SWR MEGA project, version 1.0

Manual version 0.2, 10.1.2010

In summer 2009 my antenna tower in my new QTH was ready for antennas, so I started to build the shack. Good SWR meter is a need of course. You can buy a lot of different models on market these days, but such simple equipment is a challenge for a HAM - build your own. Most homemade meters use classic analog meter, why not, but it is also possible to engage new microcontroller technology little bit more and display the values on the LCD display. The analog indication is in most cases sufficient – you see immediately whether everything's OK or something happened with your PA, you don't need to crunch numbers. But in other situation more precise values expressed as numbers for power and SWR are useful. I have some experience with Atmel AVR controllers, so I tried…

The target is:

- SWR meter/indicator as independent unit with external RF probe (I will talk about probe later)
- LCD 2 line display; 1st line used for analog bar indicator, 2nd for numerical data. It is recommended to use LCD model with 40 characters per line, but version for 20 characters exists as well (there is a parameter in source file that compiles the right version, or take SWRM20.HEX instead of SWRM40.HEX; it shows less information and the bar resolution is lower)
- Backlit display for good visibility (disadvantage high current, shack use only - batteries would be empty soon)
- Maximum possible resolution for analog bars, for precise adjustment of PA for maximum output, fast indication (10 times per second)
- Switching between peak and average mode for bars and power
- The bar line must display both forward and reverse voltage amplitudes simultaneously
- Possibility to use several probes with different ranges (frequency, max. power, …) with automatic probe identification (up to 8 probes)
- Software compensation of detector non linearity (diods in probe are usually less sensitive for low signals, without compensation the power indicated would be lower than the real one). One set of parameters for each probe, stored in EEPROM
- Calibration for maximum power range including unit for each probe, stored in EEPROM
- Possibility to calibrate the meter from PC using dedicated program

Hardware

I took Atmel AVR type ATMega8 as a central piece. It has enough IO pins for LCD display connection (the display works in 4 bit mode with pooling, for minimum pin requirement and maximum speed), Analog-digital converter with 10 bit resolution for probe signals reading and also for probe identification signal, USART fro serial communication with PC, free pins for “in-circuit” programming and timers that turn the wheels.

I tried to keep it simple, so all things are as expected. The analog inputs for probe signals are just RF filtered and over voltage protected. The probe identification signal is derived from power supply voltage / there is a resistor coming from plus, you need just another resistor in probe connected to ground. Both create a voltage divider, the resulting voltage identifies the probe.

USART is connected using standard MAX232 interface. The serial data transfer is used during calibration, I send there also measured power and SWR during normal operation. In my opinion it is not very useful, but… It doesn’t eat a bread, people say. Maybe some people decide not to use the serial communication at all, so just simply do not use the MAX232 and the things around. You can put the values for calibration curve directly in source code and download EEPROM configuration using programming interface (*.EEP file).

LCD display is working in 4 bit mode (display’s pins D0-D3 are not connected) in order to spare processor output pins (and less wires = less possible mistakes). But I use RW pin to get signal about LCD readiness instead of time looping (like in AVROT project) – one reason is it can make the transfer faster (no useless time) and it should guarantee the slow modules work as well. The bars are updated 10 times per second. Just one note about pins. The majority of LCD modules use pin numbering identical to that I use. But there are modules having same signals on different pins (esp. crossed power and ground pins). Please check your particular model data in advance. Another note: because of polling for LCD not busy signal the presence of display is necessary, otherwise you are in endless loop (if the display is not connected). But in fact without display the meter has no sense :-}

Programming

Another challenge for me was to try programming in C language. All my former projects has been written in assembler directly; it is hard work but you have full control on program flow and you can easily debug the application in AVR studio. I used C like languages in my job so it is not strange for me, but writing C for microcontrollers is something different than for Windows. I find free GNU tool WinAVR, on first view I suppose there is all I need. I was also surprised how good is integration of that tool in AVR Studio. Unfortunately the GNU debugger doesn’t work on my PC (it looks like the AVR simulator works strange, crashes and reports strange errors). So finally I prepared the program in AVR studio and in Programmers notepad (included in WinAVR), both editors are good. Compiler is the common for both. I did some debugging in AVR studio, but I was disappointed so as soon as the display was working I did the debugging using the device directly with outputs on display.

First I was happy I don’t need to care about register numbers and assignments (the most boring in assembler), passing parameters to functions, saving something on stack etc., later I found it is not so easy - you have to explain to the compiler where are the borders it can work in: some data is untouchable (cannot be used for other purposes, or must always be evaluated despite the fact it looks useless on first view), especially because it is used in interrupt routines; the other problem was arithmetic – I don’t use floating point so I need defined arithmetic operation sequence in order not to lose precision, some numbers are signed, other unsigned, some 8, other 16 or 32 bit long. Unfortunately the compiler creates a code that is hard to debug in AVR studio (mostly because of optimization), you cannot simply jump from one instruction to the other.

There are lot of possibilities how to program the AVR chip. I still use the SP12 program, it does what I need so I don’t search further. I have 2 batch files – one to install driver (called once before I start work) and the other for actual data burning (setting switches and writing EEPROM and flash). Example for both is included in user package, along the SP12 program.

The main advantage of such programming is the chip can be inserted on board during burning, there is a dedicated programming connector (identical to that in AVROT project) where you connect very simple adapter connected to LPT port of PC (see SP12 documentation for details).

The burning sequence is

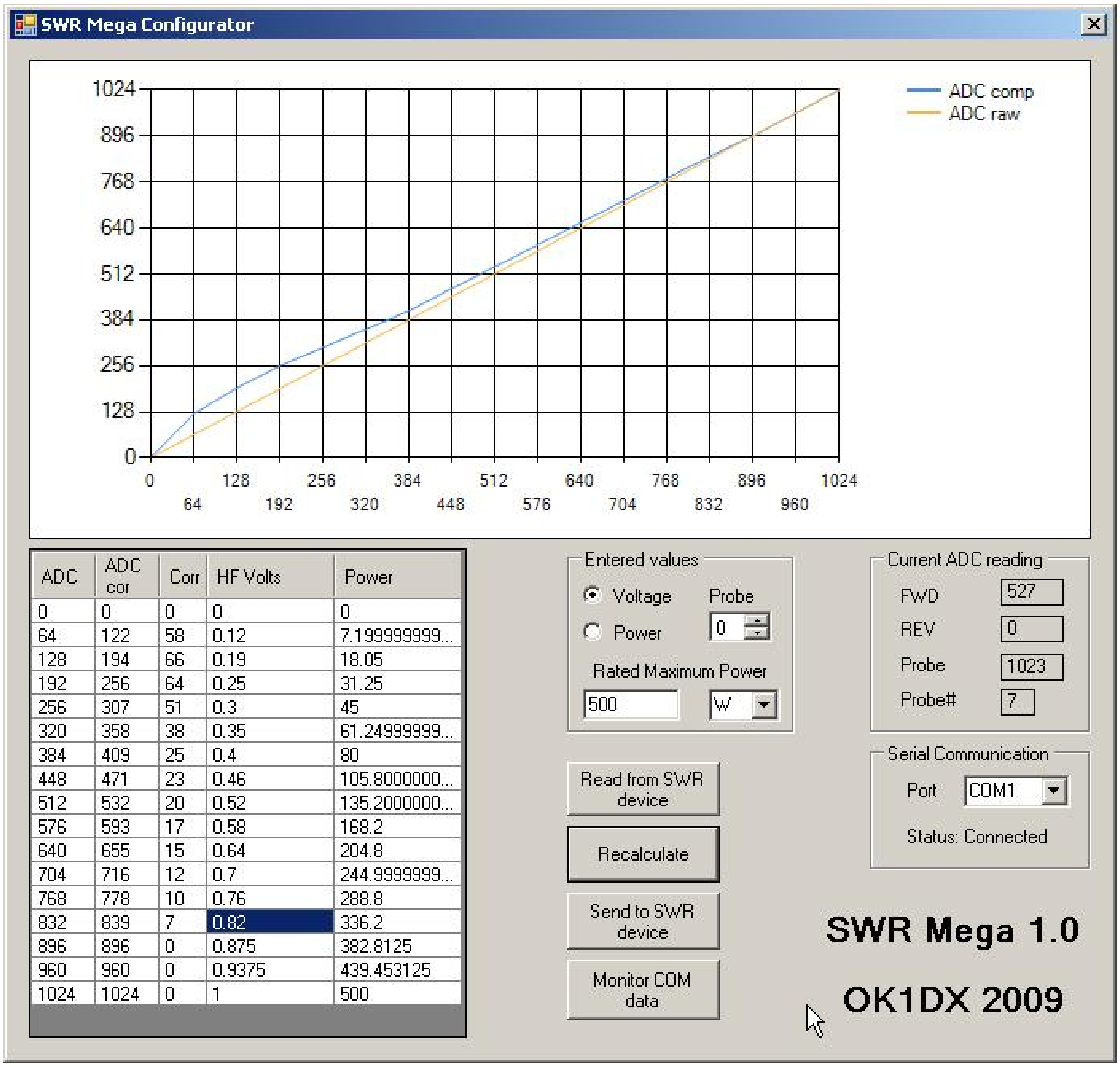
- Install SP12 driver (if not yet done)
- switch off the power of SWR meter
- connect programming cable
- switch on the power of SWR meter (LCD display remains not initialized, you can see just shadowed first line)
- start programming batch file for burning (you can start SP12 from command prompt too, but I don’t recommend – you can make a mistake easily)
- you are informed about programming process in console window (takes about 10 seconds)
- when done switch off the power of SWR meter
- disconnect programming cable
- switch on the SWR meter power again, the meter should start working now!!!

Note: the SP12 should calibrate time loop for given PC model (speed) before first usage. It creates a file that is used for subsequent operations.

Calibration

There is a dedicated program written in C# VS2008 that makes it easy. To run it you need to have .net framework package version 3.5 SP1 on your PC (you probably already have) and you need to install additional free Microsoft component MSChart (included in user package). The sources are included as well, you can use MS Visual Studio Express edition for tests.

The program shows one window with all the parameters.



ADC	ADC cor	Corr	HF Volts	Power
0	0	0	0	0
64	122	58	0.12	7.199999999...
128	194	66	0.19	18.05
192	256	64	0.25	31.25
256	307	51	0.3	45
320	358	38	0.35	61.24999999...
384	409	25	0.4	80
448	471	23	0.46	105.8000000...
512	532	20	0.52	135.2000000...
576	593	17	0.58	168.2
640	655	15	0.64	204.8
704	716	12	0.7	244.9999999...
768	778	10	0.76	288.8
832	839	7	0.82	336.2
896	896	0	0.875	382.8125
960	960	0	0.9375	439.453125
1024	1024	0	1	500

Entered values
● Voltage
○ Power
Probe: 0
Rated Maximum Power: 500 W

Current ADC reading
FWD: 527
REV: 0
Probe: 1023
Probe#: 7

Serial Communication
Port: COM1
Status: Connected

As first step you need to establish communication with SWR meter:

- connect serial cable to COM port
- Set correct COM port in the program
- Switch off the SWR meter
- start the SwrCnfg application
- Switch on the SWR meter (it is important the SwCnfg be running before the power is applied as SWR meter sends identification on startup only)
- The SwrCnfg should report state connected

The tool allows you to do 3 things: set range for power calculation, identify the probe and compensate detector non linearity.

Range:

The standard SWR probe has 2 outputs, both DC voltage. One probe signal is proportional to forward RF signal and the other to reverse. I don’t go to details of theory, for us is sufficient to know that 5V signal corresponds to rated probe power, that means when the probe signal is 5V the display show full bar and the power value displayed is the rated power. So let say we have a transmitter and dummy load (SWR=1) connected through the probe. When the probe signal is 5V (or we see full forward bar on SWR meter) the power the transmitter is giving is exactly rated power. That value is to be entered in program, together with unit (mW, W, kW etc.)

Probe identification:

There is a resistor in each probe connected between probe Id pin and ground. The SWR meter reads its value and uses it for probe identification. How to determine what resistance is for what probe? Easily. Connect potentiometer instead of fixed resistor, let say 10 kOhm. The program shows actual values directly from ADC on the chip. One value correspond to probe ADC value and one is probe Id decoded from that signal {you should see the same number on LCD display). When you turn the potentiometer the ADC values changes together with probe Id (higher resistance means higher probe Id). Try to find position where the ADC reading is approx. in middle of the range where the probe Id is what you want (let say we decide to assign probe id = 3 for currently tested probe). Disconnect he potentiometer, measure its value by ohmmeter and try to find a fixed resistor with same value (you can combine more resistors, too). Connect that resistor and you should see the correct probe Id displayed. When OK, put that resistor inside the probe (or just inside connector wired to the probe). Done.

Detector linearity compensation:

The semiconductor diodes used in probes show some non linearity because of their characteristics. That means, the DC voltage after detection is not linear proportional to RF voltage applied to the detector. The detection is especially poor when the RF voltage is low, bellow 1V. To improve the situation the SWR meter uses compensation. How it works: The whole ADC range is divided into 16 equally wide subranges. The calibration table contains values for compensation at subranges borders. When the ADC value is read, the program determines the range. Then it takes lower and higher table value and calculates the compensation using linear interpolation. Finally the correction value is added to ADC value, so we have a value that should correspond to RF voltage applied.

How to do the calibration. Of course we need another meter we can rely on to do so. It can be a RF voltmeter that allows measurement of RF voltage across the dummy load, another possibility is RF wattmeter. In the program you can select which meter you use. If you select voltmeter, you have to enter RF voltage (it is not required it is real RF voltage, sufficient is just voltage proportional to RF voltage) and rated power (you have to know that).

If you use wattmeter, enter power values (real power). Don’t forget to select unit…

The calibration sequence is:

1. Select probe Id for probe you calibrate (the displayed probe Id of actual probe is not used for that, but usually you set both to same number)
2. Observe actual ADC reading for forward signal
3. Adjust RF power in way the ADC reading is the same like reading in ADC column of particular row in the table
4. Enter voltage or power into corresponding row
5. Repeat steps 2 to 4 for all table rows

6. Click Recalculate button. The program tries to calculate the compensation values. If the compensation value is out of range (-128 to 127) the program report a problem. The reason can be too high non linearity of detector. Check.
7. Look at the chart. The straight line is non-compensated ADC value and the other line is the signal after compensation. It should be smooth line.
8. Click the button Send to transfer both the range and correction to the SWR meter.
9. Just for sure click Read to read back all that parameters from meter – the values in table should not change significantly (just some changes because of rounding).

The changes are immediate when Send succeeded so you can check the calibration immediately, no need for restart SWR meter.

Repeat the procedure for different probes as necessary, you can repeat the procedure as many times you want. The parameters are stored in EEPROM so they are persistent when the SWR meter power is switched off.

Note: you can easily transfer the calibration curve to more probes (different ID's) if you know all use the same detectors. Just adapt the rated power…

Displayed data

The first line on the display is used for bars (maybe you noticed the version information is displayed there shortly too, during startup only). The upper part of line displays forward signal, the lower reverse signal. The are 2 "pixels" per one character, so the resolution for 2x40 chars display model is 80 values.

The second line shows

A. for 2x40 LCD: letter A or P indicating average or peak mode (depending on switch position), probe Id, forward power, reverse power, real power (forward minus reverse), calculated SWR.
B. for 2x20 LCD: letter A or P indicating average or peak mode (depending on switch position), probe Id, real power (forward minus reverse), calculated SWR.

Note: the SWR is calculated only when the voltage is more than 10% of the range (that menas power more than 1% of rated power), bellow that level the precision would be very poor. The maximum calculated SWR is 10. No useful antenna should have such high SWR…

Operation

Here is nearly nothing to say. When everything's OK (the SWR meter is working, probe calibrated) – just connect power and probe and enjoy… There is only one thing you can control: a switch that changes between average and peak mode. You are perhaps familiar with such functionality on analog SWR meter. The peak mode evaluates maximum reading in certain time window, so if you use SSB and talk to mike it shows peak power. The average mode calculates floating average, so the indication gives overview about what average power goes to antenna (but, to be more precise, the averaging is done on voltage signal, not the power signal that depends on voltage square. So maybe instead of average power I should use term effective power…). Both apply to bar indicators and displayed power value.

The SWR value is calculated using average values only to filter possible strange peaks.

If you connect a PC to serial port you can see the measured data, just if you need it…

Serial port communication

The gear has a standard serial port with female DB9 Canon connector (strait cable can be used) to connect to serial port on PC. The communication parameter are 9600 Bd, no parity, 2 stop bits, no hardware flow control.
There are reasons: online measured data and setup parameters transfer.
The SWR mega sends also firmware version information when the power is applied (once at beginning).

Version data:
Sent once when the power is applied. The message from SWRmega looks like:

```
SWR10
```

Where the number indicates firmware version (here 1.0). This string should help other programs to identify SWRmega.

Measured data:
The gear sends periodically (every 500 msec) a block of data similar to this one:

```
Pw230.5 W
SWR1.45
```

Where Pw is the power consumed by antenna and SWR is measured SWR value. Note: the SWR value is not sent when the power is not sufficient for the SWR measurement.

Setup data:
Setup mode is used for calibration of the gear. It is activated by command only. The gear is out of setup mode after power up sequence.

Command to initiate setup mode (sent form PC to SWRmega)

```
set
```

Command to leave (quit) setup mode (sent from PC to SWRmega)

```
q
```

When the setup mode is activated the SWRmega sends every 500 msec following message:

```
P730
F840
R23
```

Meaning: probe ADC value 730, forward ADC value 840, reverse ADC value 23. The values are raw ADC values without any correction applied.

Command to read setup data (PC->device, only one):

```
r3
```

Reads block of parameters of probe number 3 – SWRmega replies by sending block of data with following meaning:

```
3G205
3S201
3CA5
3CB10
3CC8
3CD7
3CE6
3CF5
3CG5
3CH4
3CI4
3CJ4
3CK3
3CL2
3CM1
3CN0
3CO0
3CP0
```

The first number indicates probe number (here 3).
Next letter is G for range, S for scale and C for correction.
For G and S the following number is directly the parameter (see later for details).
C means correction point. We have 16 correction points indicated by letter A to P. The following number is the correction value at given point (see later for details).

Commands to write setup data:

```
ws31234
```

Writes scale word 1234 into probe 3 set. SWRmega replies by “WS”

```
wr3205
```

Writes range byte 205 into probe 3 set. SWRmega replies by “WR”

```
wp3c205
```

Writes one point into correction table. Probe number = 3, point number 3 (letter C), correction value 205. SWRmega replies by “WP”.

Meaning of parameters:
Scale – word indicating maximum reading scale. The value has decimal point shifted as necessary in order to get this value into range 1000 – 9999. For example, if the probe maximum range is 24.5 W the scale word is 2450.
Range – a byte indicating 2 things: decimal point and unit used.

The lower 4 bits indicates position of decimal point. This value together with scale word creates value that is displayed on LCD display.
The higher 4 bits indicates engineering unit displayed (0 = pW to 6 = MW).
Correction byte - a byte used to correct ADC values (forward and reverse signals). The value is interpreted as signed byte value (1-127 positive values, 128-255 negative values (128 means -128, 255 means -1). This value is added at correction point to value received from AD converter.

Copyright

The project is hamware, free for radio amateurs and non commercial usage (that means to build one copy for you, not for sale). For all other purposes please contact me.
I have an experience some HAM's ask me for help to get the microcontroller, to program it etc. I try my best to help them, but it is just a hobby and I have limited resources... About the PCB - I have no possibility to produce the boards. Anyway, if you are in position you can arrange the PCB for others, please let me know and I will publish contact to you here.
You can use the data or make web links to it as you want as long as the origin is noted.

73 Pavel OK1DX
Email: ok1dx@amsat.org
Web: http://ok1dx.dyndns.org