ISSN: 2277-9655
Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

ATMEL AVR Based 32-BITS Data

Ms. Shanila Mahreen
Assistant Professor in Electronic & Communication Engineering at Nawab Shah College Of Engineering & Technology (Affliated to JNTUH), Malekpet, Hyderabad-500024, A.P, India

Abstract

In Embedded web technology a smart devices are controlled from a remote location via internet, through a web server. A client which may be a desktop computer or a mobile phone with the web browser, connected to the internet can control and monitor all actions of the smart device with special permissions and authentication. The combination of smart device and web server is Internet appliance. Handling of heavy machinery equipments in several places like, sugar industries, cotton mills, and substation is strenuous task. Collecting information of such heavy equipments like, noting down their operating parameters, their operating conditions is risky. Hence an effective system has been evolved to overcome such difficulties, which is known as SCADA.

SCADA abbreviation is supervisory control and data acquisition. It has data acquisition feature as major role than controlling. It is mostly operated through Internet. The purpose of SCADA is to provide information regarding the equipments attached to it. It is the new technology evolved to overcome problems regarding data collection, decreasing human labor making dynamic decisions, communication with other substations. SCADA is used in generating stations and substations to communicate with other substations and control center via satellite. It's application in substation; is being presented in this project. The threats that it is undergoing are also being presented. The protective measurements involved to protect it are clearly elucidated.

Keywords: SCADA, INTERNET,SATELLITE.

Introduction

Protocols such as TCP/IP, UDP, DHCP and ICMP form the backbone of internet communications a large bulk of which consists of Hyper Text Transfer Protocol (HTTP) traffic for the World Wide Web.

A HTTP or web server is a server process running at a web site which sends out web pages in response to HTTP requests from remote browsers. While high performance 32 bit desktop computers are used for serving websites, much smaller and cheaper 8 or 16 bit microcontrollers, though not as powerful in terms of processing power, can do the job as well. This report details the workings of the embedded web server built for the project.

The ATMega32 controller, both being versatile and adequate in terms of capability was chosen for this project. Building a HTTP server involved implementing several protocols, namely, UDP, TCP/IP, DHCP and ARP. ICMP was also implemented for testing.

A Microchip based ENC28J60 Ethernet controller chip was used to interface the microcontroller with Ethernet. A RJ45 Ethernet jack was used to connect the Ethernet controller to a router. ATmega32 interfaces to the Ethernet controller using SPI communication.

The web server should be able to fulfil the following requirements:

- Send and receive Ethernet packets
- Differentiate between and respond to ARP and IP packets
- Respond to a ping (ICMP)
- Send and receive TCP and UDP packets
- Perform the appropriate checksums and acknowledgements for TCP
- Have enough TCP functionality to serve web pages.
- Have enough versatility such that another user can change and modify the webpage or add web pages.

The 8-bit ATMEL Mega32 was chosen for this project since it has a sizable amount of SRAM (2kb) and Flash (32kb) and is one of the more current microcontrollers in the market. It also had an inexpensive price tag and came with comprehensive documentation and software support.

ISSN: 2277-9655
Impact Factor: 1.852

This paper shows a miniature SCADA implementation on the Web Server by showing a Voltage variation using a Potentiometer, a Temperature Sensor LM35 and a ON/Off control from the Web Page served up by the Web server. These Webpages can be viewed by any Internet Browser client.

Architecture

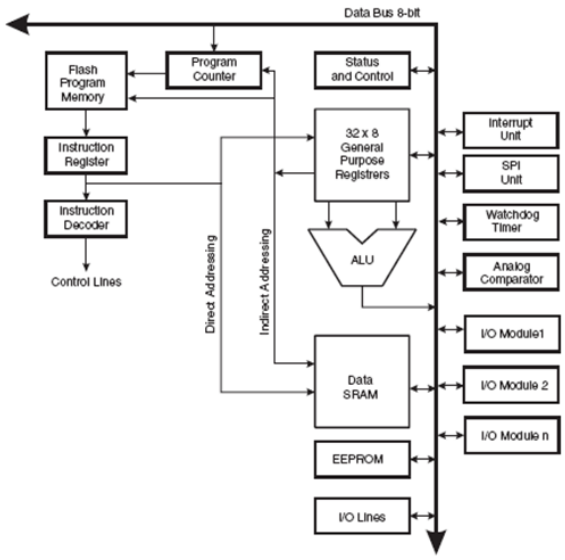

Fig 1 . AVR Architecture

I/O Ports: All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both VCC and Ground

A lower case "x" represents the numbering letter for the port, and a lower case "n" represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. i.e., PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn.

Three I/O memory address locations are allocated for each port, one each for the Data Register – PORTx, Data Direction Register – DDRx, and the Port Input Pins – PINx. The Port Input Pins I/O location is read only, while the Data Register and the Data Direction Register are read/write. In addition, the Pull-up Disable – PUD bit in SFIOR disables the pull-up function for all pins in all ports when set.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b11) as an intermediate.

Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx Register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activate d. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when a reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled ({DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedant environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the SFIOR Register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b11) as an intermediate step.

ISSN: 2277-9655
Impact Factor: 1.852

DDxn	PORTxn	PUD (In SFIOR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

Table 1 Port pin configurations

Paper Objectives

- Implementation of a miniature Web Server on a Microcontroller
- Utilization of this Web Server in SCADA implementation
- Providing the access control by WPA2 encryption algorithm.

Paper Scope

This paper shows the SCADA (Supervisory Control and Data Acquisition) implementation using a Web Server on Microcontroller by showing some analog quantities like Voltage/Temperature acquisition and real time display of these monitored quantities on a Web page, and also a Digital control of signal from the Web Page served up by the Web server. These Web pages can be viewed by any Internet Browser client.

Though signals shown in this case is only a Few, but can be expanded as required by the SCADA system with the only limitation being the I/O pins on the microcontroller being used.

This implementation using the Web Server has got some of the best advantages. Most of the IEC/ANSI protocols are now getting revised to Ethernet based systems as Ethernet based systems get more reliable and faster with lesser wiring required. Added to this is the remote monitoring capability through well established LAN/Internet based networks. In Distributed Control systems used in Plant Area wide networks it makes perfect sense to add Web Server capability to non-critical Distributed Control Modules which can be easily monitored and controlled on LAN based networks

SCADA Overview

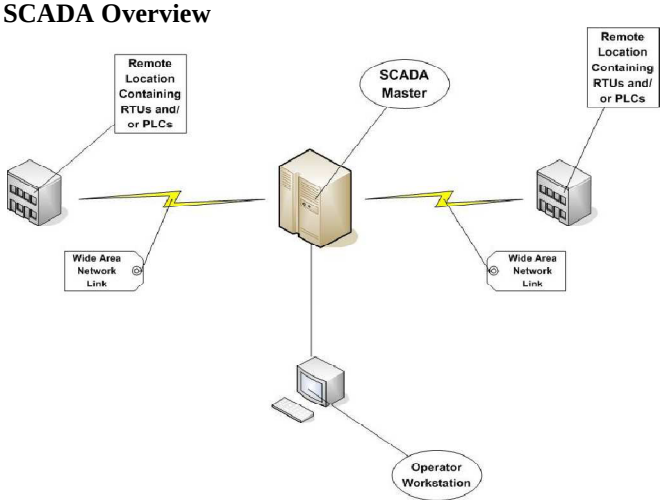

Fig 2 Typical SCADA System

SCADA is not a specific technology, but a type of application. SCADA stands for Supervisory Control and Data Acquisition — any application that gets data about a system in order to control that system is a SCADA application.

A SCADA application has two elements:

1. The process/system/machinery you want to monitor and control — this can be a power plant, a water system, a network, a system of traffic lights, or anything else.
2. A network of intelligent devices that interfaces with the first system through sensors and control outputs. This network, which is the SCADA system, gives you the ability to measure and control specific elements of the first system. You can build SCADA system using several different kinds of technologies and protocols.

SCADA can be used to manage any kind of equipment. Typically, SCADA systems are used to automate complex industrial processes where human control is impractical — systems where there are more control factors, and more fast-moving control factors, than human beings can comfortably manage. Around the world, SCADA systems control:

- Electric power generation, transmission and distribution: Electric utilities use SCADA systems to detect current flow and line voltage, to monitor the operation of circuit breakers, and to take sections of the power grid online or offline.
- Water and sewage: State and municipal water utilities use SCADA to monitor and regulate water flow, reservoir levels, pipe pressure and other factors.

- Buildings, facilities and environments: Facility managers use SCADA to control HVAC, refrigeration units, lighting and entry systems.
- Manufacturing: SCADA systems manage parts inventories for just-in-time manufacturing, regulate industrial automation and robots, and monitor process and quality control.
- Mass transit: Transit authorities use SCADA to regulate electricity to subways, trams and trolleybuses; to automate traffic signals for rail systems; to track and locate trains and buses; and to control railroad crossing gates.
- Traffic signals: SCADA regulates traffic lights, controls traffic flow and detects out-of-order signals.

SCADA is used in nearly every industry and public infrastructure project — anywhere where automation increases efficiency What's more, these examples don't show how deep and complex SCADA data can be. In every industry, managers need to control multiple factors and the interactions between those factors. SCADA systems provide the sensing capabilities and the computational power to track everything that's relevant to your operations. Real-Time Monitoring and Control Increases Efficiency and Maximizes Profitability.

Here are few of the things that can be done with the information and control capabilities you get from a SCADA system:

- Access quantitative measurements of important processes
- both immediately and over time
- Detect and correct problems as soon as they begin
- Measure trends over time
- Discover and eliminate bottlenecks and inefficiencies
- Control larger and more complex processes with a smaller, less specialized.

A SCADA system gives you the power to fine-tune your knowledge of your systems. You can place sensors and controls at every critical point in your managed process (and as SCADA technology improves, you can put sensors in more and more places). As you monitor more things, you have a more detailed view of your operations — and most important, it's all in real time. So even for very complex manufacturing processes, large electrical plants, etc., you can have an eagle-eye view of every event while it's happening — and that means you have a knowledge base from which to correct errors and improve efficiency. With SCADA, you can do more, at less cost, providing a direct increase in profitability.

A SCADA system performs four functions:

1. Data acquisition
2. Networked data communication
3. Data presentation
4. Control

These functions are performed by four kinds of SCADA components:

1. Sensors (either digital or analog) and control relays that directly interface with the managed system.
2. Remote telemetry units (RTUs). These are small computerized units deployed in the field at specific sites and locations. RTUs serve as local collection points for gathering reports from sensors and delivering commands to control relays.
3. SCADA master units. These are larger computer consoles that serve as the central processor for the SCADA system. Master units provide a human interface to the system and automatically regulate the managed system in response to sensor inputs.
4. The communications network that connects the SCADA master unit to the RTUs in the field.

The Simplest SCADA System

The simplest possible SCADA system would be a single circuit that notifies you of one event. Imagine a fabrication machine that produces widgets. Every time the machine finishes a widget, it activates a switch. The switch turns on a light on a panel, which tells a human operator that a widget has been completed. Obviously, a real SCADA system does more than this simple model. But the principle is the same. A full-scale SCADA system just monitors more stuff over greater distances. Let's look at what is added to our simple model to create a full scale SCADA system:

Data Acquisition

First, the systems you need to monitor are much more complex than just one machine with one output. So a real-life SCADA system needs to monitor hundreds or thousands of sensors. Some sensors measure inputs into the system (for example, water flowing into a reservoir), and some sensors measure outputs(like valve pressure as water is released from the reservoir).Some of those sensors measure simple events that can be detected by a straightforward on/off switch, called a discrete input (or digital input). For example, in our simple model of the widget fabricator, the switch that turns on the

light would be a discrete input. In real life, discrete inputs are used to measure simple states, like whether equipment is on or off, or tripwire alarms, like a power failure at a critical facility. Some sensors measure more complex situations where exact measurement is important. These are analog sensors, which can detect continuous changes in a voltage or current input. Analog sensors are used to track fluid levels in tanks, voltage levels in batteries, temperature and other factors that can be measured in a continuous range of input. For most analog factors, there is a normal range defined by a bottom and top level. For example, you may want the temperature in a server room to stay between 60 and 85 degrees Fahrenheit. If the temperature goes above or below this range, it will trigger a threshold alarm. In more advanced systems, there are four threshold alarms for analog sensors, defining Major Under, Minor Under, Minor Over and Major Over alarms.

Data Communication

In our simple model of the widget fabricator, the "network" is just the wire leading from the switch to the panel light. In real life, you want to be able to monitor multiple systems from a central location, so you need a communications network to transport all the data collected from your sensors.

Early SCADA networks communicated over radio, modem or dedicated serial lines. Today the trend is to put SCADA data on Ethernet and IP over SONET. For security reasons, SCADA data should be kept on closed LAN/WANs without exposing sensitive data to the open Internet. Real SCADA systems don't communicate with just simple electrical signals, either. SCADA data is encoded in protocol format. Older SCADA systems depended on closed proprietary protocols, but today the trend is to open, standard protocols and protocol mediation. Sensors and control relays are very simple electric devices that can't generate or interpret protocol communication on their own. Therefore the remote telemetry unit (RTU) is needed to provide an interface between the sensors and the SCADA network. The RTU encodes sensor inputs into protocol format and forwards them to the SCADA master; in turn, the RTU receives control commands in protocol format from the master and transmits electrical signals to the appropriate control relays.

Data Presentation

The only display element in our model SCADA system is the light that comes on when the switch is activated. This obviously won't do on a large scale — you can't track a light board of a thousand separate lights, and you don't want to pay someone simply to watch a light board, either. Areal SCADA system reports to human operators over a specialized computer that is variously called a master station, an HMI (Human-Machine Interface) or an HCI (Human-Computer Interface).

The SCADA master station has several different functions. The master continuously monitors all sensors and alerts the operator when there is an "alarm" — that is, when a control factor is operating outside what is defined as its normal operation. The master presents a comprehensive view of the entire managed system, and presents more detail in response to user requests. The master also performs data processing on information gathered from sensors — it maintains report logs and summarizes historical trends.

Conclusion and Future Work

This paper required detailed and extensive knowledge about the workings of computer networks as well as internet protocols. It also required some expertise in C programming. Having no prior knowledge of the former and mediocre experience in the latter, I was fortunate to have access to many reference books on the internet and source code. The fact that the RFCs were open source and easily available on the internet was also a great help in this project. The initial phase of the project was simply to familiarize myself with the internet protocols, computer networks, as well as programming in C. Despite several setbacks encountered early in the project such as failed attempts to integrate the Ethernet driver for the Mega32, I was able to start writing my own code, using an open source barebones Ethernet driver code. Debugging the project without having an actual internet interface was very difficult since it was impossible to know what the microcontroller was doing with the packets. Connecting and debugging the hardware took a considerable amount of time since there was very little documentation. Once that was done, I could use Ethereal (packet analyzer) to check and debug the packets that were being assembled in the microcontroller. Future improvements to the webserver could include a full implementation of TCP as well as more WebPages and perhaps even picture and graph files. External flash memory could also be added to the web server for added storage space. There might also be other innovative applications for this embedded web servers that could be implemented. Through this project, I have gained immense knowledge and familiarity with internet protocols such as TCP/IP since I had to actually write code that executes the protocol. This project has also given me a glimpse on the workings of computer networks although that aspect of the web server was almost wholly handled by the Ethernet controller.

ISSN: 2277-9655
Impact Factor: 1.852

In this paper I concentrated on authenticated based services. In future this can be upgraded to fully encrypted level by putting code both in the server and browser system.
The main limitation of the wide area network is the undefined delays in the packet transaction. This project is based entirely on the wide area network; this is also one of the constraints.

References

[1] *Research and Implementation of Embedded Web Server (paper published in IEEE computer society in the year 2008 by Mr. Zhan mei-qiong from Liaoning Technical University and Mr. Ji chang-peng from Liaoning Technical University, Huludao, China).*
[2] *Behrouz A. Forouzan, "Digital Communications and Networking", Second Edition, published by Tata Mc Graw-Hill, New Delhi, [2001].*
[3] *John L. Hennessy and David A. Patterson, "Computer Architecture – A Quantitative Approach", Third Edition, published by Morgan Kaufmann, USA, [2003].*