DOI: 10.35598/mcfpga.2021.001

Features of the Digital Filters Implementation on STM32 Microcontrollers

Oleg Zubkov
ORCID 0000-0002-8528-6540
Department of Microprocessor Technologies and Systems
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
oleh.zubkov@nure.ua

Iryna Svyd
ORCID 0000-0002-4635-6542
Department of Microprocessor Technologies and Systems
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
iryna.svyd@nure.ua

Oleksandr Vorgul
ORCID 0000-0002-7659-8796
Department of Microprocessor Technologies and Systems
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
oleksandr.vorgul@nure.ua

***Abstract*—The purpose of this work is to study the efficiency of compilation of digital filters software implementation for STM32 microcontrollers. The research included: the stage of filter synthesis according to its amplitude-frequency characteristic and order, analysis of the signal at the filter output for various types of the program code optimization. As a result of the research, practical recommendations were given for choosing a compiler and the level of optimization in devices containing digital filters.**

Keywords—compiler, microcontroller, optimization, low pass filter, frequency response

I. Introduction

Digital signal filtering is widely used in modern electronic technology. It is used in: positioning and navigation devices, radar signal processing, multimedia and communication technology, etc. [1, 2] Digital signal filtering allows you to isolate a useful signal from the background of noise or suppress interference spectral components. To implement a digital filter, time sampling and quantization of the analog signal amplitude is performed using an analog-to-digital converter (ADC). After that, the obtained samples are processed using filters with a finite or infinite impulse response (FIR or IIR). The filtering result can be written to a information storage or fed to a digital-to-analog converter (DAC).

Digital filters today are implemented in hardware or software. For hardware implementation, FPGAs are most often used [3-5], which allows achieving high performance and processing signals at frequencies up to several GHz. However, such devices are expensive and require significant financial resources for software development. In most devices for industrial and domestic use, digital filters are implemented in software on specialized DSP processors or general-purpose controllers [6].

In recent years, STM32 controllers have been widely used in industrial, communication, multimedia devices [7, 8]. The F4, F7, H7 series of these controllers have a DSP module that allows you to perform floating point calculations at the hardware level, which reduces signal processing time. The microcontroller manufacturer provides information out filter performance for various microcontroller series. However, the effectiveness of the filter depends on the result of the program compilation. Various compilers are used in modern software development environments STM32CubeIDE, IAR, etc. The user is also given the opportunity to select the levels of program optimization. However, when optimizing a program, the compiler can simplify the program code, which can affect the accuracy of calculations or cancel certain actions. Therefore, the purpose of these studies was to study the effect of using different compilers and optimization modes on the performance and implementation correctness of the filter.

II. Designing a Digital Filtering Device on STM32

A. Filter type selection and its parameters calculation

For practical implementation, a low-pass FIR filter was chosen. For high-quality suppression of interference or out-of-band emissions in practical tasks, high-order filters from the 30th to the 200th are used, depending on the technical requirements for the filter. To calculate the filter coefficients, we used the Filter Builder utility from the Matlab R2014b package [9-11]. 2 filters of the 50th and 100th orders were designed. The mathematical description of the 100th order filter has the form

$$y(n) = \sum_{i=n}^{n-100} x(i) \cdot b(i-n), \qquad (1)$$

where y (n) is the signal value at the filter output, x (i) are the input signal samples, which are stored in the filter memory cells, b (i-n) are the filter coefficients.

Fig. 1 shows the amplitude-frequency characteristic (AFC) of the synthesized filter of the 100th order.

The sampling frequency of the signal at the filter input is 44 kHz. The cutoff frequency of both filters is 2 kHz, the frequency of filter suppression is 4 kHz. Suppression of spectral components at frequencies above 4 kHz in the 50th order filter is 43dB, and in the 100th order filter at least 79dB. The calculated filter coefficients are 32 bit real numbers.

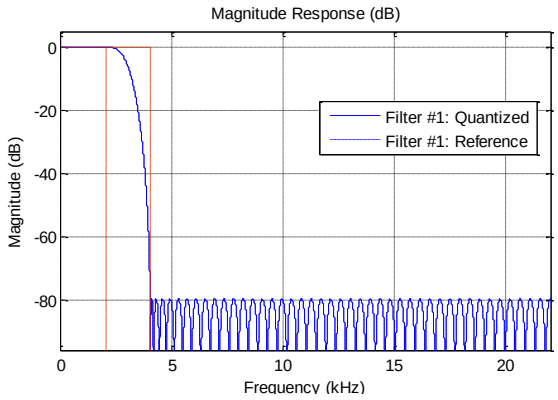

Fig. 1. Frequency response of the synthesized low-pass filter.

B. Hardware implementation of the filter

The STM32F407VG microcontroller with a core clock frequency of 168 MHz was chosen as the hardware platform. This microcontroller contains a built-in 12-bit ADC and DAC. Also, due to the presence of a built-in multi-channel controller for direct memory access (DMA), it is possible to transfer the results of analog-to-digital conversion to RAM and samples from the filter output to the DAC in parallel with the execution of the main task. The timing of the start of converting the analog signal to digital form and digital samples to analog voltage is set by the timer. It generates internal events, according to which the ADC and DAC work synchronously. Then the functional diagram of the digital filtering device on the STM32F407VG has the form (Fig.2).

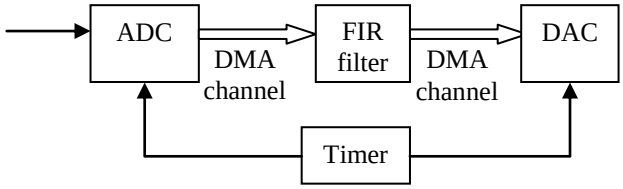

Fig. 2. Functional diagram of the filtration device.

During the research, the developed digital filter worked in two modes: without input data buffering and with input data buffering. In the first mode, when one sample of the analog signal arrived from the ADC, it was fed to the input of the filter and the result of filtering through the DMA channel was fed to the input of the DAC. In the second mode, 128 samples of the input signal were transferred to the RAM buffer via the DMA channel. Upon completion of the buffer filling, an interrupt from the DMA channel was generated and the contents of the buffer were filtered. The filtered data array was transferred to the DAC. The second mode is more efficient, since it takes a certain amount of time to enter and exit the interrupt when processing single samples.

During the research, generator Siglent sdg 2122 was used to generate signals at the ADC input. The signal at the DAC output was investigated using an Intrustar ISDS 220B USB oscilloscope and an Siglent SSA3021 spectrum analyzer.

III. Research Results

To study the software implementation of the filter, the two most popular development platforms IAR Embedded Workbench 8.3 and STM32CubeIDE 1.4 were selected. The first platform has a certified compiler and the ability to choose one of 4 optimization modes: None - no optimization, Low, Medium, High. Produce microcontrollers offers its own development environment STM32CubeIDE, which uses the free popular GCC compiler. In this platform, the user can also select the optimization level: O0 - no optimization, O1 - the main optimization level tested on many tasks, Og - the lowest optimization level for debug mode, O2 - speed optimization with increasing code, O3 - speed optimization with increasing code, Os - optimization for speed without increasing the code, Ofast - maximum performance.

The main parameters for the study were: the speed of the filter (loading the microcontroller to perform filtering operations), the correspondence of the filter characteristics to those obtained during synthesis

Table 1 shows the results of studying the filter speed and the results of measuring the attenuation in the suppression band after compiling the program in the IAR Embedded Workbench.

TABLE I. Research Results for IAR Embedded Workbench

Parameter	**Optimization**					
	None		***Low***		***Medium***	
	Filter order					
	50	***100***	***50***	***100***	***50***	***100***
Efficiency without/with buffer, µs	12,3/ 11,9	24,1/ 23,5	11,9/ 11,6	23,97/ 22,9	6,45/ 6,13	13,05/ 12,09
Attenuation on 4kHz, dB	39	-70	39	-65	-9	-20

When using the optimization levels Medium and High, the suppression of spectral components above 4 kHz does not match the synthesis results. Therefore, when using the IAR Embedded Workbench to compile digital filter code, optimization levels higher than Low should not be used.

Table 2 shows the results of studying the speed of the filter and the results of measuring the attenuation in the suppression band after compiling the program in the STM32CubeIDE environment.

TABLE II. Research Results for STM32CubeIDE

Parameter	**Optimization**					
	O0		***Og***		***O1***	
	Filter order					
	50	***100***	***50***	***100***	***50***	***100***
Efficiency without/with buffer, µs	20/ 17,6	37,9/ 28,9	7,9/ 6,3	15,5/ 12,8	4,2/ 3,8	8,3/ 6,5
Attenuation on 4kHz, dB	39	-70	39	-40	-12	-23

When using STM32CubeIDE to compile a software implementation of a digital filter, it is unacceptable to use optimization levels higher than Og, since the measured filter characteristics do not correspond to the synthesis results.

The usage of data buffering can reduce processing time by up to 18%.

IV. CONCLUSIONS

In the absence of optimization, the program compiled in the STM32CubeIDE environment is inferior in performance to the program compiled in the IAR Embedded Workbench environment. However, at the first level of optimization, the controller manufacturer's GCC compiler provided a significant performance advantage over the IAR Embedded Workbench compiler. Application of any optimization types in STM32CubeIDE leads to a significant deviation of the filter characteristics from the calculated ones.

REFERENCES

[1] Tirthadip Sinha,JaydebBhaumik, “Design of Computationally Efficient Sharp FIR Filter Utilizing Modified Multistage FRM Technique for Wireless Communications Systems” *Journal of Electronic Science and Technology*, Volume 17, Issue 2, 2019, pp. 185-192.

[2] Yuan Xu, Yuriy S Shmaliy, Luchi Hua, Liyao Ma and Yuan Zhuang, “Decision tree-extended finite impulse response filtering for pedestrian tracking over tightly integrated inertial navigation system/ultra wide band data”, *Measurement Science and Technology*, Volume 32, Number 3, 2021, pp.217-228.

[3] Sumbal Zahoor, Shahzad Naseem, “Design and implementation of an efficient FIR digital filter” *Cogent Engineering* vol. 4, 2017, pp. 123-135.

[4] Pandey, B., Jain, A., Kumar, P., Hussain, A., Levy, J. y Chowdhry, B. S. “Energy Efficient and High–Performance FIR Filter Design on Spartan–6 FPGA”. *3C Tecnología. Glosas de innovación aplicadas a la pyme*. Edición Especial, Mayo 2019, pp. 36–49.

[5] I. Svyd, O. Maltsev, L. Saikivska and O. Zubkov, "Review of Seventh Series FPGA Xilinx", *I International Scientific and Practical Conference*, 2019. doi: 10.35598/mcfpga.2019.008.

[6] Moutaman Mirghani, “Implementation of Matched Filters Using Microcontrollers”, *Conference of Basic Sciences and Engineering Studies* (SGCAC), 2016, pp. 62-66.

[7] O. Zubkov, “Teaching trends in the design of electronic devices on microcontrollers”. Specialized show “KharkivProm Days. Wi-Fi and Efficiency ". *Collection of materials in the forum of the section “Automation, electronics and robotics*. *Development strategy and innovation technologies*". - Kharkiv, KNURE, Vistavkova company ADT, 2019 .-- pp. 40-42.

[8] O. Vorgul, O. Zubkov, I. Svyd and V. Semenets, "Teaching microcontrollers and FPGAs in Quarantine from Coronavirus: Challenges and Prospects", *MC&FPGA-2020*, 2020. doi: 10.35598/mcfpga.2020.005.

[9] Zena Ez Dallalbashi, “MatLab Based Design and Implemetation of Digita Filter” *International Journal of Computer Science and Network Security*, VOL.20 No.1, 2020, pp. 91–101.

[10] I. Svyd, O. Maltsev, O. Zubkov and L. Saikivska, "Matlab Use in Design of Digital Systems on the FPGA in CAD Xilinx VIVADO", *I International Scientific and Practical Conference*, 2019. doi: 10.35598/mcfpga.2019.010.

[11] Jie Zhao, “Modeling and Simulation of Digital Filter” *4th National Conference on Electrical, Electronics and Computer Engineering*, 2015, pp.1333-1338.