Development of SDI-12 Data Logger with BME680 and BH1750FVI

Chun Wai Leong
Department of Engineering
Swinburne University of Technology
Melbourne, Australia
105239948@student.swin.edu.au

***Abstract*— This project aim is to develop an SDI-12 sensor and data logger to monitor patient's body temperature and the ambient light of the room then save it in a SD Card for further analysis. The data will be converted into graphs and displayed on the Arduino Due LCD Screen. The given SDI-12 sensors are BME680 and BH1750FVI. The group has discussed with a conclusion to utilize BME680 sensor for reading body temperature and BH1750FVI sensor for reading digital light intensity.**

Keywords—SDI-12, Arduino Due, BME680, BH1750FVI, Data Logger, Software

I. Introduction

This report is a brief report about the group project planning and literature review based on Bachelor of Engineering majoring in Software. This project is required for the ENG20009, Engineering Technology Inquiry Project to provide students with the opportunity to learn and design protocol-based sensor and data logger. The requirement of the project is to read data from provided sensors which are BME680 and BH1750FVI and transmit it to computer through the Arduino board using UART to SDI-12 Converter and SDI-12 to USB Converter. Further requirements such as standalone data logger and applying interrupt will be applied to the project too. This report will go through the project planning, task division of the group, literature review of SDI-12 sensor on Software major and end with conclusion.

A. Project Planning

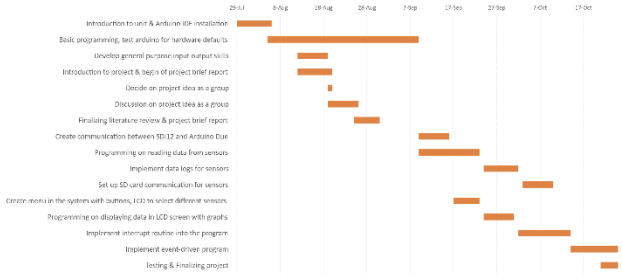


Fig. 1 Project Gantt Chart

B. Task Division

Table I. Task Division

Task	Member
Read data from BME680	Lachlan Mee
Read data from BH1750FVI	Benedictus Hamdoko
Convert / Save data from BME680	Chun Wai Leong
Convert / Save data from BH1750FVI	Benjamin Ciccocioppo
LCD Screen and Read data from sensors	Chao Yang Lau

The tasks are divided equally into 5 members as shown in Table I., the tasks in general are handling read data from sensor, save and convert data to graphs and display graphs data to the Arduino Due LCD Screen. Tasks are broken down into smaller pieces and arranged in Fig. 1, project members are expected to follow the Gantt Chart strictly to ensure the project can be completed on time, which is the week of 17 October 2024. Lachlan and Benedictus are in charge of collecting the data read from the sensors. Benjamin and I will be in charge of saving and converting the data from sensors in text files to graphs. Lastly, Chao Yang will be in charge of displaying data graphs to the Arduino Due LCD Screen.

C. Save and Convert Data from Sensor

To save data into a text file, SD library and RTC library are needed. First, setting up the SD Card is needed, the correct pin must be initialized according to the Arduino board used. SD library consists of many functions to interact with text files including open file, write line, close file and so on [1]. The data collected can be organized properly into a single line before writing it into the text file for better efficiency. Besides that, RTC library is imported to know the exact time when the data is collected [2]. Therefore, timestamping is done by adding the data collected time with the help of RTC. After the data is organized, it can finally be written into the exited text file. Formatted data can simplify the step of extracting data from text file and display it in the LCD Screen. Adafruit_GFX library is used to handle the graphic functions of the Arduino board [3]. Simple graph that made up of lines can be done in the Arduino Due LCD Screen, by only extracting the numerical value from the text file.

II. Application Of The SDI-12 Sensor Based On Software Major

SDI-12 (Serial/Digital Interface at 1200 baud) sensors are widely used in various types of environment monitoring systems to collect data with the assist of SDI-12 protocol. SDI-12 protocol has the capability to communicate in long distance with a low power consumption. Therefore, it has become a standard for remote sensing application [4]. Software plays an important role in developing an application that uses SDI-12 protocol to communicate with others device such as PC. The software implementation of SDI-12 protocol including:

A. Firmware Development

SDI-12 sensors are often paired with microcontrollers such as Arduino. Therefore, SDI-12 protocol is essential in the firmware of microcontroller for the device to communicate with SDI-12 sensors. For Arduino, there is a library named "SDI-12" developed by Kevin M. Smith and maintained by Sara Damiano which offers pre-built functions that simplify the communication between the

devices and SDI-12 sensors [5]. This allows developers to focus on high-level programming.

B. Data Acquisition

SDI-12 protocol is a command-response model which means it will only return data from specific sensor when the user sends a specific command [6]. The command has to be pre-setup to the SDI-12 sensor, so it understands the command sent and retrieves the desired data from the correct sensor. Currently, SDI-12 has a unique set of commands, but user can also implement additional command to the SDI-12. Software also helps in handling unexpected errors that occur to the sensor such as returning unprocessed data. This can be prevented by adding data validation to the software so it will parse the unprocessed data to the desired type of data and return it to user. Different errors require different error handling. Therefore, it is essential to implement all the possible error handling to the software.

C. Data Logging

After collecting the data, the data can be stored locally within the built-in memory or a memory card. This ensures the data stored is available even if it is powered off. Data logging is performed in a way called timestamping which every piece of data is recorded with the exact time when it is collected. It is extremely useful with timestamping which allows users to monitor the changes of the sensors from time to time. However, using the built-in memory to store data might lead to the crash of the embedded software when it exceeds the amount of memory available. Therefore, memory management is important and should be well programmed to the software.

D. Data Analysis

Other than collecting data periodically, software is also capable of real-time data visualization with the help of external tools such as Python with "Matplotlib" library [7]. LabView is also a helpful software to visualize the data collected in graphs [8]. By visualizing the data collected, users can analyze the data easily and help in decision making.

E. Cloud Integration

For remote monitoring to work on SDI-12 sensors, cloud integration is compulsory. The real-time data visualization mentioned above is achieved by transmitting data through the Internet to the cloud server. For example, Arduino cloud allows industry to synchronize data by using the cloud backend service [9]. It also allows industry to control and monitor the Arduino board that is connected to the SDI-12 sensors with graphical tools like mobile app. Application Programming Interface (API) is developed and used for system to share the data among different type of systems.

III. Conclusion

Application of SDI-12 sensors from the software aspect is essential and critical for performing all the tasks required for the industry. Regardless of the hardware that implements the SDI-12 sensor, embedded software-based systems are compulsory to make sure all the sensors are collecting accurate data and display to the users for future decision-making processes.

References

[1] Arduino, "SD," Arduino Reference, n.d. [Online]. Available: https://www.arduino.cc/reference/en/libraries/sd/

[2] David Meaney, "What Is A Real Time Clock (RTC)?," n.d. [Online]. Available: https://ecsxtal.com/what-is-a-real-time-clock-rtc

[3] Arduino, "Adafruit GFX Library," Arduino Reference, n.d. [Online]. Available: https://www.arduino.cc/reference/en/libraries/adafruit-gfx-library/

[4] NEXSENS, "SDI-12 Protocol," Overview, n.d. [Online]. Available: https://www.nexsens.com/knowledge-base/software/ichart/sdi-12-protocol.htm

[5] Arduino, "SDI-12," Arduino Reference, n.d. [Online]. Available: https://www.arduino.cc/reference/en/libraries/sdi-12/

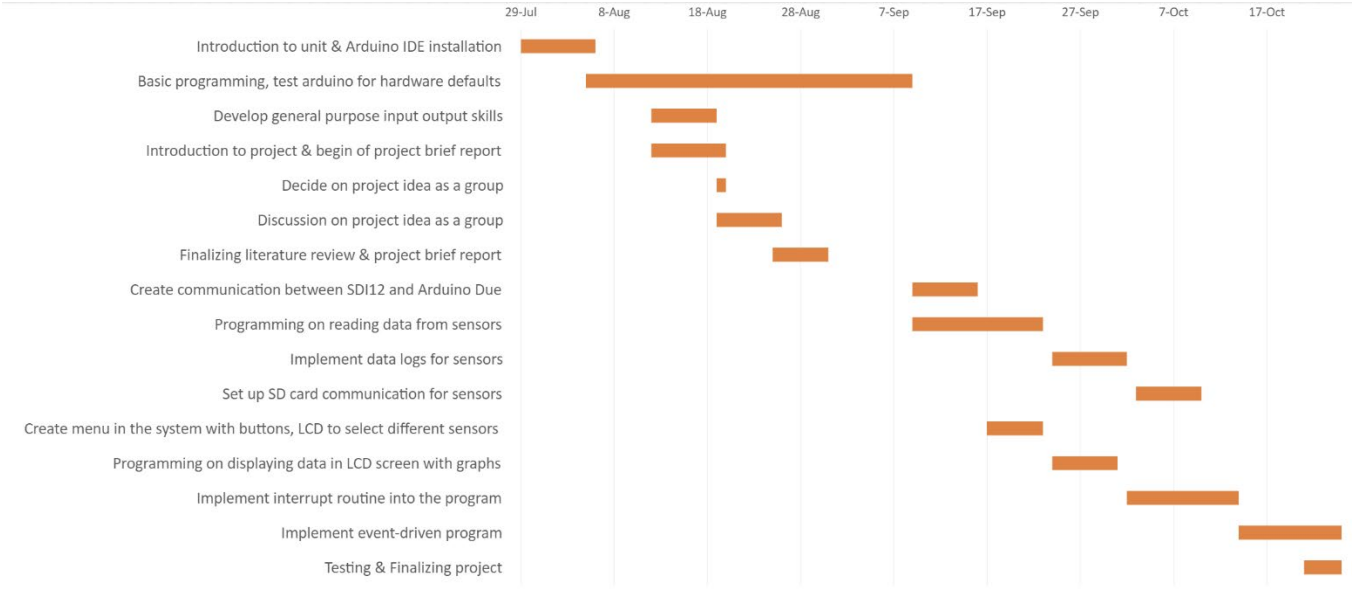
[6] SDI-12 Support Group, "SDI-12," n.d. [Online]. Available: https://www.sdi-12.org/current_specification/SDI-12%20Specification%201.4%20February%2020%202023.pdf

[7] Paul McCarter, "Arduino With Python," n.d. [Online]. Available: https://toptechboy.com/python-with-arduino-lesson-2-installing-the-software/#google_vignette

[8] Joe Peck, "Getting Started with Arduino and LabView Community Edition," Explore an Example, n.d. [Online]. Available: https://learn.ni.com/learn/article/getting-started-with-arduino-and-labview-community-edition#TARGET_4

[9] Karl Söderby, "Getting Started with Arduino Cloud," n.d. [Online]. Available: https://docs.arduino.cc/arduino-cloud/guides/overview/

APPENDICES


29-Jul
8-Aug
18-Aug
28-Aug
7-Sep
17-Sep
27-Sep
7-Oct
17-Oct
Introduction to unit & Arduino IDE installation
Basic programming, test arduino for hardware defaults
Develop general purpose input output skills
Introduction to project & begin of project brief report
Decide on project idea as a group
Discussion on project idea as a group
Finalizing literature review & project brief report
Create communication between SDI12 and Arduino Due
Programming on reading data from sensors
Implement data logs for sensors
Set up SD card communication for sensors
Create menu in the system with buttons, LCD to select different sensors
Programming on displaying data in LCD screen with graphs
Implement interrupt routine into the program
Implement event-driven program
Testing & Finalizing project