sup>)

[RfAvrHowto] "AVR Microcontrollers in Linux HOWTO"

<http://www.tldp.org/HOWTO/Avr-Microcontrollers-in-Linux-Howto/x207.html> (retrieved 2016 Apr 5<sup>th</sup>)

See in particular section 2: <http://www.tldp.org/HOWTO/Avr-Microcontrollers-in-Linux-Howto/x207.html>

[RfAtmProd] "Atmel Products"

## APPENDIX

<http://www.atmel.com/products/AVR/> (retrieved 2016 Apr 5<sup>th</sup>)

[RfUsbCdc] “USB communications device class”

[https://en.wikipedia.org/wiki/USB\\_communications\\_device\\_class](https://en.wikipedia.org/wiki/USB_communications_device_class) (retrieved 2016 Apr 5<sup>th</sup>)

# Index

## A

AVR Application Note 109, [3](#)  
AVR Application Note 910, [3](#)  
avrdude, [3](#), [5](#)

## B

Binary File, [3](#), [3](#)

## C

Commercial Use, [1](#)  
Connections, [4](#)

## D

DASA-3, [2](#)  
DE-9 Connector, [4](#), [8](#)  
Delay, [5](#), [6](#)  
Driver, [2](#)

## E

Examples, [5](#)

## F

Flash Program, [3](#), [3](#)

## H

HEX File, [3](#), [3](#)  
Host Computer, [3](#)

## I

ICSP, [10](#)  
IdxRS232Interface, [3](#)  
Intel HEX File, [3](#), [3](#)  
ISP, [10](#)  
ISP Cable, [4](#), [4](#)  
ISP Connector, [4](#), [4](#), [7](#), [7](#), [9](#)

## L

License, [ii](#)  
Linux, [3](#), [6](#)

## M

Microprocessor, [3](#)

## N

Null Modem Cable, [4](#)

## P

Port, [2](#), [5](#), [5](#), [6](#)  
Power Connector, [4](#), [9](#)  
Power Supply, [2](#), [4](#), [7](#), [7](#), [7](#), [7](#), [7](#), [9](#), [9](#)  
Professional Use, [1](#)

## R

RS-232, [4](#)

## S

Serial Port, [2](#), [5](#), [5](#), [6](#)  
Serial Protocol, [2](#)  
SPI, [7](#), [10](#)  
Supply Voltage, [5](#)

## U

uisp, [3](#), [5](#)  
USB CDC, [4](#)  
USB RS-232 Adapter, [4](#), [6](#)

## V

VCC, [4](#), [7](#), [9](#), [9](#), [10](#)  
Verification, [5](#), [6](#)  
Voltage, [5](#)

## W

Windows, [2](#), [3](#)