Kyaukse University Research Journal, 2025, Vol.IV

DESIGN AND DEVELOPMENT OF ARDUINO BASED DATA LOGGING SYSTEM WITH SENSOR AND SD CARD

Ei Ei Mon[1], Soe Yu Mon[2], Pyae Phyo Ko[3]

Abstract

The environmental temperature and humidity were analyzed with DHT11 sensor and the obtained data were logged with Arduino and SD memory card. This research deals with hardware electronics section, which implemented Arduino Uno development board, data logger shield, SD card, DHT11 temperature and humidity sensor, and software section that controls the microcontroller to acquire the sensor data and to communicate the host PC with serial communication. The obtained value of temperature and humidity can be expressed on the serial monitor and can also be stored in the SD card. The logged data were written in SD card as the ".txt" file format and it can be read by any computer and mobile devices which have the serial communication interface.

Keywords: Arduino, DHT11, SD card, Serial Monitor

Introduction

Data logging refers to the process of collecting and storing data for use at a later time. A data logger is essentially an electronic device designed to collect and store data continuously over a period of time. An Arduino Uno board, data logger shield, DHT11 sensor, program, personal computer (PC), USB cable, and Arduino IDE software are required for logging the temperature and humidity. The program is developed in the C language and serves as the code that is uploaded to the Arduino board for execution.

The SD card logger shield uses SD cards to store data over long periods. Although the Arduino chip includes built-in EEPROM memory, its capacity is only a few hundred bytes-very small when compared to an SD card with a capacity of 4 GB. SD cards are cheap in price and simple to interface; it is an obvious choice for long time storage.

Arduino is an open-source electronic prototyping board designed with flexible, beginner-friendly hardware and software. Arduino can sense the temperature by receiving input from DHT11 temperature and humidity sensors. The temperature and humidity can be displayed on the serial monitor and logged into the SD card. Its function was mainly designed to acquire the send messages from a computer to an Arduino board (over USB) and also to follow the commands from the Arduino. As shown in figure 1, the block diagram represents the constructed system.

[1] Demonstrator, Department of Physics, Kyaukse University
[2] Demonstrator, Department of Physics, Kyaukse University
[3] Demonstrator, Department of Physics, Kyaukse University

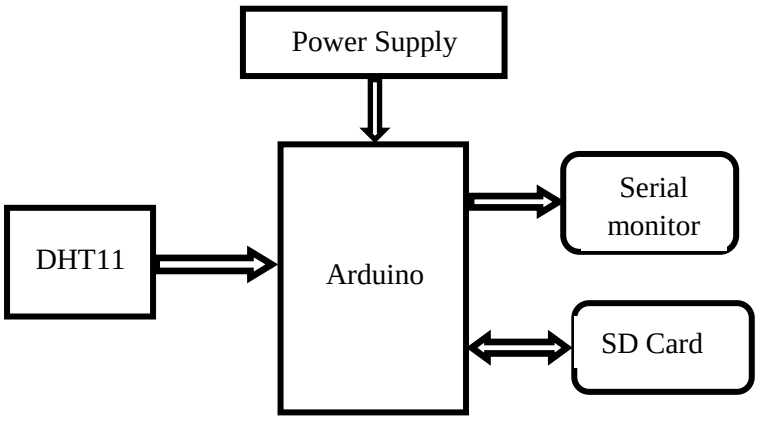

Figure 1 Block diagram of data logging system

In this research work, Design and Development of Arduino Based Data Logging System with Sensor and SD Card are studied. This work is conducted from August, 2023 to August, 2024.

Hardware and Software Requirements

Arduino Uno board, DHT11 temperature and humidity sensor, SD card breakout as data logger, micro SD memory card, and USB communication cable, are main parts in the temperature and humidity logger. SD and SPI libraries are required to implement as the data logging protocol. The hardware that can run the program, which was coded and compiled with Arduino IDE (integrated development environment) software.

Arduino Uno

Based on the ATmega328 chip, the Arduino Uno is a microcontroller board. The board features an 8-bit MCU and includes 14 digital input/output pins, with 6 of them supporting PWM output(Kumar et al., 2015). It also has 6 analog pins from A0 to A5**.** The Uno board is equipped with internal peripherals that support SPI, UART, and I2C communication protocols. The Uno board, as shown in figure 2, has a memory capacity of up to 30 Kbytes and a crystal oscillator of 16MHz**,** which is its operating frequency.

Figure 2 Arduino Uno and USB printer cable

The Arduino circuit diagram as shown in figure 3, consists of two principal sections. The left section is the power supply section that drops the voltage to a steady 5 V. There is an LED that is lit to show that power is available(Ibrahim, 2012). The right section contains the microcontroller, a reset button, programming pins, and an additional LED(Ibrahim, 2012). This specific LED is linked to the ATmega 328 pin referred to as Arduino pin 13.

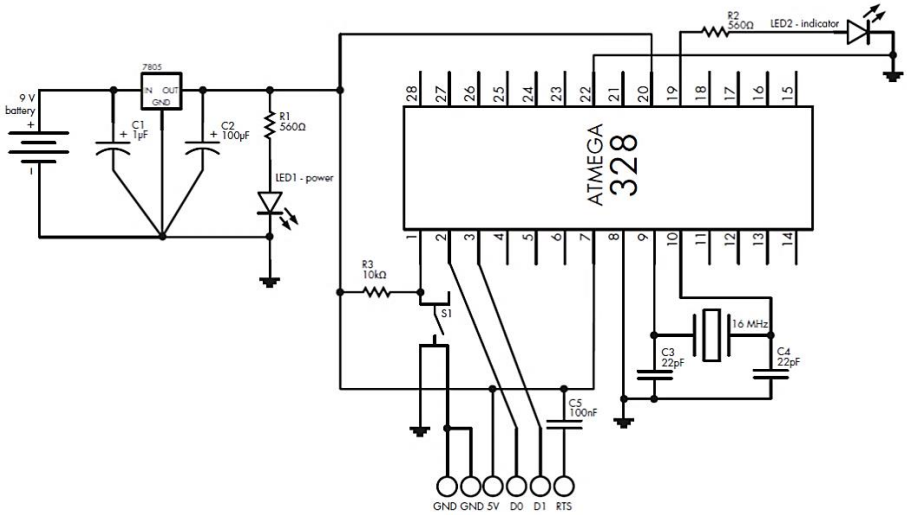

Figure 3 Schematic circuit of Arduino board

MicroSD Card Module

SD cards or microSD cards are widely used in numerous applications, such as data logging, data visualization, and many more. A microSD card reader and writer with an involved 3.3 V regulator and level shifters that make it safe to interface with most 5 V and 3.3 V microcontrollers, including the Arduino Uno, Mega, Leonardo, Due, etc(Ibrahim, 2010). The module uses an SPI interface for communication and is also compatible with the standard Arduino SD card library in Arduino IDE for normal use(Ibrahim, 2010).

Configuring it is as easy as connecting the header pins of the microSD card adapter to the Arduino (Ashwin, 2019). The micro SD card adapter, as shown in figure 4, is connected to the Arduino board. Digital pins 4, 11, 12, and 13 are connected to the CS, MOSI, MISO, and SCK of the microSD card adapter(Ashwin, 2019).

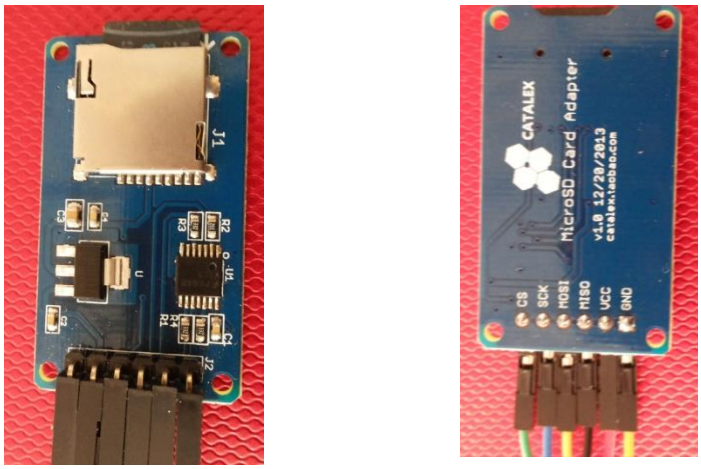

Figure 4 MicroSD card adapter

The MicroSD Memory Card

By interfacing microSD cards with Arduino, it becomes possible to log and save data with expanded storage space from different sources, including temperature sensors. Moreover, the microSD card can function as storage for web server data or other file types. To record and store the collected data, a microSD memory card is

needed to place in the shield(Sen et al., 2019). This memory interface shield is compatible with the micro SDHC cards with a capacity of up to 32 GB. A microSD card with a 4 GB capacity is shown in figure 5.

Figure 5 A microSD card with a 4GB capacity.

The secure digital (SD) card is a non-volatile memory card format developed by the SD card association for use in portable devices(Nath et al., 2022). The micro SD card has been formatted as FAT32 before interfacing with the MCU. The SD card only accepts the standard SD commands. By using these commands, the MCU can read the SD card registers and also read/write the memory core(Satheesh et al., 2016). It is based on flash memory technology and widely used in digital cameras, cell phones, ebook readers, tablet computers, netbook computers, media players, GPS receivers, and video game consoles(Lu et al., 2021).

File allocation table (FAT) is a computer file system architecture widely used on most computer systems and memory cards, owing to its comparative simplicity(Rusbarsky & City, 2012). The FAT file system is quite straightforward technically and supported by virtually all existing operating systems for personal computers, which makes it a useful format for flash memory cards and a convenient way to share data between operating systems(Rusbarsky & City, 2012).

DHT11 Temperature and Humidity Sensor

DHT11 temperature and humidity sensor, as illustrated in figure 6, was used as temperature and humidity sensor, and it could produce a calibrated digital signal output. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability, and cost-effectiveness(Manghnani et al., 2017a).

DHT sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of ±2°C and ±5%RH(Kuria et al., 2020). For measuring temperature, the sensor uses an NTC thermistor, which means that the resistance decreases with an increase in temperature, as shown in the graph of figure 7(Atanasijevic & Mihailovic, 2019). The humidity sensing capacitor in the DHT sensor has two electrodes with a moisture-holding substrate as a dielectric between them, as shown in figure 8. Change in the capacitance value occurs with a change in humidity level(Kuria et al., 2020).

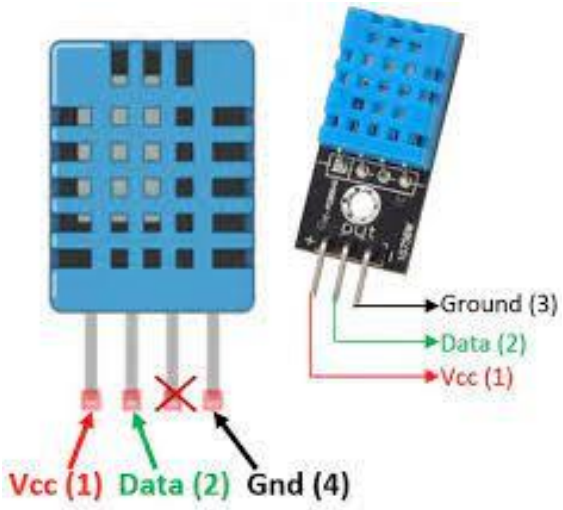

Figure 6 DHT11 temperature and humidity sensor

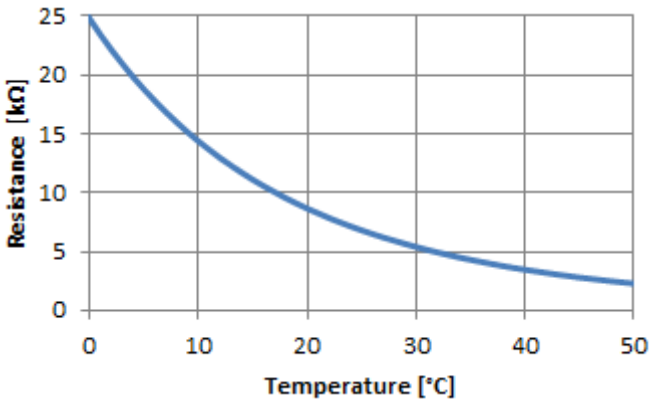

Figure 7 NTC thermistor characteristics curve

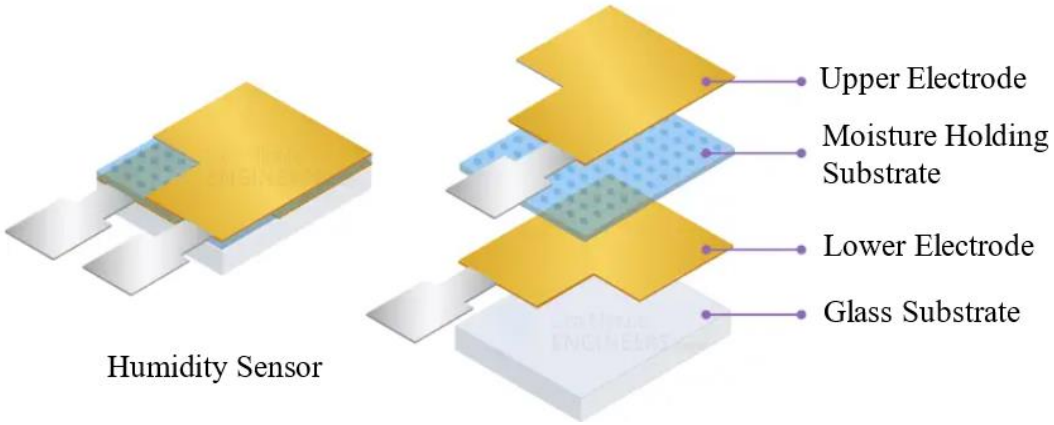

Figure 8 Internal structure of humidity sensor

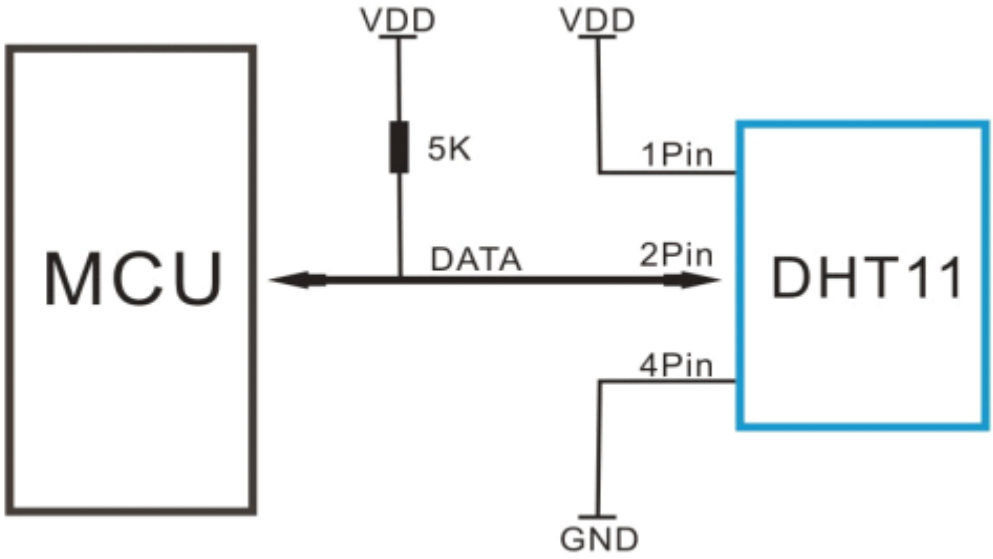

Figure 9 Communication process: serial interface (Single-Wire Two-Way)

DHT11 is a single-wire digital humidity and temperature sensor, which provides humidity and temperature values serially with one-wire protocol. In figure 9, a single-wire data protocol is employed for communication and synchronization between the MCU and the DHT11 sensor(Manghnani et al., 2017b). A single communication cycle takes around 4 ms. Data consists of decimal and integral parts. A complete data transmission is 40 bits, and the sensor sends higher data bits first(Manghnani et al., 2017b).

Data format: 8-bit integral RH data, 8-bit decimal RH data, 8-bit integral T data, 8-bit decimal T data, 8-bit check sum. If the data transmission is right, the checksum should be the last 8-bit of the following data flow: "8-bit integral RH data, 8-bit decimal RH data, 8-bit integral T data, 8-bit decimal T data"(Rohini & Venkatasubramanian, 2015). When the MCU sends a start signal, DHT11 changes from the lower-power mode to the running mode, and it waits for the start signal from the MCU. Once it obtains the trigger pulse, DHT11 sends a response signal of 40-bit data that includes the relative humidity and temperature information to the MCU (Dung, 2017).

In this condition, the obtained data could be selected for temperature or humidity sensing data. Without the start signal from the MCU, DHT11 will not give the response signal to the MCU (Alam, 2019). Once data is collected, DHT11 returns to the low-power mode until it receives a start signal from the MCU again. Overall sensing and communication process is as shown in figure 10 (Alam, 2019).

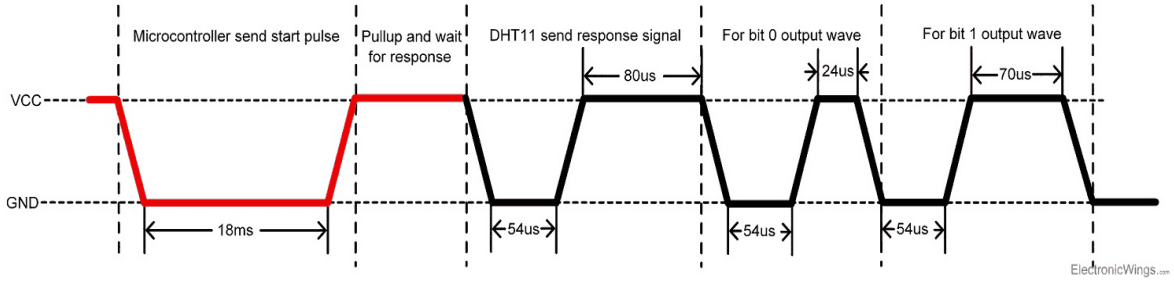

Figure 10 Overall Communication Process

Arduino Programming Language

The setup() function is used to start a sketch program. The setup() function initialized the variables, pin modes, invocation of libraries, etc. The setup() function will only run once, after each power-on or reset of the Arduino board(Evans, 2011b).

After defining a setup() function, the loop() function does just what its name suggests, and loops continuously, allowing the program to change and respond while it runs. Code in the loop() section of the sketch is used to control the Arduino board actively(Evans, 2011a). The code below won't actually do anything, but its form is useful for copying and pasting as a starting point for any sketch(Evans, 2011b). Arduino IDE contains a C/C++ library called "Wiring" which implements many common input/output functions. Arduino programs are written in C/C++, although users only need to define two functions to make a runnable 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(Montironi et al., 2017).

The Serial Monitor

To open the Serial Monitor, begin the IDE and click the Serial Monitor icon button on the toolbar. The Serial Monitor contains an input field at the top, with one line and a Send button, and an output field below, where the Arduino data will appear(Agarwal-Hollands & Andrews, 2001). If the Autoscroll box is checked, the most recent output is displayed, and if the screen becomes full, old data scrolls off the screen as soon as there is new output(Agarwal-Hollands & Andrews, 2001).

To begin the Serial Monitor, it is necessary to start it by placing this line into our sketch within void setup(): Serial.begin(9600);

9600 is the speed at which the data will travel between the computer and the Arduino, otherwise known as baud(Brooks, 2016a). This value must match the speed setting in the bottom right of the Serial Monitor(Brooks, 2016b). To print text to the Serial Monitor so that it will display in the output window, one uses Serial.print: Serial.print ("To display the contents of the variable results, it use "Serial.println (results);

Circuit Design and Sketch

Circuit Design

Temperature and humidity logger system consisted of Arduino UNO, data logging shield, microSD memory card, USB cable. Pin 1 (+5 V), pin 3 (GND), and pin 2 (data pin) of DHT11 were connected to the 5 V, GND and Digital pin D7 of Arduino, respectively. MicroSD card adapter was interfaced to the Arduino board. Digital pin 4, 11, 12, and 13 were connected to the CS, MOSI, MISO and SCK of microSD card adapter. Data logger circuit is shown in figure 11. Circuit connection is drawn in Fritzing software and it is shown in figure 12 and figure 13.

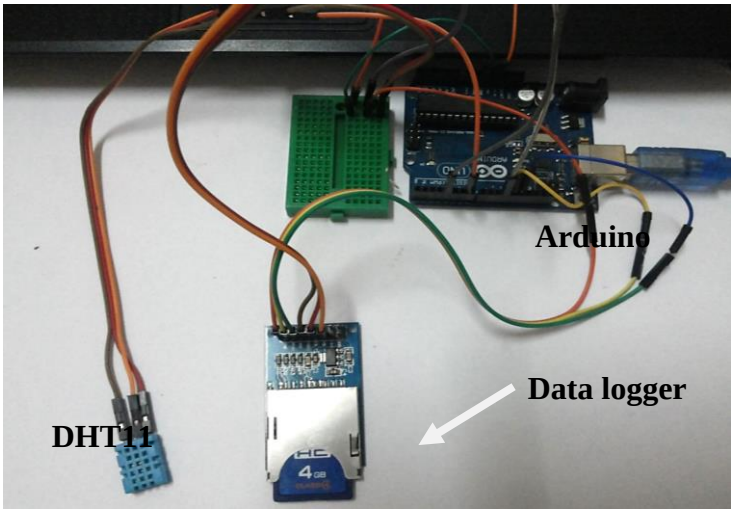

Figure 11 Photograph of data logging system

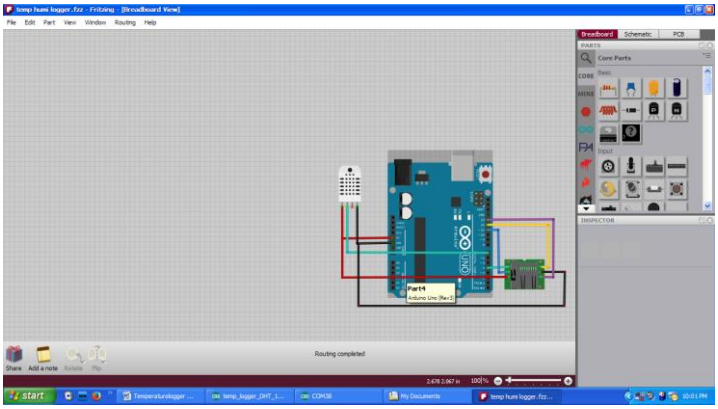

Figure 12 Data logger circuit drawing on the fritzing window

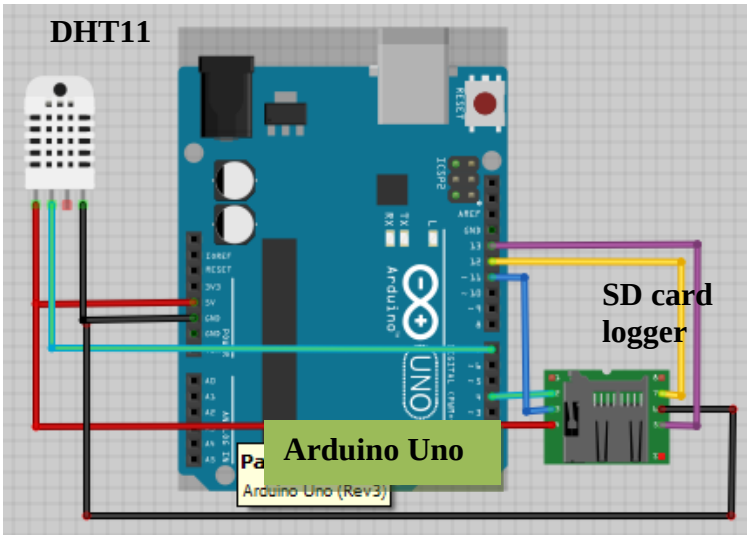

Figure 13 Connection of data logger circuit

Results and Discussion

Data Logging Analysis

In the SD card logger, the DHT11 temperature and humidity sensor were connected to the Arduino microcontroller. The temperature and humidity data would be written on the serial monitor and shown on the serial monitor for every prescribed time, but it could change this time period by changing the value in delay. At once, these data were stored in the microSD card, which was fitted to the SD card logger. The sketch on the Arduino window is shown in figure 14. The data stored and the data on the serial monitor in the SD card logger are shown in figures 15 and 16.

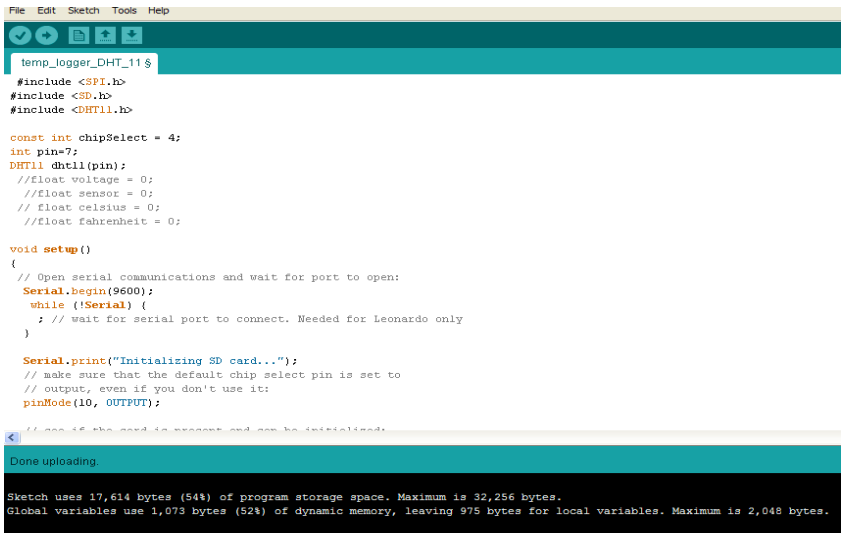

Figure 14 Sketch on the serial monitor

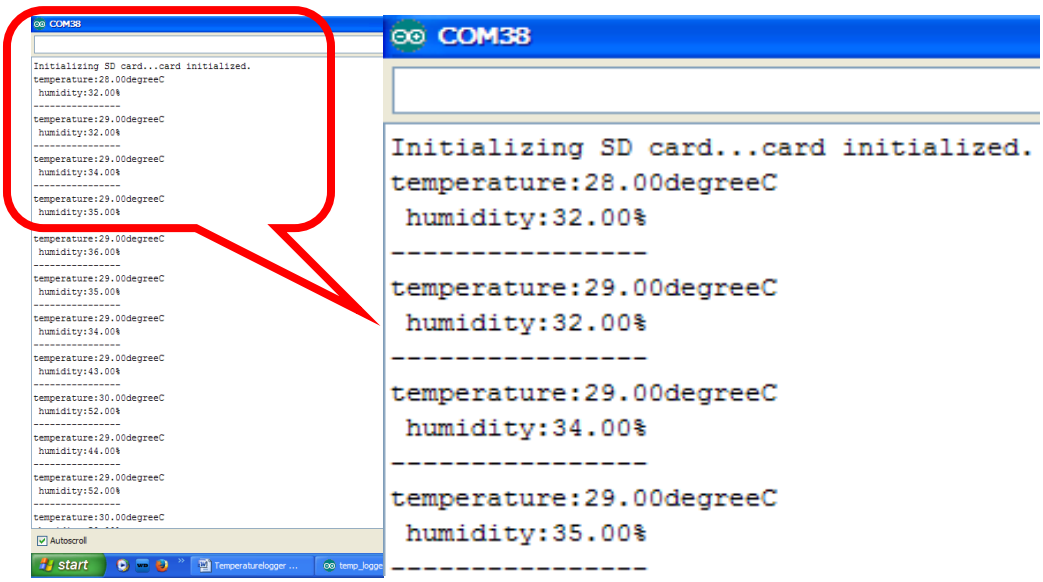

Figure 15 Data display on the serial monitor

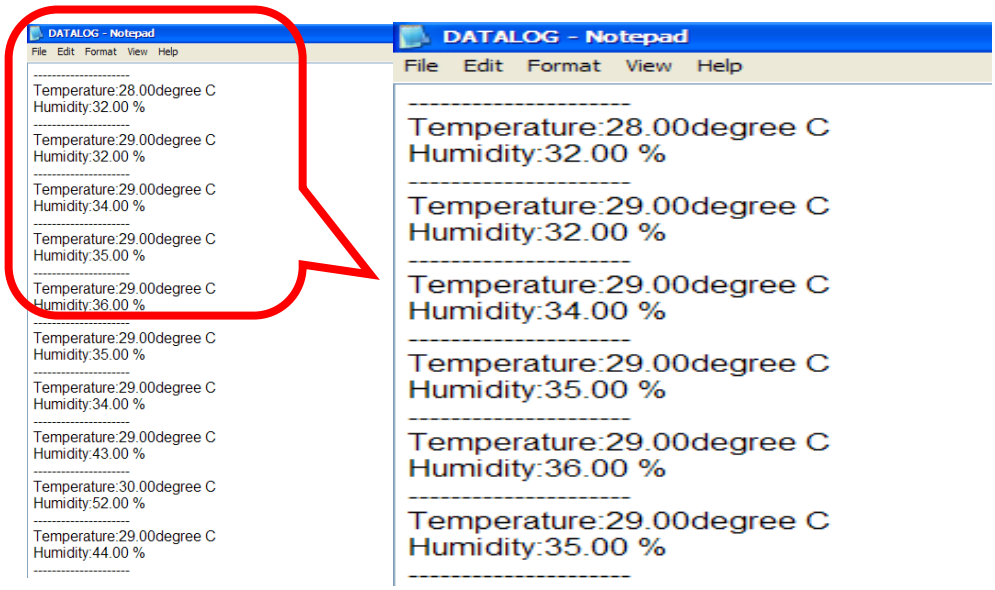

Figure 16 Data stored in SD card logger

Conclusion

In the constructed data logging system with SD memory card, the output of DHT11 sensor were detected and received by the Arduino MCU. The system could measure the environmental temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of ±2°C and ±5%RH. The constructed data logging system avoids the time and cost of sending someone to take temperature and humidity measurements in a remote location, and it enables much higher data density than was possible through manual recording. The control data logging program provided the ability to configure acquisition parameters and format data outputs, which was more accurate because there is no likelihood of human error. And also, it was designed continuously without interruption, even in the worst industrial environments, and the specified accuracy was maintained throughout the period of application.

Acknowledgements

We would to express our deepest gratitude to Dr Lai Lai Wai, Rector of Kyaukse University and Dr Seine Nyoe Nyoe Ko, Pro-Rector of Kyaukse University for their kind permission to carry out this research . Moreover, we would like to give special thanks to Professor Dr Than Than Swe, Head of Department of Physics and Professor Dr Myat Myo Myo Aye, Department of Physics, Kyaukse University for their guidance, discussion and valuable advice to complete this research.

References

Agarwal-Hollands, U., & Andrews, R. (2001). From scroll... To codex... And back again. *Education, Communication & Information*, *1*(1), 59–73.

Alam, K. M. M. (2019). *Development of microcontroller based air quality monitoring system for data centre*.

Ashwin, P. (2019). *Arduino Made Simple*. BPB Publications.

Atanasijevic, P., & Mihailovic, P. (2019). Temperature compensation of NTC thermistors based anemometer. *Sensors and Actuators A: Physical*, *285*, 210–215.

Brooks, D. R. (2016a). Arduino-Based Dataloggers: Hardware and Software. *Institute for Earth Science Research and Education*, *1*(2014), 2015–2016.

Brooks, D. R. (2016b). Arduino-Based Dataloggers: Hardware and Software. *Institute for Earth Science Research and Education*, *1*(2014), 2015–2016.

Dung, N. K. (2017). Design of a wireless air temperature and humidity monitoring system. *Vietnam Journal of Agricultural Sciences*, *15*(3), 306–317.

Evans, B. (2011a). *Beginning arduino programming*. Apress.

Evans, B. (2011b). *Beginning arduino programming*. Apress.

Ibrahim, D. (2010). *SD card projects using the PIC microcontroller*. Newnes.

Ibrahim, D. (2012). *Using LEDs, LCDs and GLCDs in microcontroller projects*. John Wiley & Sons.

Kumar, R., Roopa, A., & Sathiya, D. P. (2015). Arduino ATMEGA-328 microcontroller. *Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng*, *3*(4), 27–29.

Kuria, K. P., Robinson, O. O., & Gabriel, M. M. (2020). *Monitoring temperature and humidity using Arduino Nano and Module-DHT11 sensor with real time DS3231 data logger and LCD display*.

Lu, J., Yu, X., Zheng, M., & Shi, L. (2021). *Optimal Design of SD Card Storage Performance Based on MCU: Hubei Province environmental protection automobile oil pipe intelligent manufacturing enterprise and school joint innovation center*. 797–801.

Manghnani, S., Chittampally, R., & Kasaram, P. (2017a). *Arduino Based Wireless System for Temperature and Humidity Monitoring*. 521–528.

Manghnani, S., Chittampally, R., & Kasaram, P. (2017b). *Arduino Based Wireless System for Temperature and Humidity Monitoring*. 521–528.

Montironi, M. A., Qian, B., & Cheng, H. H. (2017). Development and application of the ChArduino toolkit for teaching how to program Arduino boards through the C/C++ interpreter Ch. *Computer Applications in Engineering Education*, *25*(6), 1053–1065.

Nath, S. S., Navaneethan, S., Sakthekannan, M., Krishnan, M. U., & Yogavignes, B. (2022). *SD Card Interface Using FPGA for Multimedia Applications*. 388–394.

Rohini, S., & Venkatasubramanian, K. (2015). Z-Wave based zoning sensor for smart thermostats. *Indian J. Sci. Technol*, *8*(20), 1–6.

Rusbarsky, K.L., & City, K. (2012). A forensic comparison of NTFS and FAT32 file systems. *Marshall Univ*, *29*.

Satheesh, M., Senthilkumar, B., Veeramanikandasamy, T., & Saravanakumar, O. (2016). Microcontroller and SD card based standalone data logging system using SPI and I2C protocols for industrial application. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *5*(4), 2208–2214.

Sen, S., Sengupta, S., Mukherjee, S., Dey, S., Ghosh, S., Das, R., & Bal, S. (2019). *Encryption in MicroSD card reader or USB data storage devices*. 232–236.