COMPUTER DESIGN SYSTEMS. THEORY AND PRACTICE

Vol. 6, No. 3, 2024

Andriy Holovatyy[1], Andrzej Lukaszewicz[2], Sofia Holovata[3], Nazariy Klym[4], Kostyantyn Kolesnyk[5]

[1] Computer Design Systems Department, Lviv Polytechnic National University, 12, S. Bandery str., Lviv, Ukraine, E-mail: andrii.i.holovatyi@lpnu.ua, ORCID 0000-0001-6143-648X,
[2] Department of Machinery Design and Operation, Faculty of Mechanical Engineering, Bialystok University of Technology, 45a, Wiejska str., Bialystok, Poland, E-mail: a.lukaszewicz@pb.edu.pl, ORCID 0000-0003-0373-4803
[3] Department of Software Engineering, The National Forestry and Wood-Technology University of Ukraine, 103, Chuprynky str., Lviv, Ukraine,E-mail: sofi.pobereyko@nltu.edu.ua, ORCID 0000-0001-5904-9683
[4] Computer Design Systems Department, Lviv Polytechnic National University, 12, S. Bandery str., Lviv, Ukraine, E-mail: nazarii.klym.mknsp.2023@lpnu.ua,
[5] Computer Design Systems Department, Lviv Polytechnic National University, 12, S. Bandery str., Lviv, Ukraine, E-mail: kostyantyn.k.kolesnyk@lpnu.ua, ORCID 0000-0001-9396-595X

DEVELOPMENT OF EMBEDDED SYSTEM FOR REAL-TIME AUDIO ACQUISITION AND PROCESSING BASED ON STM32 MICROCONTROLLER

Received: November 03, 2024/ Revised: November 20, 2024 / Accepted: November 25, 2024

https://doi.org/10.23939/cds2024.03.176

Abstract. In the paper, the embedded system for real-time acquisition and processing audio data has been developed using the STM32F407G-DISC1 Discovery kit with the STM32F407VGT6 ARM Cortex-M4 32-bit MCU. The basic modules of the STM32F4DISCOVERY kit are used for its operation. The STM32F407VGT6 MCU I2S2 peripheral module has been configured in half-duplex master mode to acquire PDM data from the MP45DT02 microphone. The STM32 USB peripheral module is configured in host mode and the MSC protocol is implemented for transmitting and receiving audio data to/from USB flash drive. The I2S3 peripheral module of the STM32F407VGT6 MCU is configured in master transmitter mode for transmitting audio data to the CS43L22 DAC. The I2S2 DMA of the STM32F407VGT6 MCU is used to transfer data from the microphone to the RAM buffer, which significantly relieves the CPU. The user buttons on the STM32F407G-DISC1 board are used for application control (playback or recording). The real-time audio acquisition and processing software for STM32 MCU has been developed in C using the BSP audio driver and PDM2PCM library.

Keywords: real-time embedded system, audio acquisition and processing, STM32F407G-DISC1 Discovery kit, STM32F407VGT6, MP45DT02 ST-MEMS microphone, CS43L22 audio DAC, LCD WH1602B-NYG-CT, I2S, SPI, PDM, PCM, WAV audio file format.

Introduction

The modern market of electronic components offers powerful 32-bit MCUs with ARM Cortex-M4 core and digital MEMS microphones, which allow us to create various embedded audio systems for industrial and household applications for high-quality sound recording and playback, speech recognition, human-machine voice interaction in real time.

Microphones have been around for decades in embedded systems, and the advent of MEMS microphones has rapidly expanded their use. This allows us to add voice functions to various projects – smart home, cars, mobile terminals, laptops, portable media players, VoIP, speech recognition systems, A/V e-learning devices, gaming devices and VR devices, digital photo and video cameras, wearable gadgets. In addition to significantly smaller dimensions, lower power requirements, and greater suppression of electrical noise, one of the main advantages of MEMS microphones is the increase in output

options, which provides more flexibility for engineers. Although, there are still analog MEMS microphones, the most popular one is a digital version with PDM audio output.

Each of these options has its own distinctive characteristics, advantages and disadvantages, which must be taken into account when implementing the project.

Digital MEMS microphones can be used in audio projects where small size, high sound quality, reliability and affordability are key requirements. A digital MEMS microphone converts sound waves into a PDM signal, which is a stream of high-frequency 1-bit digital samples. This output signal is received in blocks using the synchronous serial interface (SPI/I2S) for processing and conversion to standard audio data in PCM format.

The purpose of the work is to develop the embedded system that acquires PDM data from a digital MEMS microphone in real time, filters and processes them, stores the processed audio data in PCM format on a USB flash drive or SD memory card for the further targeted application. For the development of the embedded system for real-time audio acquisition and processing, it is proposed to use the STM32F4-DISCOVERY board with the STM32F407VGT6 MCU, MP45DT02 ST-MEMS microphone, CS43L22 audio DAC, USB flash drive/SD memory card [1].

Study object – process of real-time audio data acquisition and processing.

Scope of research – methods and tools for real-time audio data acquisition and processing.

Goal of research – development of the STM32-based embedded system that in real time acquires audio data in PDM format from the MEMS microphone, processes and converts them into PCM sound format using the PDM2PCM library.

Main Material Presentation

Overview of existing technologies for audio acquisition. There are two different technologies for audio acqusition: traditional ECM (Electret Condenser) and MEMS. The ECM microphones are composed of dicrete components, and usually include variable capacitance based on permanent electret component, a transistor is used to amplify and buffer variations in capacitance voltage. The ECM microphones output analog signal which must be further processed and filtered.

The MEMS microhone is a dual-die device consisting of membrane based on silicon processes, embedded ASIC, analog or digital output [2].

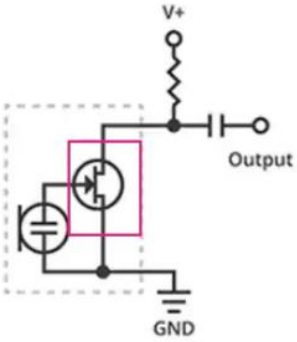

Fig. 1. ECM microphone circuit

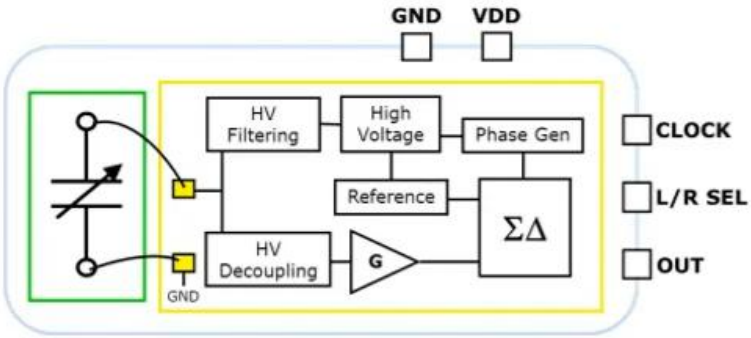

Fig. 2. Components of digital MEMS microphone

The MCU-based embedded system receives raw data from the microphone in PDM format, processes and converts them into audio data in standard PCM format.

PDM is a form of modulation used to represent an analog signal in the digital domain. It is a high-frequency stream of 1-bit digital samples. In a PDM signal, the relative pulse density corresponds to the amplitude of the analog signal. A large cluster of ones corresponds to a high (positive) amplitude value, while a large cluster of zeros will correspond to a low (negative) amplitude value, and alternating ones and zeros will correspond to a zero amplitude value.

In PCM signal, specific amplitude values are coded into pulses. The PCM stream has two main properties that determine the accuracy of the output analog signal stream: sampling frequency and bit rate.

The sampling rate is the number of samples of a signal taken per second to represent it digitally. The bit depth determines the number of bits of information in each sample.

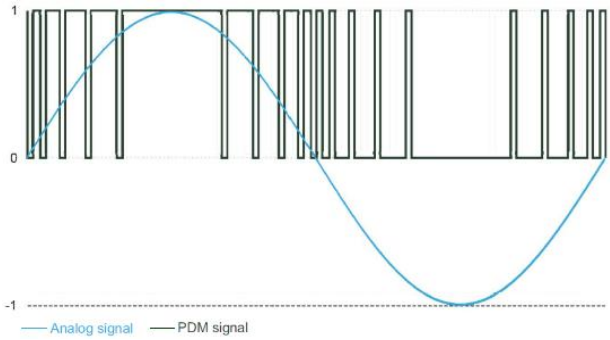

Fig. 3. PDM signal

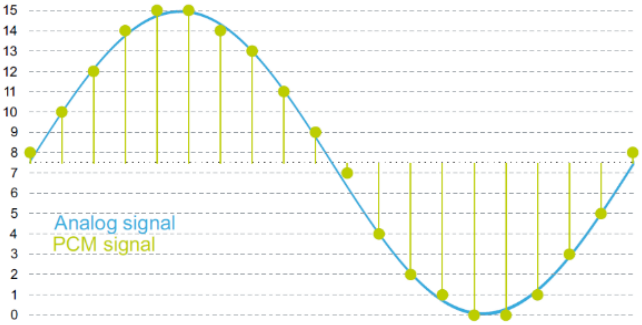

Fig. 4. PCM signal

For conversion of PDM stream to PCM samples, the PDM stream must be filtered and decimated. In the decimation step, the sample rate of the PDM signal is reduced to the target audio sample rate (for example, 16 kHz). By selecting 1 out of every M samples, the sample rate is reduced by a factor of M. Thus, the PDM data rate (microphone clock rate) is M times larger than the target audio sample rate required for the application, where M is the decimation factor.

PDM frequency = audio sampling frequency × decimation factor.

For example, PDM frequency = 16 kHz × 64 = 1024 kHz. The decimation factor is usually in the range from 48 to 128. The decimation step is preceded by a low-pass filter to avoid distortions due to overlapping spectra.

Hardware interface for audio acquisition from MEMS microphones

In Fig. 5 the block diagram of digital PDM microphone is shown. The main parts of the digital microphone are a MEMS transducer, an amplifier and a PDM modulator [2].

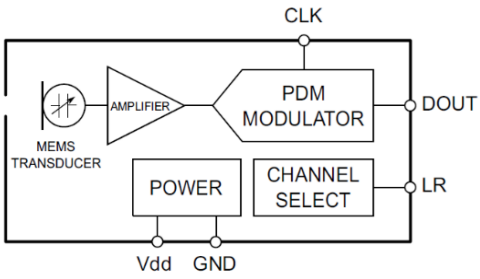

Fig. 5. Block diagram of digital microphone with PDM output

MEMS transducer is a variable capacitance that converts the change in air pressure, caused by sound waves, into a voltage.

The amplifier buffers the voltage supplied by the MEMS converter and provides a sufficiently powerful signal for the PDM modulator.

PDM modulator converts a buffered analog signal into a serial PDM signal. The clock input (CLK) is used to control the PDM modulator.

The clock frequency range of ST digital microphones is from 1 MHz to 3.25 MHz. This frequency defines the sampling rate at which the amplifier analog output signal is sampled to create a discrete time representation (PDM bitstream).

The microphone output goes to the appropriate level on the selected timing (clock) edge and then goes to a high impedance state for the other half of the clock cycle. Channel selection determines the timing edge at which the digital microphone outputs valid data. The LR pin must be connected to Vdd or GND.

Table 1

Channel selection for outputting a PDM signal at the DOUT output of the microphone

LR	DOUT	
	CLK low	CLK high
GND	Valid data	High impedance
Vdd	High impedance	Valid data

The Vdd and GND pins provide power to the various components of the digital microphone. The power supply must be correctly connected to the microphone, as any ripples can generate noise at the output.

Table 2

Pin description of MP45DT02 MEMS microphone

Pin name	Function	Description
Vdd	3.3 V power supply	Input
GND	0 V	Input
LR	Left/right (L/R) selection	Input
CLK	Synchronization clock	Input
DOUT	L/R PDM data output	Output

In the mono mode, the LR pin can be connected either to Vdd or to GND. In the Fig. 6, the LR pin is connected to Vdd – data generation on right channel.

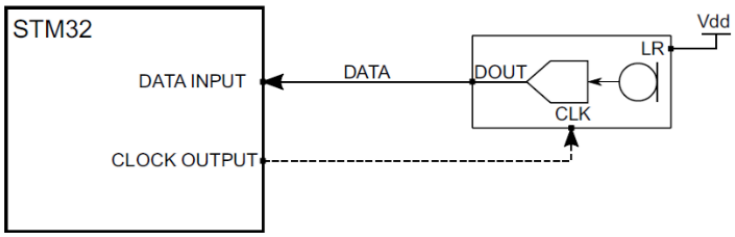

Fig. 6. Mono mode configuration – data generation on right channel

On the rising clock edge, the microphone generates valid data for the half of the clock period, and then goes to a high impedance state for the other half.

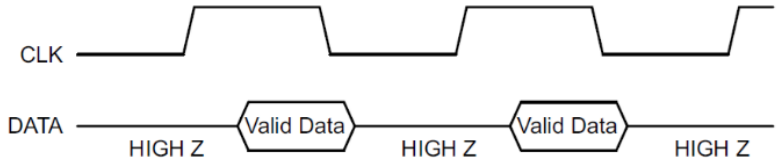

Fig. 7. Right channel data pattern

In the Fig. 8, the LR pin is connected to GND – data generation on left channel.

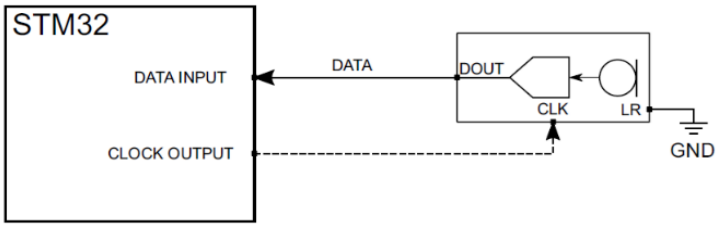

Fig. 8. Mono mode configuration – data generation on left channel

On the falling edge of the clock, the microphone generates valid data for the half of the clock period and then goes into a high impedance state for the other half.

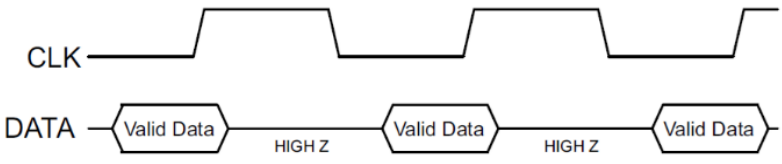

Fig. 9. Left channel data pattern

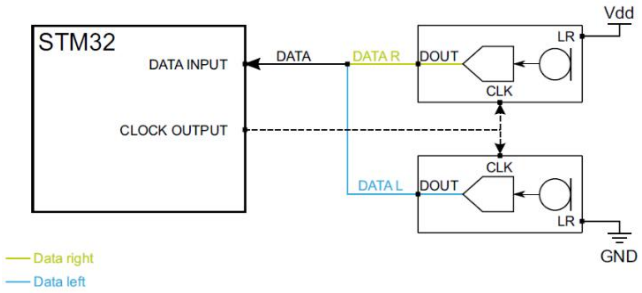

Fig. 10. Stereo configuration: sharing one data line

Two different digital MEMS microphones are connected to the same data line. The first microphone is configured to generate data on the rising edge of the clock signal where the the LR pin connected to Vdd, and the second on the falling edge, where the LR microphone pin connected to GND.

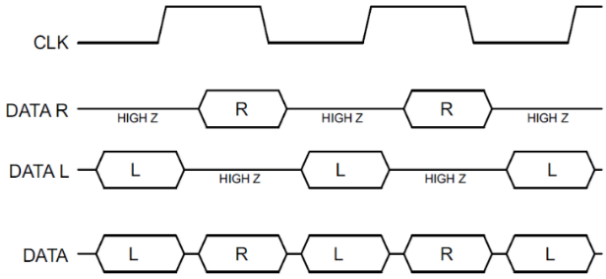

Fig. 11. Data transfer in stereo configuration

The STM32 microcontrollers offer extensive audio processing capabilities with rich connectivity options, including serial and enhanced voice capture interfaces, allowing a user to easily create applications using microphones [3].

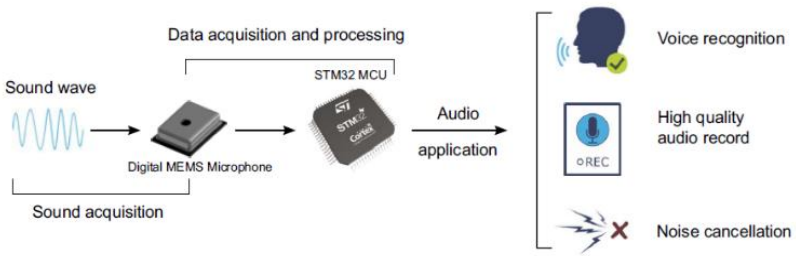

Fig. 12. Audio data acquisition and processing using a MEMS microphone and STM32 MCU

Digital MEMS microphones can be connected to SPI/I2S, SAI and DFSDM (Digital Filter for Sigma Delta Modulators) peripherals of the STM32 MCU in both mono and stereo configurations [3].

The MP45DT02 ST-MEMS microphone outputs a PDM signal, which is a high-frequency stream (1 to 3.25 MHz) of 1-bit digital samples. This output signal is received in blocks of 8 samples using the STM32 I2S synchronous serial interface. The PDM output of the microphone is synchronized with its input clock signal, so the STM32 SPI/I2S peripheral device generates a clock signal for the microphone (Fig. 13).

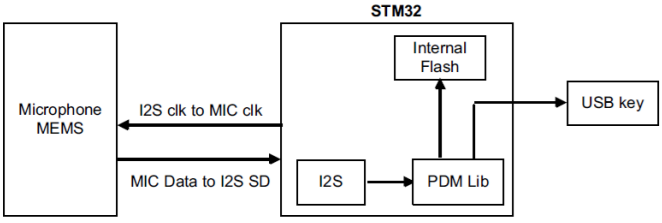

Fig. 13. Block diagram of the microphone connection to the STM32 MCU via I2S

The PDM data are captured via the STM32 I2S serial interface. The data are transferred using DMA to the RAM buffer for processing. After conversion, the PCM data can be processed depending on the application implementation (saving as wave/compressed data on the data storage, transfering to an external DAC).

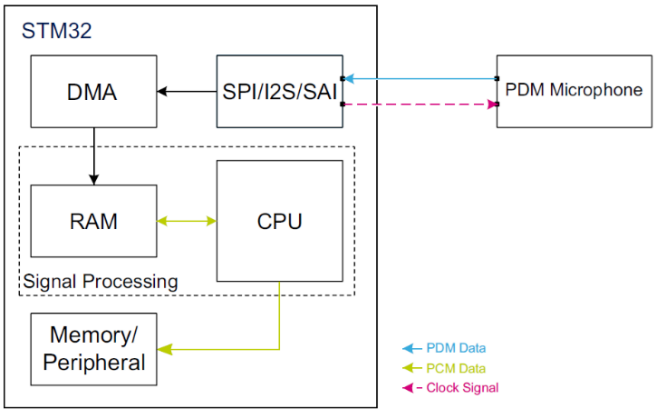

Fig. 14. Block diagram of embedded system
for real-time audio acquisition and processing

The project is based on the STM32F407G-DISC1 Discovery kit with the STM32F407VGT6 MCU and uses a digital ST-MEMS microphone MP45DT02, which is connected via the I2S interface. The MP45DT02 microphone is designed to capture stereo sound. Two microphones can operate simultaneously on the same common bus. If the LR pin is connected to GND, the MP45DT02 is put into left channel mode.

On the STM32F407G-DISC1 board, the LR pin of the MP45DT02 microphone is connected to GND, so the device is always in the left channel mode. The data is valid when the clock is low. Thus, the MP45DT02 microphone expects samples between clock transitions when the data line is stable. In I2S, data is loaded on the falling edge of the clock signal to be valid for sampling on the rising edge of the clock signal.

According to the timing information on the MP45DT02 microphone, it can take up to 16 ns to clear the data from the PDM pin on the I2S bus on the rising clock edge. The STM32F407 clock rise time varies with load capacitance, but is rated at 4 ns at 30 pF. Using I2S works because, in the vast majority of cases, the MP45DT02 takes more than 4 ns to respond to a rising clock, allowing the STM32F4 to sample data before it is flushed.

The connection between the STM32F407 MCU and the MP45DT02 PDM microphone is as follows: the GPIOB10 pin is connected to the CLK pin of the microphone, and the GPIOC03 pin is connected to the DOUT pin of the microphone. The used pins are controlled by the I2S2 hardware interface.

To receive the PCM data with a sample rate of 16 kHz, the microphone clock frequency generated by the interface should be 1.024 MHz for mono mode and 2.048 MHz for stereo mode. To receive the PCM data at a sampling rate of 48 kHz, the microphone clock frequency generated by the interface must be 3.072 MHz for mono mode and 6.144 MHz for stereo mode.

I2S2 peripheral configuration in STM32CubeMX

From the list of MCU peripherals on the Pinout & Configuration –> Multimedia tab, we need to select the I2S2 peripheral device and configure it to operate in Half-Duplex Master Mode. In Fig. 15, it is shown how to enable the I2S2 interface in Half-duplex master mode.

The enabled I2S2_SD, I2S2_CK and I2S2_WS pins are highlighted with green after the I2S2 peripheral GPIO configuration is correct. The clock signal for I2S2 can be configured in mono or stereo mode for 16 kHz and 48 kHz streams. The external HSE = 8 MHz clock generator is used as a clock signal source. In stereo mode, the timer and I2S2 have to use the same clock source, so a PLLR is chosen as the clock source for I2S2. At the input f(VCO clock, PLLI2S) = 1 MHz, the following values for PLLN and PLLR parameters depending on the sampling rate have to be set:

– for the sampling rate of 11 kHz, parameters PLLN 429, PLLR 4, PLLI2SCLK (MHz) 107.25.
– for the sampling frequency of 16 kHz, parameters PLLN 213, PLLR 4, PLLI2SCLK (MHz) 53.25.
– for the sampling frequency of 22 kHz, parameters PLLN 429, PLLR 4, PLLI2SCLK (MHz) 107.25.
– for the sampling frequency of 32 kHz, parameters PLLN 426, PLLR 4, PLLI2SCLK (MHz) 106.5.
– for the sampling frequency of 44 kHz, parameters PLLN 271, PLLR 6, PLLI2SCLK (MHz) 45.16.
– for the sampling frequency of 48 kHz, parameters PLLN 258, PLLR 3, PLLI2SCLK (MHz) 86.

For proper operation, the clock frequency of I2S2 must be higher than its APB frequency. In Fig. 16, the I2S clock configuration in mono mode is shown.

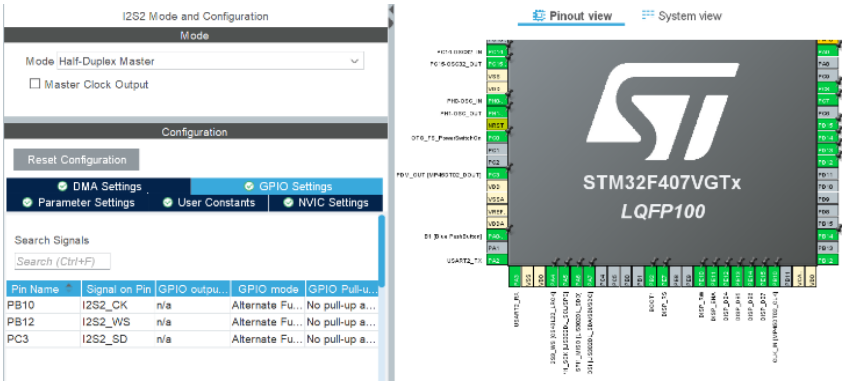

Fig. 15. Mode configuration for I2S2 peripheral

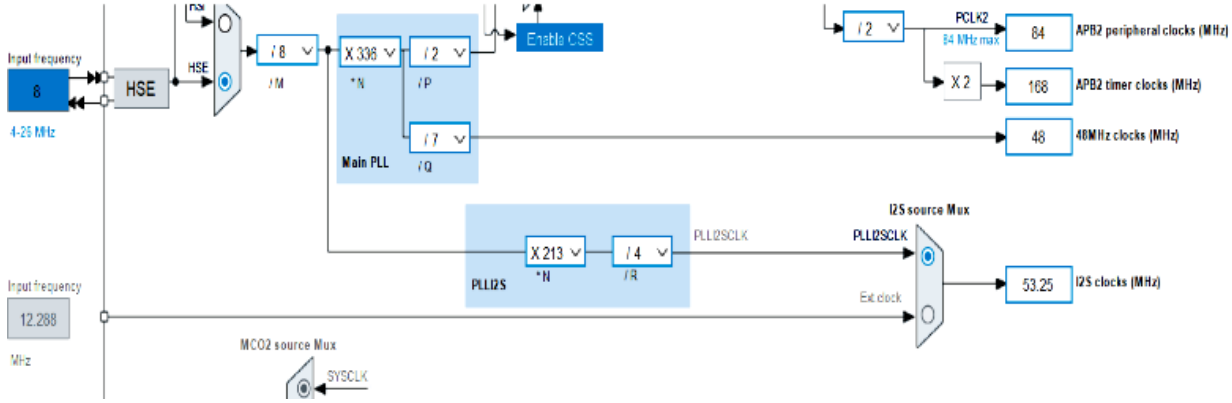

Fig. 16. I2S2 clock configuration

a) I2S2 parameter settings

In the I2S2 configuration window, select the Parameter Settings tab and configure the I2S parameters as follows:

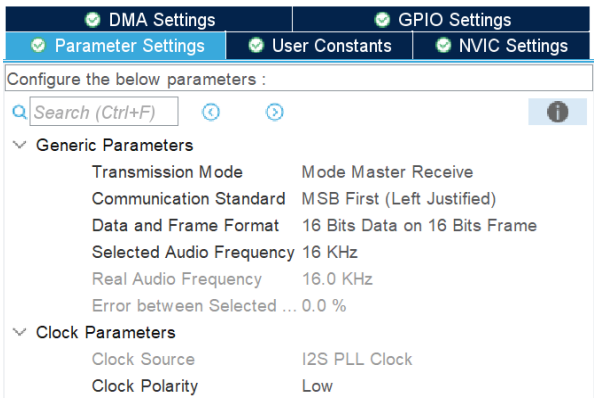

Fig. 17. I2S2 parameter settings

Selected Audio Frequency = AUDIO_SAMPLING-_FREQUENCY.

• Communication Standard = MSB First (Left Justified): I2S reads data on the falling edge of the clock signal. In mono configuration, the LR (L/R) microphone pin must be connected to GND. In Fig. 17, the I2S parameter settings for the 16 kHz audio sampling rate in mono mode is shown.

b) I2S2 DMA settings

In this project, DMA processes the PDM data transfer from I2S2 to the RAM buffer. In the I2S2 configuration window, select the DMA Settings tab and add DMA request. In Fig. 18, the selected I2S DMA settings are shown.

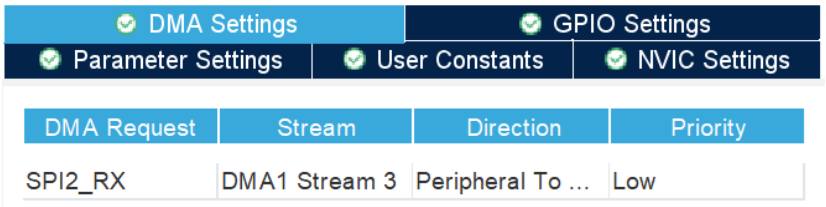

Fig. 18. I2S2/SPI2 DMA settings

Software for real-time audio processing

In order to process audio data, the special software for the STM32F407VGT6 has been developed using the STM32 BSP audio driver and the PDM2PCM library. The PDM2PCM library converts the PDM bit stream from the MEMS microphone into the audio stream in PCM format. The PDM2PCM sound decoding library is an optimized software implementation for decoding a PDM signal and reconstructing an audio signal for digital MEMS microphones connected to the STM32 MCU. This library implements several filters for a 1-bit high-frequency PDM signal output from a digital microphone and converts it to a 16-bit PCM signal at the appropriate sampling rate. The PDM2PCM library is a part of the STM32CubeIDE firmware package, it allows the audio data to be processed from the digital PDM microphones on the STM32F4, STM32F7 and STM32H7 series MCUs [4, 5].

The PDM2PCM library has functions to decimate and filter a PDM stream from a digital microphone to convert it into an output PCM signal stream. The output PCM stream is implemented with a resolution of 16 bits. The sampling frequency of PCM is 16 kHz. Different decimation factors can be configured to adapt to different PDM clock frequencies. The high-pass filter and digital gain can also be adjusted.

The PDM2PCM library accepts as input a PDM signal stream (from 768 kHz to 2.048 MHz) of 1-bit digital samples. This signal is collected in blocks of 8 samples using the STM32 I2S interface. The library is available in different versions, for different MCU cores and development tools.

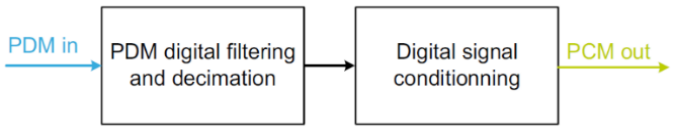

Fig. 19. Digital signal processing

At the first stage, the PDM signal from the microphone is filtered and decimated to obtain the audio signal with the required frequency and resolution. In the second stage, the digital audio signal obtained from the filter is further processed to properly shape the signal using low-pass and high-pass filters. Both of these filters can be turned on/off and adjusted (cutoff frequencies) using the filter initialization function.

In the stereo configuration, if two microphones share a single data line and the PDM data are bit-by-bit interleaved, a software deinterlacing step is necessary to separate the signal from the two microphones before PDM to PCM signal conversion.

PDM2PCM library API calls

Step 1. Creation of a PDM2PCM handler, memory allocation for the filter configuration structure, for the input PDM and output PCM buffers. HW CRC must be enabled to unlock the library.

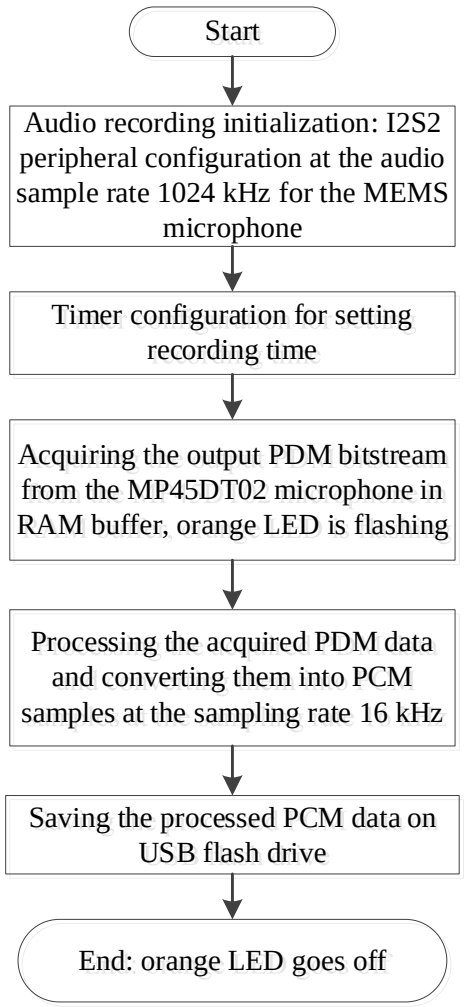

Fig. 20. Algorithm of audio data acquisition and processing

Step 2. Setting the PDM filter parameters and calling the PDM_Filter_Init() function.

Step 3. Setting the required PDM filter configuration values and calling the PDM_Filter_setConfig() function.

Step 4. Reading the input PDM bit stream from the I2S2 interface with byte packing.

Step 5. Invoking the PDM_Filter() function that executes the PDM2PCM algorithm.

Step 6. Recording the output PCM audio stream on the drive.

Step 7: User can change the configuration parameters and call the PDM_Filter_setConfig() function to update the library configuration.

Step 8. If the application and the input PDM stream are still running, the processing cycle returns to step 5, otherwise it terminates.

Step 9: After the processing cycle is complete, the allocated memory must be freed.

The I2S driver is configured to capture audio data from the microphone into STM32 RAM buffer using the DMA controller. The DMA controller generates a DMA1 stream3 global interrupt when the buffer is full and calls the functions HAL_I2S_TxHalfCpltCallback (the first half of the buffer is full) and HAL_I2S_TxCpltCallback when the second half is full, which notifies about the readiness to read the I2S sample buffer.

The operation algorithm and prototype of the STM32-based embedded system for real-time audio acquisition and processing are shown in Fig. 20, 21, respectively.

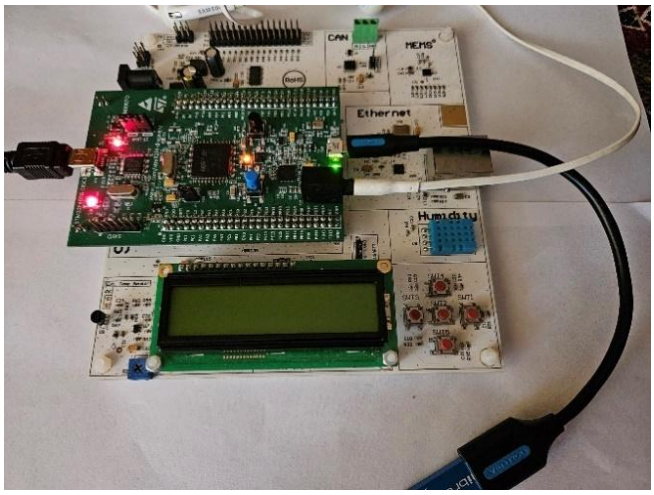

Fig. 21. Prototype of the embedded system for real-time audio acquisition and processing

Conclusions

The prototype of the embedded system for real-time audio data acquisition and processing has been developed. It is based on the STM32F4-DISCOVERY Discovery kit with 32-bit ARM Cortex-M4 STM32F407VGT6 microcontroller. In the embedded system, MP45DT02 ST-MEMS microphone, CS43L22 audio DAC and LCD WH1602B-NYG-CT modules of the STM32F4-DISCOVERY Discovery kit are used.

The STM32F407VGT6 MCU I2S2 peripheral has been configured in Half-Duplex Master Mode to acquire PDM data from the MP45DT02 microphone. It also generates the input clock signal for the MEMS microphone. The I2S2 DMA transfers PDM data from the microphone to the RAM buffer independently of the CPU. The STM32 USB peripheral is configured as a host and MSC (Mass Storage Class) protocol has been implemented for transmitting and receiving audio data from/to USB flash drive. The STM32 I2S3 peripheral is configured in master transmitter mode for transmitting audio data to the CS43L22 DAC. The CS43L22 DAC is used for the recorded audio playback. The embedded software for real-time audio acquisition and processing has been developed in C using the STM32 BSP audio driver and the PDM2PCM library.

References

[1] Electronic resource UM1472 User manual. Discovery kit with STM32F407VG MCU. [Access mode]: https://www.st.com/resource/en/user_manual/um1472-discovery-kit-with-stm32f407vg-mcu-stmicroelectronics.pdf.

[2] Electronic resource AN5027. Interfacing PDM digital microphones using STM32 MCUs and MPUs. [Access mode]: https://www.st.com/resource/en/application_note/an5027-interfacing-pdm-digital-microphones-using-stm32-mcus-and-mpus-stmicroelectronics.pdf.

[3] Електронний ресурс AN3997. Audio playback and recording using the STM32F4DISCOVERY. [Access mode]: https://www.st.com/resource/en/application_note/an3997-audio-playback-and-recording-using-the-stm32f4-discovery-stmicroelectronics.pdf.

[4] Electronic resource AN3998. PDM audio software decoding on STM32 microcontrollers. [Access mode]: https://www.st.com/resource/en/application_note/an3998-pdm-audio-software-decoding-on-stm32-microcontrollers-stmicroelectronics.pdf.

[5] Carmine Noviello. Mastering STM32 – 2nd Edition. A step-by-step guide to the most complete ARM Cortex-M platform, using the official STM32Cube development environment, Lean Publishing, 2018, p. 852.

Андрій Головатий[1], Анджей Лукашевіч[2],
Софія Головата[3], Назарій Клим[4], Костянтин Колесник[5]

[1] Кафедра систем автоматизованого проєктування, Національний університет “Львівська політехніка”, вул. С. Бандери, 12, Львів, Україна, E-mail: andrii.i.holovatyi@lpnu.ua, ORCID 0000-0001-6143-648X,
[2] Кафедра проєктування та експлуатації машин, Білостоцький технологічний університет, вул. Вєйска, 45а, Білосток, Польща, E-mail: a.lukaszewicz@pb.edu.pl, ORCID 0000-0003-0373-4803
[3] Кафедра програмної інженерії, Національний лісотехнічний університет України, вул. Г. Чупринки, 43, Львів, Україна, E-mail: sofi.pobereyko@nltu.edu.ua, ORCID 0000-0001-5904-9683
[4] Кафедра систем автоматизованого проєктування, Національний університет “Львівська політехніка”, вул. С. Бандери, 12, Львів, Україна, E-mail: nazarii.klym.mknsp.2023@lpnu.ua,
[5] Кафедра систем автоматизованого проєктування, Національний університет “Львівська політехніка”, вул. С. Бандери, 12, Львів, Україна, E-mail: kostyantyn.k.kolesnyk@lpnu.ua, ORCID 0000-0001-9396-595X

РОЗРОБЛЕННЯ ВБУДОВАНОЇ СИСТЕМИ ЗБИРАННЯ ТА ОБРОБЛЕННЯ ЗВУКУ В РЕАЛЬНОМУ ЧАСІ НА БАЗІ МІКРОКОНТРОЛЕРА STM32

Отримано: Листопад 03, 2024/ Переглянуто: Листопад 20, 2024 / Прийнято: Листопад 25, 2024

Анотація. Вбудовану систему для збирання та оброблення аудіоданих у реальному часі розроблено із використанням набору STM32F407G-DISC1 Discovery із 32-розрядним мікроконтролером STM32F407VGT6 ARM Cortex-M4. Для його роботи використано базові модулі комплекту STM32F4

DISCOVERY. Периферійний модуль STM32F407VGT6 MCU I2S2 налаштовано у напівдуплексному головному режимі для отримання даних PDM від мікрофона MP45DT02. Периферійний USB-модуль STM32 налаштовано у режимі хоста, а протокол MSC реалізовано для передавання та отримання аудіоданих на/з USB-накопичувач. Периферійний модуль I2S3 мікроконтролера STM32F407VGT6 налаштовано у режимі головного передавача для передавання аудіоданих на ЦАП CS43L22. I2S2 DMA STM32F407VGT6 MCU використовується для передавання даних із мікрофона в буфер оперативної пам'яті, що істотно розвантажує процесор. Кнопки користувача на платі STM32F407G-DISC1 використовуються для управління додатками (відтворення або запис). Програмне забезпечення для збирання та оброблення звуку в режимі реального часу для STM32 MCU розроблено на C із застосуванням аудіодрайвера BSP та бібліотеки PDM2PCM.2

Ключові слова: вбудована система реального часу, збір і обробка аудіо, комплект STM32F407G-DISC1 Discovery, STM32F407VGT6, мікрофон MP45DT02 ST-MEMS, аудіо ЦАП CS43L22, LCD WH1602B-NYG-CT, I2S, SPI, PDM, PCM, формат аудіофайлу WAV.