Near East University

Faculty of Electrical and Electronic Engineering

Home Appliances Control via Internet

Prepared by: Wassim Orabi

Supervisor: Assoc. Prof. Dr. Özgür C. Özerdem

Eng. Mohammad Kmail

NICOSIA – 2014

Special Thanks:

To all Instructors who have been teaching me for years. And a special thanks to Eng. Mohammad Kamil, who made a huge contribution in guiding and helping me to make this project see the light.

Index:

Chapter 5

Chapter One

Objective

The aim of this project is to design and implement a control system over the internet. The system can present data about installed systems and apply controls via internet connection.

Introduction

Thanks to the huge development in technology today, we have the ability to control machines in our homes or offices even if we are thousands of miles away. It is important to mention that the development of communication systems has simplified our life.

This control system will allow us to control a circuit over the internet and can be used in a countless number of applications, since it could be connected to a variety of devices and control those devices such as printers, temperature control units, security systems etc. the system can allow us also to monitor the data about different systems via internet protocols.

For example, the circuit could be connected to any electrical device and the user could turn it on/off over the internet from anywhere the user wishes. It can also monitor the case of the device, log information about it, and many other different tasks.

Project Overview

This project offers a way to control a system over the internet. It interacts with the user via web page that provides several options to control the system. For example, if the system controls an electrical lamp in a room, the web page can allow the user to turn this lamp on/off by simply choosing one of the icons that appear in the web page.

The connection between the web page and the system passes through many stages. First, system is connected to a microcontroller, Arduino Duemilanove, which controls the operations in the system. Second, this microcontroller is connected to an Arduino Ethernet shield; this device makes it possible for the microcontroller to be accessed through the internet. Third, a router is responsible of making sure that the Ethernet shield is given a stable internet connection and can access the web. Since this router is given a dynamic IP by the internet service provider, we had to run a DNS-IP-Tracking software to make sure that the user interface is kept up to date with the changing IP of the router. Finally, on the other side of the network, a dedicated web page, written using HTML, contains several options for the user to control the system.

In the following chapters, I introduce the basic components of my project with a brief illustration about each of them in addition to some general concepts that help establishing a strong background for the project, for example I mentioned some general concepts about IP, Routers, Microcontrollers, and Darlington Pair Transistors. The components that make up my project are: Arduino microcontroller, Arduino Ethernet shield, Relays, and ULN2003.

Chapter Two

Microcontrollers

A microcontroller (sometimes abbreviated µC, uC or MCU) is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. Program memory in the form of NOR flash or OTP ROM is also often included on chip, as well as a typically small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes. Mixed signal microcontrollers are common, integrating analog components needed to control non-digital electronic systems.

Some microcontrollers may use four-bit words and operate at clock rate frequencies as low as 4 kHz, for low power consumption (single-digit milliwatts or microwatts). They will generally have the ability to retain functionality while waiting for an event such as a button press or other interrupt; power consumption while sleeping (CPU clock and most peripherals off) may be just nanowatts, making many of them well suited for long lasting battery applications. Other microcontrollers may serve performance-critical roles, where they may need to act more like a digital signal processor (DSP), with higher clock speeds and power consumption.

Embedded Design

A microcontroller can be considered a self-contained system with a processor, memory and peripherals and can be used as an embedded system. The majority of microcontrollers in use today are embedded in other machinery, such as automobiles, telephones, appliances, and peripherals for computer systems. While some embedded systems are very sophisticated, many have minimal requirements for memory and program length, with no operating system, and low software complexity. Typical input and output devices include switches, relays, solenoids, LEDs, small or custom LCD displays, radio frequency devices, and sensors for data such as temperature, humidity, light level etc. Embedded systems usually have no keyboard, screen, disks, printers, or other recognizable I/O devices of a personal computer, and may lack human interaction devices of any kind.

Interrupts

Microcontrollers must provide real time (predictable, though not necessarily fast) response to events in the embedded system they are controlling. When certain events occur, an interrupt system can signal the processor to suspend processing the current instruction sequence and to begin an interrupt service routine (ISR, or "interrupt handler"). The ISR will perform any processing required based on the source of the interrupt, before returning to the original instruction sequence. Possible interrupt sources are device dependent, and often include events such as an internal timer overflow, completing an analog to digital conversion, a logic level change on an input such as from a button being pressed, and data received on a communication link. Where power consumption is important as in battery operated devices, interrupts may also wake a microcontroller from a low power sleep state where the processor is halted until required to do something by a peripheral event.

Programs

Typically microcontroller programs must fit in the available on-chip program memory, since it would be costly to provide a system with external, expandable, memory. Compilers and assemblers are used to convert high-level language and assembler language codes into a compact machine code for storage in the microcontroller's memory. Depending on the device, the program memory may be permanent, read-only memory that can only be programmed at the factory or program memory that may be field-alterable flash or erasable read-only memory.

Manufacturers have often produced special versions of their microcontrollers in order to help the hardware and software of the target system. Originally these included EPROM versions that have a "window" on the top of the device through which program memory can be erased by ultraviolet light, ready for reprogramming after a programming ("burn") and test cycle.

Other versions may be available where the ROM is accessed as an external device rather than as internal memory, however these are becoming increasingly rare due to the widespread availability of cheap microcontroller programmers.

The use of field-programmable devices on a microcontroller may allow field update of the firmware or permit late factory revisions to products that have been assembled but not yet shipped. Programmable memory also reduces the lead time required for deployment of a new product.

Where hundreds of thousands of identical devices are required, using parts programmed at the time of manufacture can be an economical option. These "mask programmed" parts have the program laid down in the same way as the logic of the chip, at the same time.

A customizable microcontroller incorporates a block of digital logic that can be personalized in order to provide additional processing capability, peripherals and interfaces that are adapted to the requirements of the application. For example, the AT91CAP from Atmel has a block of logic that can be customized during manufacture according to user requirements.

Other microcontroller features

Microcontrollers usually contain from several to dozens of general purpose input/output pins (GPIO). GPIO pins are software configurable to either an input or an output state. When GPIO pins are configured to an input state, they are often used to read sensors or external signals. Configured to the output state, GPIO pins can drive external devices such as LEDs or motors.

Many embedded systems need to read sensors that produce analog signals. This is the purpose of the analog-to-digital converter (ADC). Since processors are built to interpret and process digital data, i.e. 1s and 0s, they are not able to do anything with the analog signals that may be sent to it by a device. So the analog to digital converter is used to convert the incoming data into a form that the processor can recognize. A less common feature on some microcontrollers is a digital-to-analog converter (DAC) that allows the processor to output analog signals or voltage levels.

In addition to the converters, many embedded microprocessors include a variety of timers as well. One of the most common types of timers is the Programmable Interval Timer (PIT). A PIT may either count down from some value to zero, or up to the capacity of the counting register, overflowing to zero. Once it reaches zero, it sends an interrupt to the processor indicating that it has finished counting. This is useful for devices such as thermostats, which periodically test the temperature around them to see if they need to turn the air conditioner on, the heater on, etc.

A dedicated Pulse Width Modulation (PWM) block makes it possible for the CPU to control converters, resistive loads, motors, etc., without using lots of CPU resources in tight timer loops.

Universal Asynchronous Receiver/Transmitter (UART) block makes it possible to receive and transmit data over a serial line with very little load on the CPU. Dedicated on-chip hardware also often includes capabilities to communicate with other devices (chips) in digital formats such as I^2C and Serial Peripheral Interface (SPI). [1]

Arduino

What is Arduino?

Arduino is an open-source electronic prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists and anyone interested in creating interactive objects for environments.

What can Arduino do?

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language and the Arduino development environment. Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, and MaxMSP).

The boards can be built by hand or purchased preassembled; the software can be downloaded for free. The hardware reference designs (CAD files) are available under an open-source license, and can be adapted to suit anyone's needs. [2]

The hardware of Arduino

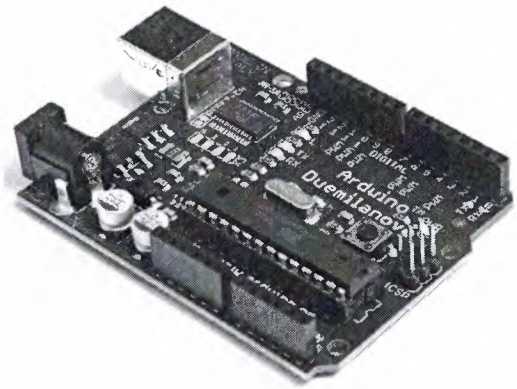


Figure (1) Arduino Circuit

An Arduino board consists of an Atmel 8-bit AVR microcontroller with complementary components to facilitate programming and incorporation into other circuits. An important aspect of the Arduino is the standard way that connectors are exposed, allowing the CPU board to be connected to a variety of interchangeable add-on modules known as shields.

Some shields communicate with the Arduino board directly over various pins, but many shields are individually addressable via an I²C serial bus, allowing many shields to be stacked and used in parallel.

An Arduino's microcontroller is also pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory, compared with other devices that typically need an external programmer. This makes using an Arduino more straightforward by allowing the use of an ordinary computer as the programmer.

At a conceptual level, when using the Arduino software stack, all boards are programmed over an RS-232 serial connection, but the way this is implemented varies by hardware version. Serial Arduino boards contain a level shifter circuit to convert between RS-232-level and TTL-level signals. Current Arduino boards are programmed via USB, implemented using USB-to-serial adapter chips such as the FTDIFT232.

Some variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods. (When used with traditional microcontroller tools instead of the Arduino IDE, standard AVR ISP programming is used.)

The Arduino board exposes most of the microcontroller's I/O pins for use by other circuits. The Diecimila, Duemilanove, and current Uno provide 14 digital I/O pins, six of which can produce pulse-width modulated signals, and six analog inputs. These pins are on the top of the board, via female 0.10-inch (2.5 mm) headers. Several plug-in application shields are also commercially available.

The Arduino Nano, and Arduino-compatible Bare Bones Board and Boarduino boards may provide male header pins on the underside of the board to be plugged into solderless breadboards.

There are many Arduino-compatible and Arduino-derived boards. Some are functionally equivalent to an Arduino and may be used interchangeably. Many are the basic Arduino with the addition of commonplace output drivers, often for use in school-level education to simplify the construction of buggies and small robots.

Others are electrically equivalent but change the form factor, sometimes permitting the continued use of Shields, sometimes not. Some variants use completely different processors, with varying levels of compatibility.

The software of Arduino

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch".

Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. Users only need define two functions to make a runnable cyclic executive program:

- setup(): a function run once at the start of a program that can initialize settings
- loop(): a function called repeatedly until the board powers off

The Arduino IDE uses the GNU tool chain and AVR Libc to compile programs, and uses avrdude to upload programs to the board.

As the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.[3]

In this project, I decided to use the Arduino Duemilanove since it is suitable for my application. Following is a description of this kind of microcontrollers and its features.

Chapter Three

Internet Protocol (IP)

Every machine on a network has a unique identifier. Just as you would address a letter to send in the mail, computers use the unique identifier to send data to specific computers on a network. Most networks today, including all computers on the Internet, use the TCP/IP protocol as the standard for how to communicate on the network. In the TCP/IP protocol, the unique identifier for a computer is called its IP address.

There are two standards for IP addresses: IP Version 4 (IPv4) and IP Version 6 (IPv6). All computers with IP addresses have an IPv4 address, and many are starting to use the new IPv6 address system as well. Here's what these two address types mean:

IPv4 uses 32 binary bits to create a single unique address on the network. An IPv4 address is expressed by four numbers separated by dots. Each number is the decimal (base-10) representation for an eight-digit binary (base-2) number, also called an octet. For example: 216.27.61.137

IPv6 uses 128 binary bits to create a single unique address on the network. An IPv6 address is expressed by eight groups of hexadecimal (base-16) numbers separated by colons, as in 2001:cdba:0000:0000:0000:0000:3257:9652. Groups of numbers that contain all zeros are often omitted to save space, leaving a colon separator to mark the gap (as in 2001:cdba::3257:9652).

At the dawn of IPv4 addressing, the Internet was not the large commercial sensation it is today, and most networks were private and closed off from other networks around the world. When the Internet exploded, having only 32 bits to identify a unique Internet address caused people to panic that we'd run out of IP addresses. Under IPv4, there are 232 possible combinations, which offers just under 4.3 billion unique addresses. IPv6 raised that to a panic-relieving 2128 possible addresses. Later, we'll take a closer look at how to understand your computer's IPv4 or IPv6 addresses.`

How does your computer get its IP address?

An IP address can be either dynamic or static. A static address is one that you configure yourself by editing your computer's network settings. This type of address is rare, and it can create network issues if you use it without a good understanding of TCP/IP. Dynamic addresses are the most common. They're assigned by the Dynamic Host Configuration Protocol (DHCP), a service running on the network. DHCP typically runs on network hardware such as routers or dedicated DHCP servers.

Dynamic IP addresses are issued using a leasing system, meaning that the IP address is only active for a limited time. If the lease expires, the computer will automatically request a new lease. Sometimes, this means the computer will get a new IP address, too, especially if the computer was unplugged from the network between leases. This process is usually transparent to the user unless the computer warns about an IP address conflict on the network (two computers with the same IP address). An address conflict is rare, and today's technology typically fixes the problem automatically.

Next, let's take a closer look at the important parts of an IP address and the special roles of certain addresses.

IP Classes

Earlier, you read that IPv4 addresses represent four eight-digit binary numbers. That means that each number could be 00000000 to 11111111 in binary, or 0 to 255 in decimal (base-10). In other words, 0.0.0.0 to 255.255.255.255. However, some numbers in that range are reserved for specific purposes on TCP/IP networks. These reservations are recognized by the authority on TCP/IP addressing, the Internet Assigned Numbers Authority (IANA). Four specific reservations include the following:

0.0.0.0 -- This represents the default network, which is the abstract concept of just being connected to a TCP/IP network.

255.255.255.255 -- This address is reserved for network broadcasts, or messages that should go to all computers on the network.

127.0.0.1 -- This is called the loopback address, meaning your computer's way of identifying itself, whether or not it has an assigned IP address.

169.254.0.1 To 169.254.255.254 -- This is the Automatic Private IP Addressing (APIPA) range of addresses assigned automatically when a computer's unsuccessful getting an address from a DHCP server.

The other IP address reservations are for subnet classes. A subnet is a smaller network of computers connected to a larger network through a router. The subnet can have its own address system so computers on the same subnet can communicate quickly without sending data across the larger network. A router on a TCP/IP network, including the Internet, is configured to recognize one or more subnets and route network traffic appropriately.

The following are the IP addresses reserved for subnets:

10.0.0.0 To 10.255.255.255 -- This falls within the Class A address range of 1.0.0.0 to 127.0.0.0, in which the first bit is 0.

172.16.0.0 To 172.31.255.255 -- This falls within the Class B address range of 128.0.0.0 to 191.255.0.0, in which the first two bits are 10.

192.168.0.0 To 192.168.255.255 -- This falls within the Class C range of 192.0.0.0 through 223.255.255.0, in which the first three bits are 110.

Multicast (formerly called Class D) -- The first four bits in the address are 1110, with addresses ranging from 224.0.0.0 to 239.255.255.255.

Reserved for future/experimental use (formerly called Class E) -- addresses 240.0.0.0 to 254.255.255.254.

The first three (within Classes A, B and C) are those most used in creating subnets.

Internet Addresses and Subnets

The following is an example of a subnet IP address you might have on your computer at home if you're using a router (wireless or wired) between your ISP connection and your computer:

IP address: 192.168.1.102

Subnet mask: 255.255.255.0

Twenty-four bits (three octets) reserved for network identity

Eight bits (one octet) reserved for nodes

Subnet identity based on subnet mask (first address): 192.168.1.0

The reserved broadcast address for the subnet (last address): 192.168.1.255

Example addresses on the same network: 192.168.1.1, 192.168.1.103

Example addresses not on the same network: 192.168.2.1, 192.168.2.103

Besides reserving IP addresses, the IANA is also responsible for assigning blocks of IP addresses to certain entities, usually commercial or government organizations. Your Internet service provider (ISP) may be one of these entities, or it may be part of a larger block under the control of one of those entities. In order for you to connect to the Internet, your ISP will assign you one of these addresses. If you only connect one computer to the Internet, that computer can use the address from your ISP.

Many homes today, though, use routers to share a single Internet connection between multiple computers. Wireless routers have become especially popular in recent years, avoiding the need to run network cables between rooms.

If you use a router to share an Internet connection, the router gets the IP address issued directly from the ISP. Then, it creates and manages a subnet for all the computers connected to that router. If your computer's address falls into one of the reserved subnet ranges listed earlier, you're going through a router rather than connecting directly to the Internet.

IP addresses on a subnet have two parts: network and node. The network part identifies the subnet itself. The node, also called the host, is an individual piece of computer equipment connected to the network and requiring a unique address. Each computer knows how to separate the two parts of the IP address by using a subnet mask. A subnet mask looks somewhat like an IP address, but it's actually just a filter used to determine which part of an IP address designates the network and node.

A subnet mask consists of a series of 1 bits followed by a series of 0 bits. The 1 bits indicate those that should mask the network bits in the IP address, revealing only those that identify a unique node on that network. In the IPv4 standard, the most commonly used subnet masks have complete octets of 1s and 0s as follows:

255.0.0.0.0 = 11111111.00000000.00000000.00000000 = eight bits for networks, 24 bits for nodes

255.255.0.0 = 11111111.11111111.00000000.00000000 = 16 bits for networks, 16 bits for nodes

255.255.255.0 = 11111111. 11111111.11111111.00000000 = 24 bits for networks, eight bits for nodes

People who set up large networks determine what subnet mask works best based on the number of desired subnets or nodes. For more subnets, use more bits for the network; for more nodes per subnet, use more bits for the nodes. This may mean using non-standard mask values. For instance, if you want to use 10 bits for networks and 22 for nodes, your subnet mask value would require using 11000000 in the second octet, resulting in a subnet mask value of 255.192.0.0.

Another important thing to note about IP addresses in a subnet is that the first and last addresses are reserved. The first address identifies the subnet itself, and the last address identifies the broadcast address for systems on that subnet.

How Routers Work

We're all used to seeing the various parts of the Internet that come into our homes and offices – the Web pages, e-mail messages and downloaded files that make the Internet a dynamic and valuable medium. But none of these parts would ever make it to your computer without a piece of the Internet that you've probably never seen. In fact, most people have never stood "face to machine" with the technology most responsible for allowing the Internet to exist at all: the router.

Let's look at what a very simple router might do. Imagine a small company that makes animated 3-D graphics for local television stations. There are 10 employees of the company, each with a computer. Four of the employees are animators, while the rest are in sales, accounting and management. The animators will need to send lots of very large files back and forth to one another as they work on projects. To do this, they'll use a network.

When one animator sends a file to another, the very large file will use up most of the network's capacity, making the network run very slowly for other users. One of the reasons that a single intensive user can affect the entire network stems from the way that Ethernet works. Each information packet sent from a computer is seen by all the other computers on the local network. Each computer then examines the packet and decides whether it was meant for its address. This keeps the basic plan of the network simple, but has performance consequences as the size of the network or level of network activity increases. To keep the animators' work from interfering with that of the folks in the front office, the company sets up two separate networks, one for the animators and one for the rest of the company. A router links the two networks and connects both networks to the Internet.

Directing Traffic

The router is the only device that sees every message sent by any computer on either of the company's networks. When the animator in our example sends a huge file to another animator, the router looks at the recipient's address and keeps the traffic on the animator's network. When an animator, on the other hand, sends a message to the bookkeeper asking about an expense-account check, then the router sees the recipient's address and forwards the message between the two networks. One of the tools a router uses to decide where a packet should go is a configuration table. A configuration table is a collection of information, including:

Information on which connections lead to particular groups of addresses Priorities for connections to be used Rules for handling both routine and special cases of traffic. A configuration table can be as simple as a half-dozen lines in the smallest routers, but can grow to massive size and complexity in the very large routers that handle the bulk of Internet messages.

A router, then, has two separate but related jobs:

The router ensures that information doesn't go where it's not needed. This is crucial for keeping large volumes of data from clogging the connections of "innocent bystanders."

The router makes sure that information does make it to the intended destination.

In performing these two jobs, a router is extremely useful in dealing with two separate computer networks. It joins the two networks, passing information from one to the other and, in some cases, performing translations of various protocols between the two networks. It also protects the networks from one another, preventing the traffic on one from unnecessarily spilling over to the other. As the number of networks attached to one another grows, the configuration table for handling traffic among them grows, and the processing power of the router is increased. Regardless of how many networks are attached, though, the basic operation and function of the router remains the same. Since the Internet is one huge network made up of tens of thousands of smaller networks, its use of routers is an absolute necessity.

Internet data, whether in the form of a Web page, a downloaded file or an e-mail message, travels over a system known as a packet-switching network. In this system, the data in a message or file is broken up into packages about 1,500 bytes long. Each of these packages gets a wrapper that includes information on the sender's address, the receiver's address, the package's place in

the entire message, and how the receiving computer can be sure that the package arrived intact. Each data package, called a packet, is then sent off to its destination via the best available route -- a route that might be taken by all the other packets in the message or by none of the other packets in the message. This might seem very complicated compared to the circuit approach used by the telephone system, but in a network designed for data there are two huge advantages to the packet-switching plan.

The network can balance the load across various pieces of equipment on a millisecond-by-millisecond basis.

If there is a problem with one piece of equipment in the network while a message is being transferred, packets can be routed around the problem, ensuring the delivery of the entire message.

Knowing Where to Send Data

Routers are one of several types of devices that make up the "plumbing" of a computer network. Hubs, switches and routers all take signals from computers or networks and pass them along to other computers and networks, but a router is the only one of these devices that examines each bundle of data as it passes and makes a decision about exactly where it should go. To make these decisions, routers must first know about two kinds of information: addresses and network structure.

When a friend mails a birthday card to be delivered to you at your house, he probably uses an address that looks something like this:

Joe Smith 123 Maple Street Smalltown, FL 45678

The address has several pieces, each of which helps the people in the postal service move the letter along to your house. The ZIP code can speed the process up; but even without the ZIP code, the card will get to your house as long as your friend includes your state, city and street address. You can think of this address as a logical address because it describes a way someone can get a message to you. This logical address is connected to a physical address that you generally only see when you're buying or selling a piece of property. The survey plot of the land and house, with latitude, longitude or section bearings, gives the legal description, or address, of the property.

Logical Addresses

Every piece of equipment that connects to a network, whether an office network or the Internet, has a physical address. This is an address that's unique to the piece of equipment that's actually attached to the network cable. For example, if your desktop computer has a network interface card (NIC) in it, the NIC has a physical address permanently stored in a special memory location. This physical address, which is also called the MAC address (for Media Access Control) has two parts, each 3 bytes long. The first 3 bytes identify the company that made the NIC. The second 3 bytes are the serial number of the NIC itself.

The interesting thing is that your computer can have several logical addresses at the same time. Of course, you're used to having several "logical addresses" bring messages to one physical address. Your mailing address, telephone number (or numbers) and home e-mail address all work to bring messages to you when you're in your house. They are simply used for different types of messages -- different networks, so to speak.

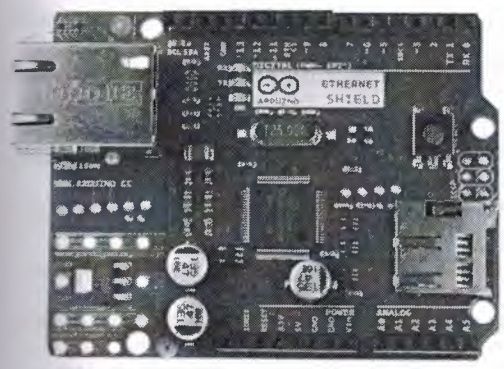
Logical addresses for computer networks work in exactly the same way. You may be using the addressing schemes, or protocols, from several different types of networks simultaneously. If you're connected to the Internet (and if you're reading this, you probably are), then you have an address that's part of the TCP/IP network protocol. If you also have a small network set up to exchange files between several family computers, then you may also be using the Microsoft NetBEUI protocol. If you connect to your company's network from home, then your computer may have an address that follows Novell's IPX/SPX protocol. All of these can coexist on your computer. Since the driver software that allows your computer to communicate with each network uses resources like memory and CPU time, you don't want to load protocols you won't need, but there's no problem with having all the protocols your work requires running at the same time.

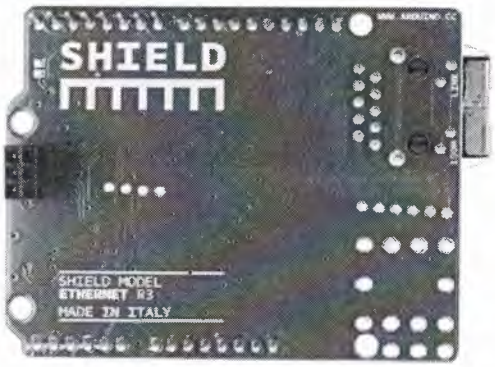
MAC Addresses

The chances are very good that you'll never see the MAC address for any of your equipment because the software that helps your computer communicate with a network takes care of matching the MAC address to a logical address. The logical address is what the network uses to pass information along to your computer.

The IP address is the logical address assigned to your connection by your ISP or network administrator. You'll see the addresses of other servers, including the DNS servers that keep track of all the names of Internet sites (so you can type "www.howstuffworks.com" rather than "216.27.61.189") and the gateway server that you connect to in order to reach the Internet. When you've finished looking at the information, click OK. (Note:For security reasons, some of the information about this connection to the Internet has been changed. You should be very careful about giving your computer's information to other people -- with your address and the right tools, an unscrupulous person could, in some circumstances, gain access to your personal information and control your system through a "Trojan Horse" program.[4]

Arduino Ethernet Shield





Arduino Ethernet Shield R3 Front *Arduino Ethernet Shield R3 Back*

Figure (2): The Arduino Ethernet Shield Circuit

Description

The Arduino Ethernet Shield allows an Arduino board to connect to the internet. It is based on the Wiznet W5100ethernet chip. The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. It supports up to four simultaneous socket connections. The Ethernet shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows another shield to be stacked on top.

The most recent revision of the board exposes the 1.0 pinout on rev 3 of the Arduino UNO board.

The Ethernet Shield has a standard RJ-45 connection, with an integrated line transformer and Power over Ethernet enabled.

There is an onboard micro-SD card slot, which can be used to store files for serving over the network. It is compatible with the Arduino Uno and Mega (using the Ethernet library). The onboard microSD card reader is accessible through the SD Library. When working with this library, SS is on Pin 4. The original revision of the shield contained a full-size SD card slot; this is not supported.

The shield also includes a reset controller, to ensure that the W5100 Ethernet module is properly reset on power-up. Previous revisions of the shield were not compatible with the Mega and need to be manually reset after power-up.

The current shield has a Power over Ethernet (PoE) module designed to extract power from a conventional twisted pair Category 5 Ethernet cable:

- IEEE802.3af compliant
- Low output ripple and noise (100mVpp)
- Input voltage range 36V to 57V
- Overload and short-circuit protection
- 9V Output
- High efficiency DC/DC converter: typ 75% @ 50% load
- 1500V isolation (input to output)

The shield does not come with the PoE module built in, it is a separate component that must be added on.

Arduino communicates with both the W5100 and SD card using the SPI bus (through the ICSP header). This is on digital pins 10, 11, 12, and 13 on the Uno and pins 50, 51, and 52 on the Mega. On both boards, pin 10 is used to select the W5100 and pin 4 for the SD card. These pins cannot be used for general I/O. On the Mega, the hardware SS pin, 53, is not used to select either the W5100 or the SD card, but it must be kept as an output or the SPI interface won't work.

Note that because the W5100 and SD card share the SPI bus, only one can be active at a time. If you are using both peripherals in your program, this should be taken care of by the corresponding libraries. If you're not using one of the peripherals in your program, however, you'll need to explicitly deselect it. To do this with the SD card, set pin 4 as an output and write a high to it. For the W5100, set digital pin 10 as a high output.

The shield provides a standard RJ45 Ethernet jack.

The reset button on the shield resets both the W5100 and the Arduino board.

The shield contains a number of informational LEDs:

- PWR: indicates that the board and shield are powered
- LINK: indicates the presence of a network link and flashes when the shield transmits or receives data
- FULLD: indicates that the network connection is full duplex
- 100M: indicates the presence of a 100 Mb/s network connection (as opposed to 10 Mb/s)
- RX: flashes when the shield receives data
- TX: flashes when the shield sends data
- COLL: flashes when network collisions are detected

The solder jumper marked "INT" can be connected to allow the Arduino board to receive interrupt-driven notification of events from the W5100, but this is not supported by the Ethernet library. The jumper connects the INT pin of the W5100 to digital pin 2 of the Arduino.[5]

Connecting the Shield

To use the shield, mount it on top of an Arduino board (e.g. the Uno). To upload sketches to the board, connect it to your computer with a USB cable as you normally would. Once the sketch has been uploaded, you can disconnect the board from your computer and power it with an external power supply.

Connect the shield to your computer or a network hub or router using a standard ethernet cable (CAT5 or CAT6 withRJ45 connectors). Connecting to a computer may require the use of a cross-over cable (although many computers, including all recent Macs can do the cross-over internally).

Network Settings

The shield must be assigned a MAC address and a fixed IP address using the Ethernet.begin() function. A MAC address is a globally unique identifier for a particular device. Current Ethernet shields come with a sticker indicating the MAC address you should use with them. For older shields without a dedicated MAC address, inventing a random one should work, but don't use the same one for multiple boards. Valid IP addresses depend on the configuration of your network. It is possible to use DHCP to dynamically assign an IP to the shield. Optionally, you can also specify a network gateway and subnet.

SD Card

The latest revision of the Ethernet Shield includes a micro-SD card slot, which can be interfaced with using the SD library.

Dynamic IP Tracking

One of the problems I faced during working on this project is how to stay updated with the continuously changing IP of the router. Since the router is given a dynamic IP by the server of the internet provider, this dynamic IP will change every few days. And keeping track with the IP of the router is extremely essential since we need it to log on into our system.

The most adequate solution was to use a dynamic IP tracker. It involves setting up a web address using a free DNS server. Instead of using an IP address, (ex 204.412.10.454) users then refer to a domain name (ex. mycompany.ath.cx) The next step is to download a small software application to place on our server that will keep this DNS listing up to date, even when the ISP changes the IP address. Whenever the router sees that its IP was changed, it sends the new IP address to our hostname in our dynamic DNS tracker account.

Many servers provide a free dynamic DNS tracker, such as NoIP.com, Passtracker.com……

In our project the user will access the system through a web page. It will provide the user with an easy interface that allows him to control the system easily and quickly by choosing from a number of icons, each controls a function that the system does. This web page is written using HTML. [6]

An image of the web page of this project is shown on page 39.

HTML

HTML is a markup language. It tells the web browser what content to display. HTML separates "content" (words, images, audio, video, and so on) from "presentation" (the definition of the type of content and the instructions for how that type of content should be displayed).

HTML uses a pre-defined set of elements to identify content types. Elements contain one or more "tags" that contain or express content. Tags are surrounded by angle brackets, and the "closing" tag (the one that indicates the end of the content) is prefixed by a forward slash. [7]

What does H-T-M-L stand for?

HTML is an abbreviation of "HyperText Mark-up Language"

- **Hyper** is the opposite of linear. Computer programs run linearly: when the program had executed one action it went to the next line and after that, the next line and so on. But HTML is different - you can go wherever you want and whenever you want. For example, it is not necessary to visit MSN.com before you visit HTML.net.
- **Text** is self-explanatory.
- **Mark-up** is what you do with the text. You are marking up the text the same way you do in a text editing program with headings, bullets and bold text and so on.
- **Language** is what HTML is. [8]

Chapter Four

Relays

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

Basic design and operation

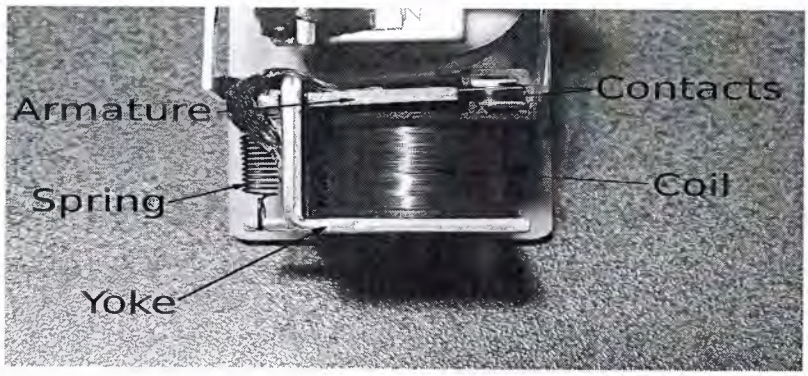


Figure (3) Relay

A simple electromagnetic relay consists of a coil of wire wrapped around a soft iron core, an iron yoke which provides a low reluctance path for magnetic flux, a movable iron armature, and one or more sets of contacts (there are two in the relay pictured).

The armature is hinged to the yoke and mechanically linked to one or more sets of moving contacts. It is held in place by a spring so that when the relay is de-energized there is an air gap in the magnetic circuit. In this condition, one of the two sets of contacts in the relay pictured is closed, and the other set is open. Other relays may have more or fewer sets of contacts depending on their function. The relay in the picture also has a wire connecting the armature to the yoke. This ensures continuity of the circuit between the moving contacts on the armature, and the circuit track on the printed circuit board (PCB) via the yoke, which is soldered to the PCB.

When an electric current is passed through the coil it generates a magnetic field that activates the armature and the consequent movement of the movable contact either makes or breaks (depending upon construction) a connection with a fixed contact. If the set of contacts was closed when the relay was de-energized, then the movement opens the contacts and breaks the connection, and vice versa if the contacts were open. When the current to the coil is switched off, the armature is returned by a force, approximately half as strong as the magnetic force, to its relaxed position. Usually this force is provided by a spring, but gravity is also used commonly in

industrial motor starters. Most relays are manufactured to operate quickly. In a low-voltage application this reduces noise; in a high voltage or current application it reduces arcing.

When the coil is energized with direct current, a diode is often placed across the coil to dissipate the energy from the collapsing magnetic field at deactivation, which would otherwise generate a voltage spike dangerous to semiconductor circuit components. Some automotive relays include a diode inside the relay case. Alternatively, a contact protection network consisting of a capacitor and resistor in series (snubber circuit) may absorb the surge.

If the coil is designed to be energized with alternating current (AC), a small copper "shading ring" can be crimped to the end of the solenoid, creating a small out-of-phase current which increases the minimum pull on the armature during the AC cycle.[1]

A solid-state relay uses a thruster or other solid-state switching device, activated by the control signal, to switch the controlled load, instead of a solenoid. An optocoupler (a light-emitting diode (LED) coupled with a photo transistor) can be used to isolate control and controlled circuits.

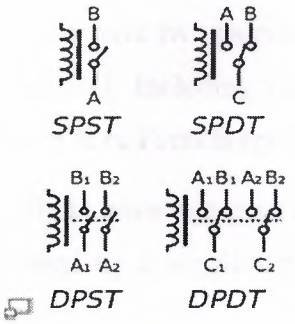


Figure (4) Circuit symbols of relays

Since relays are switches, the terminology applied to switches is also applied to relays; a relay switches one or more poles, each of whose contacts can be thrown by energizing the coil in one of three ways:

Normally-open (NO) contacts connect the circuit when the relay is activated; the circuit is disconnected when the relay is inactive. It is also called a Form A contact or "make" contact. NO contacts may also be distinguished as "early-make" or NOEM, which means that the contacts close before the button or switch is fully engaged.

Normally-closed (NC) contacts disconnect the circuit when the relay is activated; the circuit is connected when the relay is inactive. It is also called a Form B contact or "break" contact. NC contacts may also be distinguished as "late-break" or NCLB, which means that the contacts stay closed until the button or switch is fully disengaged.

Change-over (CO), or double-throw (DT), contacts control two circuits: one normally-open contact and one normally-closed contact with a common terminal. It is also called a Form C contact or "transfer" contact ("break before make"). If this type of contact utilizes a "make before break" functionality, then it is called a Form D contact.

The following designations are commonly encountered:

SPST – Single Pole Single Throw. These have two terminals which can be connected or disconnected. Including two for the coil, such a relay has four terminals in total.

It is ambiguous whether the pole is normally open or normally closed. The terminology "SPNO" and "SPNC" is sometimes used to resolve the ambiguity.

SPDT – Single Pole Double Throw. A common terminal connects to either of two others. Including two for the coil, such a relay has five terminals in total.

DPST – Double Pole Single Throw. These have two pairs of terminals. Equivalent to two SPST switches or relays actuated by a single coil. Including two for the coil, such a relay has six terminals in total. The poles may be Form A or Form B (or one of each).

DPDT – Double Pole Double Throw. These have two rows of change-over terminals. Equivalent to two SPDT switches or relays actuated by a single coil. Such a relay has eight terminals, including the coil.

The "S" or "D" may be replaced with a number, indicating multiple switches connected to a single actuator. For example 4PDT indicates a four pole double throw relay (with 12 terminals). [9]

Darlington transistor

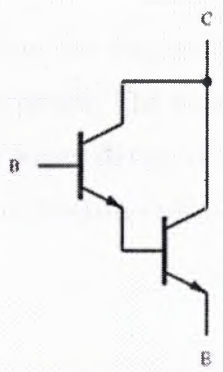


Figure (5) Circuit diagram of a Darlington pair using NPN transistors

In electronics, the Darlington transistor (often called a Darlington pair) is a compound structure consisting of two bipolar transistors (either integrated or separated devices) connected in such a way that the current amplified by the first transistor is amplified further by the second one.

This configuration gives a much higher common/emitter current gain than each transistor taken separately and, in the case of integrated devices, can take less space than two individual transistors because they can use a shared collector. Integrated Darlington pairs come packaged singly in transistor-like packages or as an array of devices (usually eight) in an integrated circuit.

Behavior

A Darlington pair is like a set of feeders with a high current gain (approximately the product of the gains of the two transistors). In fact, integrated devices have three leads (B, C and E), broadly equivalent to those of a standard transistor.

A general relation between the compound current gain and the individual gains is given by:

$$\beta_{\text{Darlington}} = \beta_1 \cdot \beta_2 + \beta_1 + \beta_2$$

If β1 and β2 are high enough (hundreds), this relation can be approximated with:

$$\beta_{\text{Darlington}} \approx \beta_1 \cdot \beta_2$$

Darlington pairs are available as integrated packages or can be made from two discrete transistors; Q1 (the left-hand transistor in the diagram) can be a low power type, but normally Q2 (on the right) will need to be high power. The maximum collector current IC (max) of the pair is that of Q2. A typical integrated power device is the 2N6282, which includes a switch-off resistor and has a current gain of 2400 at IC=10A.[10]

ULN2003

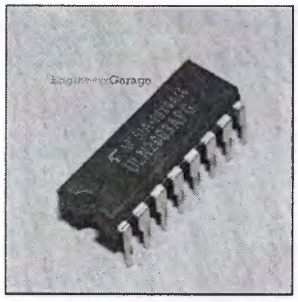


Figure (6) the IC of ULN2003

ULN2003 is a high voltage and high current Darlington array IC. It contains seven open collector Darlington pairs with common emitters. A Darlington pair is an arrangement of two bipolar transistors.

ULN2003 belongs to the family of ULN200X series of ICs. Different versions of this family interface to different logic families. ULN2003 is for 5V TTL, CMOS logic devices. These ICs are used when driving a wide range of loads and are used as relay drivers, display drivers, line drivers etc. ULN2003 is also commonly used while driving Stepper Motors.

Each channel or Darlington pair in ULN2003 is rated at 500mA and can withstand peak current of 600mA. The inputs and outputs are provided opposite to each other in the pin layout. Each driver also contains a suppression diode to dissipate voltage spikes while driving inductive loads.

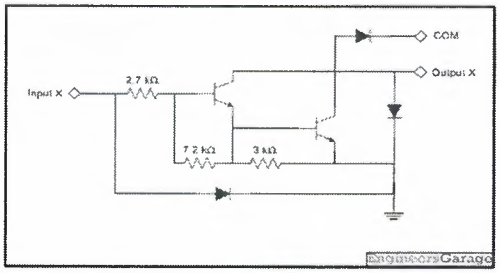


Figure (7) the schematic for each driver

ELECTRICAL CHARACTERISTICS (T_{amb} = 25°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit	Fig.
I_{CEX}	Output Leakage Current	V_{CE} = 50V T_{amb} = 70°C, V_{CE} = 50V T_{amb} = 70°C for ULN2002A V_{CE} = 50V, V_i = 6V for ULN2004A V_{CE} = 50V, V_i = 1V			50 100 500 500	µA µA µA µA	1a 1a 1b 1b
$V_{CE(sat)}$	Collector-emitter Saturation Voltage	I_C = 100mA, I_B = 250µA I_C = 200 mA, I_B = 350µA I_C = 350mA, I_B = 500µA		0.9 1.1 1.3	1.1 1.3 1.6	V V V	2 2 2
$I_{i(on)}$	Input Current	for ULN2002A, V_i = 17V for ULN2003A, V_i = 3.85V for ULN2004A, V_i = 5V V_i = 12V		0.82 0.93 0.35 1	1.25 1.35 0.5 1.45	mA mA mA mA	3 3 3 3
$I_{i(off)}$	Input Current	T_{amb} = 70°C, I_C = 500µA	50	65		µA	4
$V_{i(on)}$	Input Voltage	V_{CE} = 2V for ULN2002A I_C = 300mA for ULN2003A I_C = 200mA I_C = 250mA I_C = 300mA for ULN2004A I_C = 125mA I_C = 200mA I_C = 275mA I_C = 350mA			 13 2.4 2.7 3 5 6 7 8	V	5
h_{FE}	DC Forward Current Gain	for ULN2001A V_{CE} = 2V, I_C = 350mA	1000				2
C_i	Input Capacitance			15	25	pF	
t_{PLH}	Turn-on Delay Time	0.5 V_i to 0.5 V_o		0.25	1	µs	
t_{PHL}	Turn-off Delay Time	0.5 V_i to 0.5 V_o		0.25	1	µs	
I_R	Clamp Diode Leakage Current	V_R = 50V T_{amb} = 70°C, V_R = 50V			50 100	µA µA	6 6
V_F	Clamp Diode Forward Voltage	I_F = 350mA		1.7	2	V	7

Figure (8) electrical characteristics of ULN2003

[11]

Connecting a 12V relay to Arduino

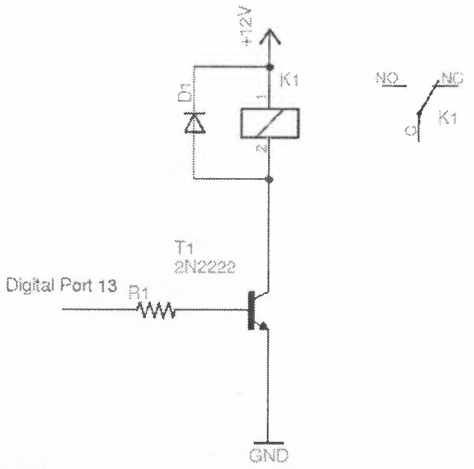


Figure (9) Connecting a 12V relay to Arduino

Step 1: Measure the coil resistance

First we must find the coil:
On some relays the pins are labeled so we can just measure at pin 2 & 5.

Otherwise we have to measure at every pin:

Between two pins we should have between 100 and 10 000 Ohm. Those are the two terminals of the coil. The coil is not polarized so it's not important which one goes to V+ or GND.

If we have found those there are only three left. Between two should be a connection. One of them is NC and one is COM. To find out which is which, we let one probe connected and connect the other to the pin that's left over. If we connect the coil to 12V DC it should make a clicking noise. If our millimeter now shows a low resistance we have found COM and NO. The one probe we didn't move is COM the other is NO.

Step 2: Calculate how much current will flow

The formula we need is a simple one:

U = R * I

I = U/R

For my our that would be:

I = 5V / 400Ohm
I = 0.03 A => 30 mA (That is Ic)

The Arduino can handle up to 20mA but its better to use a transistor even if our current is only 20mA. So for 30mA we definitely need one.

Step 3: Choose our transistor

First find the Datasheet of our transistor. For example search for "2N2222 datasheet".

Wer transistor should comply to the following things:

- It has to be NPN not PNP !!

- Ic should be bigger than the value we calculated in step 2

- Vceo should be bigger than the supply voltage

Step 4: Calculating R1

We can find the value of hfe in our datasheet:
Mine says for BC548 its 75 at 10mA at 10V. It's not very precise cause its very difficult to build transistor with a accurate hfe.

hfe = Ic / Ib

We know hfe and Ic so let's calculate Ib:

Ib = Ic / hfe

For BC548:

Ib = 0.03 A / 75
Ib = 0.0004 A => 0.4 mA

Due to Ohms Law:

R1 = U / Ib
R1 = 5V / 0.0004 A
R1 = 12500 Ohm
This is not very accurate to so we use 10kOhm.

Step 5: Choosing our diode

The diode is needed because the voltage will rise high if we suddenly change the voltage at the inductor. The formula for the voltage is:

U_L = - L * delta I/delta t

So theoretically if delta t equals zero U will be infinite.

But due to the minus in front we can add a diode in the "false direction" parallel to the relay. So the current can flow till its zero so the voltage is also zero.[12]

Chapter Five

This chapter includes figures and images related to the project. Below are images for the circuit, images show how the Arduino code is written, and Block diagram of the project.

Block Diagram of project

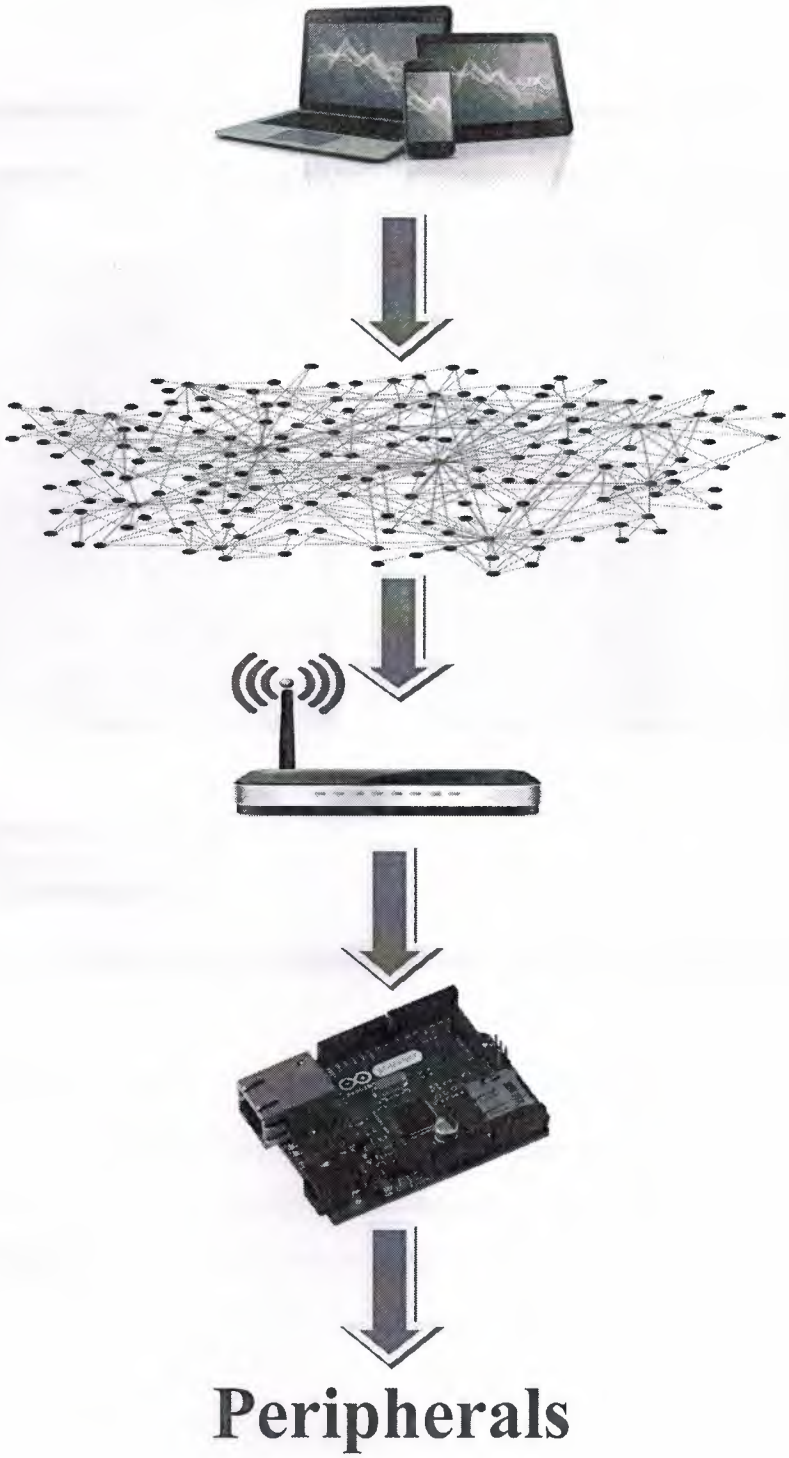


Figure (10) Block diagram of the project

Image of the web page

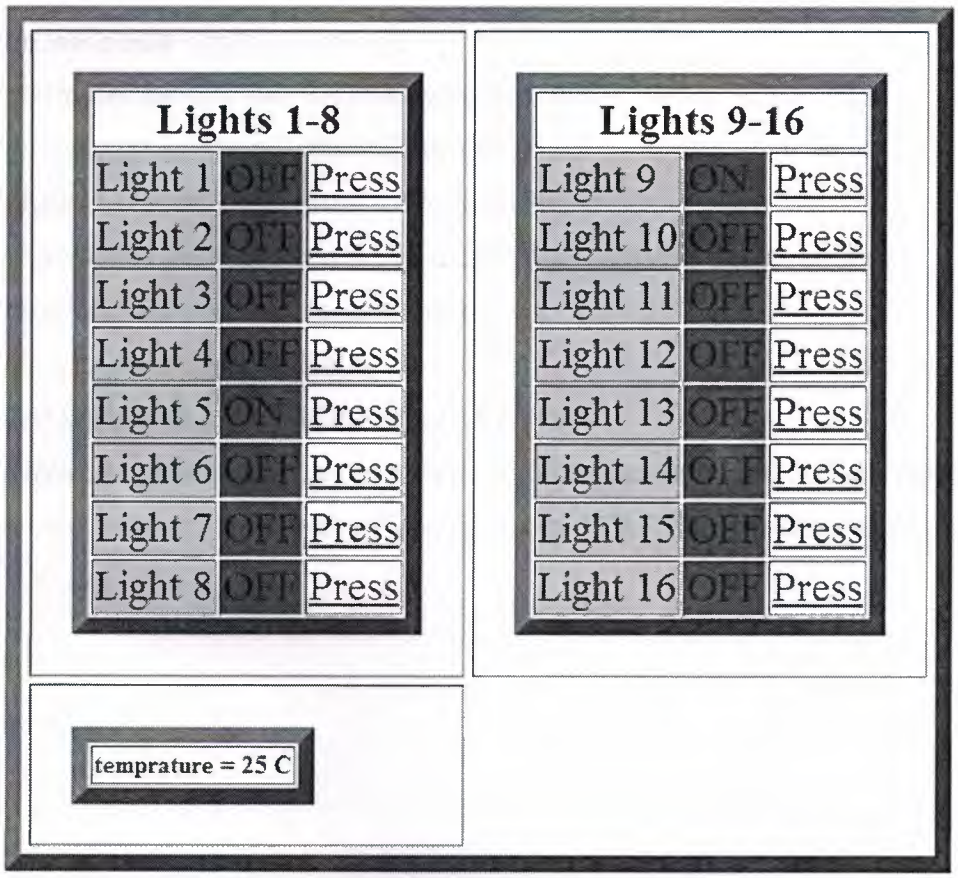


Figure (11) Image of the webpage interface

- Above is an image of the web page that allows the user to turn on/off a number of lights in the room. If a light is originally off and the user simply clicks on the "Press" icon, the corresponding light turn on/off and the color changes to green, and vice versa.

References

[1] http://en.wikipedia.org/wiki/Microcontroller

[2] http://www.arduino.cc/

[3] http://en.wikipedia.org/wiki/Arduino

[4] http://computer.howstuffworks.com/ethernet.htm

[5] http://arduino.cc/en/Main/ArduinoEthernetShield

[6] http://arduino.cc/en/Guide/ArduinoEthernetShield

[7] https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Introduction

[8] http://html.net/tutorials/html/lesson2.php

[9] http://en.wikipedia.org/wiki/Relay

[10] http://en.wikipedia.org/wiki/Darlington_transistor

[11] http://www.engineersgarage.com/electronic-components/uln2003-datasheet

[12] http://www.instructables.com/id/Connecting-a-12V-Relay-to-Arduino/?ALLSTEPS