Environmental Monitoring of Smart Cities Using IoT-Based Sensor Networks for Air Quality, Noise, and Urban Health Surveillance

Dr. Ahmed Al-Mansouri[1]*
[1]United Arab Emirates University, College of Medicine and Health Sciences, Department of Biomedical Engineering and Smart Urban Health Systems, Al Ain, UAE

ABSTRACT

This project is to design and implement an IOT based smart city using Arduino Uno. To create an urban IOT system that helps to achieve the smart city and to solve the problems. Now a day's automation plays important role. Energy consumption on street lights is the major problem; Over a Trillion Kilowatts are expanded on it. By using Automatic Street light control system it can be overcome. Air pollution has significant influence on the concentration of constituents in the atmosphere leading to effects on environment. Other physical parameters like temperature and humidity are also affected by atmospheric conditions. In Environmental Monitoring system, by using sensors we can monitor them.

Keywords: *Arduino Uno, LCD, Temperature sensor, Humidity sensor, IR sensor IoT, Buzzer.*

I. INTRODUCTION

1.1 Introduction of embedded system

An Embedded System is a combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a specific function. This is in direct contrast to the personal computer in the family room. As its name suggests, Embedded means something that is attached to another thing. An embedded system can be thought of as a computer hardware system having software embedded in it. An embedded system can be an independent system or it can be a part of a large system. An embedded system is a microcontroller or microprocessor based system which is designed to perform a Processor is the heart of an embedded system. It is the basic unit that takes inputs and produces an output after processing the data.

It is comprised of computer hardware and software and mechanical components (disk drives, for example). However, a personal computer is not designed to perform a specific function rather; it is able to do many different things. Many people use the term general-purpose computer to make this distinction clear. As shipped, a general-purpose computer is a blank slate; the manufacturer does not know what the customer will do wish it. One customer may use it for a network file server another may use it exclusively for playing games, and a third may use it to write the next great American novel.

Frequently, an embedded system is a component within some larger system. For example, modern cars and trucks contain many embedded systems. One embedded system controls the anti-lock brakes, other monitors and controls the vehicle's emissions, and a third displays information on the dashboard. In some cases, these embedded systems are connected by some sort of a communication network, but that is certainly not a requirement. At the possible risk of confusing you, it is important to point out that a general-purpose computer is itself made up of numerous embedded systems. For example, my computer consists of a keyboard, mouse, video card, modem, hard drive, floppy drive, and sound card-each of which is an embedded system. Each of these devices contains a processor and software and is designed to perform a specific function. For example, the modem is designed to send and receive digital data over analog telephone line. That's it and all of the other devices can be summarized in a single sentence as well.

If an embedded system is designed well, the existence of the processor and software could be completely unnoticed by the user of the device. Such is the case for a microwave oven, VCR, or alarm clock. In some cases, it would even be possible to build an equivalent device that does not contain the processor and software. This could be done by replacing the combination with a custom integrated circuit that performs the same functions in hardware. However, a lot of flexibility is lost when a design is hard-cooled in this way. It is much easier, and cheaper, to change a few lines of software than to redesign a piece of custom hardware.

An embedded system has three components:

- It has hardware.
- It has application software.
- It has Real Time Operating system (RTOS) that supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small scale embedded system may not have RTOS.

So we can define an embedded system as a Microcontroller based, software driven, and reliable, real-time control system.

1.1.1 History and Feature

Given the definition of embedded systems earlier is this chapter; the first such systems could not possibly have appeared before 1971. That was the year Intel introduced the world's first microprocessor. This chip, the 4004, was designed for use in a line of business calculators produced by the Japanese Company Busicom. In 1969, Busicom asked Intel to design a set of custom integrated circuits-one for each of their new calculator models. The 4004 was Intel's response rather than design custom hardware for each calculator, Intel proposed a general-purpose circuit that could be used throughout the entire line of calculators. Intel's idea was that the software would give each calculator its unique set of features.

The microcontroller was an overnight success, and its use increased steadily over the next decade. Early embedded applications included unmanned space probes, computerized traffic lights, and aircraft flight control systems. In the 1980s, embedded systems quietly rode the waves of the microcomputer age and brought microprocessors into every part of our kitchens (bread machines, food processors, and microwave ovens), living rooms (televisions, stereos, and remote controls), and workplaces (fax machines, pagers, laser printers, cash registers, and credit card readers).

It seems inevitable that the number of embedded systems will continue to increase rapidly. Already there are promising new embedded devices that have enormous market potential; light switches and thermostats that can be central computer, intelligent air-bag systems that don't inflate when children or small adults are present, pal-sized electronic organizers and personal digital assistants (PDAs), digital cameras, and dashboard navigation systems. Clearly, individuals who possess the skills and desire to design the next generation of embedded systems will be in demand for quite some time.

1.1.2 Real Time Systems

One subclass of embedded is worthy of an introduction at this point. As commonly defined, a real-time system is a computer system that has timing constraints. In other words, a real-time system is partly specified in terms of its ability to make certain calculations or decisions in a timely manner. These important calculations are said to have deadlines for completion. And, for all practical purposes, a missed deadline is just as bad as a wrong answer.

The issue of what if a deadline is missed is a crucial one. For example, if the real-time system is part of an airplane's flight control system, it is possible for the lives of the passengers and crew to be endangered by a single missed deadline. However, if instead the system is involved in satellite communication, the damage could be limited to a single corrupt data packet. The more severe the consequences, the more likely it will be said that the deadline is "hard" and thus, the system is a hard real-time system. Real-time systems at the other end of this discussion are said to have "soft" deadlines. All of the topics and examples presented in this book are applicable to the designers of real-time system who is more delight in his work. He must guarantee reliable operation of the software and hardware under all the possible conditions and to the degree that human lives depend upon three system's proper execution, engineering calculations and descriptive paperwork.

1.1.3 Application Areas

Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, data communication, telecommunications, transportation, military and so on.

1.1.4 Consumer appliances

At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video recorders etc. Today's high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems

1.15 Office automation

The office automation products using embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

1.1.6 Industrial automation

Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

1.1.7 Medical electronics

Allmost every medical equipment in the hospital is an embedded system. Thisequipment's include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonoscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

II. BLOCK DIAGRAM

2.1 Block Diagram

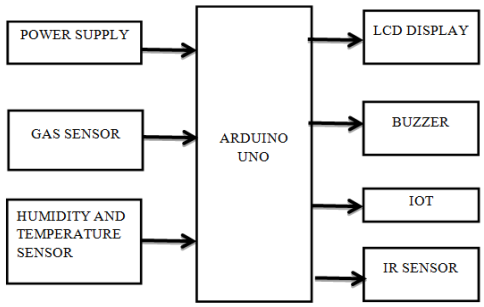

Fig: 2.1 Block Diagram of IOT BASED SMART CITY

2.2 working

Now a day, street lighting systems in industries or cities are growing rapidly. It offers safety and comfort during the dark hours. But it creates some problems like high energy consumption. Several Trillion Kilowatts are expanded on it. In this Automatic street light control system is used to control and decrease energy consumption. It does not need manual operation to turn ON/OFF the street lights. By using sensor we can operate the street lights automatically. It detect the movement of a vehicle on highways or roads to turn ON the lights when the vehicle is ahead of the lights, and turns OFF the glowing light whenever the vehicle passes away from the lights. This process is done by using an IR sensor. It has IR LED acts as transmitter and IR photodiode acts as receiver. Transmitter radiates radiation when any object comes in front of it reflects and the receiver detects it, then an obstacle is created. By using this, the street lights are automatically turned ON/OFF.

In Environmental monitoring, there are many problems like pollutions, etc. Now a day's air pollution is the major problem caused due to harm full gases released in the atmosphere. Inhaling this air causes several health problems. And it is necessary to know the temperature and humidity conditions in the environment and industries etc. In

Environmental monitoring, the harm full gases and physical parameters like Temperature and Humidity are monitored and collects the data by sensors. By using temperature and humidity (DHT 11 sensor) sensor we can know the values of them. We can see the values in LCD display. Similarly gases present in atmosphere are monitor by MQ-2 sensor and values displayed in LCD. These values are represented in graph. We can see the graph by using mobile phones.

III. HARDWARE REQUIREMENTS

The following are the hardware components used in the IOT based smart city

- ARDUINO UNO
- Humidity Sensor
- IR Sensor
- Gas Sensor
- LCD Display
- IOT
- Buzzer
- Power Supply

3.1 ARDUINO UNO

Arduino is open source physical processing which is based on a microcontroller board and an incorporated development environment for the board to be programmed. Arduino gains a few inputs, for example, switches or sensors and control a few multiple outputs, for example, lights, engine and others. Arduino program can run on Windows, Macintosh and Linux operating systems (OS) opposite to most microcontrollers' frameworks which run only on Windows. Arduino programming is easy to learn and apply to beginners and amateurs. Arduino is an instrument used to build a better version of a computer which can control, interact and sense more than a normal desktop computer. It's an open-source physical processing stage focused around a straightforward microcontroller board, and an environment for composing programs for the board. Arduino can be utilized to create interactive items, taking inputs from a diverse collection of switches or sensors, and controlling an assortment of lights, engines, and other physical outputs. Arduino activities can be remaining solitary, or they can be associated with programs running on your machine. The board can be amassed by hand or bought preassembled; the open-source IDE can be downloaded free of charge. Focused around the Processing media programming environment, the Arduino programming is an execution of Wiring, a comparative physical computing platform.

3.1.1 Why choosing Arduino

There are numerous different microcontrollers and microcontroller platforms accessible for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and numerous others offer comparative usefulness. These apparatuses take the chaotic subtle elements of microcontroller programming and wrap it up in a simple to-utilize bundle. Arduino additionally rearranges the methodology of working with microcontrollers; moreover it offers some advantages for instructors, students, and intrigued individuals:

Inexpensive

Arduino boards are moderately cheap compared with other microcontroller boards. The cheapest version of the Arduino module can be amassed by hand, and even the preassembled Arduino modules cost short of what $50.

Cross-platform

The Arduino programming runs multiple operating systems Windows, Macintosh OSX, and Linux working frameworks. So we conclude that Arduino has an advantage as most microcontroller frameworks are constrained to Windows.

Straightforward, clear programming method

The Arduino programming environment is easy to use for novices, yet sufficiently versatile for cutting edge customers to adventure as well. For educators, it's favorably engaged around the Processing programming environment, sounder studies finding ways to understand how to program in that environment will be familiar with the nature of Arduino.

Open source and extensible programming

The Arduino program language is available as open source, available for development by experienced engineers. The lingo can be reached out through C++ libraries, and people expecting to understand the specific purposes of different interests can make the leap from Arduino to the AVR C programming language on which it is based. Basically, you can incorporate AVR-C code clearly into your Arduino programs if you have to. Open source and extensible hardware - The Arduino is concentrated around Atmel's Atmega8 and Atmega168 microcontrollers. The plans for the modules are circulated under a Creative Commons license, so experienced circuit designers can make their own particular interpretation of the module, extending it and improving it. Slightly inexperienced customers can build the breadboard variation of the module remembering the finished objective to perceive how it capacities and save money.

3.1.2 Circuit Explanation

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). 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 connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform;

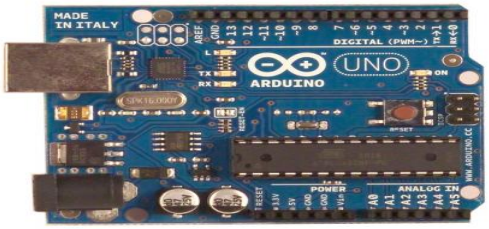

Fig: 3.1 circuit diagram of Arduino

3.1.3 Technical specifications of Arduino:

- Microcontroller : ATmega328
- Operating Voltage : 5V
- Input Voltage (recommended) : 7-12V
- Input Voltage (limits) : 6-20V
- Digital I/O Pins 14 (of which 6 provide PWM output)
- Analog Input Pins 6
- DC Current per I/O Pin 40 mA
- DC Current for 3.3V Pin 50 mA
- Flash Memory 32 KB of which 0.5 KB used by
- Boot loader
- SRAM 2 KB
- EEPROM 1 KB
- Clock Speed 16 MHz

3.1.4 Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Ground and VIN pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five

volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5VThe regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND. Ground pins.

3.1.5 Memory

The Atmega328 has 32 KB of flash memory for storing code (of which 0, 5 KB is used for the boot loader); it has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

3.1.6 Input/Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pin Mode (), digital Write (), and digital Read () functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 ohms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt () function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analog Write () function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, each of which provides 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analog Reference () function. Additionally, some pins have specialized functionality:
- I 2C: 4 (SDA) and 5 (SCL). Support I2C (TWI) communication using the Wire library. There are a couple of other pins on the board:
- AREF. Reference voltage for the analog inputs. Used with analog Reference () Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

IV. RESULTS

4.1 circuit diagram

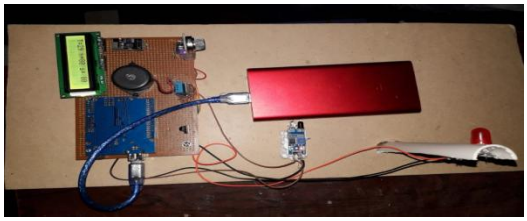

Fig: 4.1 Experimental Circuit

When any obstacle is created near the IR sensor then automatically the LED will be ON. When it is away LED will be OFF.

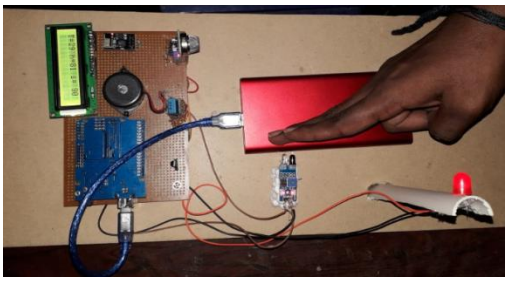

Fig: 4.1.1 When LED is ON due to obstacle

Temperature, humidity and smoke values are displayed in LCD.

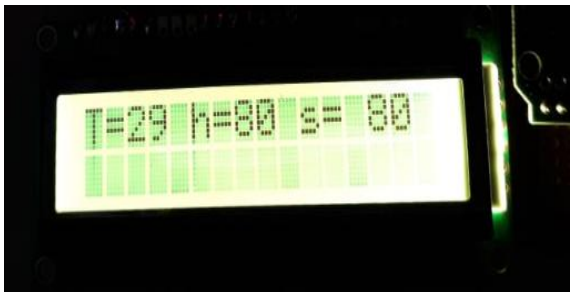

Fig: 4.1.2 values are displayed in LCD

4.2 Graphs

Temperature, humidity and smoke variations in the atmosphere.

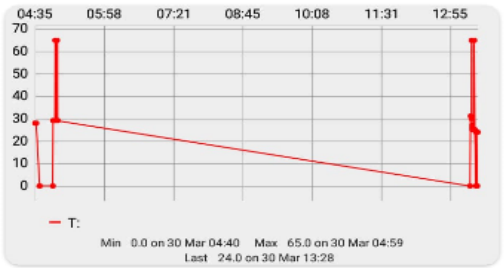

Fig: 4.2.1 temperature /time

Fig: 4.2.2 humidity/time

Fig: 4.2.3 smoke/time

V. CONCLUSION AND FUTURE SCOPE

Conclusion

Automatic street light control, it detect the movement of a vehicle on highways or roads to turn ON the lights when the vehicle is ahead of the lights , and turns OFF the glowing light whenever the vehicle passes away from the lights. In environmental monitoring, by using temperature, humidity and gases present in atmosphere is monitor by using sensors.

Future scope

The above project can be developed by solar street light system with Automatic Street light control process. It operates and control the lights either by a mobile or computer from any location.

We can Interface more number of sensors to know detail content of all gases present in air, physical parameters like temperature and humidity. Designing of webpage and upload data with time and date. We can interface module.

REFERENCES

1. *A. S. Jalan, "A survey on automatic street lightning system on indian streets using Arduino," Int. J. Innovative Res. Sci. Eng. Technol., vol. 6, no. 3, pp. 4139-4144, 2017.*
2. *H. Satyaseel, G. Sahu, M. Agarwal and J. Priya, "Light intensity monitoring & automation of street light control by Iot," Int. J. Innovations Adv. Comput. Sci., vol. 6, no. 10, pp. 34-40, 2017.*
3. *A. Rao and A. Konnur, "Street light automation system using arduinouno," Int. J. Innovative Res. Comput. Commun.Eng., vol. 5, no. 11, pp. 16499-16507, 2017.*
4. *Krishna, V. Siva, and S. Arun. "Embedded System Based Air Pollution Detection in Vehicles." (2015).*
5. *RiteekaNayak, Malaya RanjanPanigrahy , Vivek Kumar Rai and T AppaRao:IOT based air pollution monitoring system Vol-3, Issue-4, 2017.*
6. *Shanko, Eriola J., and Michalis G. Papoutsidakis. "Real time health monitoring and wireless transmission: A μController application to improve human medical needs." E-Health and Bioengineering Conference (EHB), 2013.IEEE, 2013.Circuitbasics, 2017.*
7. *How to set up the DHT11 humidity sensor. [Online] Available at: http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-onan-arduino/ [Accessed 25 10 2017].*