Using the AVR microcontroller based web server

***Abstract*:**

There are two related articles which describe how to build the AVR web server discussed here:

1. **An AVR microcontroller based Ethernet device**
2. **HTTP/TCP with an atmega88 microcontroller (AVR web server)**

The first article describes the hardware and the second introduced the web server software.

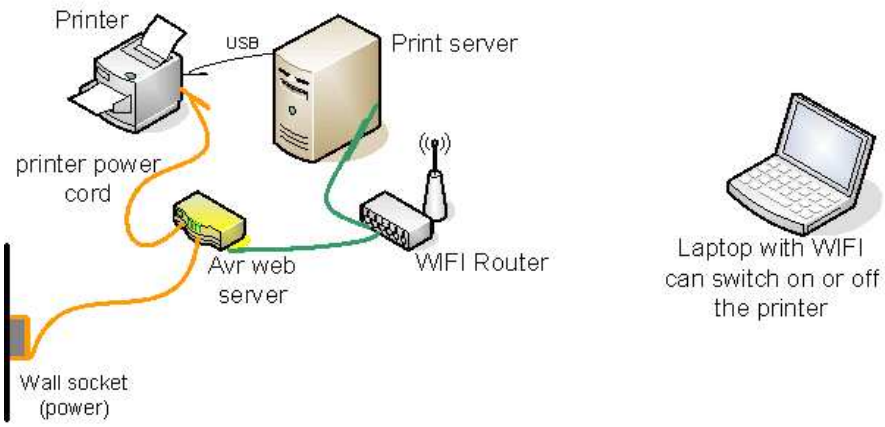
In this article we will discuss now some ideas on how to use this hardware and the web server software. The web server can be used towards the open Internet if some additional safety measures are taken.

The components for building this avr web server can be ordered from http://shop.tuxgraphics.org

Switching on and off a printer

I like my xerox laser printer because it is reliable and prints in good quality. However normally I print a couple of pages in a row and then for a long time nothing. The printer will go into sleep mode after 15 minutes but until then it has a loud fan an consumes a lot of power. I could walk to the printer, switch it on, go back to the computer, print, walk back to the printer and switch it off but it would be much nicer if one could switch on/off the printer remotely.

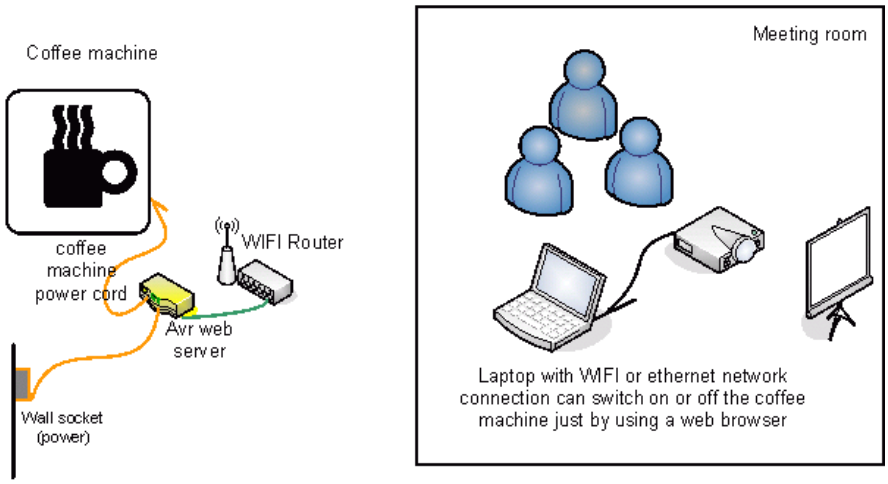
Remote control is especially convenient when I am sitting on a computer in a different room or with a laptop in the garden. This is how I use the avr web server:



Preparing fresh coffee for the break while you are giving a presentation

This is an idea which a customer had who bought the kit at shop.tuxgraphics.org.

You attach a coffee machine to the avr web server and fill the coffee machine with water, coffee powder,... Now you can give a nice and inspiring presentation to your customers. Shortly before the end of your presentation you can switch on the coffee machine directly from your computer and without leaving the room. .... and you can serve fresh coffee for the break.



Note: Be careful and use proper insulation in a possibly "wet" environment near the coffee machine.

Control your hardware over the public internet

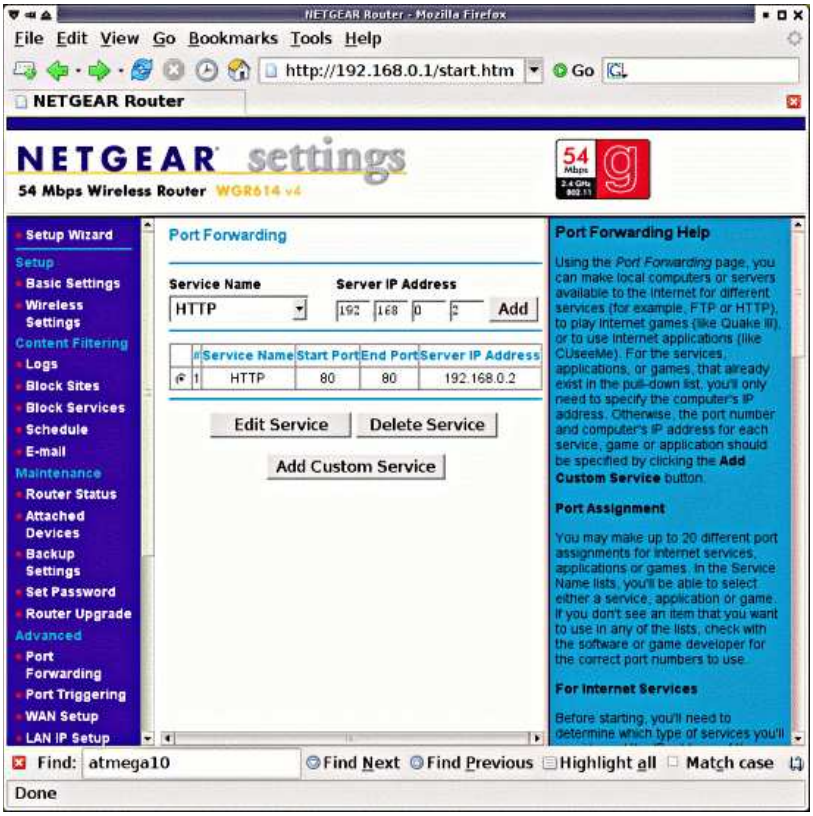
With a network enabled microcontroller such as our avr web server you can of course control anything from anywhere in the world if you connect this device to the internet.

Mobile phones have these days integrated web browsers. In other words you don't even need a computer to switch on/off a coffee machine, control the heating in your house,...

All DSL routers include these days already a firewall and something called NAT (network address translation). This NAT feature allows you to use the internet from several computers at the same time. Inside you network you use a private address range (e.g 192.168.0.1 up to 192.168.0.254) while your internet service provider has actually given you only a single public address e.g 216.109.112.2. The translation (NAT) between private and public addresses is possible because udp/tcp uses also two (client and server) 16 bit numbers called port numbers

for each connection.

A web server runs normally at port 80 but you can run it at any port. The avr web server offers also a port number configuration (look at main.c of eth_rem_dev_tcp-2.X.tar.gz). Typing http://your.domain:80/ is the the same as http://your.domain/. In order to make a web server in your local network available through the NAT firewall of your DSL router you need to tell the router to forward a given external port to an internal IP address and port number. Many DSL routers can do that. Here is an example of the port forwarding configuration page of a netgear router.



It will forward with those settings a request to the external IP address and port 80 to the internal IP address 192.168.0.2 and port 80. E.g like this:

```
http://216.109.112.2:80/  --->  192.168.0.2:80
```

We could now run the avr web server at 192.168.0.2:80 and it would work:

First tests with a mobile phone.

Security aspects

The tuxgraphics TCP/IP stack dates back to the year 2006 and has in the mean time been used for many applications by many people. It is at this point pretty stable and well tested.

This tiny tuxgraphics webserver does actually out-perform many other web servers. Even an apache server on a small PC will have trouble to keep up with it when it comes to number of users and speed. The tuxgraphics webserver does not have hard limit on the number of parallel users and it can serve hundreds of web pages per second. There is no other embedded TCP/IP stack that can do this on just an atmega88/168 microcontroller.

This performance comes of course at a cost. It can only serve small web pages and it does not do much validation of received IP packets.

It is therefore recommended to use always a firewall as provided by most home DSL routers.

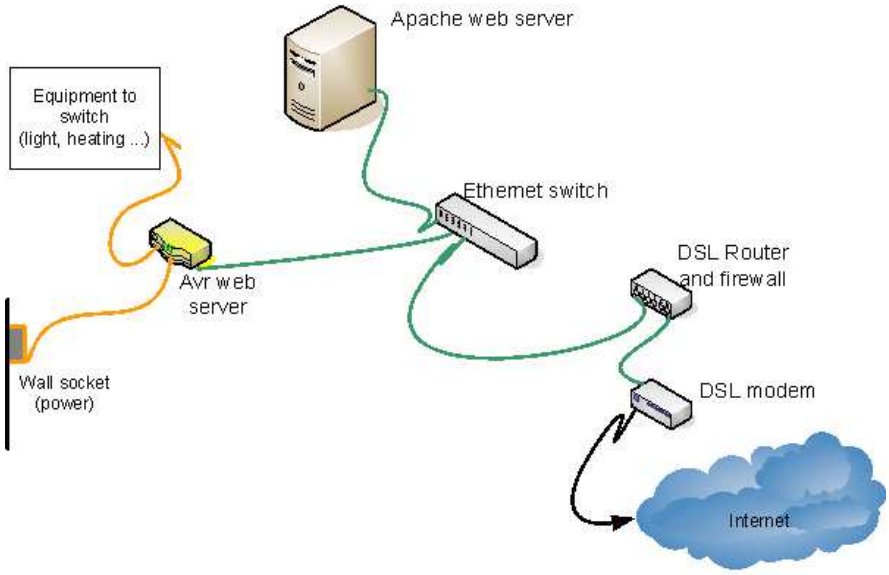
If you want even more control, especially if you want to log every request to file, then you can use the proxy-cgi as described in the next chapter.

Additional security using a proxy-cgi script

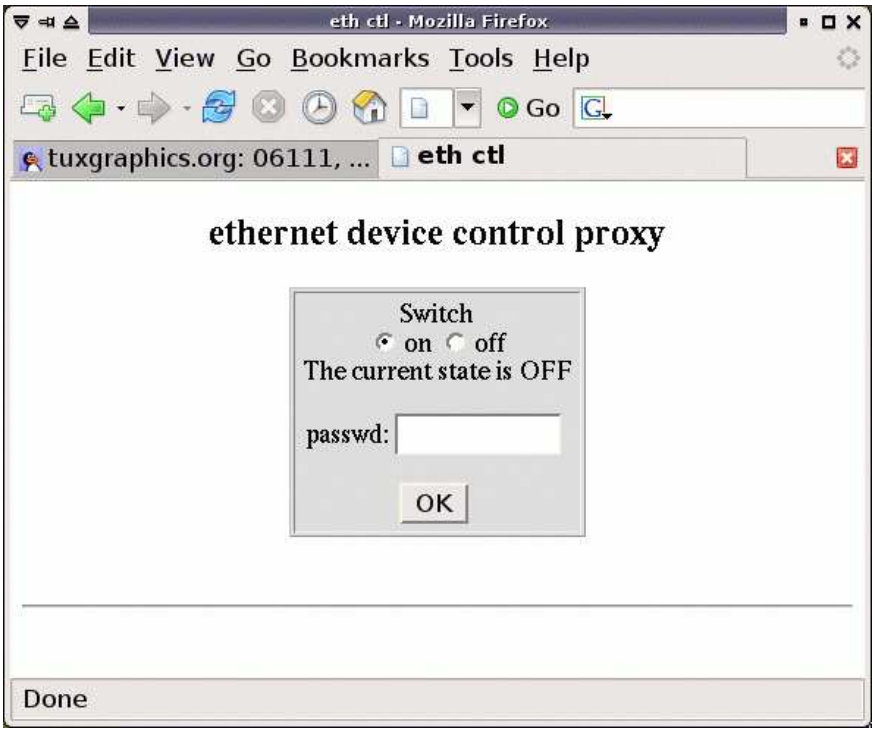
One way to protect an embedded server from direct access is to hide it behind a different server. One can e.g put an apache web server in-front and let it forward the request via a cgi script to the actual avr web server. This has several advantages.

- **You get automatically a log file written where you can see what was done at what time.**
- **You get better access control as you can change the password. The external password towards the cgi-script can be different from the internal password towards the avr web server.**

The setup looks now like this:



Here is a screen shot of how the web page generated by the cgi-script looks like.



The file which you can download at the end of the article contains two cgi scripts written in perl. One uses udp for the requests towards the avr ethernet device and the other http. A README file is included which explains further details.

Dynamic DNS

Most ISPs will give you a dynamic IP address. That is: your external IP address changes every time you switch on/off your router or even worse most telecom companies will break the connection every 24 house or every 12 hours or every 3 hours... It depends on the fine print in your contract. If you can get a static IP address from your ISP/DSL provider then go for this solution. It makes things much easier. If you can not get it the you can use a service like http://www.dyndns.com/services/dns/dyndns/. Some DSL routers do even have build-in support for dyndns.com. A list of supported hardware is available from their web site. dyndns.com works basically such that a small "update-client" software runs on the server in your local LAN and reports the current IP address to dyndns.com. This IP address is then aliased to a DNS name.

If your not sure how often your external IP address changes or if you even have a static IP then check your external IP address once in a while using one of these pages:

- **http://checkip.dyndns.org/**
- **http://tuxgraphics.org/cgi-bin/checkip**

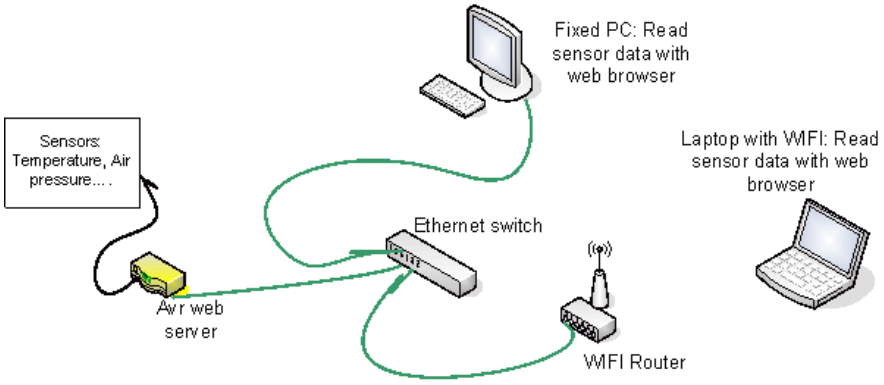
It tells you your current external IP address. You can use this also during first tests to find your external IP address.

Finally a word on reliability: My personal experience with the local telecom company which provides DSL connection here is that their service is pretty unreliable. It happens quite often that DSL is down for 2-3 hours per month especially at night. In some cases you even have to call them to do a DSL port reset on their equipment because it died again. I know from discussions with friends that it is not better in other countries. In other words don't use such a setup when 100% reliability is important.

More than just switching on/off

All the above ideas and examples where about switching on or off a single device. This is because the current example software (eth_rem_dev_tcp-2.X.tar.gz) has this functionality. The hardware has however a number of free IO-ports and ADC input lines and could switch multiple devices and even measure voltages.

The next extension to this circuit will add two temperature sensors and an air pressure sensor. This will allow you to build a little weather station. You can have the data from the sensors then instantly available in your whole network readable with any web browser.



Automation and scripting

With a simple web interface we have not only a user interface for humans. It is also possible to use it as a machine interface. It is so easy that this can even be done from a unix shell script. All we need for this is a command line ASCII only web browser:

- **w3m (http://sourceforge.net/projects/w3m/)**
- **lynx (http://lynx.isc.org/) are popular**
- **wget (http://www.gnu.org/software/wget/). Wget downloads just the plain html but can display it also to stdout.**
- **links (http://links.twibright.com)**
- **elinks (http://elinks.or.cz/)**

On top of all those web browsers you have also the option to modify the cgi perl script (see download at the end of this article).

Any of the following 4 commands will e.g switch the hardware connected to the avr web server on:

```
w3m -dump http://10.0.0.24/secret/1
or
lynx -dump http://10.0.0.24/secret/1
or
wget -q -O - http://10.0.0.24/secret/1
or
elinks -dump 1 http://10.0.0.24/secret/1

 (please change IP address and password as needed. This example
  refers to the switch on/off function as provided by the
  eth_rem_dev_tcp-2.X.tar.gz avr web server software).
```

To query the status you can e.g write a UNIX shell script like this one:

```
#!/bin/sh
# Query the status of the relay connected to the avr web server.
# Needs the eth_rem_dev_tcp-2.X.tar.gz software on the avr web server.
#
# the following just retuns the html line where is says "Output is:" on or off
htmlcode=`w3m -dump http://10.0.0.24/secret/| grep "Output is"`
#
if  echo "$htmlcode" | grep -i "on" >/dev/null ; then
    echo "The output is on"
elif echo "$htmlcode" | grep -i "off" >/dev/null ; then
    echo "The output is off"
else
    echo "Error: no output state, avr web server may be disconnected"
fi
```

```
#
```

You can save this in a text file called getavrdevstatus, make it executable (chmod 755 getavrdevstatus) and you have a command which report the status.

This is of course a very simple example as such but it can be modified for different things. E.g give water to your flowers every day. Just connect a little pump.... The beauty of a simple web interface is that you can use it as a machine interface and a interface for humans at the same time to give e.g you plants some extra water ;-).

Conclusion

I find it really fascinating what one can do with a small web server. Suddenly the microcontroller is not anymore a standalone piece of electronics. It becomes part of a network.

It is however not only a web sever providing web pages. It is an interface to other hardware. This makes the number of applications almost endless and it is completely different from a large web server running on a PC.

Download and links

- **Download page for this article: the proxy cgi-bin scripts**
- **The previous article with the description of the hardware: An AVR microcontroller based Ethernet device.**
- **The previous article with the avr web server software (eth_rem_dev_tcp-2.X.tar.gz): HTTP/TCP with an atmega88 microcontroller (AVR web server).**
- **The tuxgraphics shop: shop.tuxgraphics.org**
  Here you can get all the components needed to build this small web-server.

2009-09-27, generated by tuxgrparser version 2.57