SLAC-TN-10-022

Intelligent Platform Management Interface (IPMI)

Derek S. Wung
Office of Science, Science Undergraduate Laboratory Internship Program

University of California Los Angeles

SLAC National Accelerator Laboratory
Menlo Park, California

August 14, 2009

Prepared in partial fulfillment of the requirement of the Office of Science, Department of Energy's Science Undergraduate Laboratory Internship under the direction of Ronald Johnson in the Controls Department of the SLAC National Accelerator Laboratory.

Participant: _____
Signature

Research Advisor: _____
Signature

Work supported in part by US Department of Energy contract DE-AC02-76SF00515.

Table of Contents

Abstract:

Intelligent Platform Management Interface (IPMI). DEREK S. WUNG (University of California Los Angeles, Los Angeles, CA 90024) RON JOHNSON (Controls Department, SLAC National Accelerator Laboratory, Menlo Park, CA 94025).

Intelligent Platform Management Interface (IPMI) is a set of specifications that defines interfaces between components of the same system, most notably in servers; these interfaces provide information on the status of environmental variables such as voltage, temperature, and fan status. IPMI, which has grown significantly within the past five years, is generally used in computers to keep the system in check. However, the Controls Department at the SLAC National Accelerator Laboratory wants to find a way to implement IPMI in its standalone chassis. First, it is imperative to study how IPMI is used in a computer before actually going about designing a system that implements IPMI. Next, the proper parts of the system must be determined before research can be conducted on finding these parts. In addition to finding the right chips, figuring out the proper connections between these parts is just as important. While the system wasn't constructed during the course of the project, the general process of designing the system from scratch is outlined.

Introduction:

When we buy our first computers, we hope that they don't break down shortly after we start to use them. Today's computers, however, can last for several years before malfunctioning. A set of specifications called Intelligent Platform Management Interface (IPMI) plays a major part in maintaining the health of modern-day computers. With the implementation of such an interface, computer owners need not worry much about their hard drives overheating or breaking down since IPMI helps prevent such an occurrence.

Introduced by companies from Dell to Intel, IPMI helps maintain the safety of a computer by defining interfaces to allow different parts of a computer to communicate with one another. Communication is done by way of a protocol called Inter-Integrated Circuit (I^2C), which transfers data using two lines: a Serial Clock Line (SCL) and a Serial Data Line (SDA). The SDA physically transmits the data from one location to another while the SCL determines when exactly that data is sent to avoid a massive influx of messages.

Over the past eleven years, the specifications of IPMI have evolved. In 1998, IPMI V 1.0 was introduced. Three years later, IPMI V 1.5 was released. Finally, in 2004, the specifications of the latest version of IPMI (V 2.0) were publicized on Intel's website. Numerous revisions have taken place over the last five years, but the general concept of IPMI has remained the same throughout its history.

With IPMI, one can physically monitor a computer's temperature, fan status, and power supply voltage; this can be done by utilizing downloadable software programs such as IPMItool, IPMIView, and OpenIPMI. However, not every program can be run on

every operating system, so it is necessary to check the compatibility of the software with the user's operating system before downloading the program.

One of the advantages of utilizing IPMI is the fact that it operates independently of the main processor as well as the operating system. Therefore, even if the operating system fails or if the processor shuts down, administrators can still use IPMI to diagnose and recover their systems. With IPMI, alert notifications will be sent to administrators before encountering hardware problems [5].

At the heart of the IPMI architecture lies a microcontroller called a Baseboard Management Controller (BMC), which serves as the intelligence of IPMI [8]. The BMC's main job is to help all elements communicate with one another by way of messages to other interfaces; the messages are sent via an Intelligent Platform Management Bus (IPMB), a computer bus that is based on the I^2C protocol [2]. Embedded on most motherboards, the BMC can interact with every other board component and keep track of logging information in a System Event Log (SEL), which is utilized by an administrator in the event that the main processor fails or the operating system crashes.

One of the advantages of using IPMB is that it can communicate with devices that do not support IPMI [6]. For example, most remote management cards do not typically communicate with other components via I^2C; instead, they are usually connected to an IPMB connector before interacting with the BMC. In addition, IPMB allows platform management to be extended by connecting more management controllers, which are microcontrollers that operate in a similar manner as the BMC; these additional management controllers are typically connected to other boards in the same system and are often called satellite controllers [1].

While IPMI is primarily used to monitor the health of computers, it can be used in other systems as well. The main objective was to study IPMI and determine how it could be implemented in standalone electronics chassis.

Methods:

This research project was conducted in the Controls Department at the SLAC National Accelerator Laboratory. No textbook was consulted during the entire course of the project, but the use of the internet was extensive. Websites that appeared on Google as well as the actual IPMI specifications posted on Intel's website were often referenced. Intel's specifications demonstrated IPMI in more detail than any other site.

After studying the basics of IPMI, a design of such a system was constructed. To plan this system, each individual part was researched; most of these parts were found via www.findchips.com. Information on numerous chips of the same type was gathered and compiled together in the form of datasheets; each chip was then compared and contrasted to determine which chip represented the best choice for its respective part.

At times, certain parts of the system could not be found on www.findchips.com; when that happened, a simple Google search was used. For example, the Find Chips database contained no information about remote management cards, but Google yielded some relatively decent hits about such hardware. Then, after entering the names of the hardware on www.findchips.com, the search would yield a few results.

Upon examination of all necessary chips, some of the datasheets gathered did not contain the pin-out information. Other times, it was discovered that the chips were

compatible with IPMI V 1.5, but not necessarily with IPMI V 2.0. In such events, different chips were found to make sure they could function properly in the system.

The architecture of a system that utilizes IPMI V 2.0 is shown in Figure 1. The figure not only shows the chips involved in the system, but it also shows how each chip interacts with the BMC. When all the chips, as well as their interfaces with the BMC, were gathered, it was time to create a block diagram. A simple sketch was drawn by hand initially to get a rough idea of what the system would look like in real life. Then, a final block diagram of the necessary hardware was developed.

Results:

Upon gathering various datasheets of chips from BMCs to numerous sensors, the chips that seemed most promising were chosen. In addition to the BMC, the system design contained sensors that monitor temperature, fan status, and voltage. While the construction of the actual design won't commence until after the project, its foundation was established.

Initially, a PC87431M mini-BMC was found, but because no pinout information was found on its datasheet, it could not be trusted. After additional research, the Renesas H8S/2167 BMC (see Figure 2) was found. This BMC contained 144 pins, of which twelve were used as I^2C ports. Because an I^2C port requires two pins for the SCL and SDA lines, up to six components can be connected to the H8S/2167 via I^2C. The H8S/2167 also contains three sets of two pins each that transmit and receive data from other components using different communication protocols.

A non-volatile storage chip retains data in a system when power is shut off; if manufactured properly, today's chips can hold data for up to ten years. Data that can be stored in non-volatile storage chips include Sensor Data Records (SDRs), the System Event Log (SEL), and Field Replaceable Unit (FRU) inventories. SDRs typically contain information about the sensors in the system; these records are kept in an SDR repository, which is managed by the BMC [7]. The SEL keeps track of logging information in the event that the main processor crashes or the operating system fails. Field Replaceable Units are interchangeable parts of a system; in other words, an FRU can easily be replaced in case it malfunctions. After researching different chips for NV storage, the PCF8582C-2 (see Figure 3) was found to be optimal in the design mainly because it has a serial input/output I^2C bus, meaning it has SCL and SDA pins. The PCF8582C-2 can also hold up to 256 x 8 bits of non-volatile storage.

A set of three board temperature sensors from Analog Devices was found: the AD7416, AD7417, and the AD7418. Although each sensor can be connected via I^2C, the AD7416 (see Figure 4) was chosen because of its simplicity and low number of pins. Whereas the AD7416 and AD7418 each have eight pins, the AD7417 has sixteen pins. In addition, the AD7418 had seemingly unnecessary pins for the matter at hand. These sensors can withstand temperatures between -40° C and 125° C.

The MAX1617A (see Figure 5) is a remote diode temperature sensor that has an SMBus two-wire serial interface; SMBus is a derivation of I^2C, so the sensor also has clock and data pins. In addition to remote temperatures, the MAX1617A can also measure local temperatures and send out alarms in events of abnormally low or high temperatures.

It was far from easy to search for a suitable voltage sensor, but one chip from Texas Instruments was discovered to sense voltages from 0 to 26 volts. The INA209EVM (see Figure 6) can also monitor current and power in a system; the device can report readings with uncertainties of at most one percent. It also has serial clock and data lines for connections via SMBus.

The ADT7473 fan controller (see Figure 7) can manage up to four fans simultaneously. Like the other sensors aforementioned, this sensor can be connected to the BMC via SMBus. In addition, the ADT7473 has two remote temperature sensors as well as an on-chip temperature sensor that can measure its own internal temperature; the chip can measure temperatures as high as 191° C.

A remote management card allows administrators to manage their servers from any spot in the world. The MegaRAC G4 (see Figure 8) is one of the few remote management cards that can operate alongside IPMI V 2.0. Unlike most other remote management cards, the G4 can directly connect to the BMC via I^2C, but it does not depend on the BMC to manage system power if it has a special Feature Connector.

Finally, a serial/modem cable as well as a LAN Network controller was needed to serve as interfaces for IPMI messaging, which is how elements of a system communicate with one another. The Microchip ENC28J60 (see Figure 9) is an Ethernet controller that does not communicate via I^2C; instead, it communicates via Serial Peripheral Interface (SPI), a four-wire serial bus. To connect the ENC28J60 to the BMC, an SC18IS602IPW from NXP (see Figure 10) was needed to convert I^2C to SPI and allow for a connection between the ENC28J60 and the H8S/2167.

Unlike all of the previous components mentioned, the Serial/Modem cable does not connect via I^2C, SMBus, or SPI. Instead, the cable is connected by a different protocol called RS-232. RS-232 serves as two-line interface, similar to the I^2C interface. However, the two lines involved in the RS-232 standard are Transmit (Tx) and Receive (Rx). The HN-210 (see Figure 11) is one cable that is actually compatible with this standard.

When everything is put together, there are a total of seven chips that require I^2C connections to the BMC. Because the Renesas H8S/2167 has only six available I^2C ports, it is necessary to find an I^2C hub that allows multiple devices to connect to the same port. The PCA9518 (see Figure 12) is an expandable hub that can take up to five connections via I^2C; it also has two sets of two expansion pins each that allow two devices on the hub to run at the same time. In this design, only one set of expansion pins are needed.

Finally, the design of the system was created using DxDesigner. The design on DxDesigner can be seen in Figure 13; a simplified version of the design can be seen in Figure 14. It is not certain whether or not this specific system design will work. However, any chip that is discovered to be unsuitable for the design can easily be replaced at a relatively low cost. The fact that these parts are interchangeable ensures reliability, availability, and serviceability in today's systems [3].

In addition to microcontrollers, Field-Programmable Gate Arrays (FPGAs) can be incorporated in the system. FPGAs are more flexible than regular microcontrollers; whereas microcontrollers have fixed pin configurations, FPGAs have programmable pins. In other words, FPGAs can be programmed for extended capabilities during the construction of the design [9].

There exist software programs that let users keep an eye on the environmental variables of their system; such software can be downloaded for free online. IPMIView, for example, is a program written in Java that allows users to communicate with the BMC via Ethernet [4]. A screenshot of IPMIView can be seen in Figure 15. Most of the programs found online, however, aren't compatible with Windows, so it is desirable to use another operating system like Linux to monitor these variables. Figure 16 displays a screenshot of OpenIPMI while Figure 17 displays five different screenshots of IPMItool.

Conclusion:

This project not only accomplished the objective of designing such a system, but it also portrayed the general process of accomplishing such a task from scratch.

Upon designing a system, it was not only important to find the right chips, but it was also important to determine how each device was connected to one another since IPMI is a standard. By researching that necessary information, it gave me an idea of what chips could be used.

This project also allowed for open collaboration among colleagues as it posed numerous problems during the research and design processes. Several aspects of the project were difficult to solve at first, but the help of fellow workers made the problem much easier to solve.

Acknowledgments:

First and foremost, I would like to thank the U.S. Department of Energy, Office of Science, and the Science Undergraduate Laboratory Internship (SULI) program for funding this wonderful research experience at the Stanford Linear Accelerator Center (SLAC). A special thanks goes to Susie Zheng for allowing me to use her office space to conduct my research as well as guiding me during the entire course of the project. Additional thanks goes to my mentor Ronald Johnson for his time and effort into helping me with the study of IPMI. I would also like to thank my fellow workers for creating a friendly working atmosphere around my office space: Jeff Olsen, Andrew Young, Dave Anderson, Dave Brown, John Dusatko, Kazuko Onaga, Evgeny Medvedko, Chuck Yee, Vernon Smith, Deborah Lilly, and Kenneth Leung. I would finally like to thank the following for running the SULI program: Program Director Stephen Rock, Residence Assistant Howard Young, Program Manager SueVon Gee, and Program Administrators Vivian Lee and Elizabeth Smith.

References:

[1]: Intel, Hewlett-Packard, NEC, and Dell Computer Co. Intelligent Platform Management Interface Specification Second Generation, V 2.0, Feb. 2004.

[2]: "Intelligent Platform Management Interface," [Online Document], [2009 Jul 7], Available at HTTP: http://en.wikipedia.org/wiki/IPMI

[3]: K. Dua, "Intelligent Platform Management," [Online Document] (Oct. 2005), [2009 Jul 6], Available at HTTP: http://pcquest.ciol.com/content/search/showarticle.asp?artid=75814

[4]: A. Singh, D. Holmgren, R. Rechenmacher, and S. Epsteyn, "Tools and Techniques for Managing Clusters for SciDAC Lattice QCD at Fermilab," Computing in High Energy and Nuclear Physics, Mar., pp. 24-28, 2003.

[5]: "IPMI Management," [Online Document] (2009), [2009 Jul 7], Available at HTTP: http://www.opengear.com/SP-IPMI.html

[6]: B. Suppanz, "Intelligent Platform Starts with Intelligent Power," [Online Document] (Jul. 2006), Available at HTTP: http://www.power-one.com/technical/articles/IPMI.pdf

[7]: Y. Fang, G. Kochhar, and R. DeRoeck, "High-Performance Computing Clusters with IPMI," Dell Power Solutions, Oct., pp. 58-63, 2004.

[8]: H. Zhou, J. Yin, and A. Rao, "Remote Management with the Baseboard Management Controller in Eighth-Generation Dell PowerEdge Servers," Dell Power Solutions, Oct., pp. 26-29, 2004.

[9]: “Actel Fusion FPGAs Supporting Intelligent Peripheral Management Interface (IPMI) Applications,” [Online Document] (Oct. 2006), Available at HTTP: http://www.actel.com/documents/Fusion_IPMI_AN.pdf

Figures:

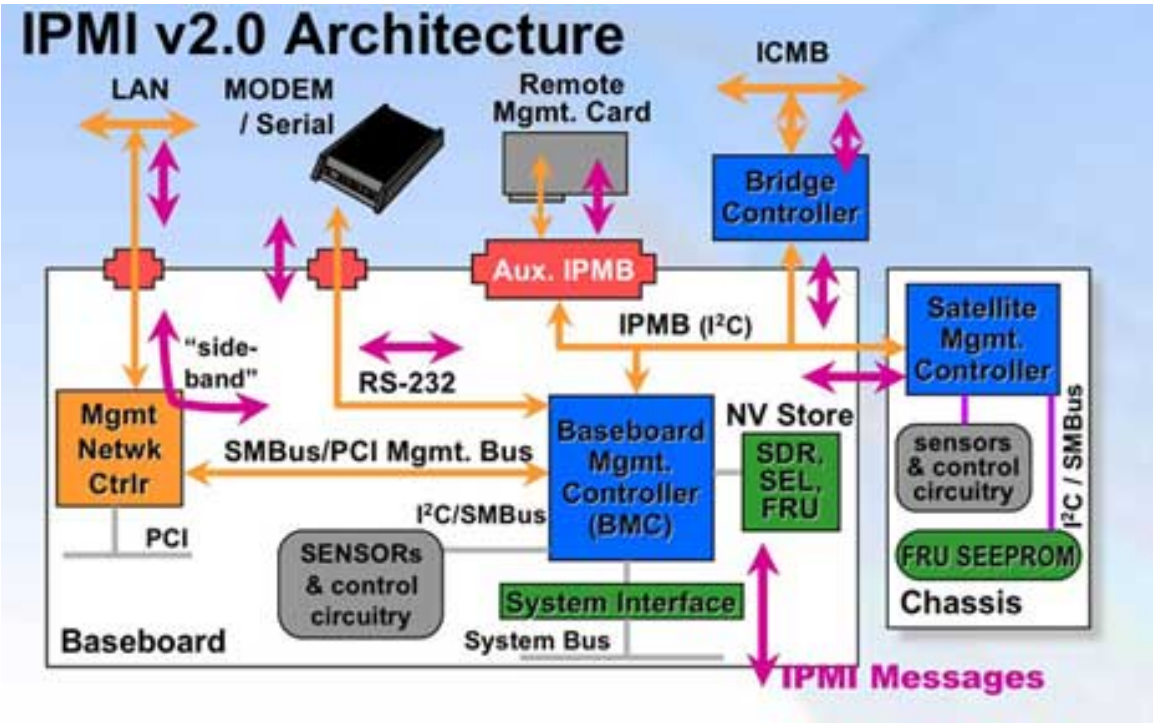

Figure 1: A block diagram of a system that utilizes IPMI. The Baseboard Management Controller (BMC) acts as the heart of this system because it interacts with every other device. The bridge controller was not included in the design because the objective is to work with just one chassis. This design is typically used on computer motherboards, so the actual design of the system in question may vary.

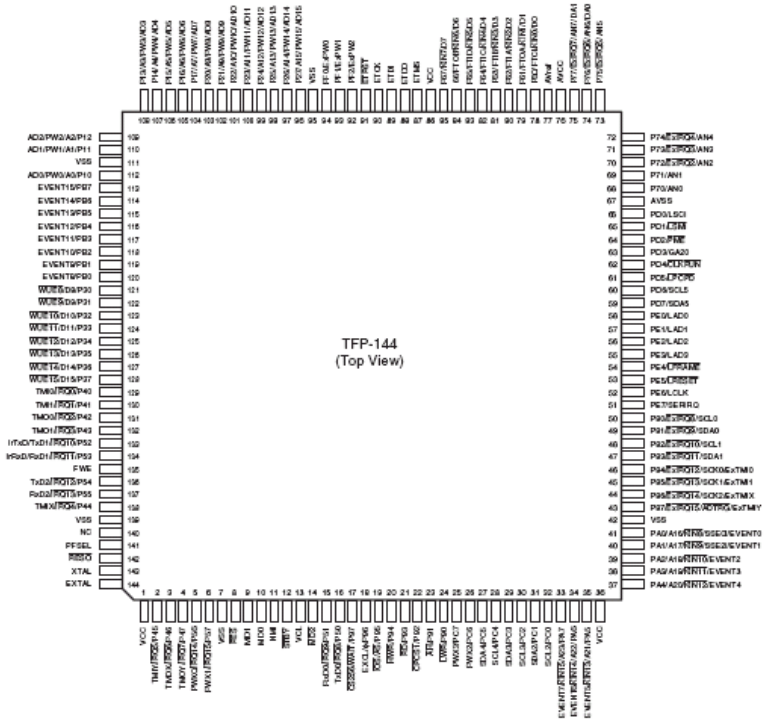

Figure 2: Pinout information of the Renesas H8S/2167 BMC. The microcontroller has a total of six I^2C ports and three ports for receiving and transmitting data.

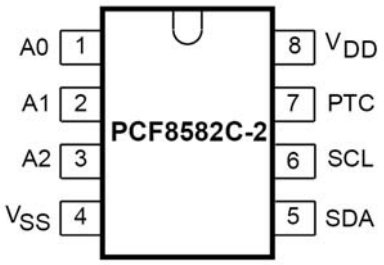

Figure 3: Phillips PCF8582C-2 Non-Volatile Storage

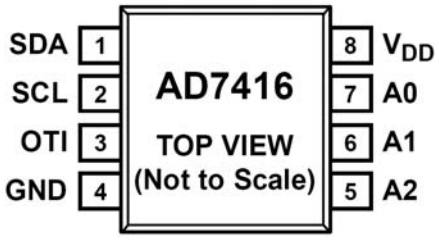

Figure 4: AD7416 Board Temperature Sensor

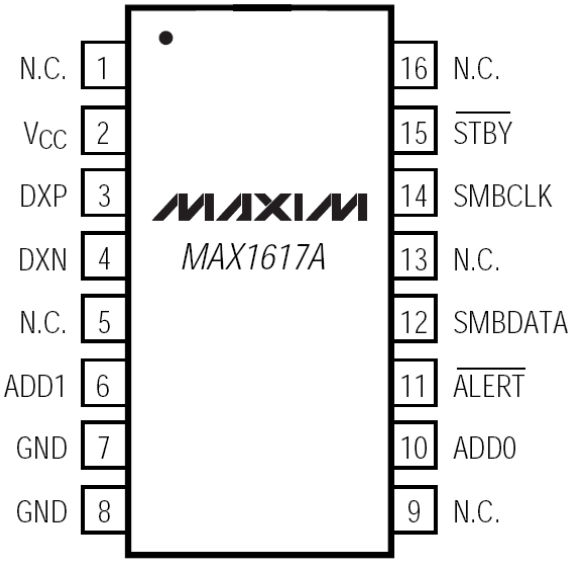

Figure 5: MAX1617A Remote Diode Temperature Sensor

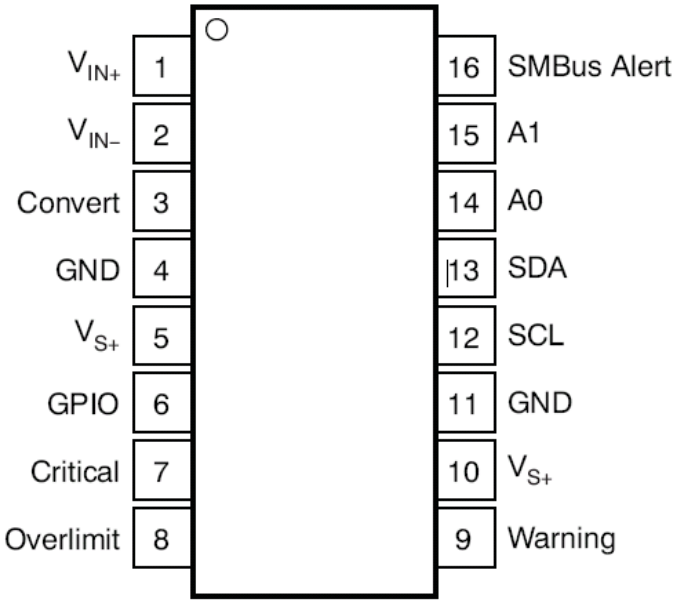

Figure 6: Texas Instruments INA209EVM Voltage Sensor

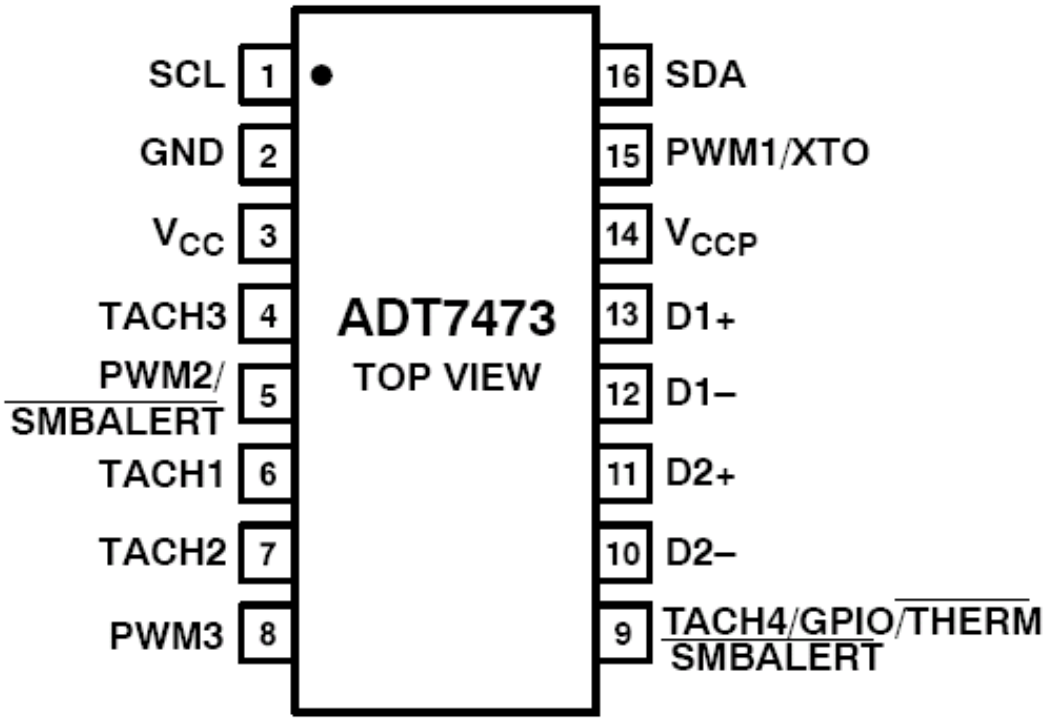

Figure 7: ADT7473 Fan Sensor

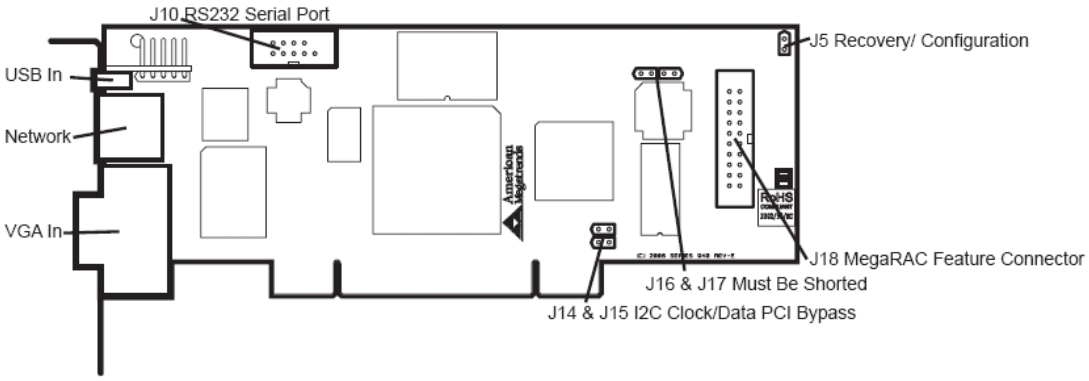

Figure 8: MegaRAC G4 Remote Management Card

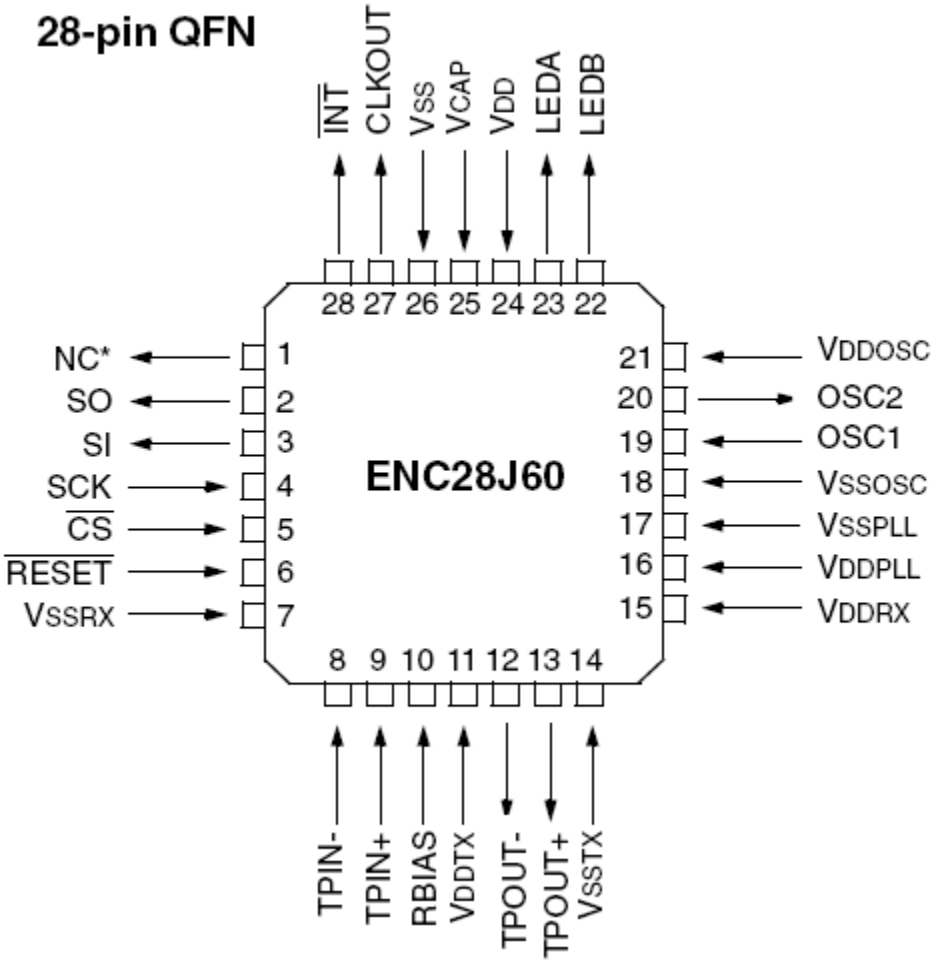

Figure 9: Microchip ENC28J60 Ethernet Controller

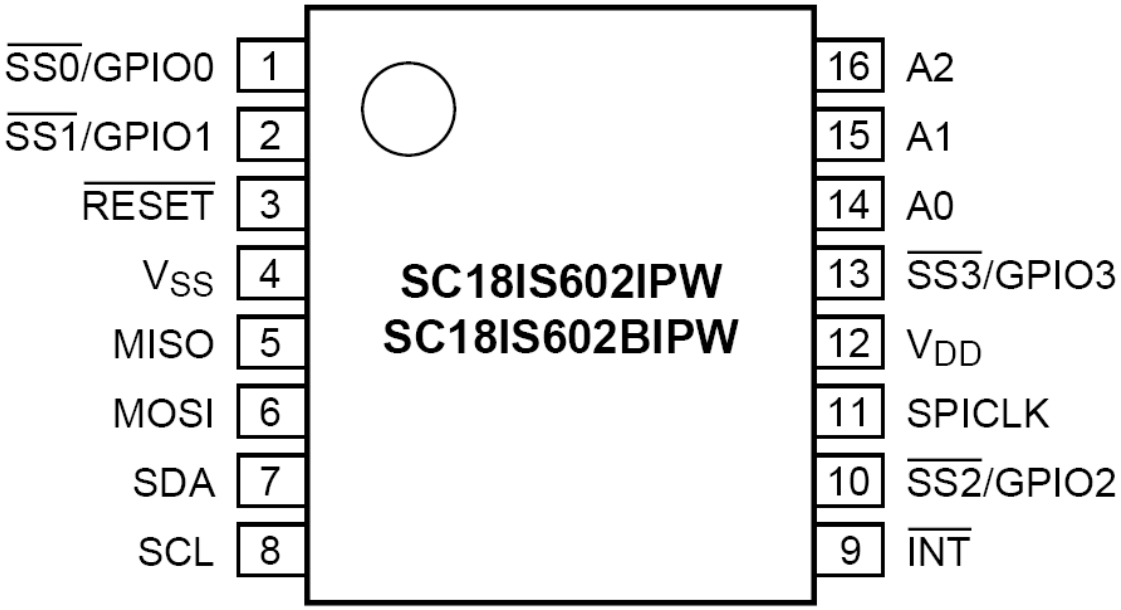

Figure 10: NXP SC18IS602IPW I^2C-bus to SPI Bridge

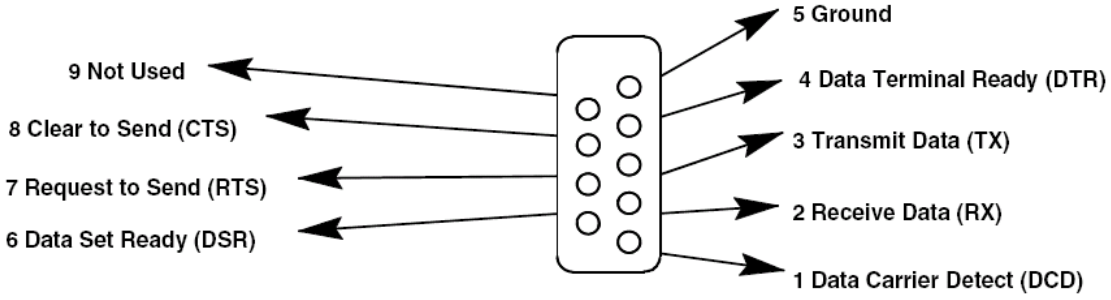

Figure 11: HN-210 Serial/Modem Cable

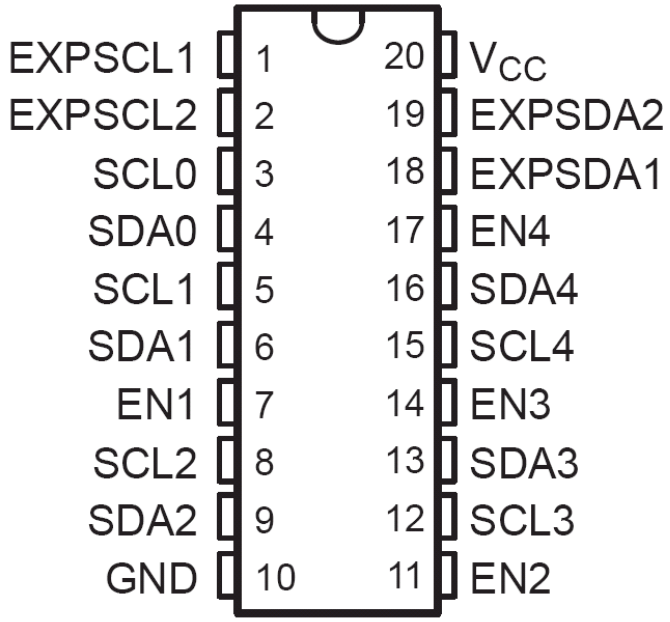

Figure 12: PCA9518 I^2C Hub

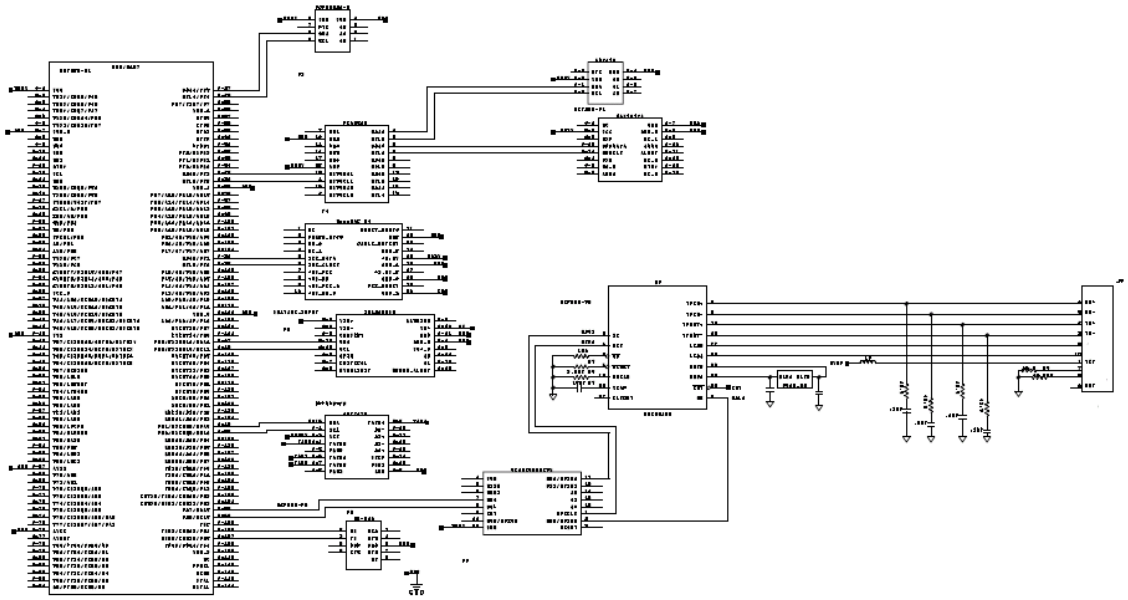

Figure 13: Block Diagram of System Design. Created using DxDesigner

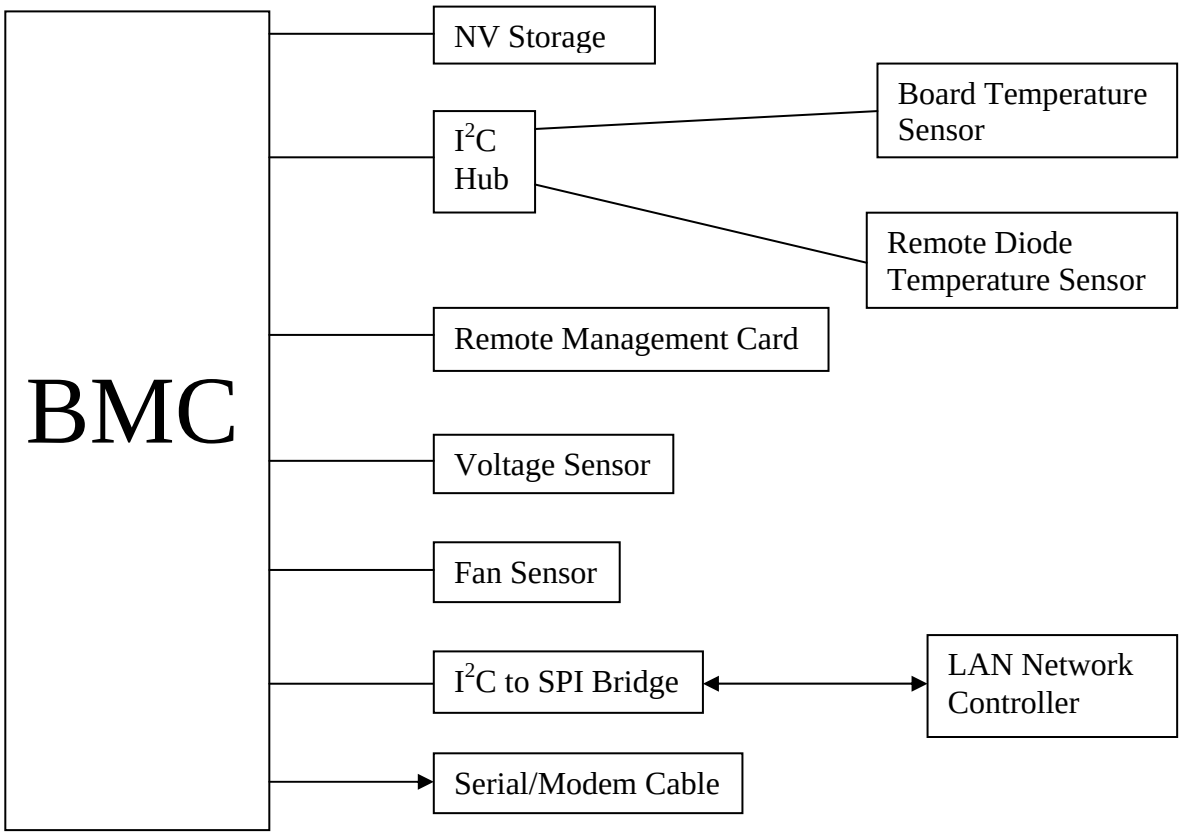

Figure 14: Simplified block diagram of Figure 13. In this diagram, segments depict I^2C connections, rays depict RS-232 connections, and lines represent SPI connections.

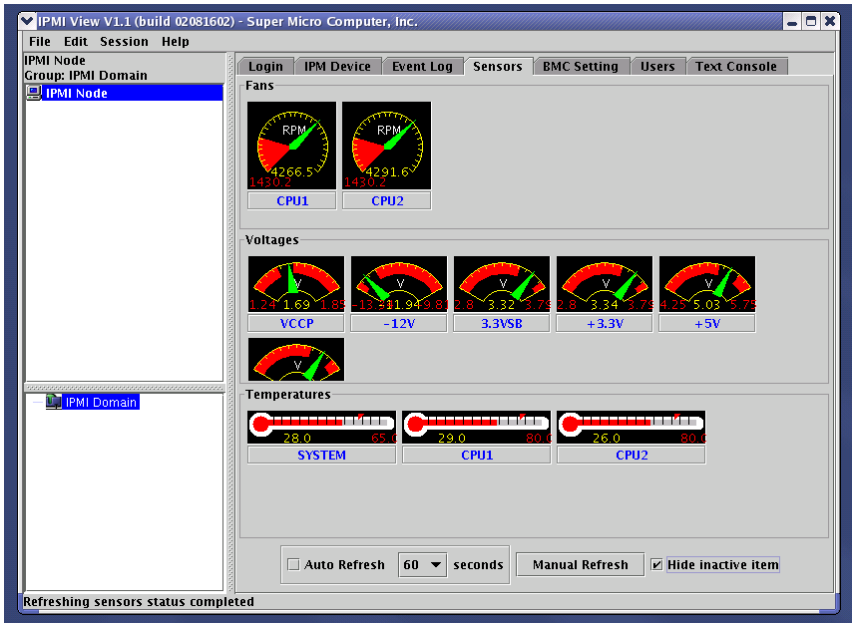

Figure 15: Screenshot of IPMIView

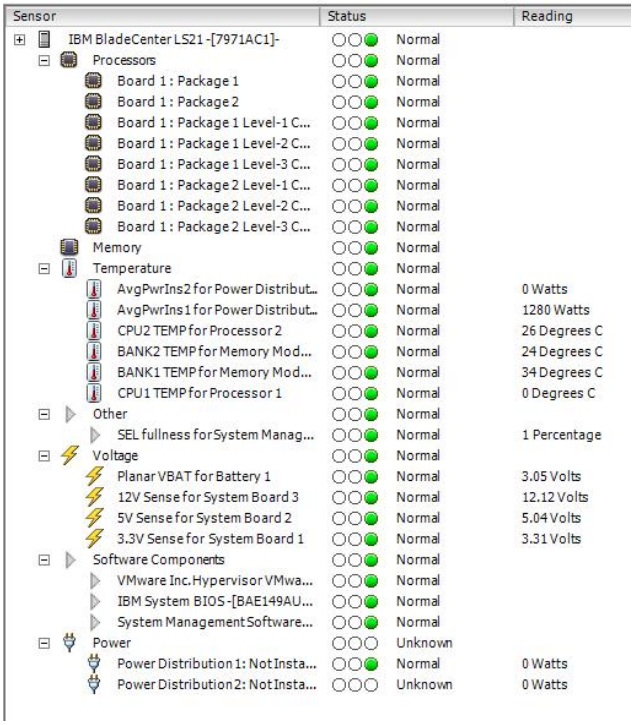

Figure 16: Screenshot of OpenIPMI

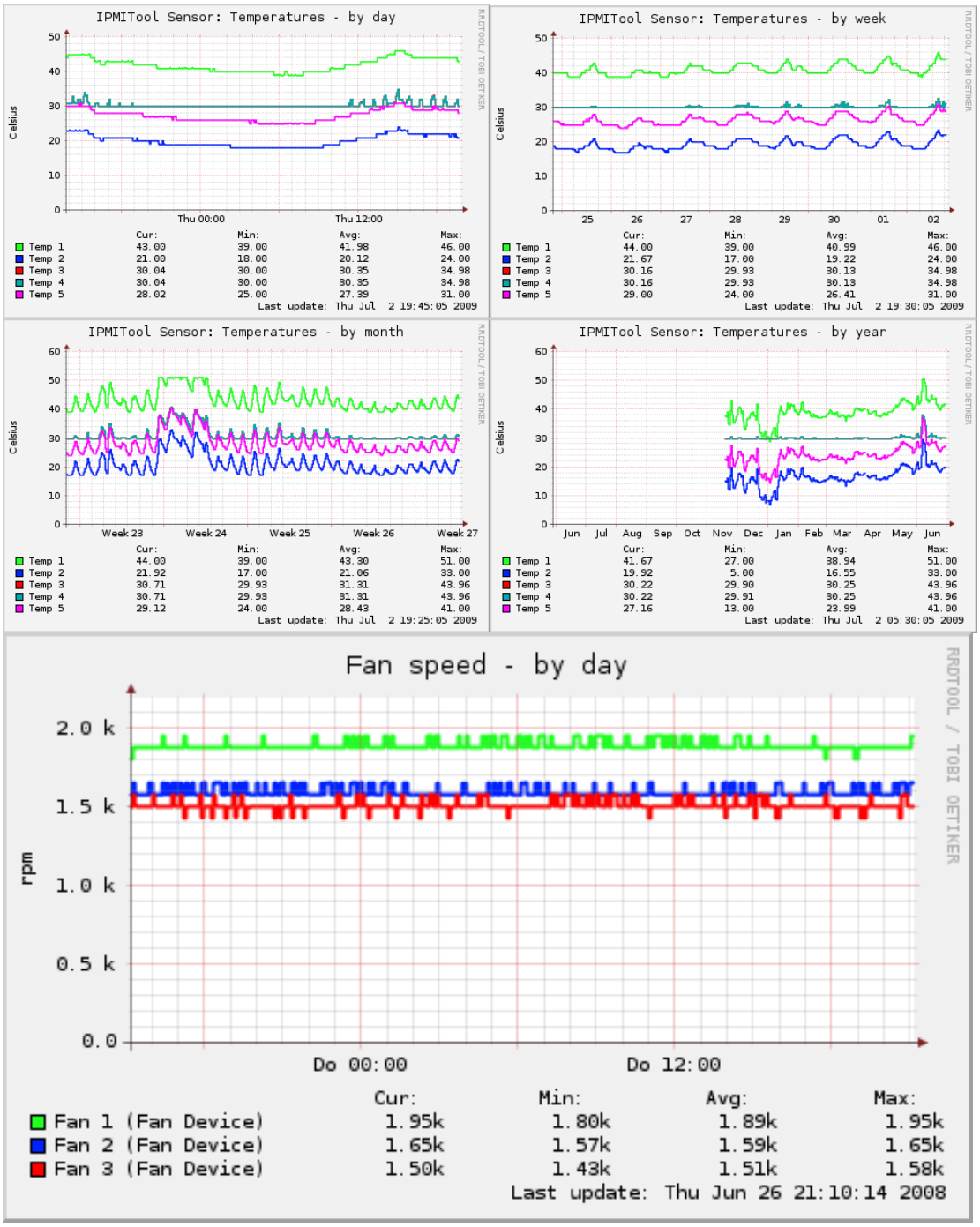

Figure 17: Five different screenshots of IPMItool