AC 2011-880: HOME AUTOMATION WITH MICROCONTROLLER AND NETWORKING

Asad Yousuf, Savannah State University
Tyler Schecklman, Savannah Technical College

Student at Savannah Technical College enrolled in the Electronics and Computer Engineering Technology Associate Degree Program. Upon completion of Associate's degree, will transfer to another institution to complete Bachelor's degree in Electrical Engineering.

Home Automation with Microcontroller and Networking

Abstract

The Internet now enables us to distribute information worldwide without time constraint, permitting information to be displayed on any client platform. This has generated great impact on the processing and control of information/knowledge acquisition in home and manufacturing/commerce automation. The ability to acquire information and even to control instruments/devices at fingertips over the Internet is becoming desirable not only to the professionals but also to the end users in general. Thanks to the development of Internet Technology, distance monitoring and control of devices are emerging realities. Individuals nowadays can access information and receive signals at home over the Internet. Network connections can be used for transmitting and receiving data from a microcontroller because in most cases, the infrastructure already exists and hence there is no need to install a medium for the data to travel. This paper will present the home automation project designed/developed by the students based on a socket connection functions from the Java.net library. The microcontroller and network connections in conjunction with sensors are used to control devices in a typical home. For simplicity this paper will explore only three outputs from the microcontroller and two analog inputs. The analog inputs will be used to measure temperature and humidity. This system could be extended to run a whole home automation suite.

Introduction

The sensor network is a technique for implementation of ubiquitous computing environment. It is an environment that enables communications with sensors through the microcontrollers. The sensor network environment is being applied in areas such as logistics, environmental control and controlling of devices at home [1]. As the use of the Internet has grown, businesses and home applications have found that the Internet is a low-cost way for mobile users and permanent sites to connect to the business and home network. Clearly it is more cost-effective to connect over the Internet to a private network, than to pay for a leased line or lines to do so [2-[3].

The home automation system as shown in Figure 1 consists of a home computer connected to the internet that has an Ardunio Microcontroller attached to it. The home computer continuously monitors the Ardunio Microcontroller board interfaced to the temperature and humidity sensor. The microcontroller is also interfaced to the opto-isolator to monitor high powered devices. The remote computer is connected to the home computer via the internet. The software processing in the home computer converts the internet command to serial command and communicates with the microcontroller through the USB port.

The subject of automation via the internet has evolved into a major industry and has become more efficient through new technologies. Automation is constantly changing due to rapid advancements and trends that are taking place in hardware and software section of the industry. The purpose of this research was to design and build an automation system that will enable students to investigate current trends in computer communication and embedded systems.

The goal of this research is to create a project that utilizes a network connection to send and receive information from a microcontroller. The paper describes the following:

- Microcontroller
- Sensors
- Opto-Isolators
- Outlets and Relays
- Software (Arduino, Processing and Java.net library)

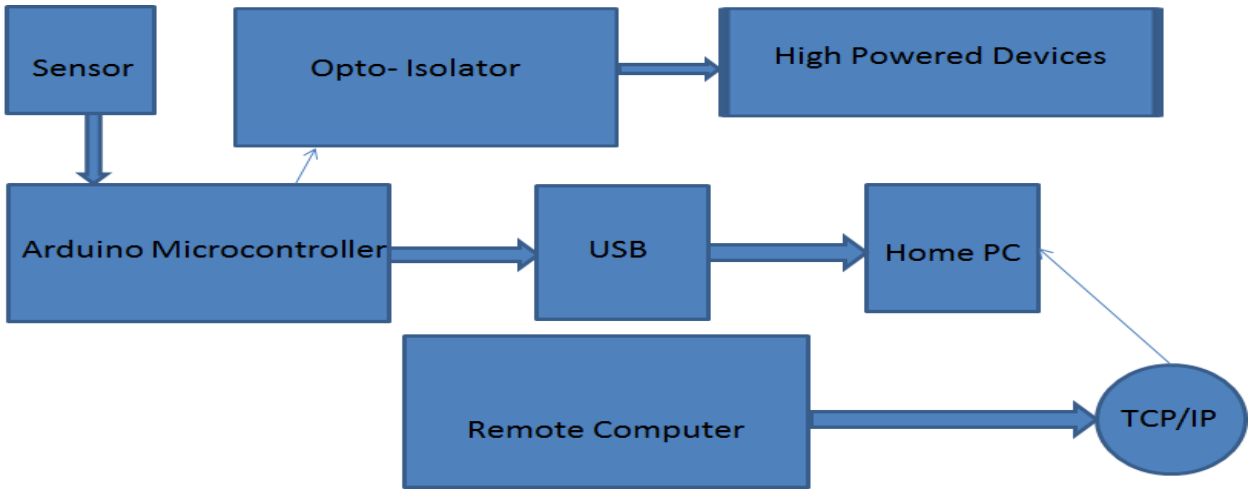

Figure 1.0 Home Automation System

Microcontroller

The microcontroller used in this project is an Ardunio Duemilanove [4]. The microcontroller is directly interfaced to the temperature and humidity sensor to acquire the data at the ADC input of the microcontroller. Furthermore, the microcontroller is also interface to the opto-isolator to control high powered devices through the relays. The Arduino Duemilanove is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB UART (Universal Asynchronous Receiver/Transmitter), a power jack, and an ICSP (In Circuit Serial Programmer) header, and a reset button. It contains everything needed to support the microcontroller. The microcontroller provides all input and output functions for the project. All communication to the computer is handled by the on board FT232RL FDDI UART. Power to the microcontroller is provided by the computer through the USB connection, which is rated at maximum of 5 volts at 500 mA. The calculated maximum current draw over the USB connection is about 114.71mA, so that the microcontroller and all attached components are well within the limits of the USB connection. This project is utilizing three of the digital outputs and two of the analog inputs. Each of the 14 digital pins can sink or source a maximum of 40mA. Each of the digital pins will source about 11.5mA to the connected devices. This is well within the limits of the ATmega 328 with plenty of room to spare. The analog inputs provide ten bits of resolution through the ATmega328's analog to digital converter. This allows the five volt reference to be divided into 1024 steps. Figure 2 shows the Ardunio board used in this project.

Figure 2.0 Ardunio Main Board

Temperature Sensors

The temperature sensor used in this project is a MCP9700T, which is a low power voltage output board mount temperature senor manufactured by Microchip [5]. This is an analog sensor that requires a 5 volt supply, and it has output line that is connected to the microcontroller ADC. The sensor has wide temperature from -40 degree centigrade to 125 degree centigrade with worst case accuracy +/- 4 degree centigrade. The output of the sensor is about l0mV per °C with an offset of about 500mV at 0°C. The equation for the output voltage of the sensor is: Vout = TCi *TA + V0°C, where Ta is the temperature coefficient of the sensor, TA is the ambient temperature, and V0-c is the output voltage at 0°C. Figure 3.0 below shows the distribution of V0-c for 108 samples and the distribution of occurrences verse Ta for 108 samples. This device is immune to parasitic capacitances and can drive large capacitive loads. This provides circuit board design flexibility by enabling the device to be remotely located from the microcontroller. Adding some capacitance at the output will improve the transient response by reducing under and overshoots.

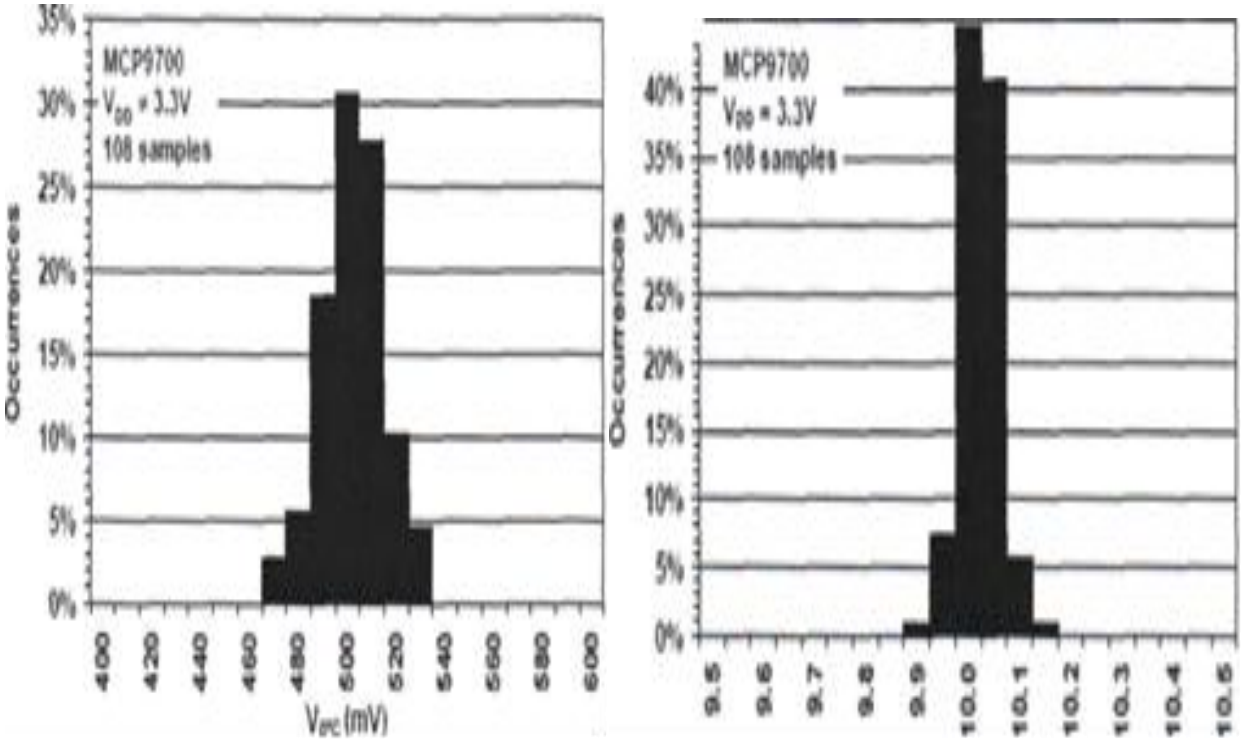

Figure 3.0 Distribution Vo and Ta vs. Occurrences

Humidity Sensor

The humidity sensor used in this project is a HIH-4030, which is a low power analog sensor manufactured by Honeywell [6]. The relative humidity sensor uses a laser trimmed, thermoset polymer capacitive sensing element with on-chip integrated signal conditioning. The sensing element's multilayer construction provides excellent resistance to most application hazards such as condensation, dust, dirt, oils and common environmental chemicals. This sensor has advantages of a fast response time combined with stable, low drift performance. This is an analog sensor that requires a 5 volt supply, and it has an output line that is connected to the microcontrollers ADC. This sensor has an accuracy of ±3.5% over the entire range. The sensor can operate from -40 to 85°C with its full scale from 0 to 100% relative humidity. Figure 4.0 shows the expected output voltage for a specific relative humidity value. The sensor can be order with a calibration sheet which would show the slope and y-intercept for the specific sensor. Unfortunately this was not an option that was offered by our chosen vendor. The values for our sensor were first established with value extrapolated from the graph, but we fine-tuned the variables using an existing standalone sensor. Overall this sensor performs better than expected, and it is very robust and easy to use.

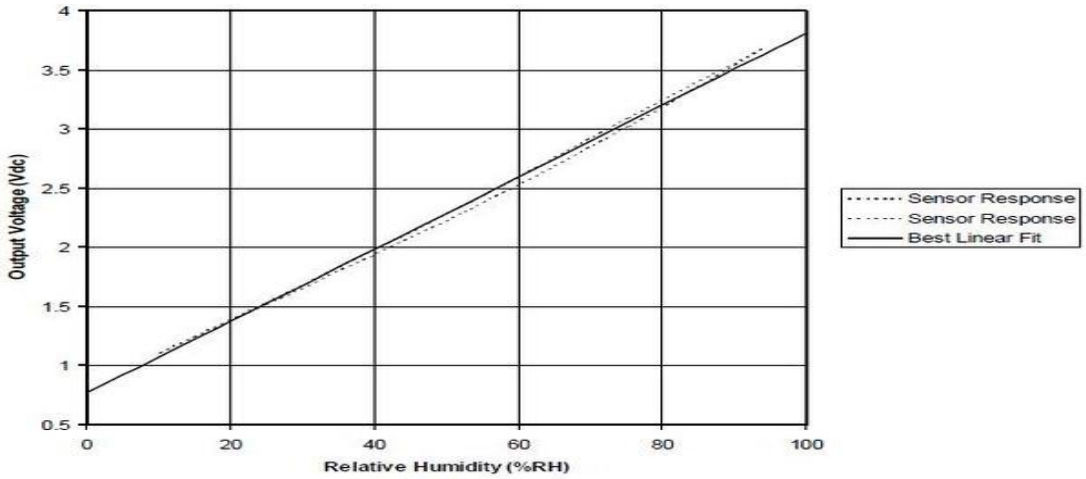

Figure 4.0 Expected Output Voltage vs. Relative Humidity

Hardware: Opto-isolators

The opto-isolator chosen for this project is a 4N35 optocoupler manufactured by Texas Instruments [7]. This is a very common part that was first manufactured over 30 years ago. It consists of a gallium-arsenide diode infrared source, which is optically coupled to a silicon NPN phototransistor. Some more advantages of this part are that it has a high direct-current transfer ratio, a very high isolation between the input and output, and it is UL approved under file number E65085. The direct current transfer ratio is the ratio between the transistors collector current, and the forward current of the infrared diode. The phototransistor can handle the 24volts required by the relays, but the collector current is a maximum of 100mA, so an additional transistor is added at the output to boost the current to a safe level for the relays. The forward voltage of the diode is about 1.5 volts, and a 330Q resistor is used to limit the current to it to

about 10.6mA. This means that it will be drawing 10.6mA from the microcontroller's digital output pin which is rated at 40mA.

The opto-isolators were necessary in order to protect the micro controller. The relays require 24 volts to activate their coil, so they required an external power supply at 24 volts. The opto-isolators were chosen so there could be no chance for the 24 volts to get to the microcontroller, which can only handle 5 volts. Figure 5.0 is the circuit that was used. The 2N3904 can handle double the current of the phototransistor, which makes it better suited to operate the relay. The diode is there to protect the transistor. Current flowing through a relay coil creates a magnetic field which collapses suddenly when the current is switched off. The sudden collapse of the magnetic field induces a brief high voltage across the relay coil which is very likely to damage the transistors. The protection diode allows the induced voltage to drive a brief current through the coil and diode so the magnetic field is dissipated more slowly. This prevents the induced voltage from becoming high enough to cause damage to the transistors.

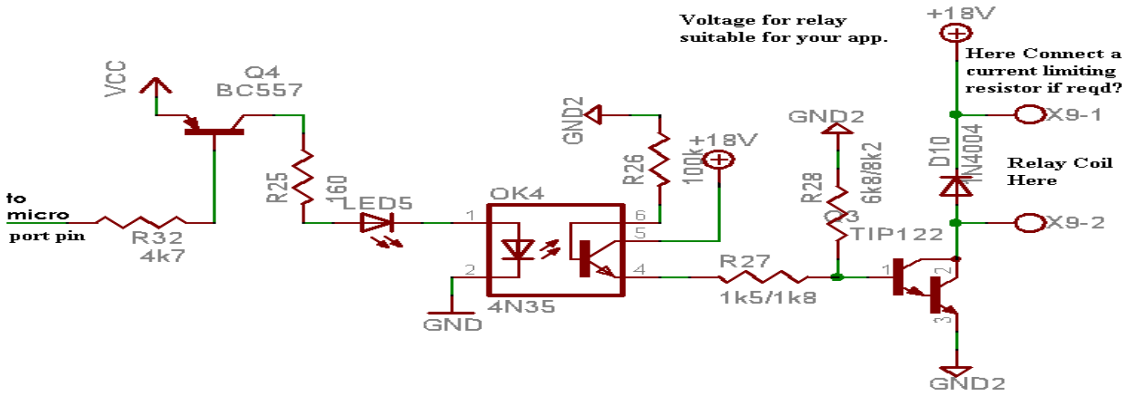

Figure 5.0 Opto coupler interface diagram

Hardware: Outlets and Relays

The relays are connected to the opto-isolators by 22 gauge copper wire. The outlets are connected to a 6 gauge, 25 foot extension cord rated at 15 Amp, with a common neutral. The outlets are safely contained in an outlet box about every 8 feet. Since all three outlets are connected to a single extension cord, the maximum current draw if all three outlets are on should stay at a maximum of 5 Amps. If only one outlet is used, then the maximum current draw is limited by the relay. The relay can handle a maximum current of 12 Amps. The hot wire is tapped at each outlet box and connected to one side of the relays contacts. The other side of the relays contacts is connected to the hot side of the outlet. This allows the outlet to be energized whenever the relays coil is powered by the opto-isolator board. The relays share a common 24 volt line, and the ground is switched by the opto-isolator board. A triac could have been used in place of the relay, but the relay is better suited for this application because of safety concerns when working with 120 volts AC. Figure 6.0 shows how the outlets and relays are wired.

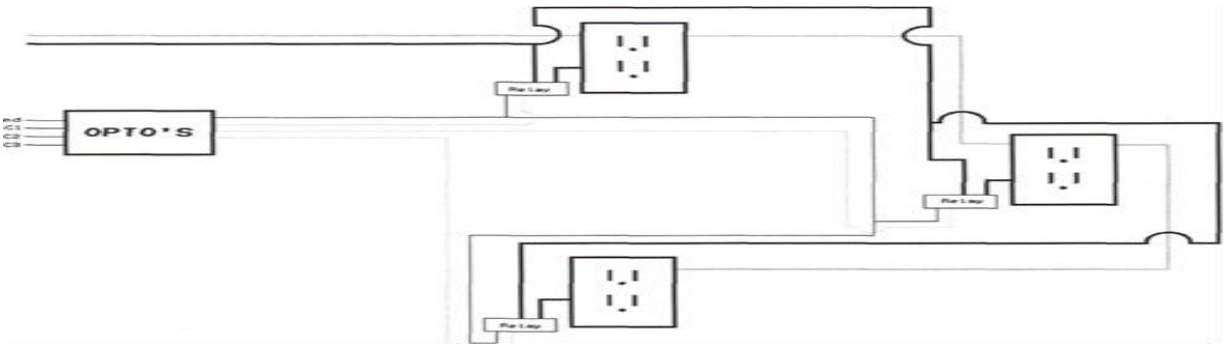

Figure 6.0 Wiring of Output and Relays

Software: Arduino

The Arduino programming language is a set of C/C++ functions that can be called from within the program code. When the code is compiled, it undergoes a minor change which automatically generates the function prototype and passes them directly to a C/C++ compiler called avr-g++. All standard C and C++ constructs that are supported by avr-g++ should work within the Arduino environment. A number of things have to happen in order for program code to get to the ATmega328. First, the Arduino environment performs some small checks and transformations to make sure that the code is correct C or C++. It then gets passed to a compiler the compiler which turns the human readable code into machine readable instructions called object files. Then, the code gets linked with the standard Arduino libraries that provide basic functions like digitalWrite() or Serial.print(). The result is a single Intel hex file, which contains the specific bytes that need to be written to the program memory of the ATmega328 on the Arduino board. This file is then uploaded to the board over the USB connection via the boot loader already on the chip.

The transformations that the Arduino environment performs to the program code include the concatenation of all the tabs in the code before passing it to the avr-g++ compiler. First, ^include "WProgram.h" is added to the top of the code. This header file includes all of the definitions needed for the standard Arduino core. Next, the environment searches for function definitions within the code and creates prototypes for them. These are inserted after any comments or pre-processor directives such as #includes or ftdefines, but before any other statements. Finally, the contents of the current target's main.cxx file are appended to the bottom of the code.

When a program is verified, it is built in a temporary directory in the system temp folder. When you upload it, it is built in the applet/ subdirectory of where the program is saved. The .c and .cpp files of the target are compiled and output with .0 extensions to this directory, as is the main program file and any other .c or .cpp files in the sketch and any .c or .cpp files in any libraries which are ^included in the sketch. These .0 files are then linked together into a static library and the main program file is linked against this library. Only the parts of the library needed for the program are included in the final .hex file, which reduces the total size of the program. The .hex file is the final output of the compilation which is then uploaded to the board. During a "Verify" the .hex file is written to the temp folder. During upload, it's written to the applet sub-directory of the program directory. Below is the main interface of the Arduino programming library [8].

Arduino - 0005 Alpha

```
*       Created 1 June 2005
*       copy left 2885 DojoDave <fittp://www.0j0,org>
*       http://ordui no.berlios.de
*
*       based on an original by H. Barragan for the Wiring i/o board
*/
int. LedPin = 13;       // LED connected to digital pin 1
void setup()
{
pinHode(LedPin, OUTPUT);    // sets the digital pin as output
}
void loop()
digitalWrite(LedPin,   HIGH);  // sets the LED on
delay(1000);    // waits for a second
digitatWrite(ledPin,    LOW);   // sets the LED off
delay (200);     /'/ waits for a second
}
```

Software: Processing

Processing is a programming language, development environment, and online community that since 2001 has promoted software literacy within the visual arts. It was initially created to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing quickly developed into a tool for creating finished professional work as well. Processing is a free, open source alternative to proprietary software tools with expensive licenses, making it accessible to schools and individual students [9]. Its open source status encourages the community participation and collaboration that is vital to Processing's growth. Contributors share programs, contribute code, answer questions in the discussion forum, and build libraries to extend the possibilities of the software. The Processing community has written over seventy libraries to facilitate computer vision, data visualization, music, networking, and electronics.

Software: Java.net

Within the processing environment, we are including a library called processing.net. This library contains the java.net package, which is a library which includes many functions to create network connections. Our project utilizes the socket connect class.

A server application normally listens to a specific port waiting for connection requests from a client. When a connection request arrives, the client and the server establish a dedicated connection over which they can communicate. During the connection process, the client is assigned a local port number, and binds a socket to it. The client talks to the server by writing to the socket and gets information from the server by reading from it. Similarly, the server gets a

new local port number (it needs a new port number so that it can continue to listen for connection requests on the original port). The server also binds a socket to its local port and communicates with the client by reading from and writing to it. The client and the server must agree on a protocol-that is, they must agree on the language of the information transferred back and forth through the socket. The java.net package in the Java development environment provides a class-Socket-that represents one end of a two-way connection between your Java program and another program on the network. The Socket class implements the client side of the two-way link. If you are writing server software, you will also be interested in the ServerSocket class which implements the server side of the two-way link.

Student Outreach

The project offered an opportunity for students to work in a team. This activity focused on important learning concepts such as Electronics, Computer Networking, Programming, Teamwork, and Cross Disciplinary Interaction. Electronics symbolize the interrelationship between various substructures of the circuit. This includes an understanding of electronic components and the manner in which all these components function together as a deterministic whole system. Integrating these components offered an opportunity for the students to understand the design/development of Microcontroller and Networking systems. The project carried out the concept of teamwork in all phases of design and implementation. The goal of linking the students into a learning community is to give the student a peer group in which they feel comfortable. The team work prepares the students to solve technical problems in a group environment in addition to meet new challenges encountered in the work place. Students experience being on successful team is to appreciate and understand the value of good team work.

The project emphasizes on the word team because team is not the same as a group. The term group implies a somewhat more than a collection of individuals but the team implies much more. The curriculum in any specific area of study tends to narrowly focus students on that area, whereas real-world multifaceted systems tend to incorporate components from multiple areas in electronics engineering technology. The development of such systems has shifted from designing individual components in segregation to working in cross-functional teams that include the variety of proficiencies needed to design an entire system .The Home Automation with Microcontroller and Networking project provides an opportunity for students interested in electronics, computer networking, design, application and troubleshooting to combine their interest in building a project. The results show that the students learned tangible lessons from each topic. The students that worked on this project enjoyed the hands on experience and being able to implement it in a real life situation in a classroom setting. Feedback from the students indicates that they were enthusiastic about implementing the concepts learned from Computer Networking and Microcontroller courses. However, this project was completed as part of the requirement for Advanced Networking course. I am planning to implement this concept in my other courses also. This concept gives an opportunity to our students to think outside the academics and implement real life situation in an academic environment.

Conclusion

In conclusion to this project there are other applications that go so far as to home automation, industrial process control, and anything in which you need to remotely turn on and off a load and read in sensor values of analog or digital values. Future additions to the project could include modifications of program code to include items such as error checking and a more aesthetically pleasing user interface. In future implementations of the project we plan to include weather control modules on a standalone platform that do not require a second computer to implement the network functions. This was a good project that included concepts from almost every class that we have taken.

References

[2] H. Kruegle, "CCTV Surveillance: Analog and Digital Video Practices and Technology ", Butterworth-Heinemann Publisher, 2006
[3] E.M. Hardwood, "Digital CCTV: A Security Professional's Guide", Butterworth-Heinemann Publisher, 2007.
[4] http://www.sparkfun.com/products/666
[5] http://ww1.microchip.com/downloads/en/DeviceDoc/21942c.pdf
[6] http://www.sparkfun.com/datasheets/Sensors/Weather/SEN-09569-HIH-4030-datasheet.pdf
[7] http://measure.feld.cvut.cz/groups/edu/osv/4N35.pdf
[8] http://arduino.cc/en/Reference/Libraries
[9] http://www.processing.org/