HOME AUTOMATION USING TINIm400 (DS80C400)

Authors: Mrs.Y.Naga Supraja,Associate Professor,Aurora Technological and research Institute,
Mr.S.Srinivas Rao, Professor, Aurora Technological and research Institute,

Abstract

In India power consumption is very costly. Now-a-days life has become mechanical. Most of us don't have time to check or forget whether we have switched off the power consumption devices. This results in a lot of power loss. After all, as they say, power saved is power created. If we can give a provision to control the appliances through Internet then we can save power. Apart from this we get some luxury benefits like we can cool our house before we reach home by switching on the AC 10 minutes before we reach home. This can be done by switching on Air conditioner before entering the house through the Internet. The next version of the Internet IPv6 allows us to have a 128 bit IP address for any network device. That means we can have 2^{128} devices. That is, IPv6 provides nearly 600 quadrillion addresses for every square millimetre on earth. That's 6x1023 addresses for every square meter of the earth's surface. So we can have 600 quadrillion devices for every square millimetre on earth. In this paper we will look at a way to control these devices

Keywords: TINI-**Tiny InterNet Interface,home automation**

Introduction

Internet Protocol version 4 was mainly designed to allow computing devices to communicate with each other. By the early 1990s, it was clear that IPv4 was not a long-term protocol. Its design did not anticipate a number of requirements that turned out to be crucial. Such requirements not only pertained to the proliferation of devices, but also the need for additional security, simpler configuration and better prioritization of some services, such as real-time services (often referred to as Quality of Service issues). But mainly we were running out of address space. Since IPv4 has only a 32 bit address space we can have only 2^{32} or only 4 billion devices. Enter Ipv6. The main improvement brought by IPv6 is the increase in the number of addresses available for networked devices, allowing, for example, each mobile phone and mobile electronic device to have its own address. IPv4 supports 2^{32} (about 4.3 billion) addresses, which is inadequate for giving even one address to every living person, let alone supporting embedded and portable devices. IPv6, however, supports 2^{128} (about 340 billion billion billion billion) addresses, or approximately 5×10^{28} addresses for *each* of the roughly 6.5 billion people alive today.

So once every device has an IP address, devices can communicate with each other and this opens up an exciting array of possibilities. We can control all the appliances through Internet and do things like place an online order before a particular grocery item is about to be over or monitor electrical devices through the Internet. In this paper we will look at a framework for developing a microcontroller based home automation system.

Controlling of home appliances through desktop computers already exist. But we have many disadvantages with this. They are:

1. Desktop computers are bulky. So, they occupy more space
2. It doesn't make sense to dedicate a computer just to monitor devices when it can do a lot more computing

In order to overcome those we can use an embedded system. An embedded system consists of a processor (microprocessor or microcontroller) and supporting software. Apart

SEEK DIGITAL LIBRARY

from that we need to select a device to control the switching operation of home appliances. This paper is organized as follows. First we will look at the task of choosing an appropriate processor and controlling device. Then we will describe the basic operation of the system. Finally we will discuss the pros and cons of the system.

1. Selection of microcontroller/microprocessor

Many microcontrollers which suit our needs are available in the market. To choose an appropriate processor the criteria are:

1. Cost: Should obviously be cheap
2. Built in TCP/IP: Now-a-days many networked microcontrollers are appearing in the market. Rather than creating a TCP/IP stack from scratch, it is better to choose a microcontroller which has built in TCP/IP stack.
3. Ethernet support: Most microcontrollers which have TCP/IP stack have this.
4. Serial port: Serial port communication should also be built into the microcontroller

We chose TINI DS80C400 as it fit most our requirements. Some other microcontroller which could fit the bill is RCM4000 RabbitCore.

Software:

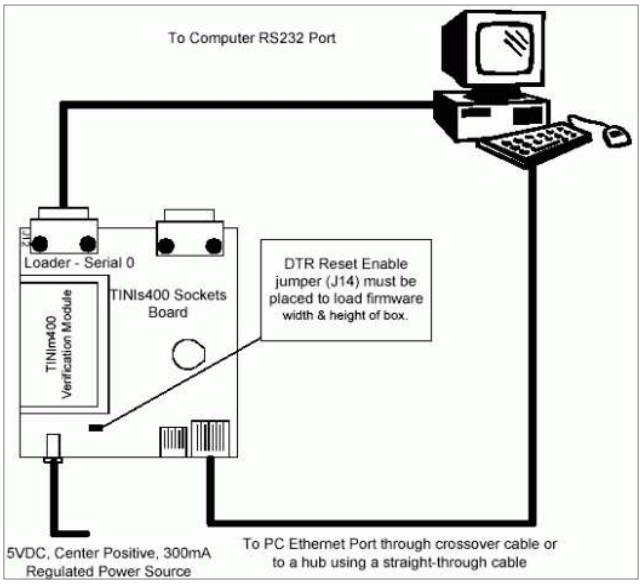
The DS80C400 processor has a 64kB ROM that contains a boot loader, network stack, memory manager, and process scheduler implemented in highly optimized 8051 assembly language. Using the latest TINI reference design, the DSTINIm400 reference module, developers can choose among using Java, C, or even coding in 8051.Here we have chosen JAVA because of the following advantages:

1. Developing in java increases portability
2. Hardware independence cuts cross-platform development costs
3. Strong OOP model increases reusability -- Class Libraries.
4. Strong integration with the Network gets you where you want to go.

2. Setup and Operation of the Automation system

Any home appliance can be controlled by the TINI processor. This TINI processor has built in TCP/IP. So we can control the device by the Internet. First we will explain how to setup the home automation system and then we will explain the technical aspects.

1. Get a TINIm400 (DS80C400) board
2. Setup the TINI board as explained in http://www.maxim-ic.com/appnotes.cfm/appnote_number/612 and connect it to the computer as shown



3.So now you have a working microcontroller. Set the IP address of the microcontroller to 192.168.0.15 using the command `ipconfig -a 192.168.0.15 -m 255.255.255.0` at the slush prompt

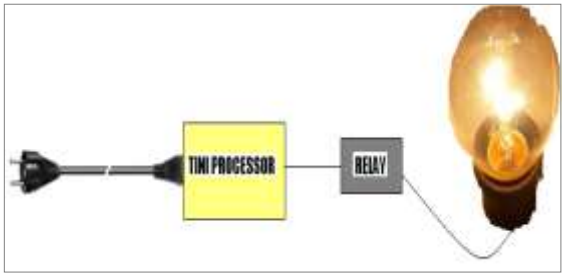
4. Copy the provided postExample.tini file to c:\
5. Open the command prompt on the computer and cd to c:\
6. Connect using ftp to the TINI microcontroller using the command "ftp 192.168.0.15" on the command prompt of your computer.
7. It will ask for username and password. Enter root as username and tini as the password. You will be connected through ftp to the microcontroller.
8. Now run the ftp command "get postExample.tini" on the ftp prompt. This command will upload postExample.tini from c:\ drive of your computer to the microcontroller.
9. Since the program is now loaded on the microcontroller we need to run the program. To run programs on TINI we need to telnet to TINI. So disconnect from ftp using the command "bye" and connect through telnet using the

SEEK DIGITAL LIBRARY

command “telnet 192.168.0.15”. Once again enter the same username and password. Now you will be connected to the telnet shell.

10. Execute the command “java postExample.tini &” on the telnet prompt. That will start the program and you should get some output saying the web server is started. Since the program is now loaded we can disconnect the microcontroller from the host computer and connect it to the network by attaching the microcontroller to a hub or switch. But to make things easier let us assume the host computer and the microcontroller are still directly connected.

11. Connect the bulb to the relay and the relay to the serial port of TINI as shown below.

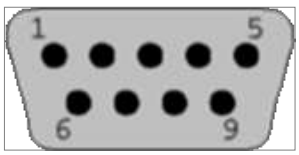


12. Open a browser on your computer and in the address bar type http://192.168.0.15 and click on go. You will get the home automation page as shown below. The bulb state is displayed. You can switch on/off the bulb by selecting the appropriate radio button and clicking on submit. When submit is clicked you will see that the bulb is appropriately switched on or off.

3. Technical Details:

Controlling Relay through Serial Port:

It is pretty easy to control a relay with a serial port. With a standard serial port we can control 2 relays. A standard PC serial port has 9 pins. Pin 4 - DTR (data terminal ready) and Pin 7 - RTS (request to send) can be used to control a relay. These two ports don't actually send data. They are used to signal the other device to tell it when to send data.



These pins can be set high or low. When set high, they each go to about +6volts. When set low they go to about -6 volts. This voltage swing is what is used to run the relay. To drive the pin, a program is written. It turns DTR high then low based on status set by user in Internet. The related java code is shown below.

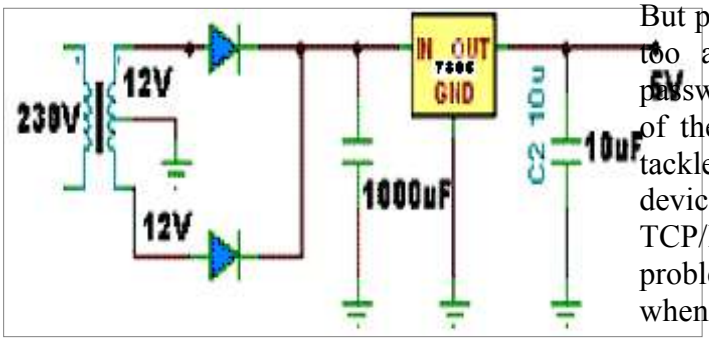
```
//open serial port. The TINI serial port we connect to is called serial4
String status="";
 CommPortIdentifier portId = CommPortIdentifier.getPortIdentifier("serial4");
      sp = (SerialPort)portId.open("testApp", 5000);
//setting serial port parameters

TINIOS.setSerial(TINIOS.SERIAL_SET_RTSCTS_FLOW_CONTROL,4,true);
//get user request
PostElement
Pe= (PostElement)data.elementAt(0);
//to switch bulb off, set DTR pin to true. That indicates to the relay to cut the current off
if ( !pe.value.equals("off") ){
        status = "Bulb switched off";
        sp.setDTR(true);
                }
//to switch bulb off, set DTR pin to false. That indicates to the relay to allow current flow
else if( !pe.value.equals("on") ){
        status = "Bulb switched on";
        sp.setDTR(false);
                }
else if( !pe.value.equals("close") ){
        status="closed";
        sp.close();
                }
        sp.close();
```

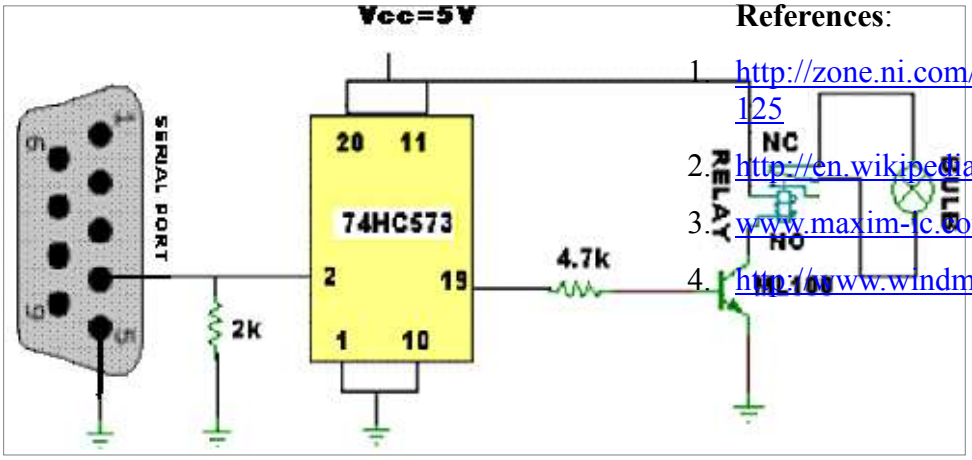
After the relay is controlled through the serial port we set the HTML page to reflect the new

SEEK DIGITAL LIBRARY

status. Please go through the code provided to get a better understanding of the system. The code is well documented and each step is explained.

But the current coming from the serial port is not sufficient to drive the relay. In order to boost up the current levels we use a device driver i.e.,IC74HC573 (octal tristate output transparent latch).To drive the IC we need 5V constant Dc supply. This supply is provided by the Voltage regulator IC7805.A transistor is used to protect the relay. The circuit must be connected as shown below:





Problems:

The major problem with this kind of application is security. When your electrical devices can be controlled by you over the Internet, then basically your electrical devices have been opened up to the entire Internet world. Since you can access the devices on the Internet, then anyone on the Internet can also access your devices unless you protect them somehow. The most common protection mechanism that comes to mind is password based protection. We can password protect the device control web pages. But password protection is not too effective, that too as we may not be able to encrypt the passwords because of the computing limitations of the microcontroller. This problem has to be tackled at a more basic level. Some kind of device control protocol which works on top of TCP/IP is needed which can resolve these problems. until that time, we need to be careful when we open up our devices to the Internet.

References:

1. http://zone.ni.com/devzone/cda/tut/p/id/4125
2. http://en.wikipedia.org/wiki/Relay
3. www.maxim-ic.com/microcontrollers.
4. http://www.windmeadow.com/node/4

Conclusion:

This is just the starting point. Going forward we can link a set of relays and control several devices at the same time. Also a variety of devices can be controlled. We can connect the TINI to a load cell which in turn could be linked to a grocery bag. When the grocery supplies go down the load cell sends a signal to the TINI microcontroller. Since TINI is connected to the internet it could be programmed to order the groceries online when it receives a signal from the load cell. The possibilities are limitless.

SEEK DIGITAL LIBRARY