УДК 621.391
DOI: 10.18799/29495407/2024/4/72
Шифр специальности ВАК 2.2.4

Обзор модуля ЦАП на базе микросхемы MCP4725 и модуля ЦАП на основе интегрирующей RC-цепи

К.А. Волобуев, И.В. Трубин, В.Г. Трубин✉

Новосибирский Государственный Технический Университет, Россия, г. Новосибирск

✉trubin@corp.nstu.ru

Аннотация. В технических кибернетических системах невозможно обойтись без обмена информацией между устройствами. Большинство современных управляющих устройств являются цифровыми, но не редки случаи, когда для управления исполнительными утройствами требуется аналоговый сигнал, который можно изменять в определённом диапазоне. Для получения аналогового сигнала из цифрового используется специальное устройство – цифро-аналоговый преобразователь. В данной статье приводится краткая информация о типах сигналов и их различиях. Рассматриваются два широко распространенных модуля цифро-аналогового преобразования, с помощью которых можно преобразовать цифровой сигнал с микроконтроллера в аналоговый управляющий сигнал 0–10 В. Управление модулями происходит с помощью отладочной платы «Blue Pill» на базе микроконтроллера STM32F103. Модуль цифро-аналогового преобразователя на микросхеме MCP4725 управляется по интерфейсу I2C. Другой модуль на основе RC-цепи преобразует ШИМ-сигнал с микроконтроллера в аналоговый сигнал 0–10 В. Статья будет полезна к ознакомлению студентам, инженерам, аспирантам, желающим изучить работу цифро-аналогового преобразователя на основе ШИМ и RC-цепи, а также подключение устройств по интерфейсу I2C.

Ключевые слова: цифро-аналоговый преобразователь (ЦАП), ШИМ, аналоговый сигнал, дискретный сигнал, квантованный сигнал, цифровой сигнал, MCP4725, RC-цепь, STM32F103, отладочная плата Blue Pill

Для цитирования: Волобуев К.А., Трубин И.В., Трубин В.Г. Обзор модуля цифро-аналогового преобразователя на базе микросхемы MCP4725 и модуля цифро-аналогового преобразователя на основе интегрирующей RC-цепи // Известия Томского политехнического университета. Промышленная кибернетика. – 2024. – Т. 2. – № 4. – С. 1–12. DOI: 10.18799/29495407/2024/4/72

UDC 621.391
DOI: 10.18799/29495407/2024/4/72

Overview of the digital-to-analog converter module based on the MCP4725 chip and the digital-to-analog converter module based on the integrating RC circuit

K.A. Volobuev, I.V. Trubin, V.G. Trubin✉

Novosibirsk State Technical University, Novosibirsk, Russian Federation

✉trubin@corp.nstu.ru

Abstract. In technical cybernetic systems, it is impossible to do without exchanging information between devices. Most modern control devices are digital, but it is not uncommon for an analog signal to be required to control executive devices, which can be changed in a certain range. To receive an analog signal from a digital one, a special device is used – digital-to-analog converter. This article provides brief information about the types of signals and their differences. Two widely used digital-to-analog conversion modules are considered. They can be used to convert a digital signal from a microcontroller into an analog control signal of 0–10 V. The modules are controlled using the "Blue Pill" debugging board based on the STM32F103 microcontroller. The digital-to-analog converter module on the MCP4725 chip is controlled via the I2C interface. Another module

based on the RC circuit converts the PWM signal from the microcontroller into an analog signal of 0–10 V. The article will be useful for students, engineers, and graduate students who want to study the operation of a PWM-based digital-to-analog converter and RC circuit, as well as connecting devices via the I2C interface.

Keywords: digital-to-analog converter, PWM, analog signal, discrete signal, quantized signal, digital signal, MCP4725, RC circuit, STM32F103, Blue Pill debug board

For citation: Volobuev K.A., Trubin I.V., Trubin V.G. Overview of the digital-to-analog converter module based on the MCP4725 chip and the digital-to-analog converter module based on the integrating RC circuit. *Bulletin of the Tomsk Polytechnic University. Industrial Cybernetics*, 2024, vol. 2, no. 4, pp. 1–12. DOI: 10.18799/29495407/2024/4/72

Введение

В наше время существует множество видов управляющих сигналов. В промышленности одним из таких является аналоговый сигнал 0–10 В, с помощью которого можно управлять различными устройствами. Например, привести клапан или заслонку в нужное положение, изменить мощность электронагревателя или скорость вращения электродвигателя с помощью частотного преобразователя.

Но микроконтроллеры и промышленные контроллеры, которые управляют периферийными устройствами, не всегда имеют достаточно аналоговых выходов. В данной статье исследуются два модуля, с помощью которых можно получить аналоговый выход 0–10 В, и проводится анализ их достоинств и недостатков.

Типы сигналов

Точные определения для аналогового и цифрового сигналов задокументированы в ГОСТ 17657-79. Для наилучшего понимания далее будут приведены упрощенные определения.

Аналоговый сигнал (рис. 1, *а*) – непрерывный по времени сигнал, имеющий неограниченно большое множество значений.

Дискретный сигнал (рис. 1, *б*) – сигнал, значение которого определено только в отдельные моменты времени. В большинстве случаев время между измеряемыми значениями сигнала имеет равные промежутки.

Таким образом, на графике дискретными являются значения напряжения в точках пересечения графика напряжения от времени (выделен синим) и вертикальных линий, обозначающих моменты времени дискретизации.

Квантованный сигнал (рис. 1, *в*) – непрерывный во времени сигнал, который разбит на конечное число равных по амплитуде участков (уровней).

Цифровой сигнал (рис. 1, *г*) – дискретный по времени и квантованный по уровню сигнал.

Стоит упомянуть, что достаточно часто под термином «цифровой сигнал» имеют в виду двоичный цифровой сигнал данных, который представляет собой прямоугольные импульсы, где логические нули (0) – это минимумы на графике, а логические единицы (1) – максимумы. В дальнейшем в данной статье за понятие «цифровой сигнал» будем принимать эту формулировку.

В электронике широко распространены два преобразования:

- Аналого-цифровое преобразование сигнала (АЦП);
- Цифро-аналоговое преобразование сигнала (ЦАП).

В табл. 1 приведено краткое сравнение свойств аналогового и цифрового сигналов.

Таблица 1. *Сравнение аналогового и цифрового сигналов*

Table 1. *Comparison of analog and digital signals*

Пункт сравнения Point comparison	Аналоговый сигнал Analog signal	Цифровой Сигнал Digital signal
Определение Definition	Непрерывный сигнал, который изменяется во времени Continuous signal that changes over time	Дискретный сигнал, который несет информацию в двоичной форме Discrete signal that carries information in binary form
Диапазон значений Range of values	Содержит огромное количество значений, которые могут быть как положительными, так и отрицательными. Из-за этого более сложен в анализе Contains a huge number of values that can be both positive and negative. Because of this, it is more difficult to analyze	Значения ограничены определенным диапазоном. Каждое значение может быть представлено в бинарном виде (последовательностью логических 0 и логических 1) Values are limited to a certain range. Each value can be represented in binary form (a sequence of logical 0 and logical 1)
Помехоустойчивость Noise immunity	Имеет большую склонность к искажению Has a high tendency to distortion	Менее подвержены искажениям Less susceptible to distortion
Примеры Examples	Колебания воздуха от речи человека, показания ртутного термометра, рисунок на бумаге Air vibrations from human speech, mercury thermometer readings, drawing on paper	Сигналы в компьютерах, звуковой файл, показания цифрового термометра, векторный или растровый рисунок Computer signals, sound file, digital thermometer readings, vector or raster image

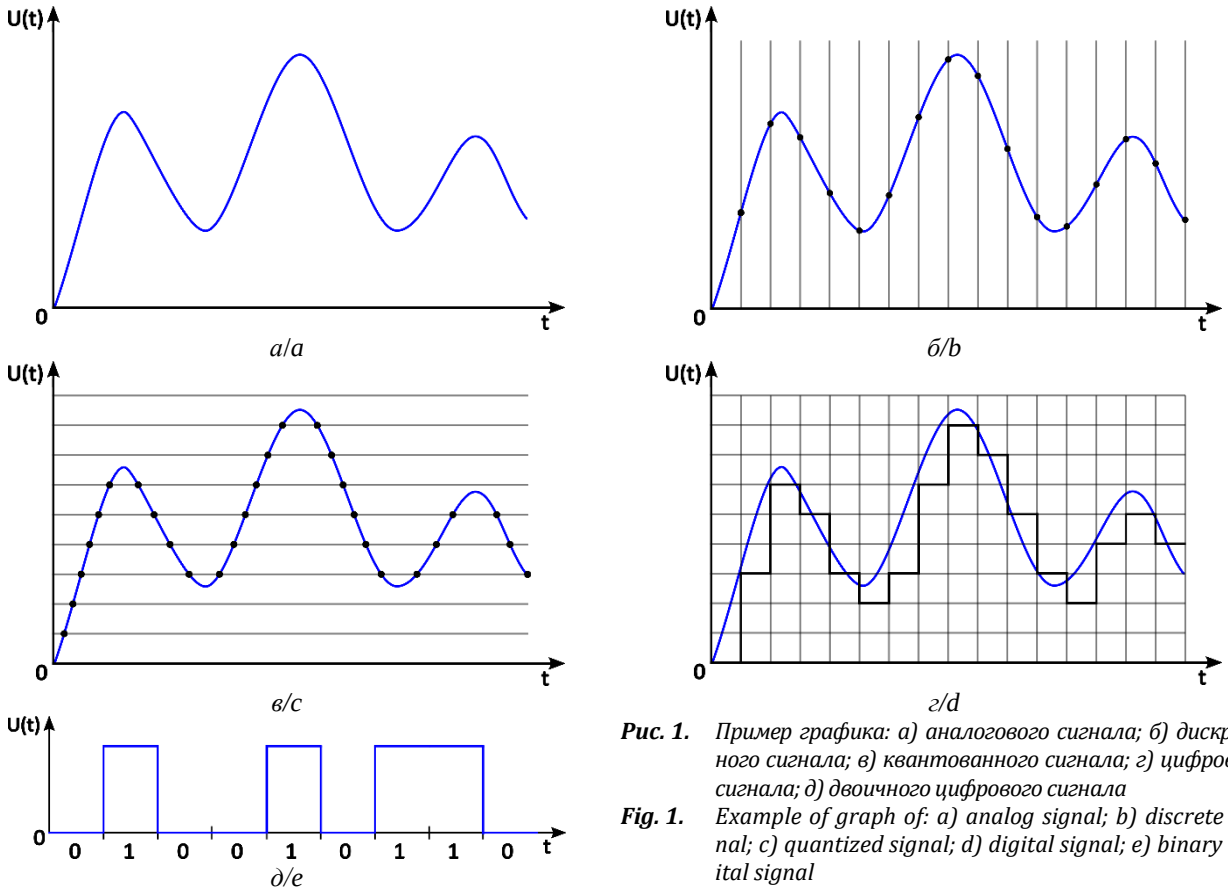

Рис. 1. *Пример графика: а) аналогового сигнала; б) дискретного сигнала; в) квантованного сигнала; г) цифрового сигнала; д) двоичного цифрового сигнала*

Fig. 1. *Example of graph of: a) analog signal; b) discrete signal; c) quantized signal; d) digital signal; e) binary digital signal*

Модуль MCP4725

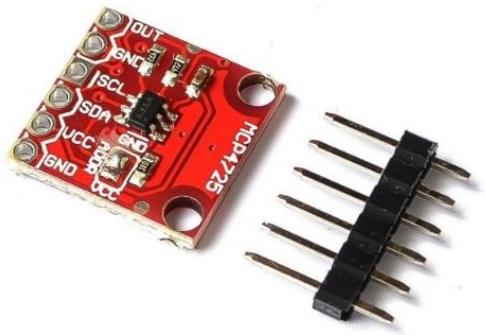

Рис. 2. *Модуль ЦАП на микросхеме MCP4725 с интерфейсом I2C*

Fig. 2. *Digital-to-analog converter (DAC) module on the MCP4725 chip with I2C interface*

Данный модуль является цифро-аналоговым преобразователем, управляемым по интерфейсу *I2C*. В табл. 2 указано назначение его выводов. Ниже приведены основные технические характеристики модуля:

- разрешение: 12 разрядов;
- встроенная энергонезависимая память: *EEPROM*;
- два режима питания: нормальный и режим пониженного энергопотребления;
- среднее время установки выходного напряжения: 6 μs;
- напряжение питания: 2,7–5,5 В;
- максимальный потребляемый ток: 400 мкА;
- максимальный выходной ток: 25 мА;
- тип выхода: *Rail-to-Rail*;
- интерфейс управления: *I2C*;
- возможность изменения I2C адреса: Микросхема MCP4725 имеет 7 бит адреса, четыре из которых (1100) всегда одинаковы и обозначают код устройства, два устанавливаются производителем по запросу заказчика (по умолчанию 00) и один бит устанавливается в зависимости от логического уровня на внешнем контакте A0;
- на модуле два доступных адреса (0×60 или 0×61);
- скорость интерфейса I2C: номинальная (100 Кбит/с), высокая (400 Кбит/с), максимальная (3,4 Мбит/с);
- широкий диапазон рабочих температур: –40…+125 °C.

Таблица 2. *Назначение контактов модуля*

Table 2. *Module contact assignment*

Контакт модуля MCP4725 MCP4725 module contact	Назначение Purpose
OUT	Выходное аналоговое напряжение Output analog voltage
GND	Земля для выходного аналогового напряжения Ground for analog output voltage
SCL	Линия тактирования I2C I2C clock line
SDA	Линия данных I2C I2C data line
VCC	Питание и опорное напряжение 2,7–5,5 В Power and reference voltage 2.7–5.5 V
GND	Земля для опорного напряжения Ground for reference voltage

Модуль *MCP4725* [1, 2] не имеет встроенного источника опорного напряжения (ИОН) и использует в качестве опорного напряжение питания. Поэтому для высокой точности ЦАП важно, чтобы источник питания на выводе *VCC* был как можно более «чистым и точным».

Как упоминалось выше, в промышленности используется сигнал 0–10 В, а модуль *MCP4725* имеет выход 0–5 В. Для того чтобы преобразовать сигнал 0–5 В к сигналу 0–10 В можно использовать операционный усилитель (ОУ) в режиме неинвертирующего усилителя с коэффициентом усиления 2. В таком случае схеме понадобится два напряжения, одно – для питания операционного усилителя, другое – для питания микроконтроллера и модуля ЦАП. Можно питать всю схему от 12 В, а для питания микроконтроллера и ЦАП понижать входное напряжение с 12 до 5 В.

На рис. 3 показана принципиальная схема подключения модуля к микроконтроллеру *STM32F103* [3], установленному на отладочной плате «*Blue Pill*» [4]. Здесь важно еще раз отметить, что модуль *MCP4725* считает напряжение питания опорным, следовательно, при отклонениях по питанию будут наблюдаться отклонения по выходу. В связи с этим в качестве понижающего стабилизатора рекомендуется использовать *TL431B* (который имеет погрешность выходного напряжения не более 0,5 %) или другие стабилизаторы с высокой точностью выходного напряжения.

В этой статье использован стабилизатор *L7805*. Данный компонент имеется в широком доступе и установлен только в качестве примера, он не является высокоточным и не рекомендуется к использованию в серьезных технических проектах. **С выхода *VOUT1* можно получать аналоговый сигнал 0–5 В. Для того чтобы получить диапазон напряжений 0–10 В необходимо подключить операционный усилитель и снимать напряжение с выхода *VOUT2*.**

При необходимости получить больший выходной ток можно добавить транзистор на выходе операционного усилителя.

Для работы с модулем ЦАП *MCP4725* необходимо передавать управляющие команды по интерфейсу *I2C*. Рассмотрим алгоритм управления устройством с компьютера при помощи микроконтроллера.

С компьютера в программе для взаимодействия с виртуальным *COM-портом* «*Terminal*» [5] будем вводить число, обозначающее желаемое напряжение на выходе ЦАП. Число по интерфейсу *UART* будет передаваться на микроконтроллер *STM32F103*, который имеет интерфейс *I2C*, необходимый для взаимодействия с модулем *MCP4725*.

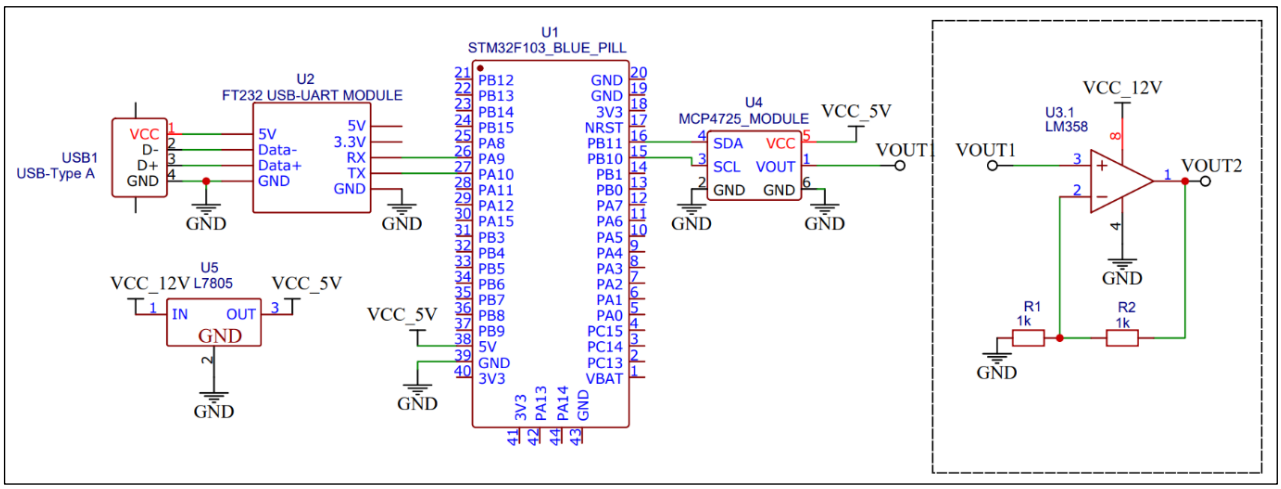

Рис. 3. *Схема подключения модуля MCP4725*

Fig. 3. *Connection diagram of the MCP4725 module*

Так как ЦАП является 12-битным, он разбивает диапазон напряжений на 4096 шагов. Примем, что ЦАП подключён к источнику питания 5 В. Тогда в идеальных условиях значению 4095 на входе будет соответствовать напряжение 5 В, следовательно 0–0 В, 2048 – 2,5 В и т. д.

Ниже приведена программа для микроконтроллера *STM32F103*.

Программный код

В файле "*main_init.c*" настроим подсистему *USART1* в режиме работы последовательного интерфейса *UART*.

Блок кода 1. *Файл main_init.c*

```
// ======== Настройка USART1 ===========
// Включаем тактирование порта GPIOA
// и блока альтернативных функций
RCC->APB2ENR |= (RCC_APB2ENR_IOPAEN |
RCC_APB2ENR_AFIOEN);
RCC->APB2ENR |= RCC_APB2ENR_USART1EN;
// Включаем тактирование USART1
// PA9 (TX1) AFIO Push-Pull
// PA10 (RX1) HiZ
// Вначале обязательно устанавливаем
// пары бит в "00"
GPIOA->CRH &= ~(GPIO_CRH_MODE9 | GPIO_CRH_CNF9
| GPIO_CRH_MODE10 | GPIO_CRH_CNF10);
// Потом нужные биты устанавливаем в '1'
GPIOA->CRH |= (GPIO_CRH_MODE9_0 |
GPIO_CRH_CNF9_1 | GPIO_CRH_CNF10_0);
// Рассчитаем частоту для связи с ПК
// PCLK2/Baud = 72000000 / 115200 = 625
USART1->BRR = 625;
// Включаем USART, передатчик и приемник
USART1->CR1 = USART_CR1_UE | USART_CR1_TE |
USART_CR1_RE;
// Разрешаем прерывание по приёму RX
USART1->CR1 |= USART_CR1_RXNEIE;
// Настройки по умолчанию: 8 инф. бит,
// 1 стоповый бит, контроля чётности нет
// Разрешаем прерывания UART1 в
// контроллере прерываний
NVIC_EnableIRQ(USART1_IRQn);
```

Затем в этом же файле настроим интерфейс *I2C*.

Блок кода 2. *Файл main_init.c*

```
// ========== Настройка I2C ============
// Включаем тактирование порта GPIOB
RCC->APB2ENR |= RCC_APB2ENR_IOPBEN;
// Включаем тактирование I2C2
RCC->APB1ENR |= 1 << 22;
// PB10, PB11-выходы с открытым стоком
// альтернативной функцией,
// частотой 50 Гц
GPIOB->CRH |= 0x0000FF00;
// Частота шины APB1 36MHz
I2C2->CR2 |= 36;
// CRR = 36 000 000 / (2* 100 000) = 180
// 100 000 - частота I2C,
// CR2 = 36 000 000
I2C2->CCR |= 180;
// Записываем максимальное время
// нарастания фронта I2C в тактах
I2C2->TRISE = 37; // CR2+1
```

Далее перейдем в файл "*main.c*". Объявим используемые библиотеки и глобальные переменные.

Блок кода 3. *Файл main.c*

```
// Используемые библиотеки и константы
#include "stm32f10x.h"
#include "main_init.c"
#include "main_functions.c"

// == Описание глобальных переменных ==
// Массив входного буфера
char Buffin[128];
// Массив выходного буфера
char Buffout[128];
// Индекс массива входного буфера
uint8_t bf = 0;
// Флаг принятых по UART данных
uint8_t flag = 0;
```

Настроим обработчик прерывания *USART1*.

Блок кода 4. *Файл main.c*

```
// ==== Обработка прерывания USART1 ====
// USART1->SR, бит RXNE сбрасывается при
// чтении USART1->DR,
// записывать в него ноль нужно только
// при мультибуферной коммуникации
void USART1_IRQHandler() {
 Buffin[bf++] = USART1->DR;
 if (Buffin[bf - 1] == '\r' ||
 Buffin[bf - 1] == '\n') {
 flag = 1;
 }
}
```

Проинициализируем настройки и в бесконечном цикле будем проверять входной буфер на наличие символов, принятых по интерфейсу *UART*. Если принятые символы являются числом, будем отправлять пакет данных модулю ЦАП.

Блок кода 5. *Файл main.c*

```
// =============== MAIN ===============
int main(void) {
// Инициализация МК
 main_init();
 while (1) {
// ========== ГЛАВНЫЙ ЦИКЛ =============
// Если что-то принято по UART
 if (flag == 1) {
// Принятые данные расшифровываем
// как число 0-4095
 uint16_t dac = 0;
 dac = Rxdecoder(bf, &Buffin[0]);
// Делаем защиту от переполнения
 if (dac > 4095) dac = 4095;
// Очищаем входой буффер и его индекс
 for (uint8_t i = 0; i < bf; i++) {
 Buffin[0] = 0;
 }
 bf = 0;
```

```
 flag = 0;
// Выведем по UART сообщение
// о начале записи и 12 битное число
 Tx1Str("write ");
 Uint16ToStr(dac, &Buffout[0]);
 Tx1Str(&Buffout[0]);
 Tx1Str("\r\n");
 I2C_write_MCP4725(dac);
// Передаем по UART успешное завершение
 Tx1Str("write ok\r\n");
 }
 delay_ms(10);
 }
}
```

Напишем функцию отправки пакета данных по интерфейсу *I2C*.

Блок кода 6. *Файл main_functions.c*

```
void I2C_write_MCP4725(uint32_t data) {
// Создаем 3 байта посылки
 uint8_t byte[3];
// Контрольный байт
 byte[0] = 0b01000000;
// Старший байт
 byte[1] = data >> 4;
// Младший байт
 byte[2] = data << 4;
// Начинаем передачу данных по I2C
// Разрешаем работу
 I2C2->CR1 |= 1;
// Формируем сигнал "Старт"
 I2C2->CR1 |= 1 << 8;
// Ожидаем окончания формирования
// сигнала "Старт"
 while (!(I2C2->SR1 & 1)) {}
// Перед передачей необходимо прочитать
// регистр SR1, для сброса бита SB.
 (void)I2C2->SR1;
// Адрес устройства, где первый бит
// слева – master1/slave0,
// последний read1/write0
 I2C2->DR = 0b11000000;
// Ожидаем окончания передачи
// адреса устройства
 while (!(I2C2->SR1 & (1 << 1))) {}
 (void)I2C2->SR1;
 (void)I2C2->SR2;
// Передаем контрольный байт
 I2C2->DR = byte[0];
// Ожидаем окончания передачи
 while (!(I2C2->SR1 & (1 << 2))) {}
// Передаем старший байт
 I2C2->DR = byte[1];
// Ожидаем окончания передачи
 while (!(I2C2->SR1 & (1 << 2))) {}
// Передаем младший байт
 I2C2->DR = byte[2];
// Ожидаем окончания передачи
 while (!(I2C2->SR1 & (1 << 2))) {}
// Формируем сигнал "Стоп"
 I2C2->CR1 |= 1 << 9;
}
```

Плюсом приведенной реализации отправки пакета данных по интерфейсу *I2C* является ее краткость. **Большим минусом является вероятность «зависания» в циклах *while* во время ожидания ответов от модуля.** Это может быть связано с помехами или плохим контактом.

Реализуем функцию расшифровки принятого массива символов (строки) по *USART1*.

Блок кода 7. *Файл main_functions.c*

```
uint16_t Rxdecoder(uint16_t Number,
char* pStr) {
// Функция сохраняет принятый массив
// символов как число
 uint16_t k = 0;
 uint16_t m = 1;
 for (uint16_t i = Number; i > 0; i--) {
 if (pStr[i - 1] >= '0' &&
 pStr[i - 1] <= '9') {
 k += (pStr[i - 1] - '0') * m;
 m = m * 10;
 }
 }
 return k;
}
```

Напишем функцию преобразования беззнакового 16-битного числа в строку.

Блок кода 8. *Файл main_functions.c*

```
void Uint16ToStr(uint16_t Number,
char* pStr) {
 static int8_t i;
 for (i = 4; i != -1; i--) {
 pStr[i] = (Number % 10) + 0x30;
 Number /= 10; }
// Добавляем символ конца строки
 pStr[5] = 0;
}
```

Реализуем функции отправки данных по интерфейсу *USART1*.

Блок кода 9. *Файл main_functions.c*

```
// Функция передачи символа
// по USART1 по готовности
void Tx1(char Symbol) {
 while ((USART1->SR & USART_SR_TXE) == 0) {};
 USART1->DR = Symbol;
}

// Функция передачи массива символов
// (строки) по USART1
// пока не встретится "0" байт
void Tx1Str(char* pStr) {
 static uint8_t i;
 i = 0; do Tx1(pStr[i++]);
 while (pStr[i] != 0);
}
```

Напишем функции для создания программных задержек.

Блок кода 10. *Файл main_functions.c*

```
// Функция задержки в мкс,
// (можно от 1 мкс)
// Результат: задержка + 1 мкс
void delay_us(uint32_t us) {
 static volatile uint32_t n;
 // SystemCoreClock/1000000 = 72
 n = us * 72;
 DWT_CYCCNT = 0;
 while (DWT_CYCCNT < n);
}
// Функция задержки в мс от 1 мс до 59 // се-
кунд (дальше переполнение)
void delay_ms(uint32_t ms) {
 delay_us(ms * 1000);
}
```

Важно отметить, что у модуля есть энергонезависимая память *EEPROM*, хранящая значение напряжения, которое должно установиться на выходе при включении питания модуля. **По умолчанию на его выходе устанавливается половина питающего напряжения (2,5 В), что может быть неудобным при управлении некоторыми устройствами.**

При необходимости это значение можно перезаписать.

В табл. 3 показаны усредненные значения на выходах модуля *MCP4725* (*VOUT1*) и операционного усилителя (*VOUT2*) в зависимости от переданного с компьютера числа. Приведенные ниже значения напряжения получены с помощью мультиметра *Mastech MS8239C*.

Таблица 3. *Напряжения на выходе модуля и операционного усилителя*

Table 3. *Voltages at the output of the module and operational amplifier*

Значение Meaning	%	VOUT1, В	VOUT2, В
0	0	0,00	0,06
41	1	0,05	0,15
205	5	0,25	0,54
410	10	0,50	1,01
1229	30	1,48	3,00
2048	50	2,50	5,00
4867	70	3,48	6,99
3686	90	4,50	8,98
3890	95	4,75	9,45
4054	99	4,94	9,86
4095	100	4,97	9,92

На рис. 4, 5 приведены осциллограммы напряжения на выходе модуля *MCP4725* (*VOUT1*). Осциллограмма, представленная на рис. 4, позволяет определить время переходного процесса при изменении входного кода с 0 на 4095. Осциллограмма, представленная на рис. 5, позволяет определить уровень пульсаций выходного сигнала.

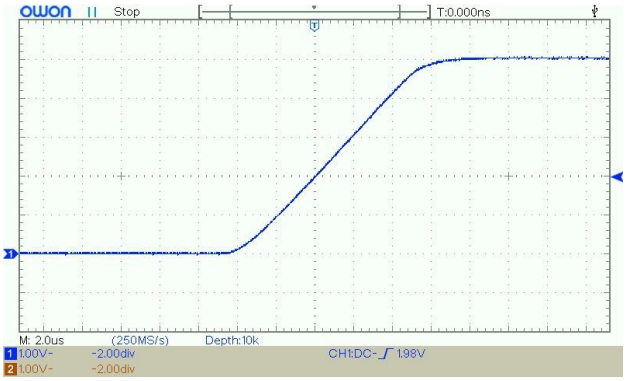

Рис. 4. *Осциллограмма переходного процесса*
Fig. 4. *Transition process oscillogram*

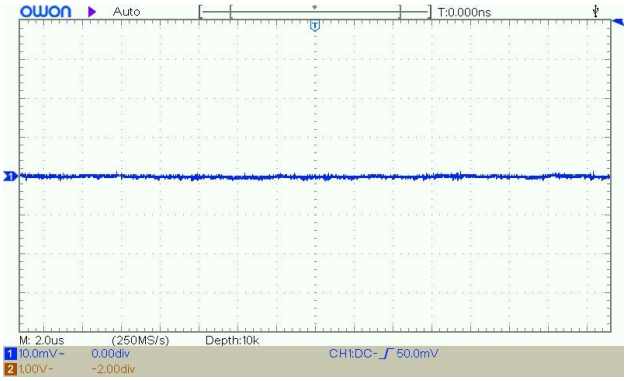

Рис. 5. *Осциллограмма выходного сигнала в режиме измерения АС*
Fig. 5. *Oscillogram of the output signal in AC measurement mode*

При рассмотрении выходного сигнала с помощью осциллографа можно заметить, что время установления сигнала занимает около 12 мкс, сигнал является достаточно стабильным, пульсации напряжения не превышают 2 мВ. Это явно относится к плюсам устройства, как и то, что при использовании шины *I2C* можно подключить несколько модулей *MCP4725* (важно при этом на модулях установить разные адреса перемычками). Минусом данного модуля является отсутствие гальванической развязки.

ШИМ-сигнал

Широтно-импульсная модуляция (ШИМ) импульсный прямоугольный сигнал, у которого фиксированные амплитуда и период, но изменяемая длительность (ширина) импульса.

ШИМ-сигнал можно описать, основываясь на двух параметрах сигнала:

- $t_и$ – время импульса (состояние логической единицы);
- $t_п$ – время паузы (состояние логического нуля);

Теперь введем еще несколько параметров:

Период импульсов T $[c]$ – время от начала одного импульса до начала соседнего

$$T = t_и + t_п.$$

Частота импульсов V $[Гц]$ – величина обратная периоду импульсов, показывает количество импульсов за секунду

$$V = 1/T.$$

Коэффициент заполнения D [%] – параметр, который является отношением времени импульса к периоду между импульсами

$$D = \frac{t_и}{T} \cdot 100\ \%\ .$$

Скважность S – параметр, обратный коэффициенту заполнения

$$S = 1/D.$$

Более наглядно параметры отражены на рис. 6.

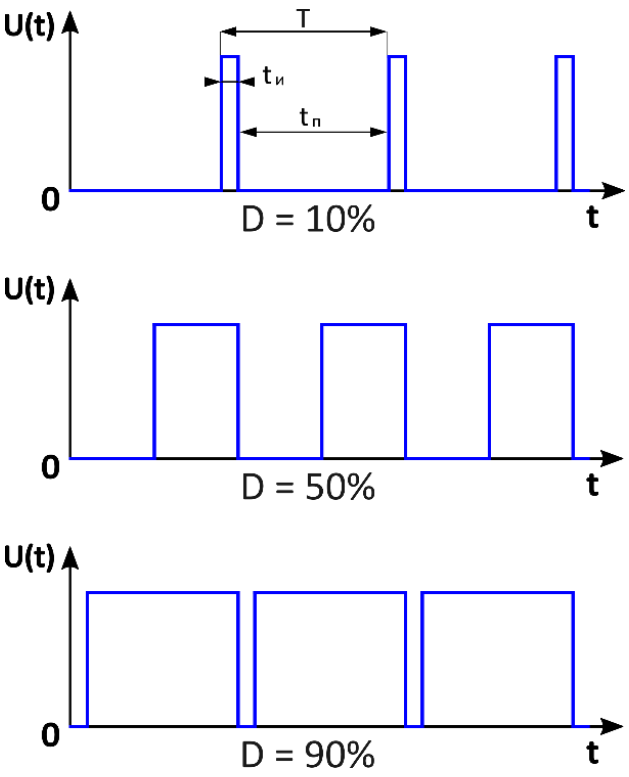

Рис. 6. *Параметры ШИМ-сигнала*
Fig. 6. *PWM-signal parameters*

Модуль преобразователя ШИМ в сигнал 0–10 В

Официальной документации на данный модуль (рис. 7) в интернете найти не удалось, поэтому ниже приведен перевод инструкции по эксплуатации модуля со страницы продавца [6].

«После включения питания при отсутствии входного сигнала напряжение на выходе равно 0 В. Выходной сигнал можно получить только при наличии входного сигнала. При первом включении питания рекомендуется произвести калибровку. Для этого необходимо подключить к входам *PWM* и *GND* входной сигнал с частотой 1–3 кГц и установить коэффициент заполнения равным 50 %. Далее необходимо измерить напряжение на выходах *OUT* и *GND* с помощью мультиметра. После этого с помощью потенциометра на регулировочной плате необходимо установить напряжение 5,00 В. После этого калибровка между вашим импульсным сигналом и этим модулем будет завершена. При изменении частоты потребуется повторная калибровка. Выходное напряжение можно изменять, регулируя рабочий цикл. Точность можно настраивать с помощью регулирующего потенциометра».

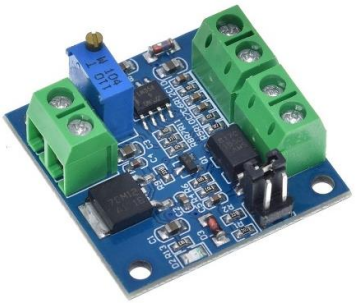

Рис. 7. *Преобразователь ШИМ в сигнал 0–10 В*
Fig. 7. *PWM to 0–10 V signal converter*

Таблица 4. *Назначение контактов модуля*
Table 4. *Module contact assignment*

Контакт модуля Module contact	Назначение Purpose
VCC	Напряжение питания 12–30 В Supply voltage 12–30 V
GND	Земля для напряжения питания Ground for supply voltage
PWM	Входной ШИМ-сигнал Input PWM signal
GND	Земля для входного ШИМ-сигнала Ground for PWM input signal
OUT	Выходное напряжение 0–10 В Output voltage 0–10 V

Рассмотрим данный модуль подробнее, для этого произведем реверс-инжиниринг модуля. Схема представлена на рис. 8.

По данной схеме можно заметить, что оптопара, установленная на модуле, не создает гальванической развязки, так как у светодиода и фототранзистора оптопары общая земля. Наличие хорошей гальванической развязки является очень важным критерием, поскольку гальваническая развязка должна разделять управляющую часть от выходной, чтобы при неполадках в выходной цепи не выходила из строя управляющая часть.

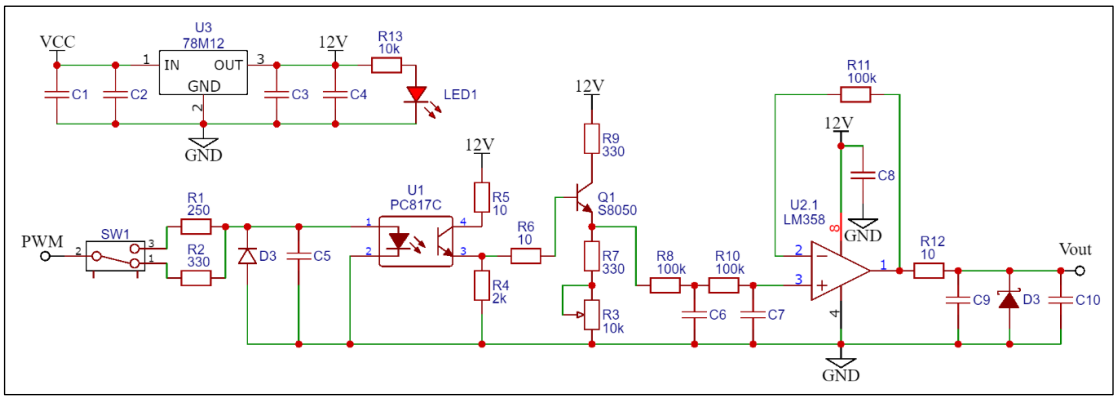

Рис. 8. *Схема модуля преобразования ШИМ в сигнал 0–10 В*
Fig. 8. *Schematic diagram of the PWM to 0–10 V signal conversion module*

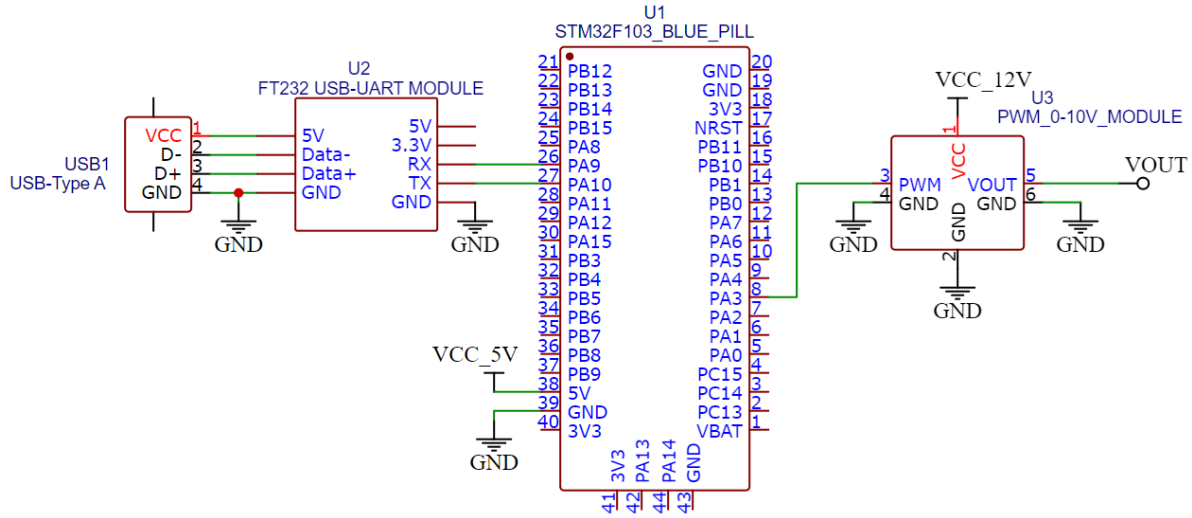

Рис. 9. *Подключение модуля преобразования ШИМ в сигнал 0–10 В*
Fig. 9. *Connecting the PWM to 0–10 V signal conversion module*

На входе питания стоит стабилизатор напряжения *7812*, а значит, диапазон напряжений питания модуля, указанный продавцом (12–30 В), является неверным. В документации на данный стабилизатор указано минимальное падение напряжения 2 В, следовательно, рабочее напряжение 12 В на выходе данного стабилизатора появятся только если напряжение на входе будет более 14 В. Также необходимо учесть, что данный операционный усилитель не является *rail-to-rail* усилителем, и значение напряжения на выходе может быть ниже напряжения питания на 1–4 вольта (чем больше напряжение питания, тем больше разница между напряжением питания и максимальным выходным напряжением), следовательно, при входном питании 12 В на выходе можно получить максимум 9 В вместо заявленных 10 В. Процесс преобразования ШИМ-сигнала с помощью *RC-цепи* подробно описан в [7]. Схема подключения данного модуля к отладочной плате «Blue Pill» приведена на рис. 9.

Программный код

За основу возьмем проект с подключением модуля *MCP4725*, программный код которого рассмотрен выше.

Настройку последовательного интерфейса *UART* в файле "*main_init.c*" оставим без изменений, настройку интерфейса *I2C* можно удалить или закомментировать.

В этом же файле настроим вывод *PA3* как альтернативный выход, для управления им с помощью таймера.

Блок кода 11. *Файл main_init.c*

```
//============== ПОРТЫ ================
// Включаем тактирование порта GPIOA
RCC->APB2ENR |= RCC_APB2ENR_IOPAEN;
// Настроим выход PA3 как
// альтернативный выход
GPIOA->CRL &= ~GPIO_CRL_CNF3;
GPIOA->CRL |= GPIO_CRL_CNF3_1;
GPIOA->CRL &= ~GPIO_CRL_MODE3;
GPIOA->CRL |= GPIO_CRL_MODE3_1;
```

После этого настроим таймер 2 на частоту 2 кГц, шкалу 0–10 В поделим на 1000 шагов. За основу данной реализации был взят код из статьи [8].

Блок кода 12. *Файл main_init.c*

```
// ============== TIM2 ================
// Разрешаем тактирование, PCLK1 = 36МГц
// TIM2 получается х 2 = 72 МГц
RCC->APB1ENR |= RCC_APB1ENR_TIM2EN;
// Выбираем внутреннее тактирование
TIM2->SMCR &= ~TIM_SMCR_SMS;
// Предделитель ставим равным 36
// следовательно частота 2 МГц
TIM2->PSC = 36 - 1;
// Переполнение через 1000 шагов,
// следовательно частота 2 кГц
TIM2->ARR = 1000 - 1;
// Настраиваем канал 4 как выход
// (PA3 - выход по умолчанию для четвертого
// канала таймера TIM2)
TIM2->CCER |= TIM_CCER_CC4E;
// Разрешаем использовать выводы таймера
// как выходы
TIM2->BDTR |= TIM_BDTR_MOE;
// При счёте вверх (+1), канал 4
// находится в состоянии логической «1»
// (пока счётчик таймера меньше значения
// регистра сравнения CCR4)
TIM2->CCMR2 = TIM_CCMR2_OC4M_2 |
TIM_CCMR2_OC4M_1;
// Коэффициент заполнения ШИМ
TIM2->CCR4 = 0;
// Таймер настроен на счёт вверх (+1)
TIM2->CR1 &= ~TIM_CR1_DIR;
// После переполнения счётчик начнёт
// счёт с нуля
TIM2->CR1 &= ~TIM_CR1_CMS;
// Разрешаем счёт
TIM2->CR1 = TIM_CR1_CEN;
```

Далее перейдем в файл "*main.c*". Подключаемые библиотеки, используемые глобальные переменные и обработку прерывания *UART* также оставим без изменений. Немного перепишем функцию *main*.

Блок кода 13. *Файл main.c*

```
// =============== MAIN ===============
int main(void) {
// Инициализация МК
 main_init();
// ========== ГЛАВНЫЙ ЦИКЛ =============
 while (1) {
// Если что-то принято по UART
 if (flag == 1) {
// Принятые данные расшифровываем
// как число 0-1000
 uint16_t dac = 0;
 dac = Rxdecoder(bf, &Buffin[0]);
// Делаем защиту от переполнения
 if (dac > 1000) dac = 1000;
// Очищаем входной буфер и его индекс
 for (uint8_t i = 0; i < bf; i++) {
 Buffin[0] = 0;
 }
 bf = 0;
 flag = 0;
// Выведем по UART принятое число
 Uint16ToStr(dac, &Buffout[0]);
 Tx1Str(&Buffout[0]);
 Tx1Str("\r\n");
// Обновляем коэффициент заполнения ШИМ
 TIM2->CCR4 = dac;
 }
 delay_ms(10);
 }
}
```

Ниже представлена таблица (табл. 5) соответствия входных значений напряжениям на выходе модуля. Модуль был откалиброван для частоты 2 кГц при скважности ШИМ 50 %. Из таблицы можно заметить, что показания напряжения на выходе модуля при значениях скважности ШИМ ниже или выше 50 % имеют отклонения.

Таблица 5. *Напряжения на выходе модуля*

Table 5. *Voltages at the module output*

Значение/Meaning	%	Ucp, B
0	0	0,01
10	1	0,02
50	5	0,43
100	10	0,93
300	30	2,98
500	50	5,00
700	70	7,02
900	90	9,05
950	95	9,58
990	99	10,07
1000	100	10,13

На рис. 10 приведена осциллограмма сигнала на выходе устройства при подаче на вход ступенчатого сигнала. На рис. 11 приведены пульсации напряжения на выходе модуля при подаче на вход ШИМ-сигнала с коэффициентом заполнения 50 %. В обоих случаях первый щуп осциллографа (синий на графике) подключён к ШИМ-входу модуля, а второй щуп (оранжевый на графике) – к выходу.

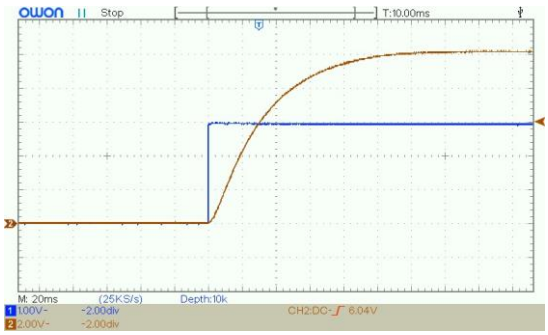

Рис. 10. *Осциллограмма переходного процесса при подаче ступенчатого сигнала*

Fig. 10. *Transient process oscillogram when a stepwise signal is applied*

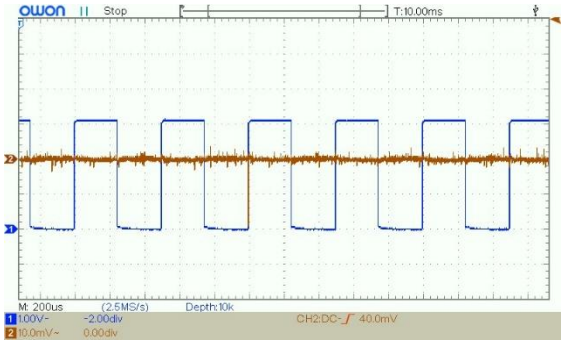

Рис. 11. *Входной ШИМ-сигнал (режим DC) и выходной аналоговый сигнал (режим AC)*
Fig. 11. *Input PWM signal (DC-mode) and output analog signal (AC-mode)*

При рассмотрении выходного сигнала с помощью осциллографа можно заметить, что время установления сигнала занимает около 100 мс, что намного больше, чем у модуля *MCP4725*.

Достоинством модуля преобразователя на основе ШИМ является простота управления.

К недостаткам данного модуля можно отнести отсутствие гальванической развязки и отсутствие расширенной документации.

Выводы

- Рассмотренные в данной статье модули являются недорогими и легко настраиваемыми устройствами.
- Модуль *MCP4725* имеет 12-битный выходной сигнал, но предъявляет повышенные требования к напряжению питания, так как использует его в качестве опорного напряжения.
- Модуль преобразователя ШИМ на основе интегрирующей *RC-цепи* в аналоговый сигнал 0–10 *В* имеет большие отклонения выходного напряжения и более зашумленный сигнал, чем модуль *MCP4725*, при этом он имеет более простое управление, для которого требуется только ШИМ-сигнал.
- У обоих модулей отсутствует гальваническая развязка, что делает их небезопасными для применения в промышленных устройствах.

СПИСОК ЛИТЕРАТУРЫ

1. Microchip. MCP472512-Bit Digital-to-Analog Converter with EEPROM Memory in SOT-23-6. URL: https://cdn-shop.adafruit.com/datasheets/mcp4725.pdf (дата обращения: 05.10.2024).
2. MCP4725 – Цифро-аналоговый преобразователь (ЦАП) // MicroPi. URL: https://micro-pi.ru/mcp4725-cifro-analogovyj-preobrazovatel/ (дата обращения: 05.10.2024).
3. Reference manual STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced Arm®-based 32-bit MCUs. URL: https://www.st.com/content/ccc/resource/technical/document/reference_manual/59/b9/ba/7f/11/af/43/d5/CD00171190.pdf/files/CD00171190.pdf/jcr:content/translations/en.CD00171190.pdf (дата обращения: 05.10.2024).
4. Blue Pill. URL: https://www.belchip.by/sitedocs/31025.pdf (дата обращения: 05.10.2024).
5. Волобуев К.А., Трубин М.В., Трубин В.Г. Управление WI-FI модулем ESP-01 с компьютера при помощи АТ-команд // Автоматика и программная инженерия. – 2023. – № 2 (44). URL: http://jurnal.nips.ru/sites/default/files/AaSI-2-2023-1_0.pdf (дата обращения: 05.10.2024).
6. Модуль преобразователь ШИМ в сигнал 0–10 В. URL: https://aliexpress.ru/item/4001225261608.html?sku_id=12000022190864449&spm=a2g2w.productlist.search_results.0.54434aa6WEAdlE (дата обращения: 05.10.2024).
7. Khaled Magdy. Convert PWM to a DAC – using PWM to generate analog waveforms. URL: https://deepbluembedded.com/convert-pwm-to-a-dac-using-pwm-to-generate-analog-waveforms/ (дата обращения: 05.10.2024).
8. Близнюк А.Е., Трубин В.Г., Саблина Г.В. Управление яркостью лампы накаливания указателя поворота автомобиля с помощью аппаратного ШИМ микроконтроллера STM32F103 // Автоматика и программная инженерия. – 2023. – № 1 (43). URL: http://jurnal.nips.ru/sites/default/files/AaSI-1-2023-4.pdf (дата обращения: 05.10.2024).

Информация об авторах

Кирилл Андреевич Волобуев, студент, кафедра автоматики, Новосибирский государственный технический университет, Россия, 630073, г. Новосибирск, пр. Карла Маркса, 20; kirya.volobuev@mail.ru
Игорь Витальевич Трубин, старший преподаватель, кафедра защиты информации, Новосибирский государственный технический университет, Россия, 630073, г. Новосибирск, пр. Карла Маркса, 20; i.trubin@corp.nstu.ru; https://orcid.org/0009-0001-1466-6001
Виталий Геннадьевич Трубин, старший преподаватель, кафедра автоматики, Новосибирский государственный технический университет, Россия, 630073, г. Новосибирск, пр. Карла Маркса, 20; trubin@corp.nstu.ru

Поступила: 10.10.2024
Принята: 10.12.2024
Опубликована: 28.12.2024

REFERENCES

1. *Microchip*. *MCP472512-Bit Digital-to-Analog Converter with EEPROM Memory in SOT-23-6*. Available at: https://cdn-shop.adafruit.com/datasheets/mcp4725.pdf (accessed: 5 October 2024).
2. MCP4725 – Digital to Analog Converter (DAC). *MicroPi*. (In Russ.) Available at: https://micro-pi.ru/mcp4725-цифро-аналоговый-преобразователь/ (accessed: 5 October 2024).
3. *Reference manual STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced Arm®-based 32-bit MCUs*. Available at: https://www.st.com/content/ccc/resource/technical/document/reference_manual/59/b9/ba/7f/11/af/43/d5/CD00171190.pdf/files/CD00171190.pdf/jcr:content/translations/en.CD00171190.pdf (accessed: 5 October 2024).
4. *Blue Pill*. Available at: https://www.belchip.by/sitedocs/31025.pdf (accessed: 5 October 2024).
5. Volobuev K.A., Trubin M.V., Trubin V.G. Controlling the WI-FI ESP-01 module from a computer using AT commands. *Automatics & Software Enginery*, 2023, no. 2 (44). (In Russ.) Available at: http://jurnal.nips.ru/sites/default/files/AaSI-2-2023-1_0.pdf (accessed: 5 October 2024).
6. *PWM to 0-10 V signal converter module*. (In Russ.) Available at: https://aliexpress.ru/item/4001225261608.html?sku_id=12000022190864449&spm=a2g2w.productlist.search_results.0.54434aa6WEAdlE (accessed: 5 October 2024).
7. Khaled Magdy. *Convert PWM to a DAC – using PWM to generate analog waveforms*. Available at: https://deepbluembedded.com/convert-pwm-to-a-dac-using-pwm-to-generate-analog-waveforms/ (accessed: 5 October 2024).
8. Bliznyuk A.E., Trubin V.G., Sablina G.V. *Controlling the brightness of the incandescent lamp of the turn signal of the car using a hardware PWM STM32F103 microcontroller*. (In Russ.) Available at: http://jurnal.nips.ru/sites/default/files/AaSI-1-2023-4.pdf (accessed: 5 October 2024).

Information about the authors

Kirill A. Volobuev, Student, Novosibirsk State Technical University, 20, Karl Marks avenue, Novosibirsk, 630073, Russian Federation; kirya.volobuev@mail.ru

Igor V. Trubin, Senior Lecturer, Novosibirsk State Technical University, 20, Karl Marks avenue, Novosibirsk, 630073, Russian Federation; i.trubin@corp.nstu.ru; https://orcid.org/0009-0001-1466-6001

Vitaly G. Trubin, Senior Lecturer, Novosibirsk State Technical University, 20, Karl Marks avenue, Novosibirsk, 630073, Russian Federation; trubin@corp.nstu.ru

Received: 10.10.2024
Revised: 10.12.2024
Accepted: 28.12.2024