디지털산업정보학회 논문지 제18권 제4호-2022년 12월
http://dx.doi.org/10.17662/ksdim.2022.18.4.047

MEMS 임베디드 시스템 설계[*]

홍 선 학[**]

MEMS Embedded System Design

Hong, Seon Hack

〈Abstract〉

In this paper, MEMS embedded system design implemented the sensor events via analyzing the characteristics that dynamically happened to an abnormal status in power IoT environments in order to guarantee a maintainable operation. We used three kinds of tools in this paper, at first Bluetooth Low Energy (BLE) technology which is a suitable protocol that provides a low data rate, low power consumption, and low-cost sensor applications. Secondly LSM6DSOX, a system-in-module containing a 3-axis digital accelerometer and gyroscope with low-power features for optimal motion. Thirdly BM1422AGMV Digital Magnetometer IC, a 3-axis magnetic sensor with an I2C interface and a magnetic measurable range of ±1200uT, which incorporates magneto-impedance elements to detect the magnetic field when the current flowed in the power devices. The proposed MEMS system was developed based on an nRF5340 System on Chip (SoC), previously compared to the standalone embedded system without bluetooth technology via mobile App. And also, MEMS embedded system with BLE 5.0 technology broadcasted the MEMS system status to Android mobile server. The experiment results enhanced the performance of MEMS system design by combination of sensors, BLE technology and mobile application.

Key Words : nRF5340, MEMS System, Bluetooth Low Energy, LSM6DSOX, BM1422AGMV

Ⅰ. Introduction

In this paper, we experimented with accurate detection technology by sensing the vibration of the fusible element using an LSM6DSOX accelerometer. The sound of the explosive noise would be detected by an MP34DT05 for MEMS audio sensor omnidirectional digital microphone built with a capacitive sensing element. BM1422AGMV magnetometer sensed the magnetic density of IoT devices.

The previous study about MEMS embedded system was largely based on the 8-bit or 16-bit

* 본 연구는 2022년 서일대학교 교내학술연구비 지원으로 수행되었음.

** 서일대학교 컴퓨터전자공학과 교수(교신저자)

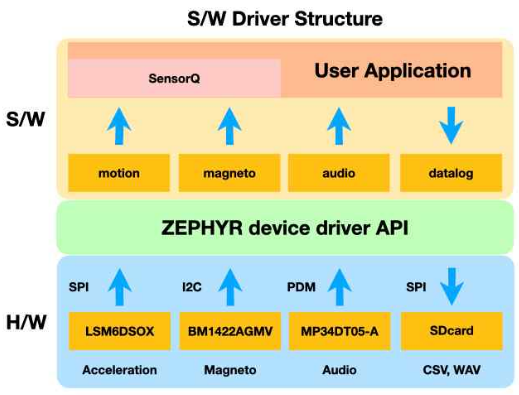
micro controller such as AVR or PIC which had deficiency of Bluetooth technology. In this paper, main micro controller for MEMS was an ARM® Cortex®-M33 core with a floating-point unit (FPU). The ARM CMSIS (Cortex Microcontroller Software Interface Standard) hardware abstraction layer of the ARM Cortex processor series was also available for the M33 CPU. In this paper, we designed the system in module(SIM) via PCB artwork and compared to the characteristics of sensors via accuracy, processing time, firmware and RAM usage. And therefore We could verify the performance of MEMS embedded system[1-5].

II. Software Design

The MEMS system had five component peripheral devices: LSM6DSOX, MP34DT05-A, BM1422AGMV, SD card, and BLE. These components are interacted with real time operating system, RTOS Zephyr device driver API.

2.1 MEMS System Design

In some cases, if no events were picked up by the sensors, the accelerometer, magnetometer, and microphone data would not be stored on the SD card. Whenever an event occurred, data between before and after the event would be collected and stored on the SD card. We designed the special queue structure to store the data at the moment events occurred. The queue saved the upcoming data from the motion driver, so we used the push function to collect the motion data. After the events were completed, the pop function working for the motion data taken out from the queue. The software driver structure, as shown in <Figure 1> showed the relationship between hardware and software, which consists of motion, audio, magneto, and datalog function.



<Figure 1> MEMS Driver Structure

SensorQ, which was a special queue included in the user application temporarily stored the data of the motion and magneto driver. However, the audio and datalog drivers connected directly to the user application. When the events were completed, the SD card stored the data from the queue and audio buffer. The data stored in CSV and WAV file formats, which could be analyzed on any personal computer. By storing data in these formats, it is easy to access and process the data in order to detect any events.

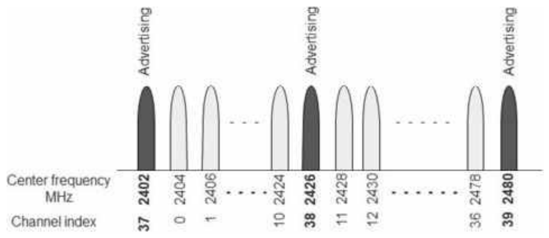
The I2C communication protocol could reach speeds of up to 400 kHz, while the SPI protocol could reach 10 MHz. Therefore, we used an SPI at 4 MHz frequency. The LSM6DSOX driver was the API that defined interactions between the SPI driver and the user Application. This driver wass used for initializing the registers of LSM6DSOX, namely the accelerometer and gyroscope. Magneto was the API that defined interactions between the BM1422AGMV driver and the user application. It functioned as an initializer and data handler that communicates with the BM1422AGMV driver. The BM1422AGMV had an output data rate of 100 SPS, so it could catch the 60 Hz AC sine wave. The BM1422AGMV driver is the API that defined the interactions between the I2C driver and the user application. BM1422AGMV driver is used for setting the registers of BM1422AGMV. It worked as an initializer and audio data processer, which is managed for the PDM driver. The audio sampling frequency was 16 kHz, which gave us an accurate signal of events. The MP34DT05-A was an ultra-compact, low-power and omnidirectional digital MEMS microphone implemented with a capacitive sensing element and an IC interface[6-10].

2.2 Zephyr System Design

We used Zephyr device tree API to easily apply the hardware device to the MEMS Embedded application. The zephyr was a scalable real-time operating system(RTOS) supporting multiple hardware architectures such as ARM, MIPS etc. The zephyr was based on a small-footprint kernel design for optimizing the resource-constrained system. Zephyr included a kernel and all component. There were two characteristics in Zephyr. One of which was the optimized device driver model that provided a consistent model for initializing all the drivers and allowed the reuse of drivers. The other feature was device tree support for describing hardware components[11].

2.3 Bluetooth Characteristics

In this paper, we used nRF BLE technology which transferring the diagnosis data to a server unit. There are several layers of the BLE architecture, with the three main blocks being the applications, the host, and the controller. The application layer referred to the implementation on top of the Generic Access Profile and Generic Attribute Profile and use-case dependent — it shows how the user's application handles the data which received from and sent to other devices. The physical layer (PHY) referred to the radio hardware that used for communication and modulating/demodulating the data. The BLE operated on the ISM band (2.4 GHz spectrum), segmented into 40 RF channels and each separated by 2 MHz (center-to-center), as shown in <Figure 2>



<Figure 2> Frequency Spectrum of RF Channels

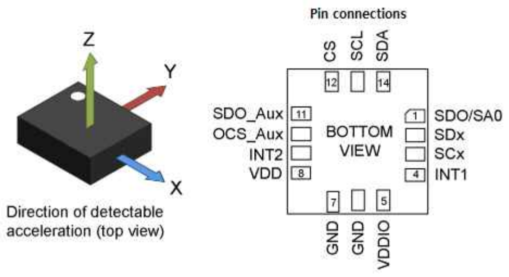
Three of these channels were primary advertising channels of BLE. The remaining 37 channels were used for extra functionality of BLE communication which were secondary advertisements and data transfer during a connection. Advertising should begin with advertisement packets going on the three primary advertising channels. This allowed the devices to scan for advertisers, to find and read their advertisement data. The scanner could initiate a connection when the advertiser was allowed. Scan requests and responses allowed the advertiser to send extra advertisement data to device which were set up to receive these data[12-16].

Ⅲ. Embedded System Design

The MEMS embedded system had five modules: an MCU, an accelerometer, a magnetometer, and a microphone, along with an SD card as a system-in-module.

3.1 MEMS Acceleration Sensor

In this paper, we used LSM6DSOX, which is a system-in-package featuring a high-performance 3-axis digital accelerometer and a 3-axis digital gyroscope that delivers the motion sensing that can detect orientation and gestures. This could make application features more sophisticated than simply



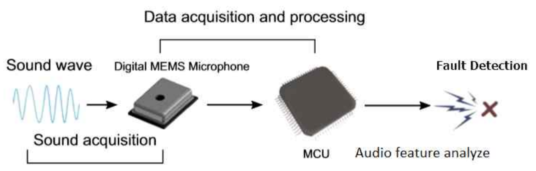
<Figure 3> LSM6DSOX Layout

orienting embedded devices to either portrait or landscape mode as shown in <Figure 3>.

The event-detection interrupts could be able to more efficient and reliable motion tracking with hardware recognition of free-fall events, and dynamic falling detection of MEMS embedded system.

3.2 MEMS Microphone

The MEMS microphone is a digital sensor that converts acoustic pressure waves into a digital signal. And then the converted digital data was delivered to the MPU via specific peripherals, which is processed and transformed into standard audio data. The acquired audio data was handled by the micro controller as shown in <Figure 4>.



<Figure 4> Sound acquisition system

The digital microphone consists of a MEMS transducer, an amplifier, and a pulse data modulator (PDM). The PDM protocol sends a serial pulse density-modulated signal which converts the buffered analog signal. The PDM control method is used to generate clock input (CLK). The frequency range of clock for digital microphone is between 1 MHz and 3.25 MHz. The sampling rate of the amplifier's analog output signal sampled to produce a discrete-time representation is defined by the clock frequency. The relative density of the pulses is corresponding to the analog signal's amplitude in a PDM signal.

3.3 MEMS Magnetometer

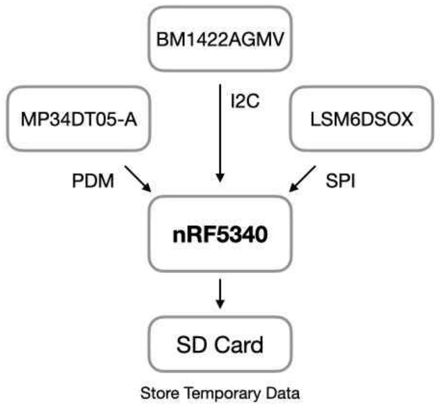
In this paper, we used a 3-axis magnetometer which produced three axis values representing the amount of magnetic field on the device's x-, y-, and z-axes. The magnetometer on MEMS system could sample 25 times per second (a rate of 25 Hz). We took these values directly as its input for the system, however there is no need for any further preprocessing. After the data had been captured, our application would determine whether a valid event was detected and would send signal over the BLE 5.0.

3.4 ARM Coretex-M33 Features

The main part of MEMS was an ARM® Cortex®-M33 core with a floating-point unit (FPU). The ARM® Cortex®-M33 core had a 32-bit instruction set which implements a superset of 16- and 32-bit instructions so it could maximize code density and performance. The ARM CMSIS (Cortex Microcontroller Software Interface Standard) hardware abstraction layer of the ARM Cortex processor series was also available for the M33 CPU. We experimented with the MEMS system using a MEMS accelerometer, magnetometer and microphone for detecting the status of the system. The nRF5340 was the world's first wireless SoC with two Arm® Cortex®-M33 processors. The combination of two flexible processors, the advanced feature set, and an operating temperature up to 105 °C, makes it the ideal choice for Audio, professional lighting, advanced wearables, and other complex IoT applications.

The application processor was optimized for performance and can be clocked at either 128 or 64 MHz, using voltage-frequency scaling. It had 1 MB Flash, 512 KB RAM, a floating-point unit (FPU), an 8 KB 2-way associative cache and DSP instruction capabilities. The network processor was clocked at 64 MHz and was optimized for low power and efficiency (101 CoreMark/mA). It had 256 KB Flash and 64 KB RAM.

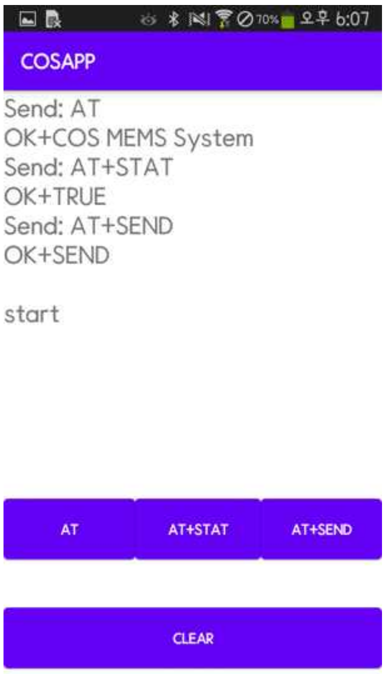
<Figure 5> showed our design of the hardware structure embedded platform for implementing the MEMS embedded system.



<Figure 5> MEMS system hardware

3.3 Bluetooth Mobile Application

We designed an Android application to display status of communication information from MEMS embedded system. The Android mobile application had several interfaces for communication with MEME system. We arranged a text view widget and buttons as shown in <Figure 6>. The text view widget showed the state of the MEMS system, which broadcasted over the BLE while the tree buttons placed below the text view widget that could request data from the MEMS system. The button AT asked the MEMS system to respond to the connection status, like OK + MEMS system. Another button, AT + STAT, requested the MEMS system's data-prepared state. If data were prepared, it responded with OK + TRUE.


Send: AT
OK+COS MEMS System
Send: AT+STAT
OK+TRUE
Send: AT+SEND
OK+SEND

start

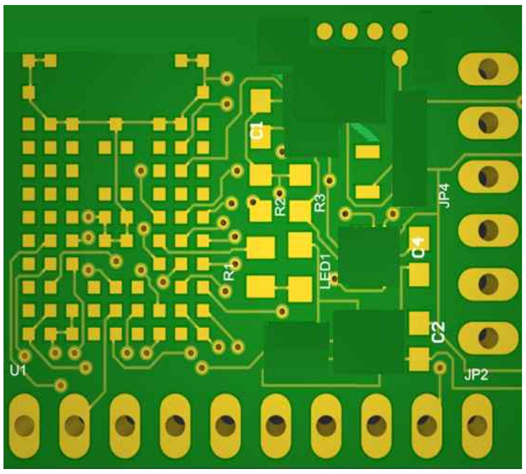
<Figure 6> BLE Mobile Application

The other button, AT + SEND, requested the MEMS system to send the prepared data and responds with OK + SEND. After responding, the system sent a series of raw data. The button CLEAN cleared the text view widget which showed the communication log.

Ⅳ. Experiment Result

4.1 Hardware Layout

The MEMS embedded system had five modules: an MCU, an accelerometer, a magnetometer and a microphone along with as a system-in-module as shown in <Figure 7>.

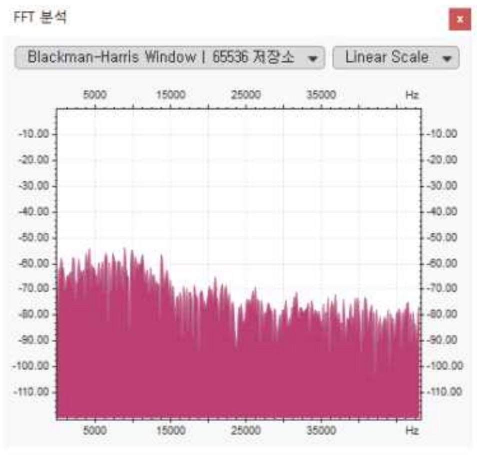


<Figure 7> PCB Layout

We used the MCU as the main unit to control the three sensors such as accelerometer with SPI protocol. the magnetometer with I2C protocol, and the sound sensor with PDM module and transmit sensor data to the SD card. The PDM module enabled the input of PDM signals from external audio front ends such as digital microphone. The PDM module generated the clock and supported both signal-channel and dual-channel(left and right) data input.

4.2 Audio Sensor Analysis

We recorded the audio waveform and 3-axis accelerometer sensor data. <Figure 8> showed the audio waveform frequency ranges form 0 to 47799Hz and the FFT analysis was based on the Blackman-Harris Window. The 3-axis accelerometer sensor data had x-axis, y-axis and z-axis features respectively.



<Figure 8> FFT analysis of audio data

4.3 MEMS Sensor Characteristics

We used three class of sensors which leveraged the performance of the MEMS embedded system in combination of Zephyr real time tools in this paper. Table 1 showed the characteristics of sensors including the accuracy, processing time, firmware size and RAM usage of the designed MEMS embedded system.

In this paper, we could find out that the magnetometer sensor was suitable for MEMS embedded system from the experimental result of <Table 1> compared to the indices such as accuracy, processing time, firmware size and RAM usage under the current flow of the IoT environment. As expected, RAM usage and processing time were similar value between accelerometer and magnetometer, but accuracy of magnetometer had the excellent feature.

<Table 1> MEMS Sensor Characteristics

	Accelerometer	Microphone	Magneto meter
Accuracy	70%	90%	100%
Processing time	1 ms	45 ms	1 ms
Firmware Size	19.2 KB	35.5KB	18.4KB
RAM Usage	1.7KB	11.8KB	1.7KB

Ⅴ. Conclusion

In this paper, we designed the MEMS embedded system for detecting events using wireless communication with transferring the sensor information vis Android mobile application. We implemented MEMS embedded system under the three sensor conditions combined with bluetooth communication and Android system. In this paper, we experimented 32 bit ARM Coretex M33 CPU with BLE 5.0 and verified that the magnetometer sensor was suitable to detect power current in the Iot environment of MEME embedded system. Afterwards we could enhance the performance of the MEMS system by making MEME embedded system predictive with machine learning.

References

[1] Seon Hack Hong, "Embedded system design with COS LoRa technology," Korea Society of Digital Industry and Information Management. 3rd14, 2018, pp.29-38.

[2] Seon Hack Hong, "Edge Impulse 기계학습 기반의 임베디드 시스템 설계," Korea Society of Digital Industry and Information Management. 3rd 17, 2021, pp.9-15.

[3] Li Na Lee, Ga Ram Lee & Ho Won Kim, "Wireless technology analysis," Korea's Information and Communications Society Summer Conference, Univ. of Busan, Korea, 2017.

[4] 이현승 · 김기덕 · 이영주 · 황동열 · 신범수, "자율 주행 조향 시스템을 위한 임베디드 제어기개발," 한국 농업기계학회, 12. 2017.

[5] Seon Hack Hong, "Real-Time Linux System Design," Korea Society of Digital Industry and Information Management, 10th, 06. 2014, pp. 13-20.

[6] Seon Hack Hong, "Mobile Arduino Embedded Platform Design," Korea Society of Digital Industry and Information Management. 4th, 12. 2013, pp. 33-41.

[7] STMicroelectronics, LSM6DSOX Datasheet, Geneva Switzerland, 01.2019.

[8] STMicroelectronics, MP34DT05-A Datasheet, Geneva Switzerland, 01.2019.

[9] Nordic semiconductor, nRF5340 Product Specification, v1, Trondheim Norway, 02. 2021.

[10] Mohammad Afaneh, Intro to Bluetooth Low Energy, the fastest way to learn BLE, NovelBits, 02. 2018.

[11] Zephyr Project, https://www.zephyrproject.org/. 11.2019

[12] Mohammad Afaneh, Bluetooth5 & Bluetooth Low Energy Developer's Guide, NovelBits, 02.2019.

[13] Nordic semiconductor, S140 SoftDevice Specification, v2.1, Trondheim Norway, 09.2019.

[14] Arm Limited, Arm Compiler User Guide, Version 6.14, Cambridge UK, 02. 2020.

[15] Arm Limited, Arm Compiler Reference Guide, Version 6.14, Cambridge UK, 02. 2020.

[16] Ata Elahi, Trevor Arjeski, ARM Assembly Language with Hardware Experiments, Berlin Germany, 2015.

■ 저자소개 ■

홍 선 학
(Hong, Seon Hack)

1992년~현재
서일대학교 컴퓨터전자공학과 교수

관심분야 : 제어응용/임베디드/객체설계
E-Mail : hongsh@seoil.ac.kr

논문접수일 : 2022년 10월 4일
수 정 일 1차 : 2022년 10월 24일
수 정 일 2차 : 2022년 11월 11일
수 정 일 3차 : 2022년 11월 24일
게재확정일 : 2022년 12월 5일