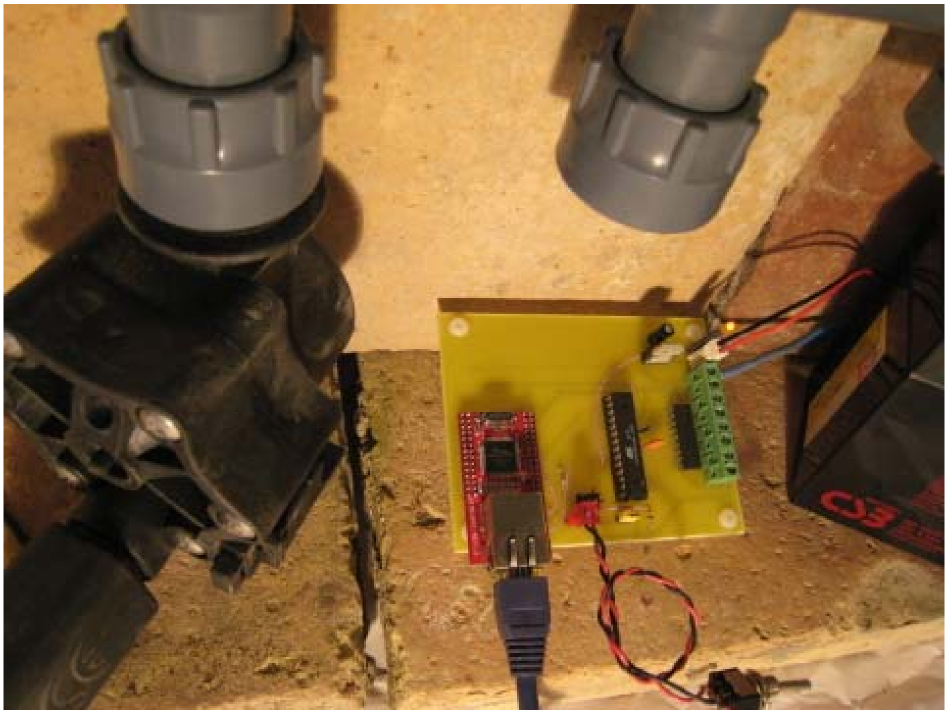
Irrigation Timer with Advanced Planning tools (ITAP)

Date: January 26, 2008
Project Number: 001142



Eligible part: WIZ810MJ

Consider the relationship between a device's function and its user interface. Among common household devices the humble irrigation timer is off the scale in terms of its ratio of UI complexity to device simplicity. The problem is the sheer volume of data that needs to be entered to tell a simple timer when to turn a solenoid on and off. A typical user quickly finds the task too complex or too tedious and resorts to ignoring the most useful functions.

Add a network interface, like the WIZ810MJ, and suddenly a completely different approach is possible. This is one instance where a web interface is not just a gee-whiz add on. Sure a networked fridge can send you email when the milk is about to spoil, but the web interface is just another bell or whistle, yet something else in the house to program. Contrast this to the typical irrigation timer. Flip to the owner's manual and you will see a large table of data to fill out. Across the top of the table are columns for things like on times, run times, odd/even day, and day of the week schedules. The rows of the table are for distinct watering zones. The data you need to enter is really the entire contents of a complex spreadsheet. For cost reasons the typical timer device has just a small LCD, a big rotary switch, and perhaps a few keys. Spreadsheets may be child's play on a laptop, but not on a tiny LCD with a single knob. Imagine doing data entry with the controls on a hand held DVM. Now to make matters much worse, any errors in entering the spreadsheet data will waste water, cost you money, and possibly ruin your yard. Oh, and you may not learn about these errors until they show up at 4am toward the end of a 10 day repeat cycle.

The networked irrigation timer (ITAP) presented here solves these problems, provides sophisticated “what if” planning tools, requires no software installation, and does it all in less than 16K of flash.

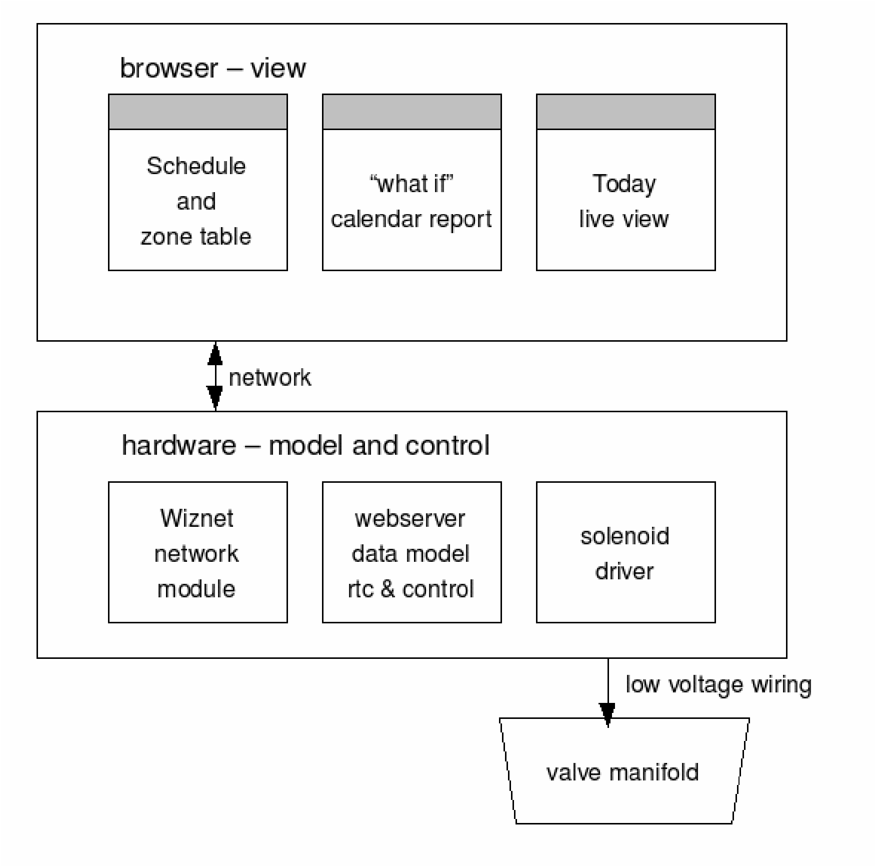
The project also demonstrates how concepts from large enterprise web development are just as useful at the micro level. It is unexpected to find such a small device so buzzword compliant: DHTML, AJAX, Fat Client, Model-View-Controller. They are all here and used appropriately, not gratuitously.

Design Summary

With the simple addition of a network module, the WIZ810MJ, it was possible to create a different type of device. There is no keyboard or LCD. Instead, all user interaction aspects of the device have been cleanly split off into a standard web browser. The hardware design is deliberately as simple and low cost as possible to showcase what can be done in software. The MCU is an ATmega168. The only storage space is the MCU's internal 16K of flash. The schematic is show in (schematic.gif).

There is no expectation of permanent network wiring. On the rare occasions when reconfiguring is desired the user will either walk out to the unit with a laptop or will unplug the field wiring and carrying the unit to the the computer. The concept is that the ITAP has a detachable programming head. Happily for the user, this expensive programming head is a computer he or she already owns.

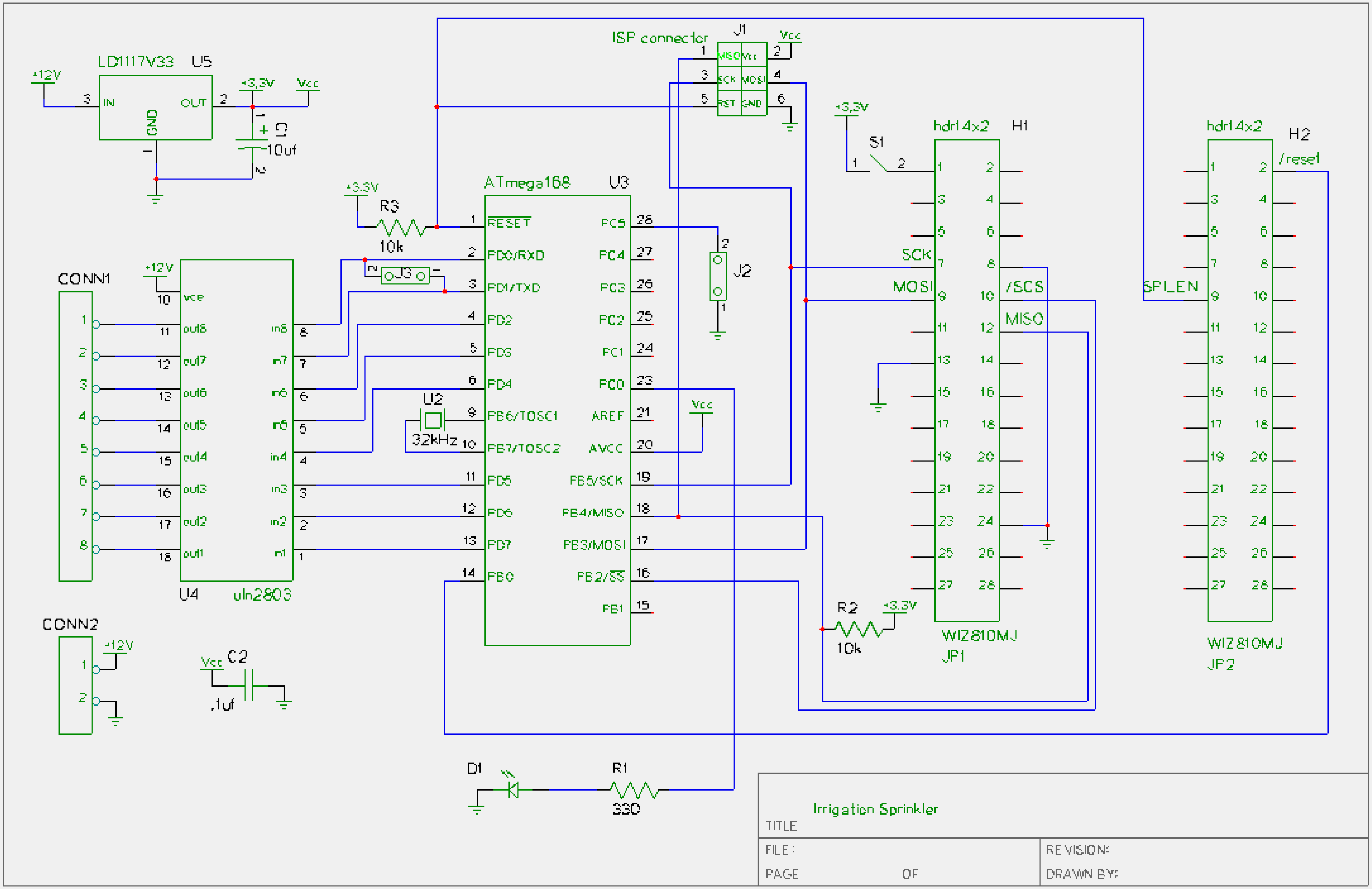
The basic design of the ITAP is shown in the block diagram (block_diagram.gif). The functionality is split between the browser-based user interface and the hardware-based web server, data model and control logic. The web server is used to read and write the internal data model. The data model is read by making specific page requests. The data model is written by specifying arguments during a general page request.



No html is generated by the firmware. All the knowledge of how the data are presented is controlled in the browser. The ITAP web server has a second role which is to return files stored in internal program memory. These files are highly compressed and are returned unmodified. The ITAP serves highly compressed files, but itself contains no compression code.

Large enterprise web development techniques, originally intended to relieve load from big servers, are used just as effectively to improve the small device's usability while at the same time reduce the memory footprint.

Planning tools present easy to read graphs of "what if" scenarios. A live view shows accurate status of the device and provides manual override controls.



TITLE Irrigation Sprinkler

FILE :

REVISION:

PAGE OF

DRAWN BY: