Voltage logger with SD-card

# What is it?

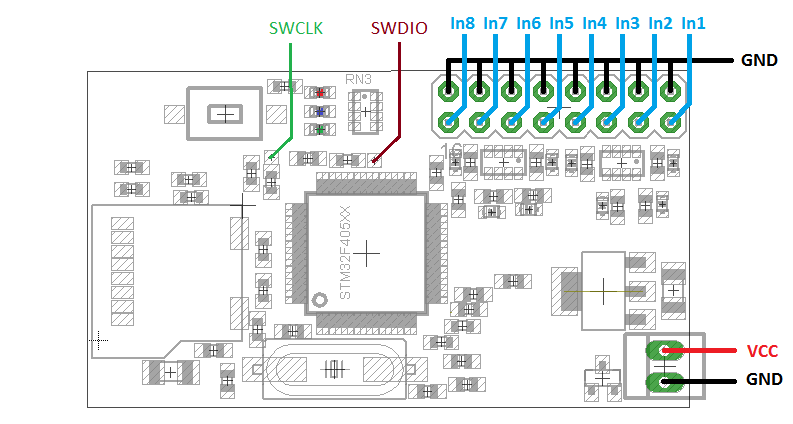
This is just a simple voltage logger which writes 8 voltages to the text file on a micro SD card. It has the following features:

* + - * Easy to use: Only one start/stop button and all the settings are stored in the configuration text file on the SD card.
      * The recorded data is stored in comma separated text file (CSV) and each entry (measurement) has a time stamp in milliseconds.
      * Configurable calibration coefficients (gain and zero offset) and moving average filter for each channel.
      * Three LEDs for indication of logger state.

Device specification:

|  |  |
| --- | --- |
| Parameter | Value |
| Power supply voltage | 5V-20V |
| Current consumption | 60mA at 5V input |
| Number of channels | 8 |
| Channel input type | Single-ended |
| Input voltage range | 0 … +3.3V |
| Sample time | 0.5 ms … 5000 ms (5 sec), 0.2 Hz … 2 kHz |
| Measurement accuracy (3σ) | 3.3 mV (0.1% of full scale) with 4th order moving average filter |
| Input impedance (max) | 50 kOhm |
| Microcontroller | STM32F405RGT6 |
| PCB size | 48.26 mm x 27.0 mm (1.9 in x 1.063 in) |

Board connections:



LEDs:

* Green: power on, also toggles eachtime the ADC samples input signals.
* Blue: blinks each time when a block of data has been written to the SD card.
* Red: fault indication (see below).

# How to start

* Input signals, make sure that each input signal is single-ended and in range from 0V to 3.3V.
* Connect power supply (make sure the voltage range is correct).
* Place **ADC.txt** file to the root folder of SD card (here is an [example](https://github.com/akpc806a/Voltage_Logger/blob/master/Tests/AC/ADC.txt) of the file for ODB). It is recommended to choose the fastest SD card (UHS Speed Class 1 / U1) and format the SD card before use in logger.
* Insert SD card.
* Press "START" button to start log
* The blue LED should blink periodically (with speed dependentonwriting rate).
* Press "START" button again to stop the log.
* The recorded data file is savedto a root folder and isnamed in the format: **HH-MM-SS.csv**, where HH-MM-SS denotestime from power cycling of the device.

# Configuration file format

All numbers in configuration text should be in decimals or floating point separated by a dot '.'

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Format | Default | Meaning |
| sample | float | no | Sampling time in milliseconds, all channels are sampled simultaneously |
| ch*X*\_en | 1 or 0 | 0 | If 1 then data from channel ***X*** is included to log file (X is from 1 to 8) |
| ch*X*\_zero | float | 0 | ADC zero calibration coefficient for channel ***X*** (X is from 1 to 8) |
| ch*X*\_gain | float | 1 | Gain calibration coefficient for channel ***X*** (X is from 1 to 8) |
| ch*X*\_filt | float | 1 | Moving average filter order, value greater or equal to 1 (X is from 1 to 8) |
| format\_str | string | %f | format of values in output file (as printf [format](http://en.cppreference.com/w/cpp/io/c/fprintf) specifies) |
| timestamp | 1 or 0 | 1 | If 1 then every record in output file has a timestamp (in milliseconds) |

The recorded value is calculated from ADC code using the following calibration equation:

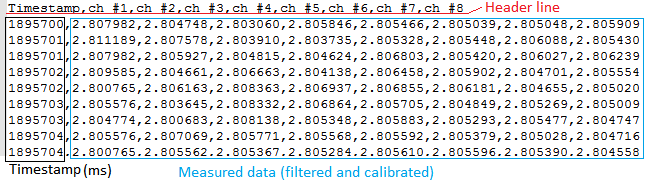
Recorded X = (ADC X - ch*X*\_zero) \* ch*X*\_gain

The moving average is applied to all channels independently from sample time, every 187 us using the following equation:

Filtered X = Filtered X \*((ch*X*\_filt - 1)/ch*X*\_filt) + ADC X/ch*X*\_filt

The unit step response time of filter order N is (0.187\*N) ms.

# Recorded data file format



# Faults and indication

* If red LED and blue LED are both on just after pressing the "START" button, then this is a configuration text file problem; check configuration file.
* If red LED is on during logging, then this is either a data buffer overflow or SD card problem. Check if there is enough free space on SD card or whether SD card has acceptable writing speed rate.

# How to calibrate zero

* Put this ADC.txt to SD card (it has ch*X*\_zero = 0 and ch*X*\_gain = 1).
* Short inputs to ground with wires.
* Start logging to collect about 1000 samples.
* Calculate average values of every channel, it will be zero offsets.
* Modify all ch*X*\_zero parameters in ADC.txt by calculated mean values.

# How to calibrate gain

* Calibrate zero offsets.
* Connect stable voltage source with known voltage to all inputs. The best voltage source is a battery with voltage about 3V (make sure that the voltage doesnot exceed 3.3V to prevent input saturation)
* Start logging to collect about 1000 samples.
* Calculate gain value for channel by equation: input voltage / mean(all channel samples)
* Modify ch*X*\_gain parameters by calculated gain values. Make sure that you copyat least 5 significant digits (for voltage, for example, it should be at least 9 digits after decimal point).