Web Enabled Monitoring and Control Device

Final Product Description

Dated: December 12, 2007

Author: Sam Hansen
Introduction

I present a proposal to design a device representing a web enabled GPIO device. The design of this device is independent of the end application to allow it to be used in a wide range of applications. The completed product will be a network addressable device, which runs a small web server capable of streaming sensing data to any remote client. A bi-directional interface will be implemented allowing remote monitoring as well as asserting a value at the device end.

The final product will consist of a circuit board enclosed in a sealed plastic housing with external ports for the various connectors (RJ45, RS232, power). There will also exist a small “push-pen” button for resetting the networking module to a known state. This “push-pen” button will be of the type that sits inside the plastic housing and requires a pen or similar device to properly activate. The final production model will be 2.5x4x0.75 inches in size. The following diagram shows the project components.

A Freescale MCF52235 microcontroller will be used to implement the majority of this project. This device has an integrated internal Ethernet physical interface which will provide the networking interface for the project. Additionally the microcontroller has an analog to digital converter and numerous GPIO ports, both of which will be used to implement the final solution. The ADC converter will supply the digital word representing the current state of the inputs. The GPIO ports will be bi-directional and have the capability to be either read or written to. The following diagram shows the MCU modules that will be utilized by the project.
When configured to run at 60Mhz, the MCU and peripherals have a peak current consumption of 340mA. This will be supplied by an AC wall transformer.

**Software Requirements**
The final implementation will be written in a combination of C and assembly and be compiled to run on the MCF52235 microcontroller. The uC/OS kernel will be used to provide a real time kernel on which to run the software. The final implementation will be comprised of four main software modules: network driver, network stack, io processing, and web server. The network driver is a small module specifically designed to allow the network stack to post and retrieve Ethernet frames to and from the on chip Ethernet hardware. This module is separate from the networking stack and is simply an interface to the Ethernet hardware. The networking stack will process Ethernet frames that have either originated at the Ethernet hardware, or are destined for it. I will be using the Micrium stack that has been made available. The io processing module will take on the task of reading the input pins or providing an output depending on the command received from the web server. The web server is responsible for serving out a very simple static web page embedded with the information coming from or going to the io pins.

Communications will take place between a remote web client and the web server running
on the device conforming to the HTTP/1.0 specification (RFC1945). The following diagram illustrates the data flow between the various software modules. The preceding modules are estimated to require 2K of flash memory.

The networking stack will be implemented using the OSI 7-layer model. The stack will implement TCP/IP connections as well as a client that works with the DHCP protocol to automatically assign the device networking information. The following diagram illustrates the 7 Layer model and shows which portions are embedded in the MCU and which portions will be implemented in software.

---

**OSI 7-Layer Model**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>HTTP</td>
</tr>
<tr>
<td>Presentation</td>
<td></td>
</tr>
<tr>
<td>Session</td>
<td></td>
</tr>
<tr>
<td>Transport</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>IP/DHCP</td>
</tr>
<tr>
<td>Data</td>
<td>Ethernet</td>
</tr>
<tr>
<td>Physical</td>
<td>EPHY</td>
</tr>
</tbody>
</table>

---
**User Interface**
The user interface will be presented in the form of a webpage that the user visits. The webpage will present basic configuration options as well as device IO. The configuration options presented will be the choice to configure the networking module manually, or respond to a DHCP offer. When the user selects a manual configuration, they will be required to enter the desired IP address, networking mask, and gateway address. If the DHCP option is selected no further input will be required. For the IO part of the user interface the user will be presented the option to configure the GPIO ports as either input or output. When the user selects input, the value on the input ports will be sent to the ADC converter and the digital word will be presented to the user. If the user selects output, they will be prompted for the digital value to put on each GPIO port. When the user informs the system that they are done, the digital values will be latched to the output ports.

The reset pushbutton will operate in the following manner. Upon pushing the button, the internal networking information will be reset to some known value (192.168.1.100/24) and DHCP will be disabled. This allows a static arp entry to be set to allow a host machine to access the webserver via the IP address. Once connected to the webserver the user can configure the device for the operating environment. The following diagram shows the back of the module and the interface presented to the user.

![Product Housing](image)

**Development Schedule**
I plan to follow the following schedule to complete this project (starting winter quarter).

- MCU Initialization and Setup: 1 week
- EPHY Initialization and Setup: 1 week
- Networking Driver Development: 2 weeks
- Networking Stack Development: 3 weeks
- IO Processing Module Development: 1 week
- HTTP Server Development: 3 weeks
- Alternate Demo Software: 1 week
- Final Testing: 2 weeks
- Additional Features: As time permits

For the final presentation all I require is a MCF52235DEMO module along with a computer and available network to demonstrate with. I only require a programming POD and networked workstation to progress through development. This project presents no sustainability issues.
The actual demonstration will consist of reading the accelerometer module on the demo board and presenting real time information to the user via the internet. It is a possibility to have the device in some remote location (other end of the room) to demonstrate the networking aspect of the project.

**Electrical Specifications**

Accuracy range:
- 12-BIT ADC conversion
- Binary (+5V) Output High (0V) Output Low
  - 8 Bits, 4 Input, 4 Output

Communications Protocol:
- TCP/IP (RFC795)
- HTTP/1.0 (RFC1945)
- ISCDHCP (RFC2131)

Power Requirements:
- 340mA Max Peak Current – Supplied via Wall-Wort

PCB Size: 2x3.75 DS-PCB: Surface Mounted Components
Operating Temp: -10 to 65 C (Nominal)

**Parts List**

- Freescale MCF52235  x1
- AC wall transformer  x1
- Housing  x1
- Ethernet Cable  x1