DOI 10.37882/2223-2966.2024.04.39

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ ИССЛЕДОВАНИЯ РАБОТЫ ТАЙМЕРОВ МИКРОКОНТРОЛЛЕРА

DEVELOPMENT OF A PROGRAM TO STUDY THE OPERATION OF MICROCONTROLLER TIMERS

S. Shchegolev
T. Efremova
A. Motkov

Summary. The article discusses the structure and modes of operation of device timers using the example of the ADuC842 microcontroller. The timers have two 8-bit registers, which can be used as independent or combined into 16-bit registers. The timers can operate in various modes, including zero mode, first mode, and reset mode. Block diagrams of timers are presented for each mode. The description of the mode and control registers serving the timers is given. The built-in microcontroller generator is designed to work with a 32.768 kHz quartz resonator. Various timer operation modes are used to generate time intervals. To form a sequence of identical time intervals, the timer operation mode is used with a reboot. A block diagram of the laboratory stand operation and the program code for controlling the on and off of LEDs with a specified frequency are shown.

Keywords: timer, program, operating mode, register, time interval, frequency generator, pulse, block diagram.

Щеголев Сергей Сергеевич
Доцент, кандидат технических наук, Балаковский инженерно-технологический институт — филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ», Балаково, Россия
SSShchegolev@mephi.ru
Ефремова Татьяна Александровна
Доцент, кандидат технических наук, Балаковский инженерно-технологический институт — филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ», Балаково, Россия
TAEfremova@mephi.ru
Мотков Александр Геннадьевич
Старший преподаватель, Балаковский инженерно-технологический институт — филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ», Балаково, Россия
AGMotkov@mephi.ru

Аннотация. В статье рассмотрена структура и режимы работы таймеров устройств на примере микроконтроллера ADuC842. Таймеры имеют два 8-битных регистра, которые могут быть использованы как независимые или скомбинированы в 16-битные регистры. Таймеры могут работать в различных режимах, включая нулевой режим, первый режим и режим с перезагрузкой. Для каждого режима представлены структурные схемы таймеров. Приведено описание регистров режима и управления, обслуживающих таймеры. Встроенный генератор микроконтроллера предназначен для работы с кварцевым резонатором 32.768 кГц. Для формирования временных интервалов используются различные режимы работы таймера. Для формирования последовательности одинаковых интервалов времени используется режим работы таймера с перезагрузкой. Изображена блок-схема работы лабораторного стенда и код программы для управления включением и выключением светодиодов с заданной периодичностью.

Ключевые слова: таймер, программа, режим работы, регистр, временной интервал, генератор частоты, импульс, структурная схема.

Таймеры предназначены для формирования временных интервалов, позволяя микропроцессорной системе работать в режиме реального времени. Таймеры представляют собой цифровые счётчики, которые подсчитывают импульсы либо от высокостабильного генератора частоты, либо от внешнего источника сигнала, в этом случае таймер называют счетчиком внешних событий.

Как правило, в микропроцессорной системе в качестве генератора частоты выступает генератор внутренней синхронизации микроконтроллера. Частота генератора задает минимальный временной промежуток, который может определять таймер. Интервалы времени, задаваемые с помощью таймера, могут иметь строго определенные дискретные значения. Разрядность цифрового счётчика таймера определяет

максимальный интервал времени, который может задать таймер.

Обычно в микропроцессорных системах используются 16-тиразрядные таймеры, для подключения такого таймера к 8-миразрядному процессору требуется два параллельных порта. Максимальное число, которое может быть записано в 16-битный счетный регистр таймера, равно $2^{16} - 1 = 65535$, что представляет собой логическую единицу в каждом разряде регистра. Таким образом, если перед запуском таймера в его счетчики были записаны нули, то переполнение таймера произойдет через 65536 машинных циклов [1]. Зная частоту задающего генератора микропроцессорной системы, следовательно период сигнала генератора T_G, можно определить время переполнения таймера в секундах:

$$T_T = 65536 \cdot T_G. \quad (1)$$

Если же требуется установить меньший интервал времени, то перед запуском таймера в его регистры можно записать начальный код, и тогда счет начнется не с нуля, а с записанного кода, и счетчику потребуется меньше времени для переполнения. В этом случае время работы таймера T_T определяется по формуле (2):

$$T_T = \left(2^n - Code\right) \cdot T_G, \quad (2)$$

где *Code* — код, записанный в таймер до его запуска; T_T — Время работы таймера; T_G — период колебаний задающего генератора; *n* — разрядность таймера.

Если же требуется сформировать интервал времени больший, чем максимальное время переполнения, то таймер можно запустить несколько раз в цикле [2]. В этом случае временной интервал определяется как:

$$\Delta T = T_T \cdot N, \quad (3)$$

где *N* — количество итераций цикла; T_T — время срабатывания таймера.

Рассмотрен учебный лабораторный стенд (рисунок 1), построенный на базе микроконтроллера ADuC842, имеющего три 16-разрядных таймера-счетчика: Таймер 0, Таймер 1 и Таймер 2. Структура и режимы работы таймеров-счетчиков соответствуют общим принципам архитектуры MCS-51. Каждый таймер-счетчик содержит по два 8-битных регистра THx и TLx (x = 0, 1, и 2) [3].

Каждый таймер-счетчик может быть запрограммирован на работу в качестве либо таймера (отсчет времени через подсчет внутренних импульсов синхронизации), либо счетчика (подсчет событий на внешнем входе). В обоих случаях переполнение счетного регистра приводит к формированию запроса прерывания и устанавливается специальный флаг переполнения.

Рис. 1. Внешний вид учебного лабораторного стенда

В режиме таймера регистр TLx увеличивает свое значение на единицу каждый машинный цикл. Поскольку машинный цикл однотактового ядра состоит из одного тактового периода, то максимальная скорость счета равна тактовой частоте ядра.

В режиме счетчика, регистр TLx увеличивает свое значение на единицу при переходе уровня из высокого в низкий на соответствующем внешнем выводе микроконтроллера: T0, T1 или T2. Когда на внешнем выводе один машинный цикл держится высокий логический уровень, а уже в следующем цикле — низкий, тогда регистр таймера увеличивает свое значение на единицу. Таким образом, для распознавания перехода из «1» в «0» требуется два такта внутреннего генератора микроконтроллера, это значит, что максимальная скорость счета может составить половину частоты внутреннего тактового генератора.

Таймеры 0 и 1 обслуживаются регистром режима TMOD (таблица 1) и регистром управления TCON (Таблица 2).

TMOD — регистр конфигурации Таймера 1 и Таймера 0. SFR адрес 0x89. Значение после подачи питания 0x00. Регистр не имеет битовой адресации.

TCON — регистр управления Таймера 1 и Таймера 0. SFR адрес 0x88. Значение после подачи питания 0x00. Регистр имеет битовую адресацию [4].

Таблица 1.

Описание бит регистра TMOD

Номер	Мнемоника	Описание
1	2	3
7	GATE	Бит управления таймером 1.
6	C/T#	Бит выбора типа событий для Таймера 1.
5	M1	MI, M0 биты определяют режим работы таймера 1
		MI M0 0 0 TH1 работает как 8-битный таймер-счетчик, TL1 выступает в качестве делителя частоты на 32 0 1 16-битный таймер-счетчик, TH1и TL1 включены последовательно.
4	M0	_1 0 8-битный таймер-счетчик с автоперезагрузкой, TH1 удерживает значение, которое загружается в TL1 всякий раз при переполнении TL1. 1 1 Таймер-счетчик 1 остановлен.
3	GATE	Бит управления таймером 0.
2	C/T#	Бит выбора типа событий для Таймера 0.
1	MI	MI, M0 биты определяют режим работы таймера 0
		MI M0 0 0 TH0 работает как 8-битный таймер-счетчик, TL0 выступает в качестве делителя частоты на 32 0 1 16-битный таймер-счетчик, TH0 и TL0 включены последовательно. 1 0 8-битный таймер-счетчик с автоперезагрузкой, TH1
0	M0	удерживает значение, которое загружается в TL0 всякий раз при переполнении TL0. 1 1 TL0 используется в качестве 8-битного таймера-счетчика со стандартными битами управления Таймера 0. TH0 используется только в качестве 8-битного счетчика, управление происходит стандартными битами управления таймера 1.

Таблица 2.

Описание бит регистра TCON

Номер	Мнемоника	Описание
1	2	3
7	TF1	Флаг переполнения Таймера 1
6	TR1	Бит запуска Таймера 1
5	TF0	Флаг переполнения Таймера 0
4	TR0	Бит запуска Таймера 0
3	IE1	Флаг внешнего прерывания 1 (INT1#)
2	IT1	Бит выбора типа активного сигнала на входе INT1 #
1	IE0	Флаг внешнего прерывания 1 (INT0 #)
0	IT0	Бит выбора типа активного сигнала на входе INT1#

Каждый таймер содержит два 8-битных регистра.

В зависимости от режима работы таймера, каждый регистр может быть использован как независимый регистр или регистры могут быть скомбинированы в одиночные 16-битные регистры.

TH0 и TL0 — старший и младший байт Таймера 0. SFR адрес — 0x8C и 0x8A, соответственно. TH1 и TL1 — старший и младший байт Таймера 1. SFR адрес — 0x8D и 0x8B, соответственно.

На рисунке 2 представлены адреса регистров SFR таймеров.

Для выбора нулевого режима следует установить биты M1 = 0 и M0 = 0 регистра TMOD. В этом режиме таймер-счетчик сконфигурирован как 13 битный счетный регистр. Этот счётчик состоит из 8 бит регистра THx и младших 5 бит регистра TLx, где x в обозначении регистра заменяется на 0 или 1 в зависимости от того таймера, которым мы управляем. Старшие 3 бита регистров TLx не определены и игнорируются. Установка запускающего таймер флага TR0 или TR1 не очищает эти регистры. Работе таймера 0 или таймера 1 в режиме 0 соответствует схема, представленная на рисунке 3.

Для выбора первого режима следует установить биты M1 = 0 и M0 = 1 регистра TMOD. В первом режиме работы таймер работает как шестнадцатиразрядный счётчик. Режим 1 похож на режим 0, за исключением того, что в регистрах таймера используются все 16 бит.

В этом режиме регистры THx и TLx также включены друг за другом. Работе таймера 0 или таймера 1 в режиме 1 соответствует схема, изображенная на рисунке 4.

Для формирования последовательности одинаковых интервалов времени используется режим работы таймера с перезагрузкой — режим 2. Для выбора второго режима следует установить биты M1 = 1 и M0 = 0 регистра TMOD. В режиме 2 регистр таймера TLx работает как 8-битный счетчик с автоматической перезагрузкой начального значения из регистра THx в регистр TLx. Переполнение регистра TLx не только устанавливает флаг TFx, но и загружает регистр TLx содержимым регистра THx, который предварительно инициализируется программно. Перезагрузка не изменяет содержимое регистра THx. Работе таймера 0 или таймера 1 в режиме 2 соответствует схема (рисунок 5) [5]:

Встроенный генератор микроконтроллера ADuC842 предназначен для работы с кварцевым резонатором 32.768 кГц. Для формирования тактов синхронизации процессора используется умножитель частоты: система фазовой автоподстройки частоты (ФАПЧ) и управляемый делитель. С помощью системы ФАПЧ частота такто-

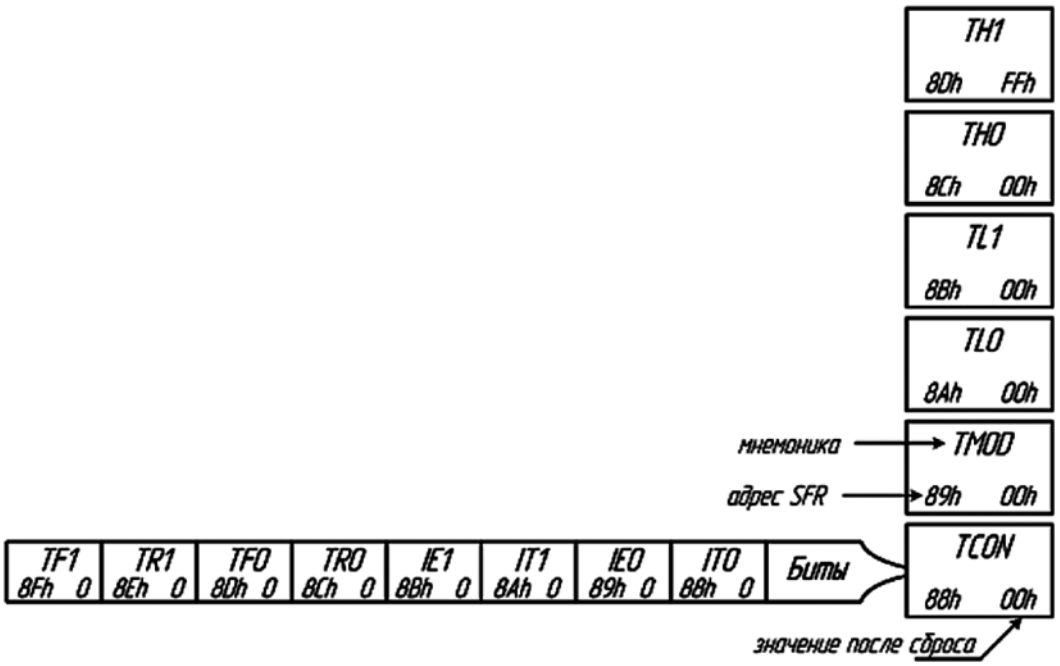

Рис. 2. Адреса 2регистров SFR таймеров

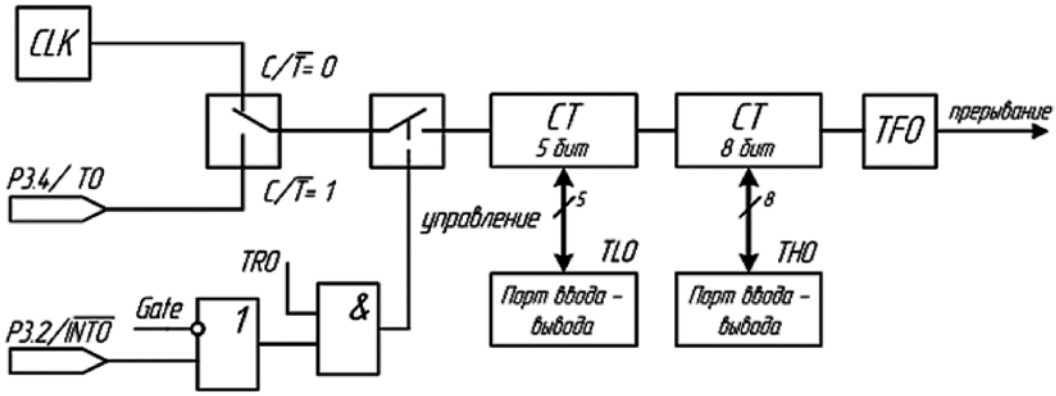

Рис. 3. Структурная схема таймера в режиме 0 (для Таймера 0)

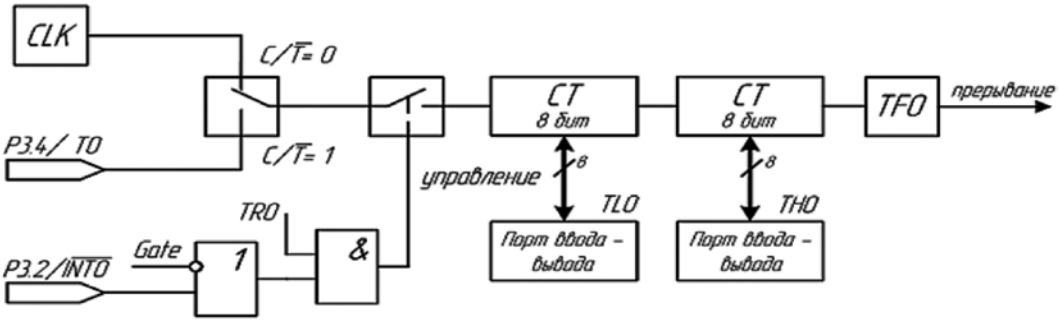

Рис. 4. Структурная схема таймера в режиме 1 (для Таймера 0)

вого генератора умножается на 512, что соответствует 16,777216 МГц. Делитель же настраивается на фиксированные коэффициенты деления (1, 2, 4, 8, 16, 32, 64, 128). При включении микроконтроллера по умолчанию установлен делитель 8, что соответствует частоте 16,777216/8 = 2,097152(МГц). Изменить этот делитель можно через регистр PLLCON.

Допустим, тактовый генератор сконфигурирован для генерации с частотой 1МГц, а нам требуется сформировать временные интервалы 3500 мкс. В этом случае на вход таймера будут поступать импульсы с периодом 1мкс. Для 16-битного счетчика максимальное время переполнения составит 65536 мкс. Для формирования меньшего интервала времени в счетчики таймера следует загрузить начальное значение Code. Из формулы 3.1 выразим Code:

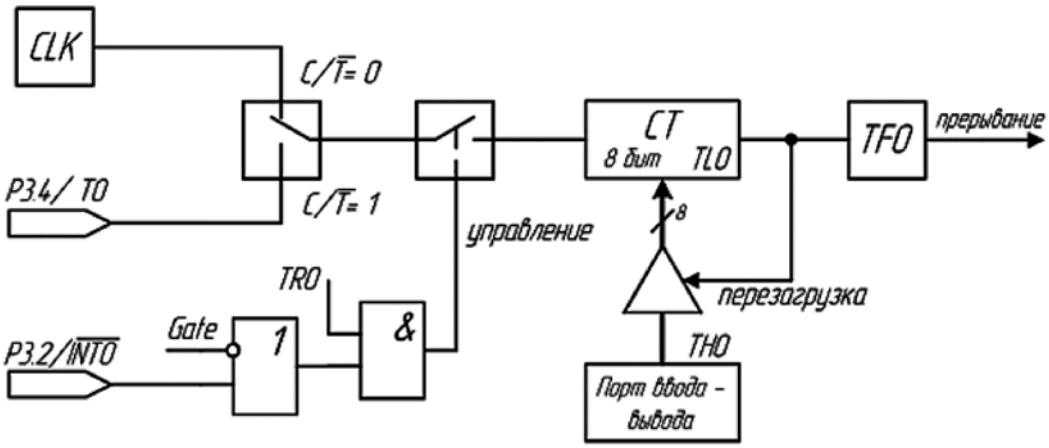

Рис. 5. Структурная схема таймера в режиме 2 (для Таймера 0)

$$Code = \frac{2^n - T_T}{T_G} = 65536 - \frac{3500}{1} = 62036$$

Счетчик таймера состоит из двух отдельных 8-битных счетчиков THx и TLx, поэтому полученное значение Code следует расщепить на два отдельных байта. Сделать это можно разными способами. Можно с помощью калькулятора перевести число Code из десятеричной системы счисления в шестнадцатеричную: 6203610 = F2 5416. Тогда старший байт THx будет равен F216, а младший TLx равен 5416 [6].

При расщеплении константы можно воспользоваться делением на 256, тогда:

```
THx = 62036/256; // заносим старший байт числа 62036
TLx = 62036; // заносим младший байт числа 62036
```

Операция деления числа на 256 эквивалентна сдвигу этого числа на 8 разрядов вправо, тогда приведенный участок программы можно записать так:

```
THx = 62036 >> 8; // заносим старший байт числа 62036
TLx = 62036; // заносим младший байт числа 62036
```

После того как регистром TMOD установлен режим работы и загружены начальные значения счетчиков, можно запускать таймер. Для этого в бит запуска TRx следует записать логическую единицу. Теперь в тело основного цикла нужно включить участок программы, который будет ожидать окончания работы таймера и только после этого приступать к выполнению следующего прохода по циклу. Это можно сделать с помощью команды, которая будет проверять флаг переполнения таймера TFx. Затем необходимо снова задать следующий интервал времени. Следует помнить, что перед запуском таймера, следует обнулить флаг переполнения. Ниже приведен один из вариантов программы, реализующей задержку на 3,5 мс.

```
TH0 = 62036 >> 8 ;// заносим старший байт числа 62036
TL0 = 62036;     // заносим младший байт числа 62036
TR0 = 1;         // запускаем таймер
while (!TF0);    // ждем переполнения таймера
TF0 = 0;         // обнуляем флаг переполнения
```

Алгоритм программы представлен на рисунке 6.

Для формирования временных интервалов большей длительности данный участок программы можно запустить в цикле необходимое количество раз.

Код программы на языке C-51 [7]:

```
#include<stdio.h>
/* BYTE Register */
sfr TCON = 0x88;sfr TMOD = 0x89;sfr TL0 = 0x8A;
sfr TL1 = 0x8B;sfr TH0 = 0x8C;sfr TH1 = 0x8D;
sfr PLLCON = 0xD7;
/* BIT Register..... */
/* TCON */
sbit TF1 = 0x8F;sbit TR1 = 0x8E;sbit TF0 = 0x8D;
sbit TR0 = 0x8C;sbit IE1 = 0x8B;sbit IT1 = 0x8A;
sbit IE0 = 0x89;sbit IT0 = 0x88;
 /* PORTS BY BIT */
sbit P0_0 = 0x80;sbit P0_1 = 0x81;sbit P0_2 = 0x82;
sbit P0_3 = 0x83;
voidzadergka () {
unsignedint H, L, rep=133;H = 2874>>8;L = 2874;TR0 = 0;
TMOD = (TMOD & 0xF0) +
while(rep--) {
                TH0 = H;TL0 = L;TR0=1;
while (!TF0);
                TF0=0;
      }
return;
}
vkl_sv0(){P0_0=1;P0_1=0;P0_2=0;P0_3=0;}
vkl_sv1(){P0_0=0;P0_1=1;P0_2=0;P0_3=0;}
vkl_sv2(){P0_0=0;P0_1=0;P0_2=1;P0_3=0;}
vkl_sv3(){P0_0=0;P0_1=0;P0_2=0;P0_3=1;}
main(){
```

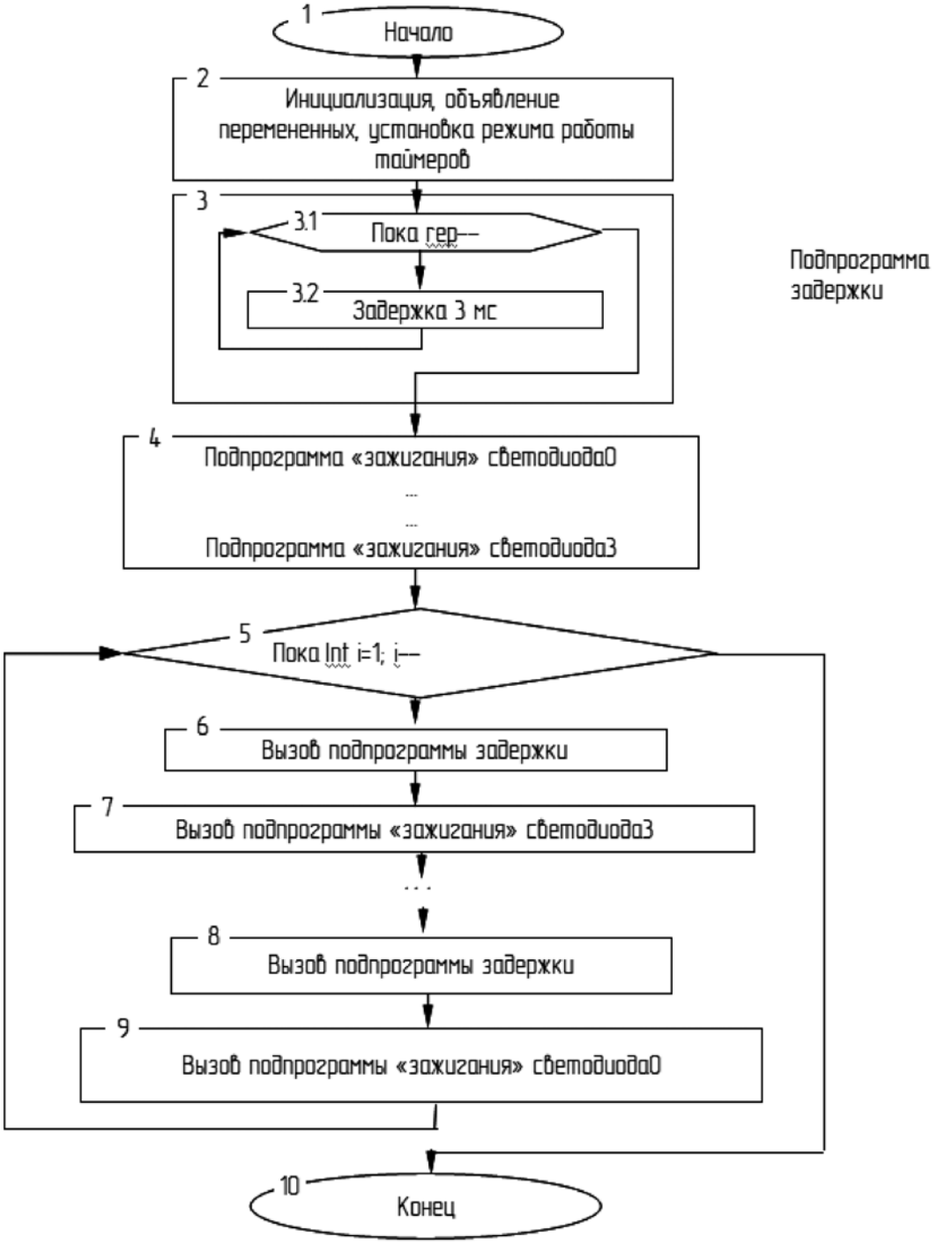

Рис. 6. Алгоритм программы

```
unsignedint i=1;
while(i--) {zadergka();vkl_sv3();}
i=1;
while(i--) {zadergka();vkl_sv2();}
i=1;
while(i--) {zadergka(); vkl_sv1();}
i=1;
while(i--) {zadergka();vkl_sv0();}
}
```

В результате выполнения данной программы на рассматриваемом стенде с периодичностью в 4 секунды, по очереди, слева направо, по циклу, будут загораться и гаснуть светодиоды.

ЛИТЕРАТУРА

1. Мендыбаев, С.А. Промышленная электроника: учебное пособие / С.А. Мендыбаев, С.С. Исенов, Г.О. Сулейменова. — Астана: КазАТУ, 2019. — 109 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/233972 (дата обращения: 15.03.2024). — Режим доступа: для авториз. Пользователей.
2. Богаченков, А.Н. Цифровые устройства и микропроцессоры: методические указания / А.Н. Богаченков. — Москва: РТУ МИРЭА, 2022. — 77 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/240125 (дата обращения: 11.03.2024). — Режим доступа: для авториз. пользователей.
3. Миронов, Б.М. Микроконтроллеры серии 8051: практикум: учебное пособие / Б.М. Миронов. — Иркутск: ИрГУПС, 2018. — 77 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/117563 (дата обращения: 15.03.2024). — Режим доступа: для авториз. пользователей.
4. Кормилин, В.А. Вычислительная техника: учебное пособие / В.А. Кормилин. — Москва: ТУСУР, 2019. — 140 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/313487 (дата обращения: 15.03.2024). — Режим доступа: для авториз. пользователей.
5. Водовозов, А.М. Микроконтроллеры для систем автоматики: учебное пособие / А.М. Водовозов. — 2-е изд. — Вологда: ВоГУ, 2015. — 164 с. — ISBN 978-5-87851-599-3. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/93084 (дата обращения: 15.03.2024). — Режим доступа: для авториз. пользователей.
6. Исаев, А.В. Программируемые цифровые устройства: микроконтроллеры: учебное пособие / А.В. Исаев, П.Г. Кривицкий, К.В. Пантелеев. — Минск: БНТУ, 2020. — 95 с. — ISBN 978-985-583-071-0. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/248408 (дата обращения: 15.03.2024). — Режим доступа: для авториз. пользователей.
7. Микушин, А.В. Программирование микропроцессорных систем на языке С-51 / А.В. Микушин. — Санкт-Петербург: Лань, 2023. — 124 с. — ISBN 978-5-507-45539-3. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/311828 (дата обращения: 15.03.2024). — Режим доступа: для авториз. пользователей.