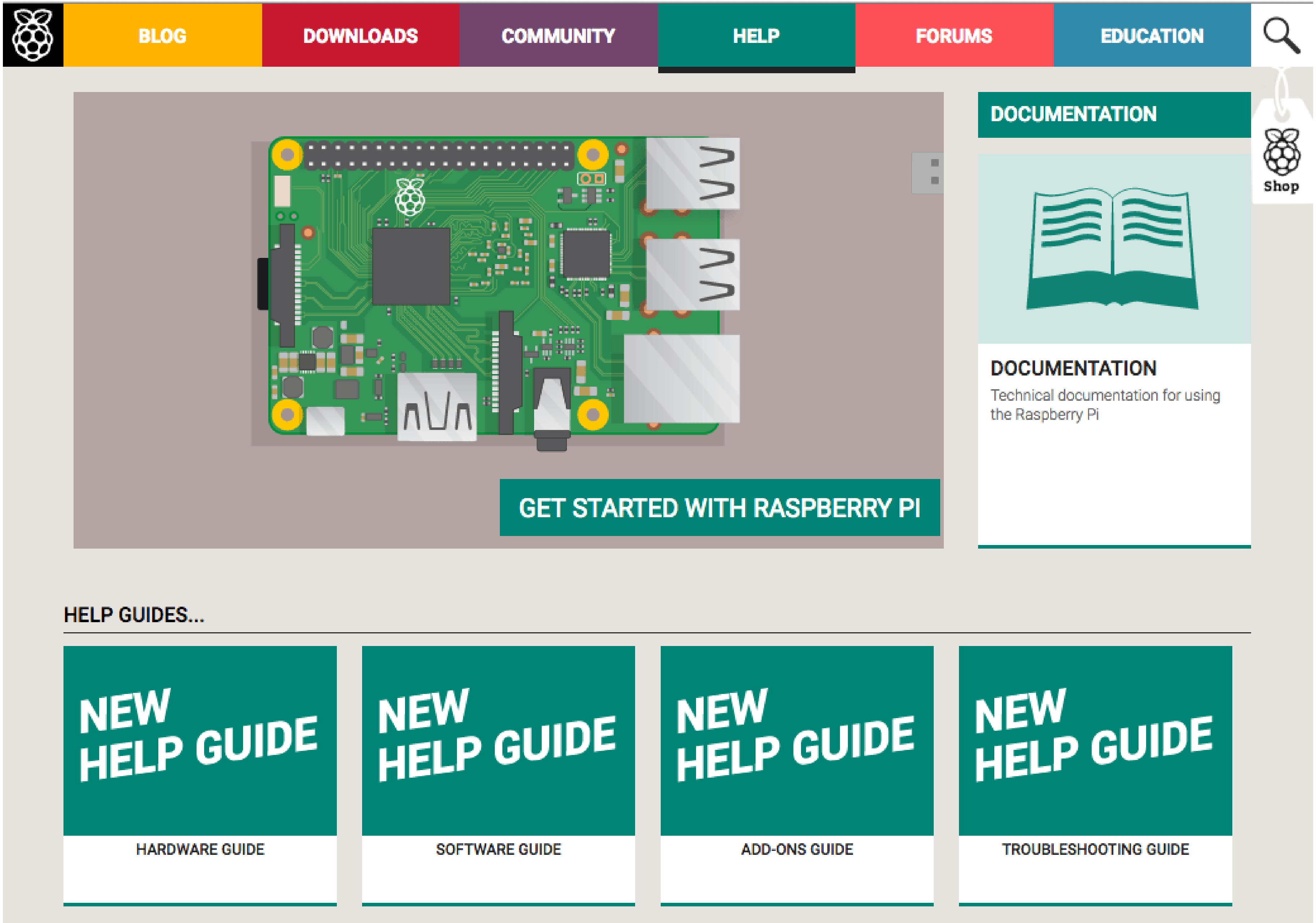
http://raspberrypi.org



DOCUMENTATION

DOCUMENTATION

Technical documentation for using the Raspberry Pi

HELP GUIDES...

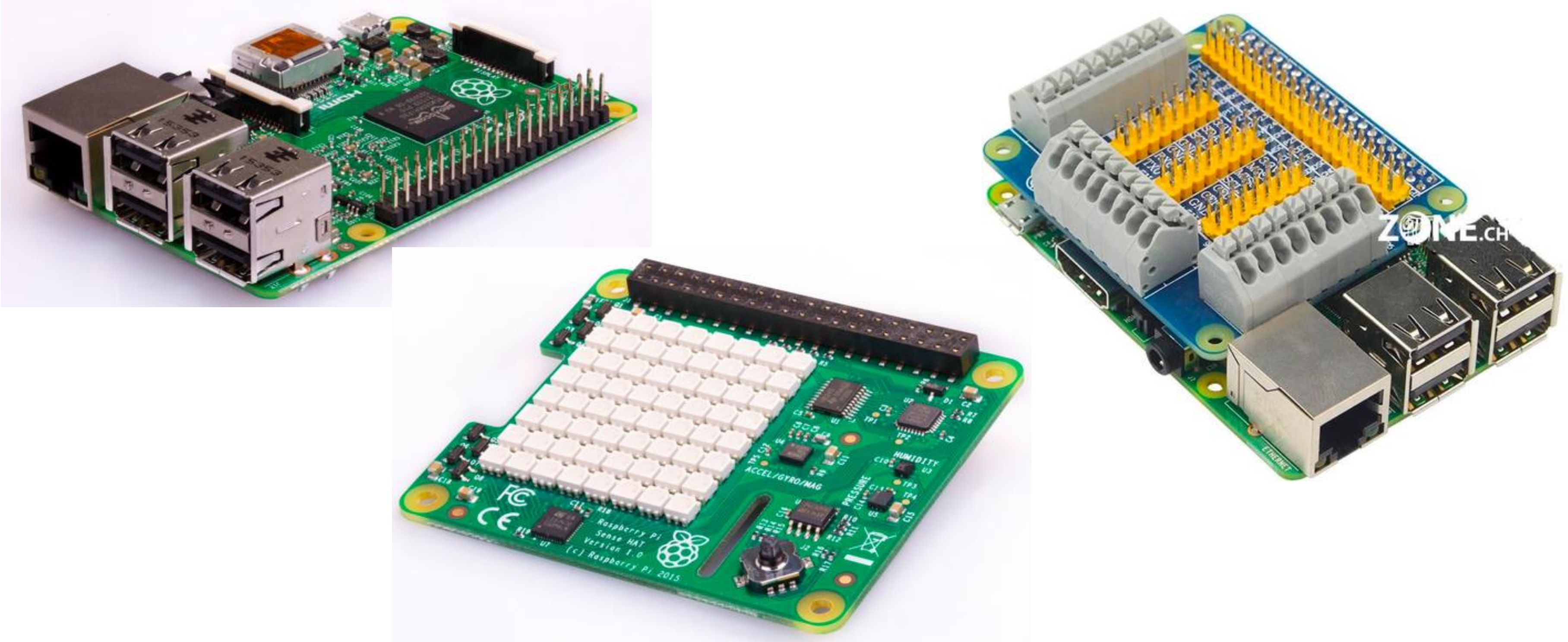
HARDWARE GUIDE

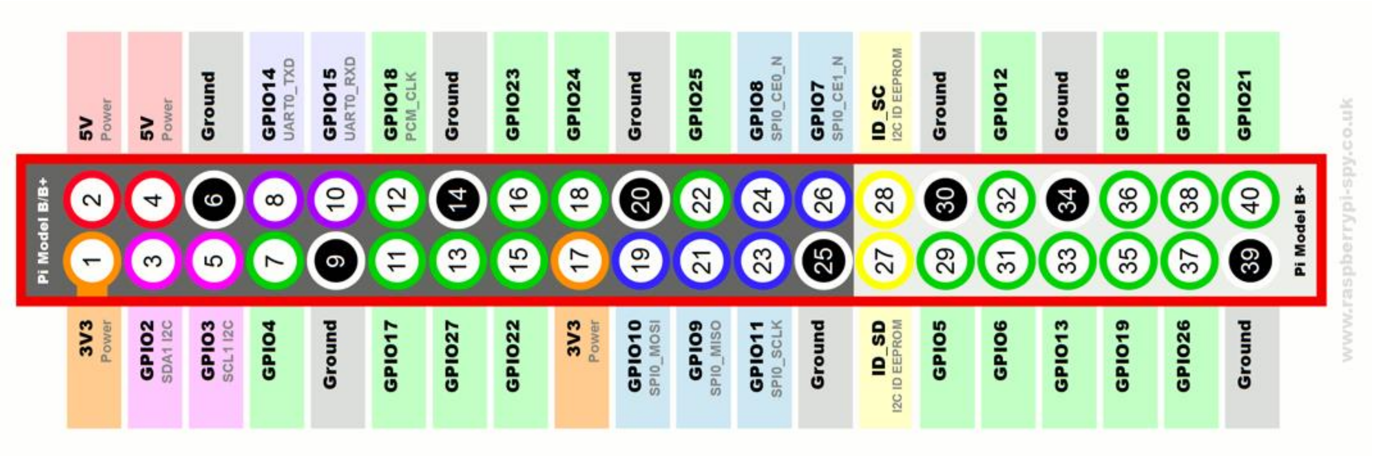
SOFTWARE GUIDE

ADD-ONS GUIDE

TROUBLESHOOTING GUIDE

http://elinux.org/RPi_Expansion_Boards





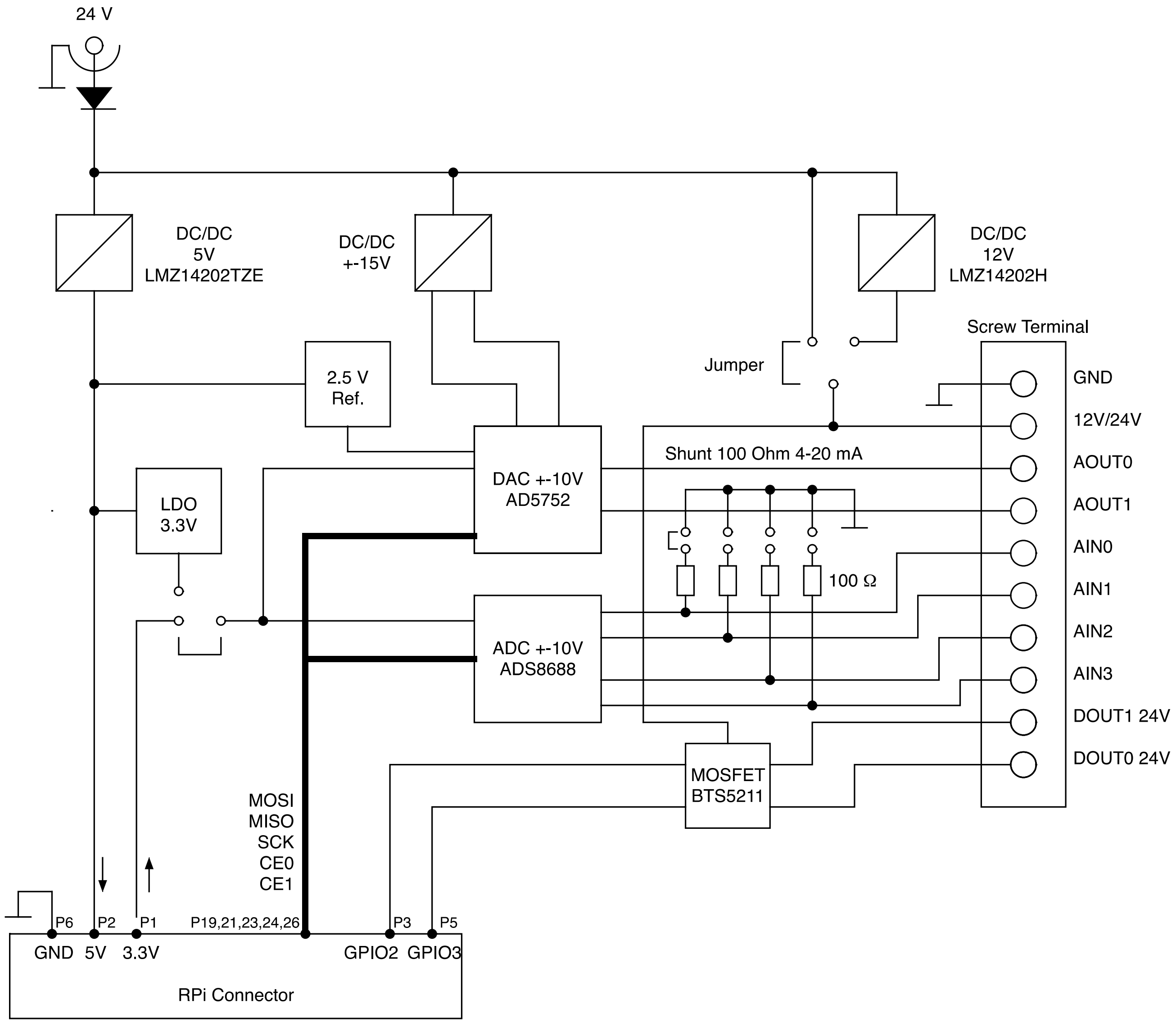
Pi Model B/B+

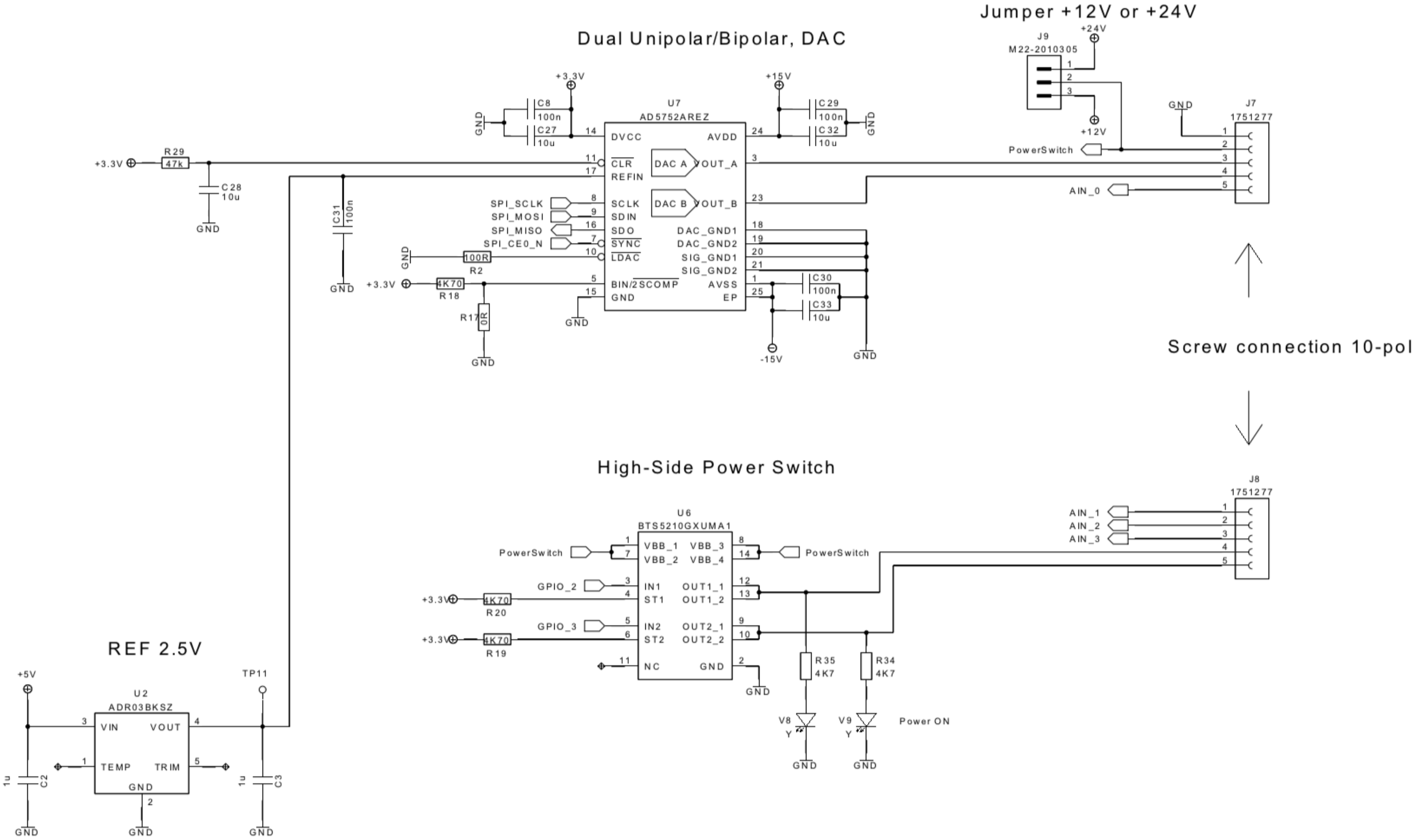
1	3V3 Power	2	5V Power
3	GPIO2 SDA1 I2C	4	5V Power
5	GPIO3 SCL1 I2C	6	Ground
7	GPIO4	8	GPIO14 UART0_TXD
9	Ground	10	GPIO15 UART0_RXD
11	GPIO17	12	GPIO18 PCM_CLK
13	GPIO27	14	Ground
15	GPIO22	16	GPIO23
17	3V3 Power	18	GPIO24
19	GPIO10 SPI0_MOSI	20	Ground
21	GPIO9 SPI0_MISO	22	GPIO25
23	GPIO11 SPI0_SCLK	24	GPIO8 SPI0_CE0_N
25	Ground	26	GPIO7 SPI0_CE1_N
27	ID_SD I2C ID EEPROM	28	ID_SC I2C ID EEPROM
29	GPIO5	30	Ground
31	GPIO6	32	GPIO12
33	GPIO13	34	Ground
35	GPIO19	36	GPIO16
37	GPIO26	38	GPIO20
39	Ground	40	GPIO21

Pi Model B+

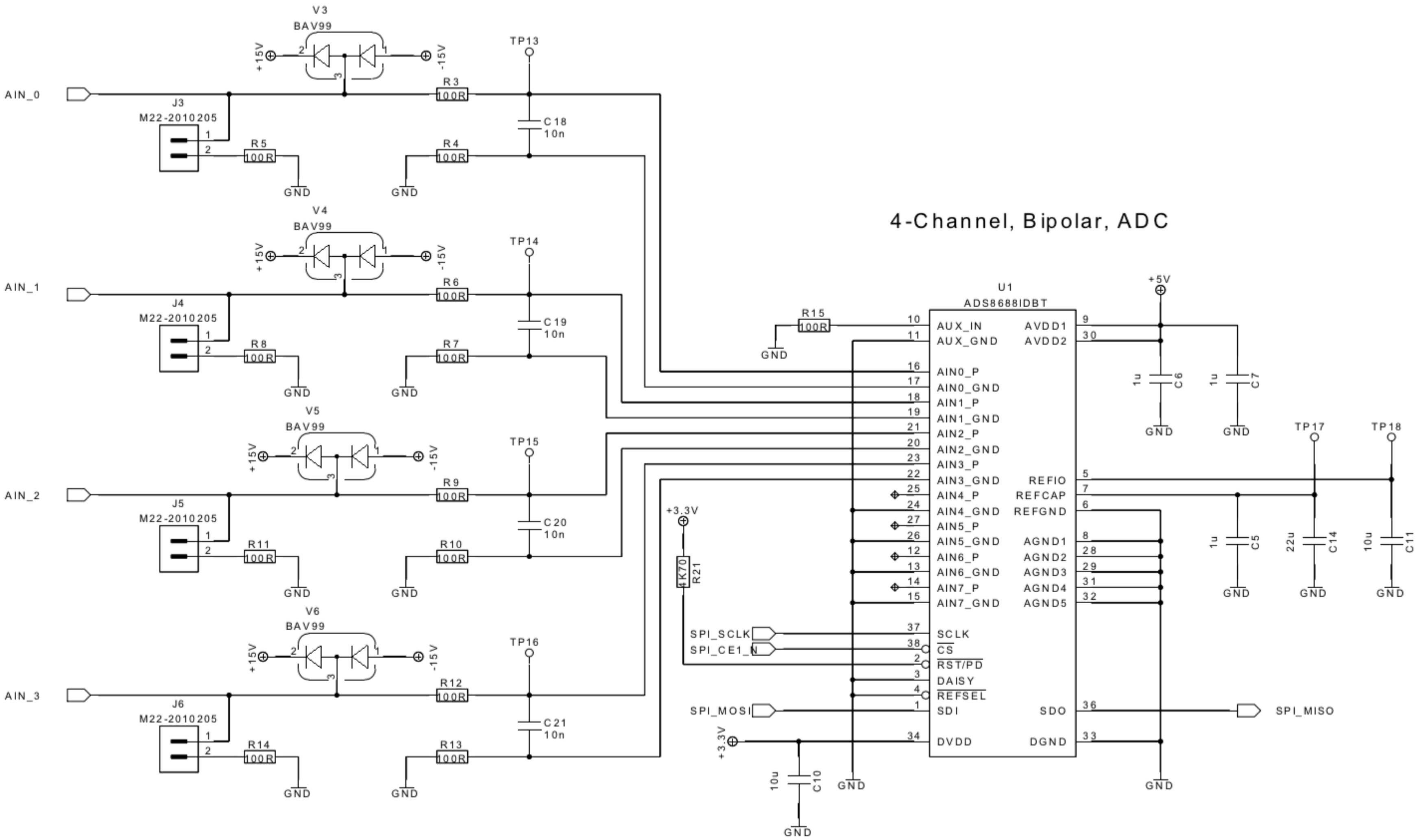
www.raspberrypi-spy.co.uk

PSI-made ADC/DAC IO board

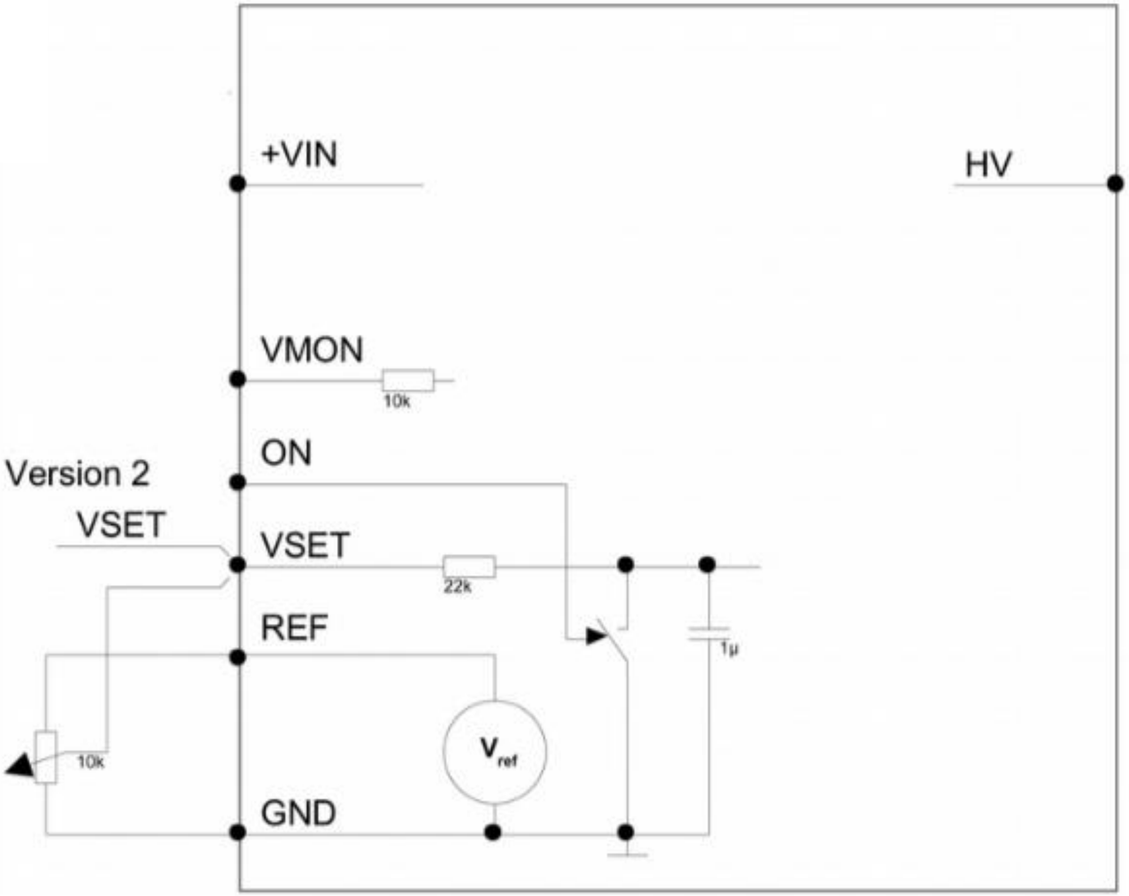



Jumper +12V or +24V
Dual Unipolar/Bipolar, DAC
J9
M22-2010305
+24V
+12V
1
2
3
J7
1751277
GND
PowerSwitch
AIN_0
+3.3V
C8
100n
C27
10u
GND
U7
AD5752AREZ
14 DVCC
24 AVDD
+15V
C29
100n
C32
10u
R29
47k
C28
10u
11 CLR
17 REFIN
DAC A
VOUT_A 3
DAC B
VOUT_B 23
C31
100n
8 SCLK
9 SDIN
16 SDO
7 SYNC
10 LDAC
5 BIN/2SCOMP
15 GND
SPI_SCLK
SPI_MOSI
SPI_MISO
SPI_CE0_N
100R
R2
4K70
R18
R17
0R
DAC_GND1 18
DAC_GND2 19
SIG_GND1 20
SIG_GND2 21
AVSS 1
EP 25
C30
100n
C33
10u
-15V
Screw connection 10-pol
High-Side Power Switch
J8
1751277
AIN_1
AIN_2
AIN_3
U6
BTS5210GXUMA1
1 VBB_1
7 VBB_2
VBB_3 8
VBB_4 14
3 IN1
4 ST1
OUT1_1 12
OUT1_2 13
5 IN2
6 ST2
OUT2_1 9
OUT2_2 10
11 NC
GND 2
PowerSwitch
GPIO_2
GPIO_3
+3.3V
4K70
R20
4K70
R19
R35
4K7
R34
4K7
V8
Y
V9
Y
Power ON
REF 2.5V
+5V
U2
ADR03BKSZ
3 VIN
VOUT 4
1 TEMP
TRIM 5
GND
2
TP11
1u
C2
1u
C3
GND

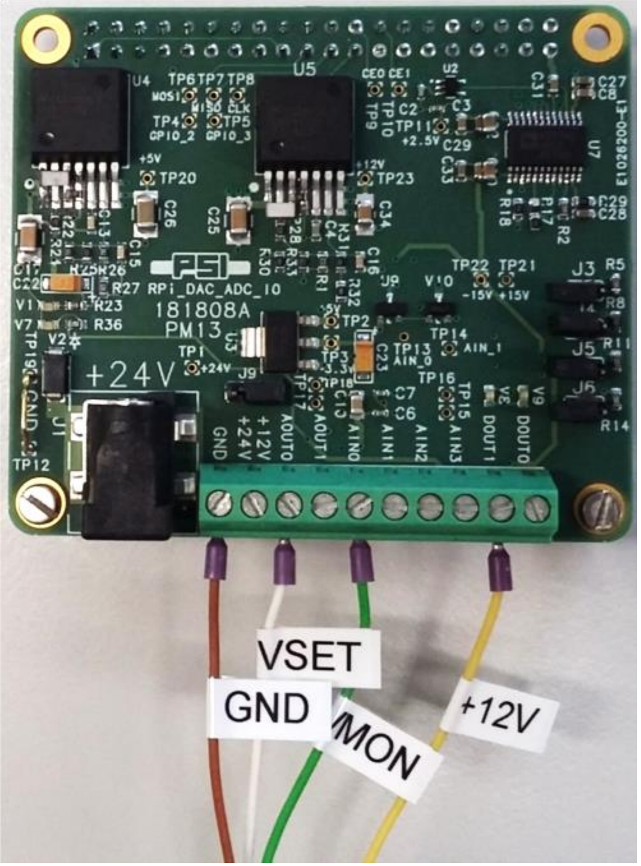
4-Channel, Bipolar, ADC



ISED BPS series DC-DC Converter

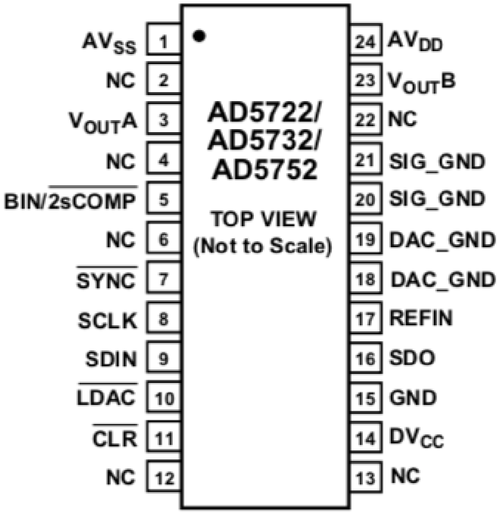


HV Supply 0…-2000V / 1.5 mA

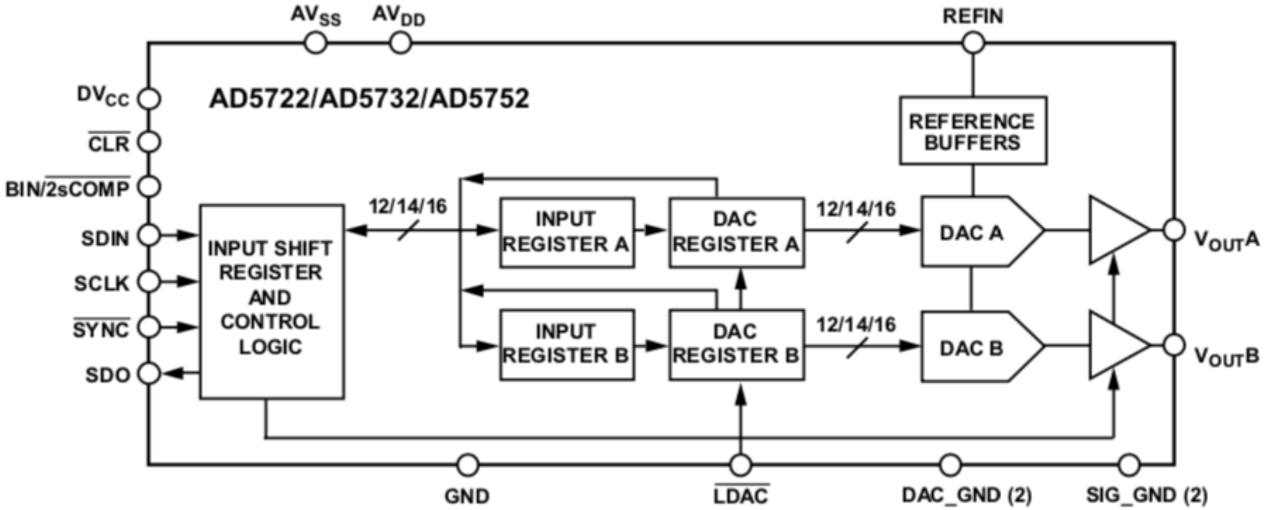


DAC programming via SPI

(https://en.wikipedia.org/wiki/Serial_Peripheral_Interface)



FUNCTIONAL BLOCK DIAGRAM



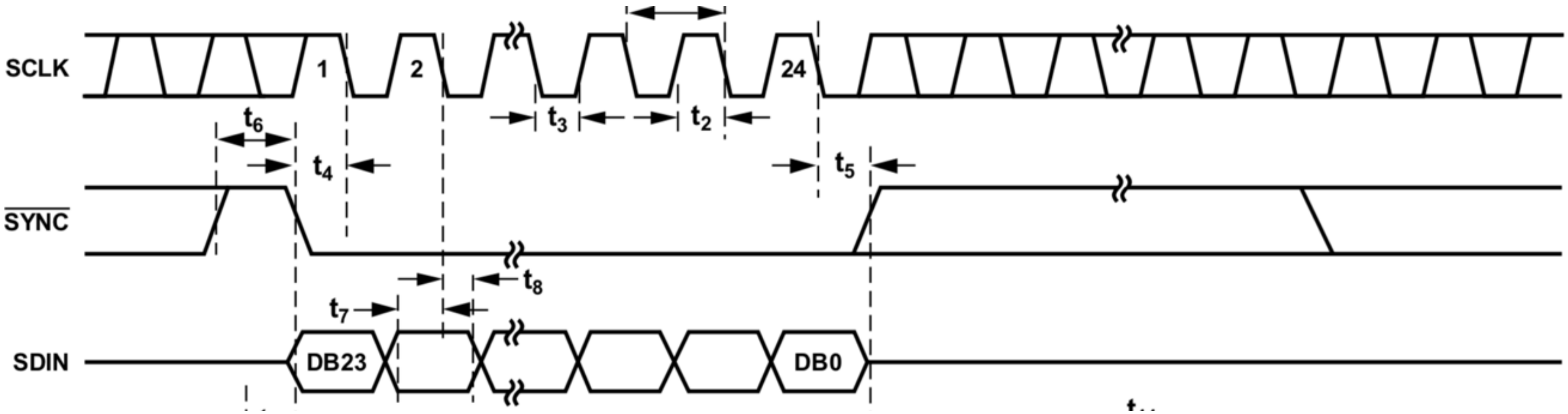
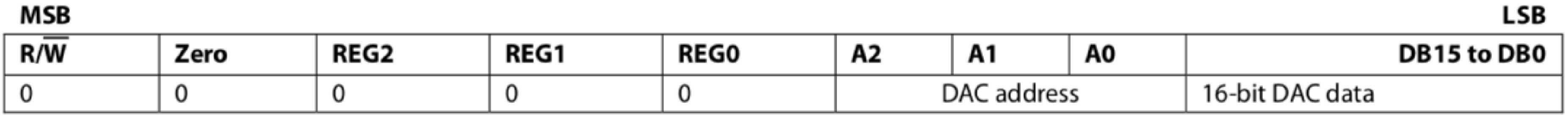


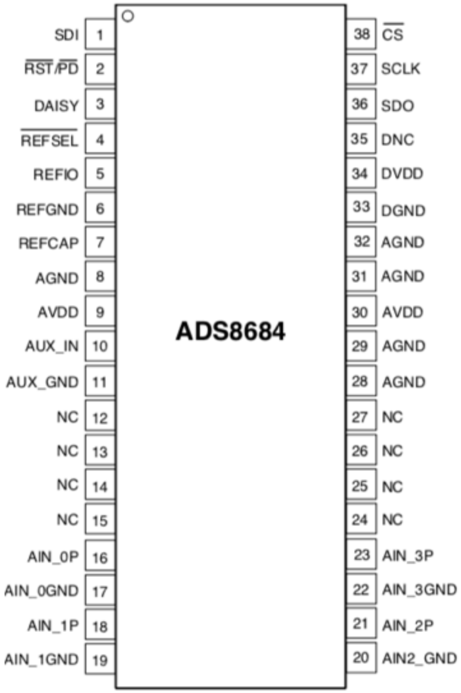
Table 18. Programming the AD5752 DAC Register

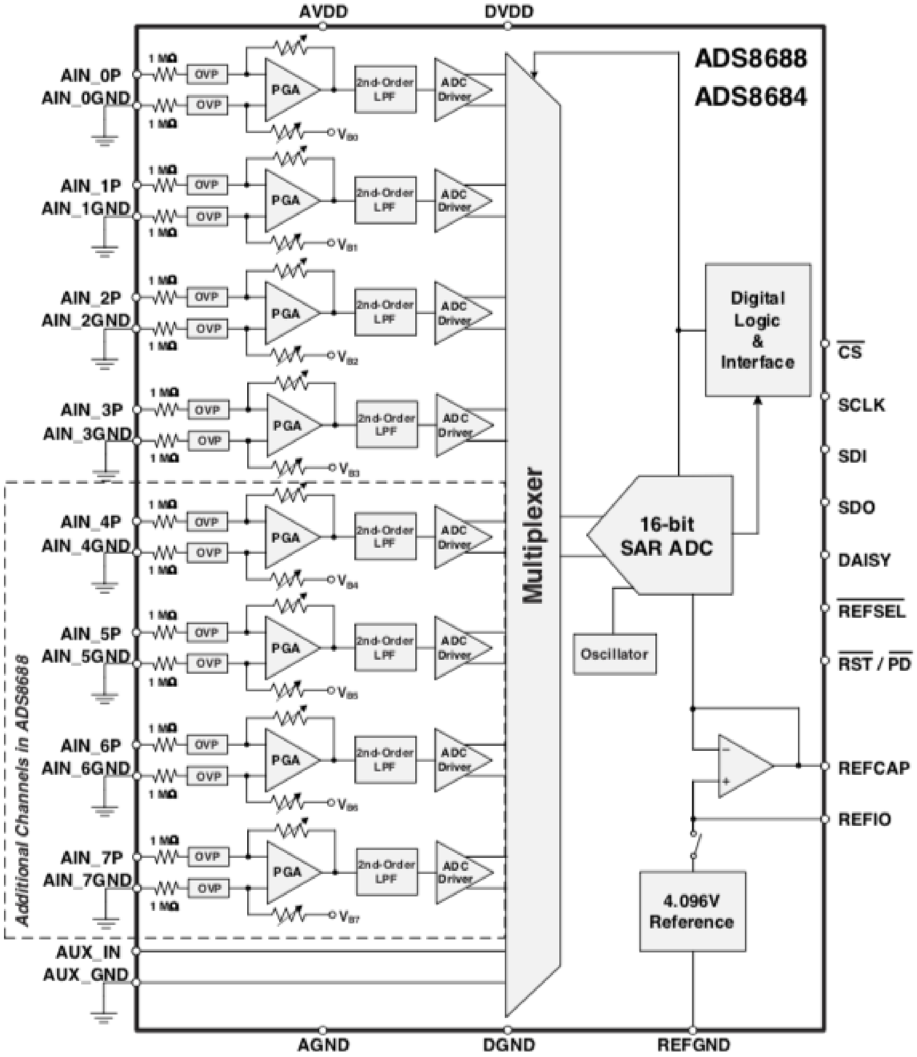


MSB LSB

R/W	Zero	REG2	REG1	REG0	A2	A1	A0	DB15 to DB0
0	0	0	0	0	DAC address			16-bit DAC data

ADC programming via SPI





ADC programming via SPI

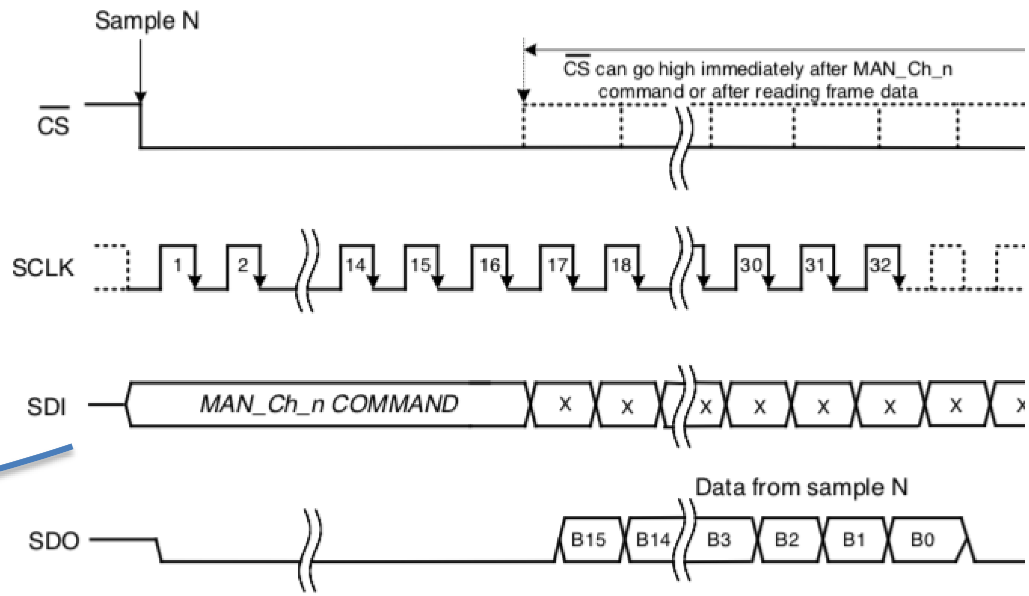
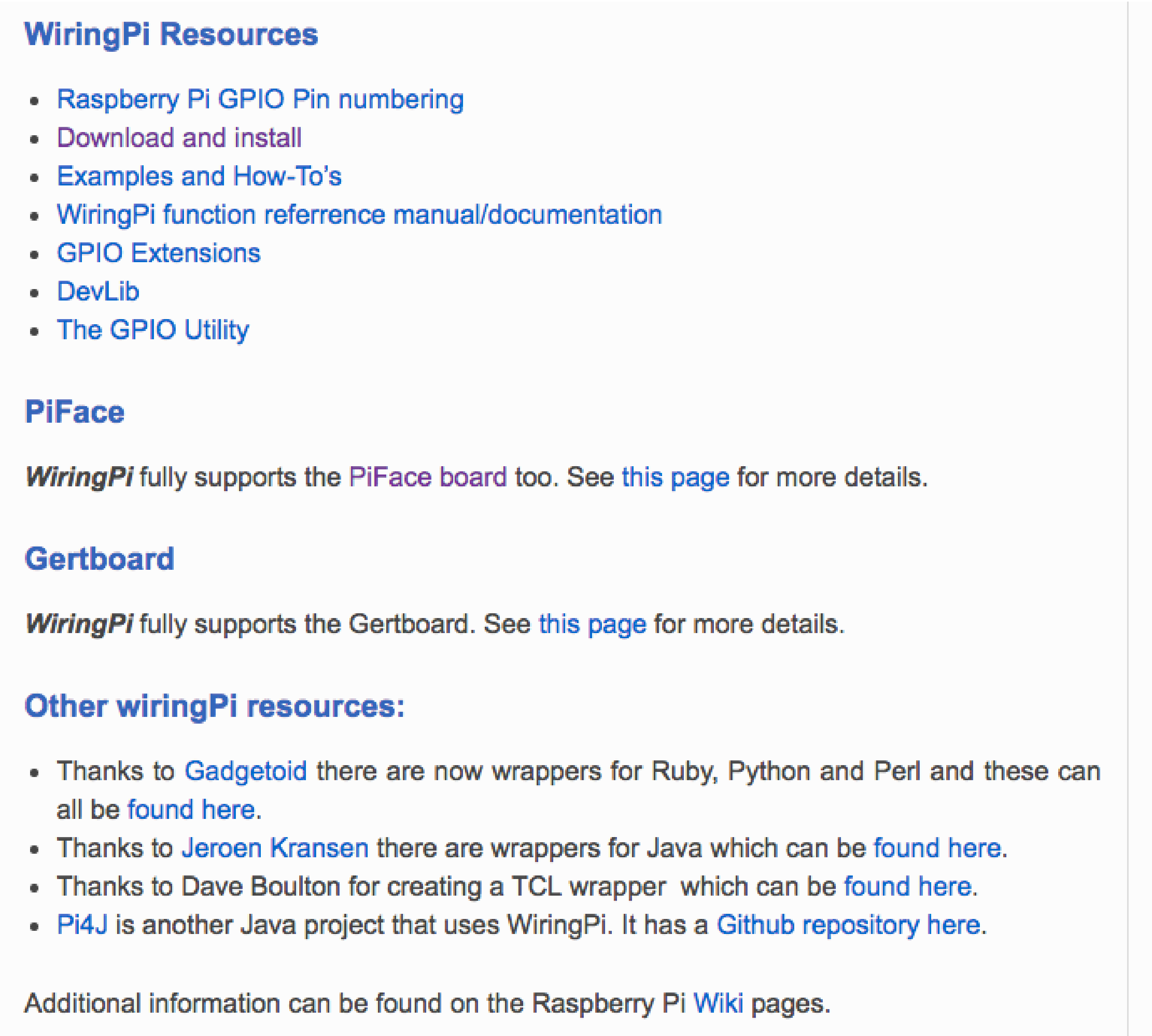


Table 6. Command Register Map

REGISTER	MSB BYTE								LSB BYTE	COMMAND (Hex)	OPERATION IN NEXT FRAME
	B15	B14	B13	B12	B11	B10	B9	B8	B[7:0]		
Continued Operation (NO_OP)	0	0	0	0	0	0	0	0	0000 0000	0000h	Continue operation in previous mode
Standby (STDBY)	1	0	0	0	0	0	1	0	0000 0000	8200h	Device is placed into standby mode
Power Down (PWR_DN)	1	0	0	0	0	0	1	1	0000 0000	8300h	Device is powered down
Reset program registers (RST)	1	0	0	0	0	1	0	1	0000 0000	8500h	Program register is reset to default
Auto Ch. Sequence with Reset (AUTO_RST)	1	0	1	0	0	0	0	0	0000 0000	A000h	Auto mode enabled following a reset
Manual Ch 0 Selection (MAN_Ch_0)	1	1	0	0	0	0	0	0	0000 0000	C000h	Channel 0 input is selected
Manual Ch 1 Selection (MAN_Ch_1)	1	1	0	0	0	1	0	0	0000 0000	C400h	Channel 1 input is selected
Manual Ch 2 Selection (MAN_Ch_2)	1	1	0	0	1	0	0	0	0000 0000	C800h	Channel 2 input is selected
Manual Ch 3 Selection (MAN_Ch_3)	1	1	0	0	1	1	0	0	0000 0000	CC00h	Channel 3 input is selected

http://wiringpi.com



WiringPi Resources

- Raspberry Pi GPIO Pin numbering
- Download and install
- Examples and How-To's
- WiringPi function referrence manual/documentation
- GPIO Extensions
- DevLib
- The GPIO Utility

PiFace

WiringPi fully supports the PiFace board too. See this page for more details.

Gertboard

WiringPi fully supports the Gertboard. See this page for more details.

Other wiringPi resources:

- Thanks to Gadgetoid there are now wrappers for Ruby, Python and Perl and these can all be found here.
- Thanks to Jeroen Kransen there are wrappers for Java which can be found here.
- Thanks to Dave Boulton for creating a TCL wrapper which can be found here.
- Pi4J is another Java project that uses WiringPi. It has a Github repository here.

Additional information can be found on the Raspberry Pi Wiki pages.

Minimal HV control program

```
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>
#include <wiringPiSPI.h>

int main(int argc, char *argv[])
{
   int spi_fd0, spi_fd1;
   unsigned char spi_buf[3];
   double hv;

   // setup IO
   wiringPiSetup();
   pinMode(8, OUTPUT); // pin #3
   spi_fd0 = wiringPiSPISetup(0, 10000000);// DAC 10 MHz
   spi_fd1 = wiringPiSPISetup(1, 10000000);// ADC 10 MHz

   // turn on HV
   digitalWrite(9, HIGH);
   hv = 1000;

   // calculate 16-bit value
   d = (unsigned int) (hv / 2000 * 65535);

   // program DAC range
   spi_buf[0] = 0x08;
   spi_buf[1] = 0x00;
   spi_buf[2] = 0x00;
   // Range = 0...+10V
   wiringPiSPIDataRW(spi_fd0, spi_buf, 3);

   // program DAC power register
   spi_buf[0] = 0x10; // REG=2, A=0 PUA = PUB = 1
   spi_buf[1] = 0x00;
   spi_buf[2] = 0x05; // PUA = PUB = 1
   wiringPiSPIDataRW(spi_fd0, spi_buf, 3);

   // set DAC output
   spi_buf[0] = 0x00;
   spi_buf[1] = d >> 8;   // MSB
   spi_buf[2] = d & 0xFF; // LSB
   wiringPiSPIDataRW(spi_fd0, spi_buf, 3);

   /*---------------------------*/

   // Read monitor output through ADC
   spi_buf[0] = 0xC0;
   spi_buf[1] = 0x00;
   spi_buf[2] = 0x00;
   spi_buf[3] = 0x00; // Manual Ch 0 Conversion
   wiringPiSPIDataRW(spi_fd1, spi_buf, 4);

   // convert to Volts
   hv = ((spi_buf[2] << 8) | spi_buf[3])/65535.0 *
         20.48 - 10.24;

   // round to one digit
   hv = (int)(hv*1000+0.5)/1000.0;

   // convert to HV
   hv = hv * 400;
   printf("%1.1lf\n", hv);
}
```

HV command line program

```
…

int main(int argc, char *argv[])
{
    if (argc < 2) {
        readvoltage();
        return 0;
    }

    if (argc == 2 && isdigit(argv[1][0])) {
        double hv = atof(argv[1]);
        sethv(hv);
        return 0;
    } else {
        printf("usage: hvcl [voltage]\n\n");
        return 1;
    }

    return 0;
}
```

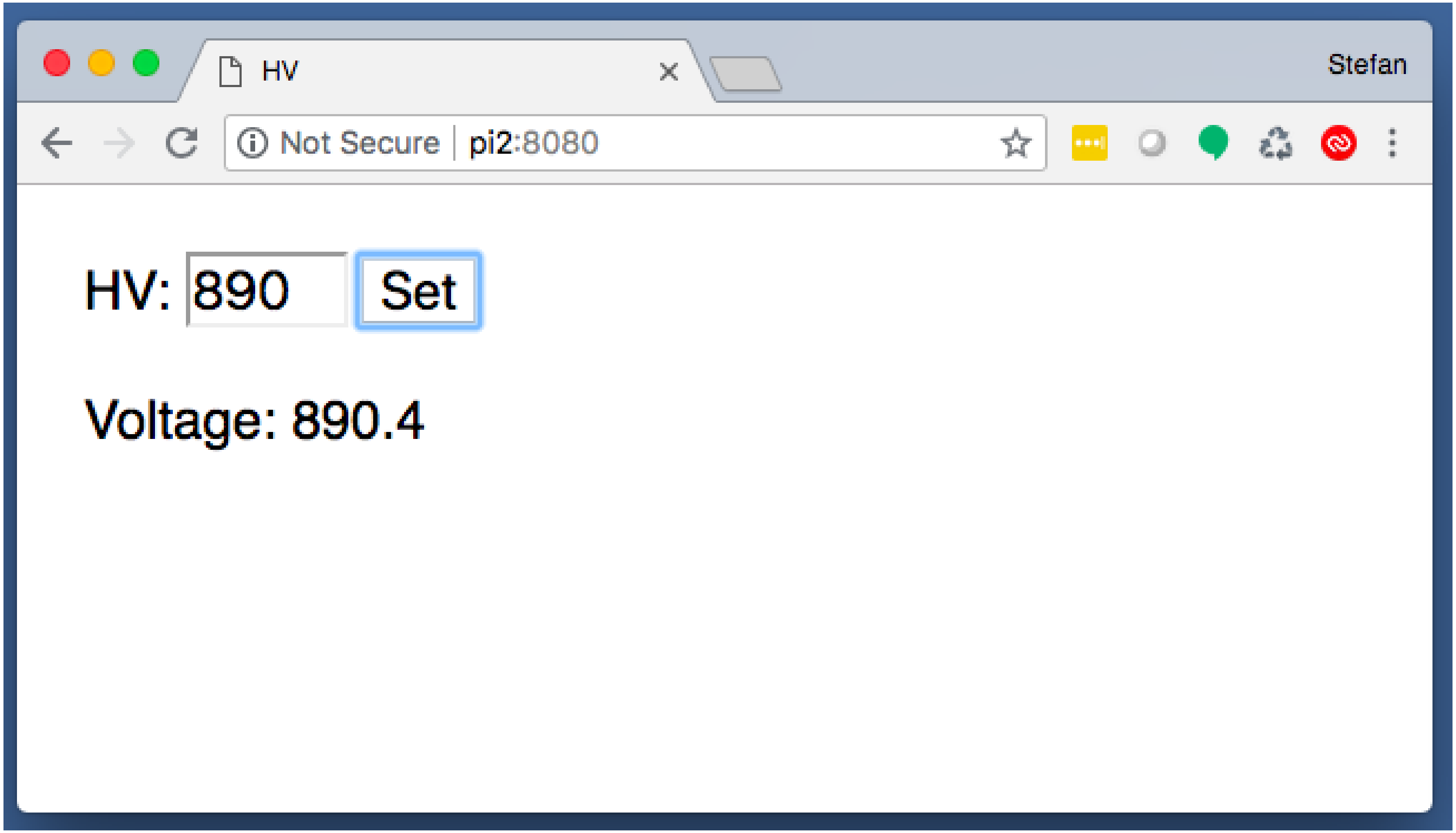
Raspberry Pi groups

- Group 1 (front table): pi01, pi02
- Group 2 (middle table): pi03, pi04
- Group 3 (back table): pi05, pi05
- Log in with Cygwin schell, usual password

Install HV software

```
~ $ wget elog.psi.ch/rts/hv.tar
~ $ tar -xvf hv.tar
~ $ cd hv
~/hv $ make
gcc server-hv.c mongoose.c -o server-hv -g -lwiringPi
gcc hvcl.c -o hvcl -g -lwiringPi
~/hv $ ./server-hv
Starting server on port 8080
```

Control HV



HV: 890 Set

Voltage: 890.4

Max voltage: 1600 V !!!

Connect HV Supply