Fire hydrant project

Soltész Péter 2016, ÓE, kvk

embedded systems project 1

**Description**

The project’s aim is to create an embedded instrument fit inside IVÓCSAP’s product, in order to measure how much water is flowing through it.

The main purpose of this is to minimize costs and time of maintenance operations, and also to provide statistical data about the usage of the tap.



**Requirements**

Vízművek has a very strict quality control for drinking water, every month they take a sample from the taps, and every two weeks they wash it inside-out to prevent the presence of bacteria.

Pending water can cause infections, so the device should send a warning if it is not operated for more than a day.

The temperature of the water should be registered. The device should count each opening of the tap by a tactile button, or a limit switch. It should calculate water flow from time of operation (open time, button pushed), and send data to a server computer using 2G.

The device must be as low power as possible as no external power source will be available whatsoever.

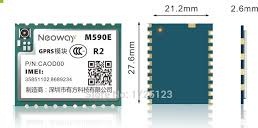
It should run for at least one season or 8-9 months without changing its batteries or any general maintenance.

**Technical description**

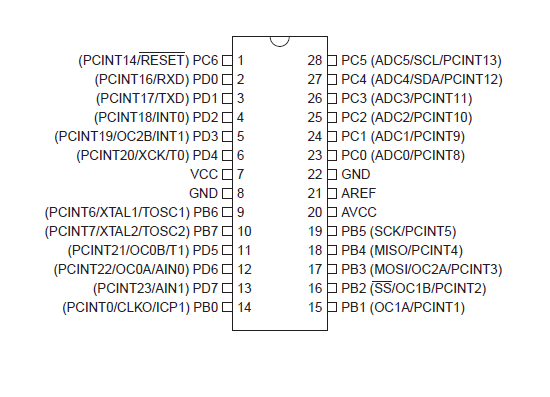
The circuit itself is hidden under the button of the tap. There is a very tiny space left for this purpose, therefore it must be minimal in dimensions.

The structural plans of the tap are kept secret, but there is a 7mm height ring shaped space inside the tap, with a 2cm radius inner, and a 4.9cm outer round, where the instrument will be placed.

The brain of the device is the AVR ATmega328p MCU, which is a low power chip. It is enhanced by a NeowayM590 GSM module with 2G communication to provide a stable connection countrywide. IP based data frame is used to get the gathered information onto the server.

I decided to use the Neoway module because it just perfectly fits in the available space. The module draws a 2A instantaneous current when carrying out a transmission, so very careful power management is required to protect the MCU core, but to feed the need of the GSM module also, which has the following dimensions:

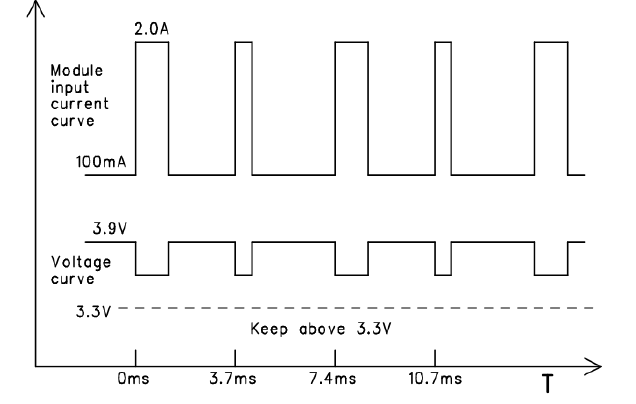
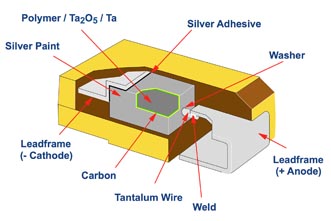
This module is capable of handling UDP/IP communication, which does not check if the transmission was received, as it has no acknowledgment line. The module will send data once each day. If no client appears on the server in 24hours, then the operator will know that the unit needs maintenance.

The GSM module has the UDP/IP communication implemented internally, the MCU communicates via USART with the MCU. The sensors are directly connected to the general purpose I/O ports of the microcontroller since it has more than enough. The pinmap of the MCU is shown below:

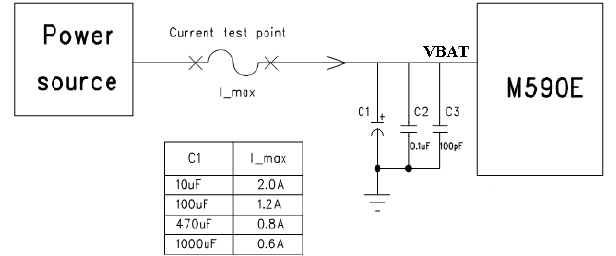
For clock signal, external 16MHz crystal is used as it is quiet low power, and accurate enough, but it needs two capacitors to maintain its stability. The MCU counts time until at least one day has elapsed between two transmissions, in order to save energy, but this period can be changed according to the available battery capacity.

**Technical details**

The MCU switches on the power of the GSM module before each transmission, and turns it of after it to conserve the battery. It also checks whether the voltage on the battery has dropped too low or not. If it does, the remaining energy should be used to send out a low battery warning, as the battery is most likely about to die soon. This is done via one of the A/D converter channels of the MCU. The whole application runs on 3.3V, the interface level is about 3V also, the GSM module cannot handle greater voltages than 3.3V at its digital interface I/O-s. The module itself runs on a 3.9V power level ideally. It consumes high power at transmissions, so the time of these is minimized.

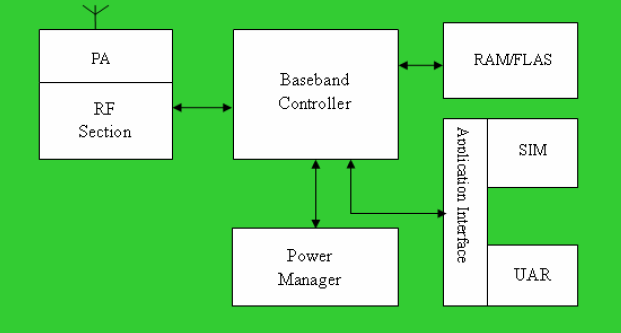
 To meet the instantaneous current demand of the module, a large, low ESR capacitor is placed as close as possible, which acts as a buffer. 1000uF, or 3x 330uF is good enough for this purpose, Ta capacitors are available at this size, in SMD package, although they are quiet expensive.

Further small size capacitors are placed near the module to filter high frequency interference at times of transmission (common digital noise).



The power source itself is a lithium battery, because those can supply very high currents in very short time without getting damaged. The 3.3V supply is given to the MCU by a simple 3 pin LDO chip. Although the ATMEGA328P can run on 5V also, the LDO provides protection against high currents getting to the VCC of the MCU, which would burn the chip. The power of the GSM module is switched off every time it is not used, because it has a 2.5mA current consumption in sleep mode, which would be a waste in this application, I could cannot afford.

The GSM communication requires a SIM card connected to the module. Luckily Hungarian Telekom agreed to give test SIM cards with internet access free of charge. First ten prototypes of the measuring instrument will run from august to the end of season, when the taps are taken down to prevent cracks caused by frozen water at wintertime. The taps are being put out at late spring usually, when temperatures don’t fall below 0 degree Celsius, even at night. This year, they are being put out in early May due to certificate delays. I will receive the test SIMs next week.

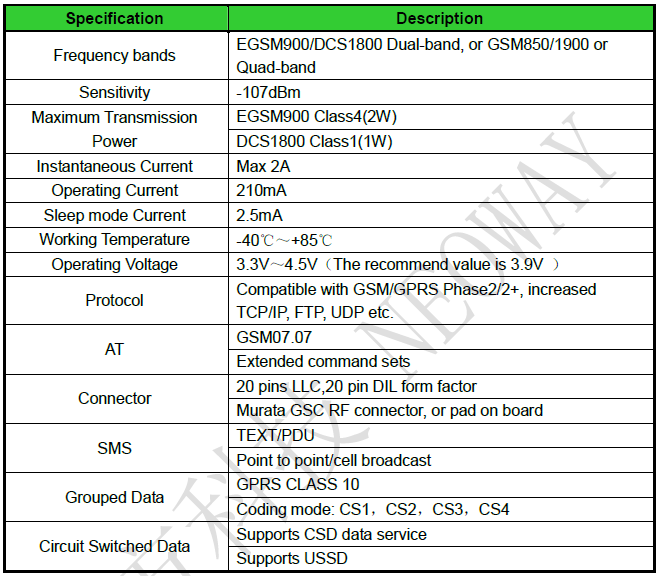


The button is connected to a general I/O pin of the MCU. It is a limit switch, that has a bending arm to bear with the mechanical stress of being operated many times, with different force acting on it each time. Also it should handle vandalism, it should hold if someone smashes the outer button.

Mechanically it is placed right under and with physical connection to the outer button, with which the users open up the tap. The outer button slides into the body of the tap, pushing the needle down with itself, to a depth determined by the user, so basically a random depth. The MCU should sense even a small indentation of the outer button, so the limit switch has to close even to a small force. The water flow is of course separated from the place of the instrument, running in a small diameter drilled hole ending up on top of the tap (see cover). The PCB is not prepared to bear water, or even high amount of moisture. The tap has a certificate from ÁNTSZ, they measured the water quality in the tap at different times, so it is not harmful of course, otherwise it could not be sold.

Separately, Vízművek also checks the water quality every two weeks, as mentioned before. They did not tell me what parameters they measure in their labs on samples taken from the taps, so it is not clear yet what sensors should be put in connection with the instrument, once I will know what to measure, hopefully the prices of maintenance could be lowered rapidly. Until then I can only offer to measure the times of operation for them, and they know the water pressure in the area, hence they can calculate the water consumption of the taps, but as it usually costs 600HUF each month from last year’s statistics (rough calculation), it is not the main concern. The lab measurements and test of the quality of the water costs more than a hundred thousand each month by tap, so that what should be reduced as much as possible, but this is a plan for the future. Another factor taken into consideration is anti-vandalism, and anti-theft actions. The instrument is planned to measure its position using both GSM data available, and an acceleration sensor. If it measures more than let’s say 1.5G, it should send a warning that it is being taken away. The outer button is made out of metal also, weighing a lot, so it should also sense if it is taken apart exposing the circuit itself visible, and vulnerable to the outer world.

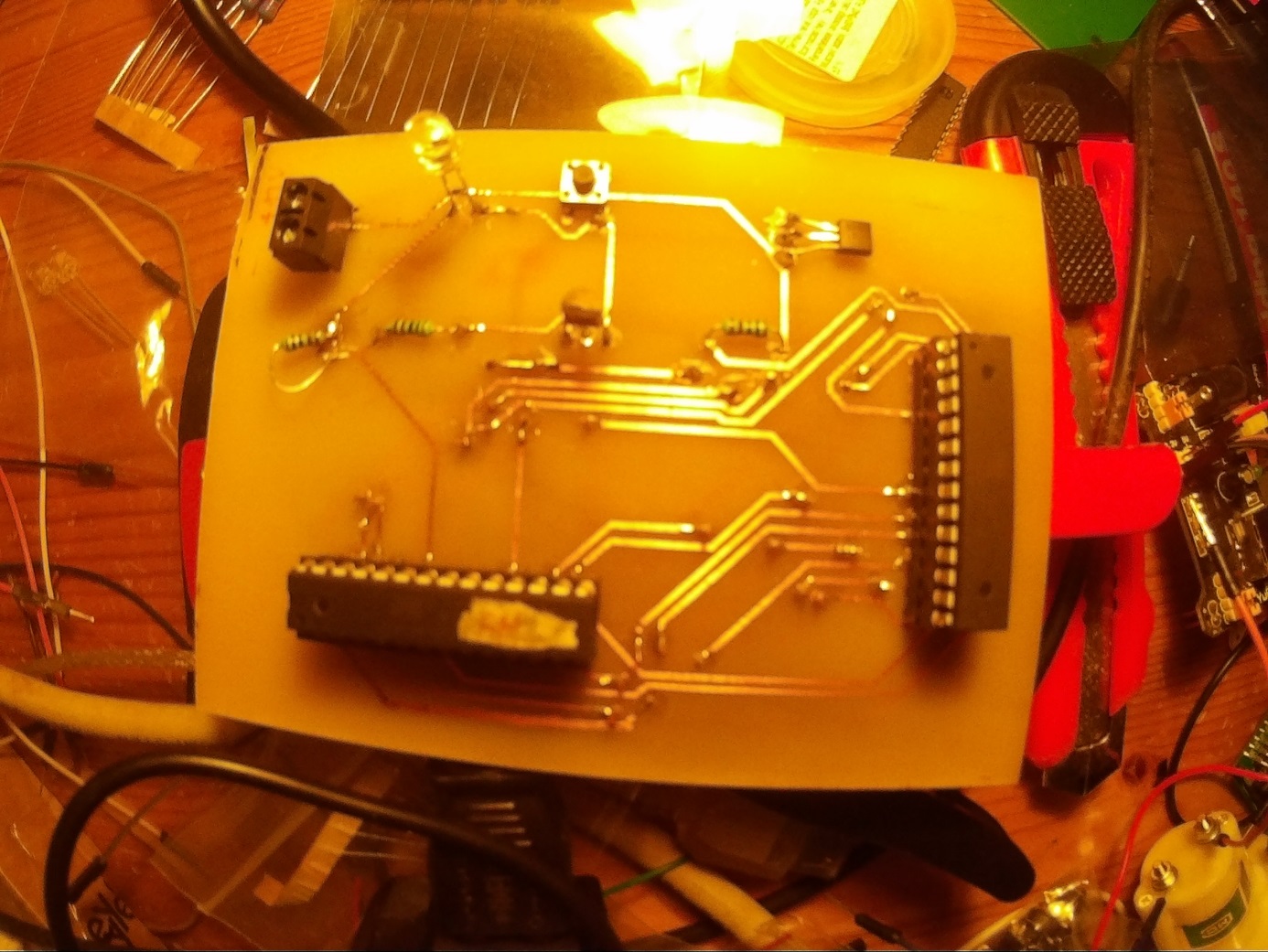
Because the body and the button, and mainly everything is made out of metal in the tap, it is quiet hard to send out any transmission from that, since it acts like a Faraday cage. The mechanic team of the company is currently working on another solution, a plastic button that would allow radio signals to enter and exit the tap, reaching the GSM module. Most likely it will be made for the next year’s season, 2017 springtime. General description of the module can be seen below, given by the manufacturer:



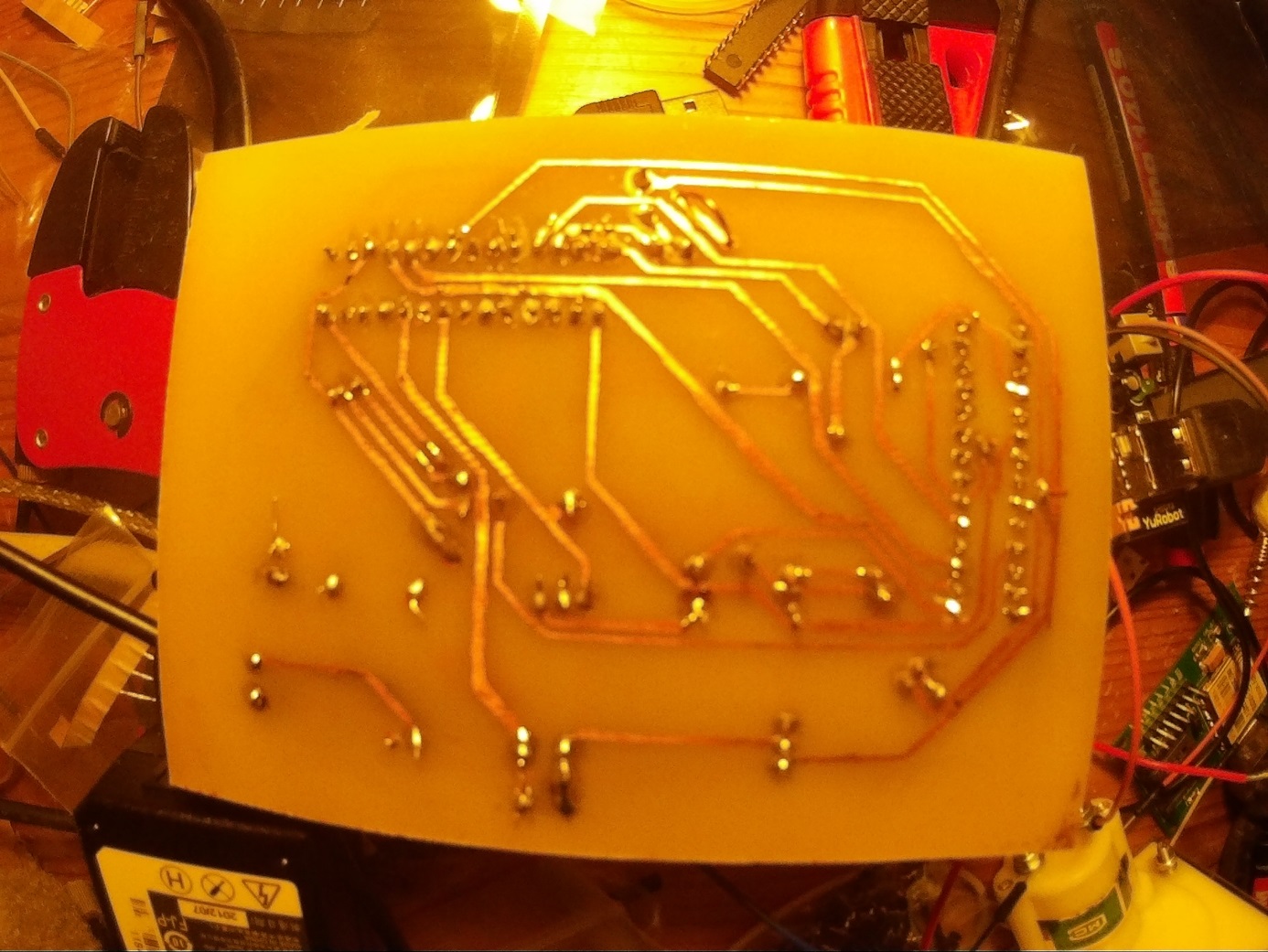
We can see that it has a -107dBm sensitivity towards radio transmission signals. I doubt that it would work in a Faraday cage like that, but further test will approve the functionality of it, hopefully we do not have to replace the button, and test the instrument before next year.

**Sideways**

When I started the whole project, I bought a very cheap, some 200-300 HUF USB ASP programmer copy made in China, and some ATMEGA328P MCUs with it. The standard baud rate in the GSM module is 115200, and the default inner clock crystal of the MCU is set to 1MHz, but this comes with a +- 10% deviation, so it is not too accurate for timing. I wanted something more precise, and obviously an external crystal is the perfect solution to this question. To enable the MCU to run from an external clock source, its fuse bits must be set accordingly. Fuse bits are like little switches inside the MCU that are responsible for switching on or off its inner modules. These must be set very carefully, because it is easy to “close ourselves” out of the MCU, if for example we disable the serial programming possibility in it. And of course as a beginner, I did exactly this with many chips, not knowing what this caused, and I ended up unable to reach my MCUs, that cost me about 800HUF each, and I didn’t have many of them anyway. I needed a solution, some way to reset the fuse bits in them so that I could continue experimenting with the clock settings. I found a useful way searching the web for solution, namely high voltage programming, which is implemented in the AVR MCUs, and its description can be found in their datasheets. The main idea is that we keep the reset pin of it at 12V, the only pin that can bear that overvoltage. Doing this the chip is vulnerable to parallel programming, and so using another similar type MCU, we can easely reset the fuse bits, or memory, on anything in it to factory default or whatever we like. I found a circuit description of this parallel programming, and I implemented, and built this design, because I could not make the MCU communicate with the GSM module without an external clock source. This was the first time I actually made a PCB of my own using a household iron, photopaper, and a double sided PCB board from lOMEX. My first attempt did not work out well, I messed up the ironing of it, and the circuit missed some essential connections, it did not work, and also the eagle design was not too accurate for home-made PCBs, because I left the wire widths too low, also I did not use enough vias for the first time. Luckily my second try was a huge success, learning from my mistakes at the first, I was able to build the circuit, and although it took three days to find all errors of it, meaning connections forgotten to soldier, or connect, etc, I managed to make it work, so now I can reset the fuse bits of my MCUs with it anytime. This is very useful, because now I can concentrate on the main task: connecting the MCU with the GSM module, and attempt to send data to a server. It was a very useful sideway of my project I learned a lot from this mistake of mine, I do not know to this day that rather the Chinese programmer burned the fuses to a wrong value, or I did a mistake when setting them up in its software before programming, I will start to try setting it up correctly. Here is the fuse resetting circuit of mine in its final stage:



The MCU on the left is the “saver” or master MCU, and the one on the left side is the one to be saved, or reset. The circuit runs on a 12V supply with a simple LDO making 5V out of it for the MCUs. The reset is started by the button pushed, and the led indicates the state of it. I did not write its software, I only modified it to set it to factory defaults described by the manufacturer. I only designed the connections, and the PCB layout. I allowed the auto router in eagle to use a lot of vias, because for the first time the wires could not get around each other, as there is a lot of them, almost all 28 pin of both MCUs is connected somewhere. I did not galvanize the vias, I only pierced a piece of copper wire in them, and soldiered the two sides of it. I used the technic of etching, because it seems reasonably simple, and cheap, one can get all tools for it in LOMEX, or from ebay, and it takes about 15minutes to etch it. Debugging of connections, and soldiering took a lot more time, I spent three days overall with this circuit to finalize. Here is the other side of it:



It is very hard to cut the PCB board to a ring shape, the outer circle is hard to cut out, I only made one so far, all the modules and sensors must fit this space:

This is where I am currently with my project, next week I will receive the test SIM cards, and so I can start transmission!