1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт компьютерных наук и кибербезопасности
5. **Кафедра «Высшая школа кибербезопасности»**

**ЛАБОРАТОРНАЯ РАБОТА № 3**

1. «РАБОТА С ТАЙМЕРАМИ-СЧЁТЧИКАМИ»
2. по дисциплине «Аппаратные средства вычислительной техники»
3. Вариант 9
4. Выполнил
5. студент гр. 5131001/30003 Шевчук Н.Е.

<*подпись*>

Преподаватель Макаров А.С.

<*подпись*>

1. Санкт-Петербург
2. 2025

**Оглавление**

[1 Цель работы 3](#_Toc193926696)

[2 Ход работы 3](#_Toc193926697)

[2.1 Схема установки 3](#_Toc193926698)

[2.2 Реализованный алгоритм 3](#_Toc193926699)

[3 Ответы на контрольные вопросы 5](#_Toc193926700)

[4 Вывод 10](#_Toc193926701)

[Приложение 11](#_Toc193926702)

# Цель работы

Получить практические навыки по работе с таймерами-счётчиками и применению механизма прерываний.

# Ход работы

## Схема установки

На рисунке й представлена схема установки, используемая при реализации задания.

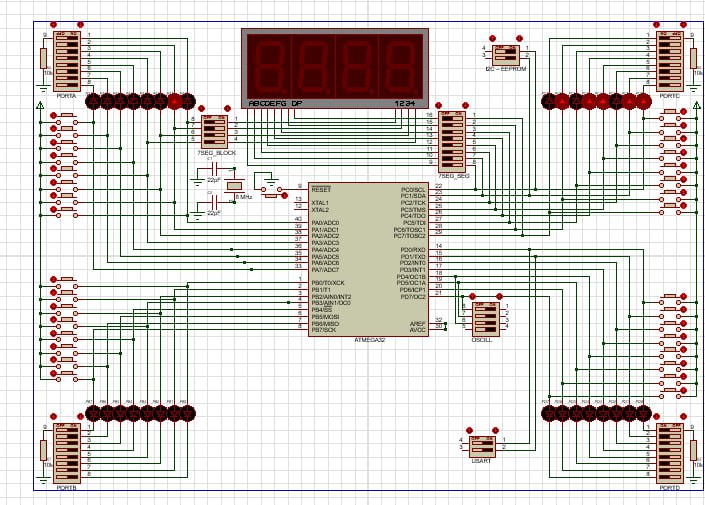


Рисунок 1. Схема установки

## Реализованный алгоритм

Была реализована программа на ассемблере (см. приложение), выполняющая следующее задание:

Модуль проверки ПИН-кода. Программа должна предоставлять возможность по проверке введённого ПИН-кода на соответствие заданному. ПИН-код должен храниться в EEPROM и считываться при запуске МК (запись ПИН-кода в EEPROM следует выполнять с помощью AVRFlash). Для отображения проверяемого ПИН-кода используется блок из четырёх семисегментных индикаторов. Настраиваемый разряд мигает с частотой 4 ГЦ. Ввод четырёхзначного ПИН-кода осуществляется поразрядно, от младшего к старшему. Для изменения значения вводимой цифры ПИН-кода используются кнопки PD0 и PD1, позволяющие соответственно увеличить или уменьшить вводимое значение, изменение циклическое, то есть при увеличении цифра 9 переходит в цифру 0 и наоборот, изменение значений происходит следующим образом: − в момент нажатия кнопки значение сразу изменяется на единицу; − если кнопка зажата дольше 2-х секунд, то, начиная со 2-й секунды, значение начинает изменяться на 1 каждые 0,25 с. При срабатывании прерывания INT0 (кнопка PD2) во время настройки 2-4 разряда ПИН-кода программа перейдёт к настройке следующего разряда. При срабатывании прерывания INT1 (кнопка PD3) во время ввода 1-3 разряда ПИН-кода программа перейдёт к вводу предыдущего разряда. Если выполнялась настройка 4 разряда – ввод ПИН-кода завершится, а программа сверит введённое значение ПИН-кода со значением в EEPROM. При вводе неправильного ПИН-кода загорается светодиод PA6 на 20 секунд, после чего программа возвращается к началу ввода ПИН-кода для проверки. При вводе неправильного ПИН-кода три раза подряд загораются светодиод PA7 и PA6, а программа перестаёт работать. При вводе правильного ПИН-кода загорается светодиод PA7, программа завершает работу (переходит в бесконечный цикл, отключаются прерывания INT0 и INT1. В начале работы программы на семисегментном индикаторе отображено значение «0000».

Для четкого измерения временных промежутков использован 16-битный Таймер1 в режиме СТС. Прерывания по совпадению срабатывают каждые 5 мс (частота 200 Гц). Предделитель равен 256, значение OCR1A выбрано согласно следующим расчётам:

С помощью этого таймера осуществляется мультиплексирование (визуально одновременный вывод всех разрядов индикатора), мигание текущего разряда и 20-секундная задержка при неправильном вводе пароля.

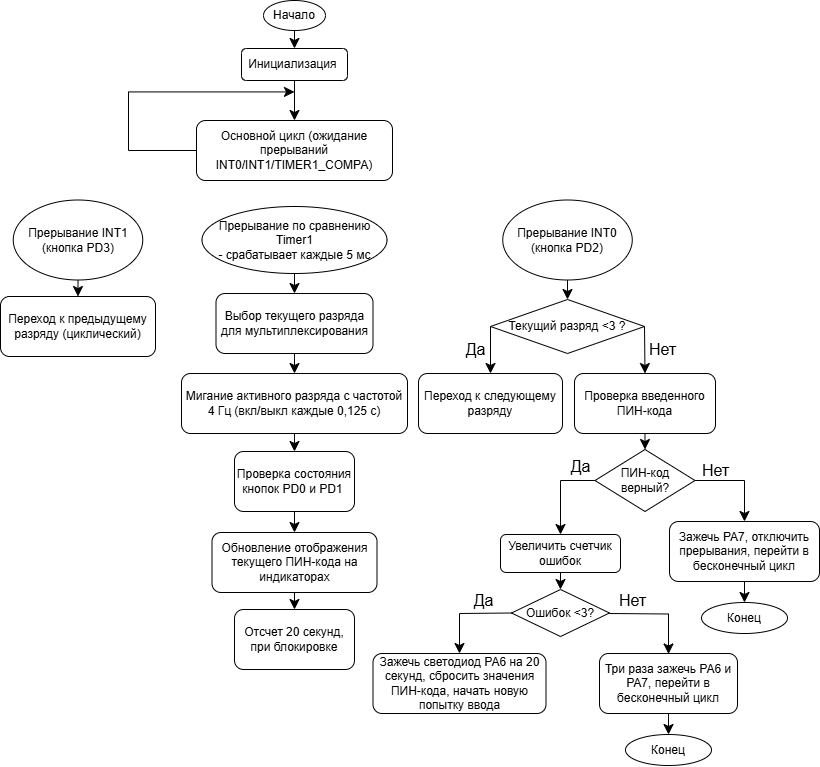


Рисунок 2. Блок-схема алгоритма программы

# Ответы на контрольные вопросы

1. *Посредством каких регистров производится конфигурирование таймера-счётчика?*

Рассмотрим все регистры на примере Timer1, так как он был использован в этой работе.

1. TCCR1A (Timer/Counter1 Control Register A)

* Адрес: 0x4F
* Назначение: Управление режимами сравнения и типами ШИМ.

|  |  |  |
| --- | --- | --- |
| Бит | Название | Описание |
| 7 | COM1A1 | Compare Output Mode for Channel A (бит 1). Определяет поведение вывода OC1A (PB1) при совпадении. |
| 6 | COM1A0 | Compare Output Mode for Channel A (бит 0). Работает в паре с COM1A1. |
| 5 | COM1B1 | Compare Output Mode for Channel B (бит 1). Аналогично для OC1B (PB2). |
| 4 | COM1B0 | Compare Output Mode for Channel B (бит 0). |
| 3 | FOC1A | Force Output Compare A. Принудительно устанавливает флаг совпадения (только в не-ШИМ режимах). |
| 2 | FOC1B | Force Output Compare B. Аналогично для канала B. |
| 1 | WGM11 | Waveform Generation Mode (бит 1). Часть настройки режима работы таймера. |
| 0 | WGM10 | Waveform Generation Mode (бит 0). |

Режимы сравнения (COM1A1/COM1A0 и COM1B1/COM1B0):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| COMA1 | COMA0 | Режим Non-PWM | Режим Fast PWM | Режим Phase Correct PWM |
| 0 | 0 | Отключено | Отключено | Отключено |
| 0 | 1 | Переключение OC1A при совпадении | Очистка OC1A при совпадении | Очистка при совпадении при счёте вверх |
| 1 | 0 | Очистка OC1A при совпадении | Установка OC1A при совпадении | Установка при совпадении при счёте вниз |
| 1 | 1 | Установка OC1A при совпадении | Инвертирование OC1A при совпадении | Инвертирование |

1. TCCR1B (Timer/Counter1 Control Register B)

* Адрес: 0x4E
* Назначение: Управление предделителем и режимами счёта.

|  |  |  |
| --- | --- | --- |
| Бит | Название | Описание |
| 7 | ICNC1 | Input Capture Noise Canceler. Включает фильтр помех для захвата сигнала. |
| 6 | ICES1 | Input Capture Edge Select. Выбор фронта срабатывания захвата (0 – спад, 1 – фронт). |
| 5 | – | Не используется. |
| 4 | WGM13 | Waveform Generation Mode (бит 3). |
| 3 | WGM12 | Waveform Generation Mode (бит 2). |
| 2 | CS12 | Clock Select (бит 2). Выбор источника тактирования. |
| 1 | CS11 | Clock Select (бит 1). |
| 0 | CS10 | Clock Select (бит 0). |

Настройки предделителя (CS12:CS10)

|  |  |  |  |
| --- | --- | --- | --- |
| CS12 | CS11 | CS10 | Режим тактирования |
| 0 | 0 | 0 | Таймер остановлен |
| 0 | 0 | 1 | Тактовая частота (8 МГц) |
| 0 | 1 | 0 | Предделитель 8 |
| 0 | 1 | 1 | Предделитель 64 |
| 1 | 0 | 0 | Предделитель 256 |
| 1 | 0 | 1 | Предделитель 1024 |
| 1 | 1 | 0 | Внешний источник (T1, фронт) |
| 1 | 1 | 1 | Внешний источник (T1, спад) |

Режимы работы таймера (WGM13:WGM10)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WGM13 | WGM12 | WGM11 | WGM10 | Режим |
| 0 | 0 | 0 | 0 | Normal (обычный счётчик, переполнение на 0xFFFF) |
| 0 | 1 | 0 | 0 | CTC (сброс при совпадении с OCR1A) |
| 1 | 0 | 0 | 0 | Fast PWM, 8-bit |
| 1 | 1 | 0 | 0 | Fast PWM, 9-bit |
| 0 | 0 | 1 | 1 | Phase Correct PWM, 8-bit |
| 1 | 1 | 1 | 1 | Phase Correct PWM, 10-bit |

1. TCNT1 (Timer/Counter1 Value Register)

* Адрес: 0x4C (младший байт), 0x4D (старший байт)
* Назначение: Хранит текущее значение счётчика.
* 16-битный регистр (TCNT1H:TCNT1L).
* Можно читать и записывать.

1. OCR1A и OCR1B (Output Compare Registers)

* OCR1A (0x4A, 0x4B) — сравнение для канала A.
* OCR1B (0x48, 0x49) — сравнение для канала B.
* Назначение: Устанавливают значение, при котором срабатывает прерывание или меняется состояние вывода OC1A/OC1B.

1. TIMSK (Timer Interrupt Mask Register)

* Адрес: 0x59
* Назначение: Включение прерываний таймера.

|  |  |  |
| --- | --- | --- |
| Бит | Название | Описание |
| 7 | - | Не используется (в ATmega32) |
| 6 | TOIE2 | Бит разрешения прерывания по переполнению таймера/счетчика 2 (Timer 2) |
| 5 | OCIE2 | Бит разрешения прерывания по совпадению канала A таймера 2 (Timer 2) |
| 4 | TICIE1 | Бит разрешения прерывания по захвату таймера 1 (Timer 1) |
| 3 | OCIE1A | Бит разрешения прерывания по совпадению канала A таймера 1 (Timer 1) |
| 2 | OCIE1B | Бит разрешения прерывания по совпадению канала B таймера 1 (Timer 1) |
| 1 | TOIE1 | Бит разрешения прерывания по переполнению таймера 1 (Timer 1) |
| 0 | TOIE0 | Бит разрешения прерывания по переполнению таймера 0 (Timer 0) |

1. TIFR (Timer Interrupt Flag Register)

* Адрес: 0x58
* Назначение: Флаги прерываний таймера (сбрасываются автоматически или вручную побитово).

|  |  |  |
| --- | --- | --- |
| Бит | Название | Описание |
| 7 | OCF2 | Timer/Counter2 Output Compare Flag Устанавливается в 1, когда значение счётчика Timer2 (TCNT2) совпадает с регистром сравнения OCR2. Сбрасывается при обработке прерывания или ручной записи 1. |
| 6 | TOV2 | Timer/Counter2 Overflow Flag Устанавливается в 1 при переполнении Timer2 (когда TCNT2 переходит из 0xFF в 0x00). Сбрасывается аналогично OCF2. |
| 5 | ICF1 | Timer/Counter1 Input Capture Flag Устанавливается в 1, когда на входе ICP1 (PB0) происходит захват значения TCNT1 в регистр ICR1 (по фронту/спаду, задаётся битом ICES1 в TCCR1B). |
| 4 | OCF1A | Timer/Counter1 Output Compare A Match Flag Устанавливается в 1 при совпадении TCNT1 с регистром OCR1A. Сбрасывается при обработке прерывания или записи 1. |
| 3 | OCF1B | Timer/Counter1 Output Compare B Match Flag Аналогичен OCF1A, но для регистра OCR1B. |
| 2 | TOV1 | Timer/Counter1 Overflow Flag Устанавливается в 1 при переполнении Timer1 (когда TCNT1 переходит из 0xFFFF в 0x0000). |
| 1 | OCF0 | Timer/Counter0 Output Compare Flag Флаг совпадения для Timer0 (аналогичен OCF2, но для OCR0). |
| 0 | TOV0 | Timer/Counter0 Overflow Flag Флаг переполнения Timer0 (аналогичен TOV2). |

1. *Какие источники импульсов могут применяться для увеличения таймера-счётчика и для каких целей?*

Источниками тактовых импульсов могут быть:

* Внутренние источники тактирования. Применения: точные временные задержки, генерация ШИМ, создание программных таймеров.
* Внешние источники тактирования (кнопка, датчик, другой микроконтроллер). Применения: измерение частоты внешнего сигнала, синхронизация с внешним устройством
* Захват сигнала (вход ICP1 Таймера1). Применения: измерение длительности импульсов, детектирование частоты, анализ цифровых сигналов.
* Тактирование от асинхронного источника (внешний кварцевый резонатор). Применения: режим сверхнизкого электропотребления, часы реального времени.
* Сравнение и генерация импульсов (ШИМ и генерация сигналов). Применение: генерация звуковых сигналов, ЦАП.

1. *В каких режимах могут работать таймеры-счётчики?*

Режимы:

* Normal Mode (Обычный). чётчик увеличивается от 0 до максимума (0xFF для 8-бит, 0xFFFF для 16-бит), затем переполняется и сбрасывается в 0.
* CTC Mode (Clear Timer on Compare). Счётчик сбрасывается не при переполнении, а при совпадении с регистром OCR1A (для Timer1) или OCR0 (для Timer0/2).
* Режимы ШИМ (PWM).
* Fast PWM. Счётчик увеличивается до TOP (максимальное значение), затем сбрасывается в 0. Для 8-бит: TOP = 0xFF (или OCR0/OCR2). Для 16-бит: TOP = OCR1A/ICR1.
* Phase Correct PWM. Счётчик увеличивается до TOP (максимальное значение), затем сбрасывается в 0. Частота в 2 раза ниже, чем в Fast PWM
* Режим захвата. При изменении сигнала на входе ICP1 (PB0) значение счетчика TCNT1 сохраняется в ICR1.
* Timer2 в асинхронном режиме (RTC). Тактируется от внешнего кварца 32 кГц (выводы TOSC1/TOSC2), работает независимо от основного тактового генератора.

1. *Как рассчитать начальное значение таймера-счётчика по заданному времени, которое должен отмерить таймер-счётчик до своего переполнения?*

Где TCNTx – начальное значение, t - желаемое время (с), MAX – максимальное значение счетчика (28/216).

*5. В чём состоит отличие работы таймера-счётчика в режиме таймера и в режиме счётчика?*

Режим таймера – тактовый сигнал от внутреннего источника, режим счетчика – тактовый сигнал от внешнего источника.

|  |  |  |
| --- | --- | --- |
| Параметр | Режим таймера | Режим счётчика |
| Источник тактов | Внутренний (системная частота) | Внешний (выводы T0/T1) |
| Предделитель | Делитель (1, 8, 64, 256, 1024) | Нет (или фильтр помех) |
| Скорость счёта | Зависит от частоты микроконтроллера и делителя | Зависит от частоты внешнего сигнала |
| Применение | Задержки, ШИМ | Подсчёт импульсов, энкодеры |
| Чувствительность | Не зависит от внешних помех | Требует защиты от дребезга |

# Вывод

В ходе лабораторной работы были получены практические навыки по работе с таймерами-счётчиками и применению механизма прерываний. Была реализована программа, которая проверяет введенный ПИН-код с помощью режима СТС Таймера1.

# Приложение

Комментированный листинг программы проверки ПИН-кода:

; Определение констант для семисегментных индикаторов

.equ null = 0b00000000

.equ zero = 0b00111111

.equ one = 0b00000110

.equ two = 0b01011011

.equ three = 0b01001111

.equ four = 0b01100110

.equ five = 0b01101101

.equ six = 0b01111101

.equ seven = 0b00000111

.equ eight = 0b01111111

.equ nine = 0b01101111

; Определение адресов в SRAM

.equ CurrentDigitDisplay = 0x0064 ; Текущий индикатор для мультиплексирования

.equ BlinkCounter = 0x0065 ; Счетчик мигания

.equ LockState = 0x0066 ; Состояние блокировки

.equ LockCounter = 0x0067 ; Счетчик задержки

.equ ButtonHoldCounterPD0 = 0x0069 ; Счётчик времени удержания для PD0

.equ ButtonHoldCounterPD1 = 0x006A ; Счётчик времени удержания для PD1

.equ HoldStagePD0 = 0x006B ; Флаг этапа удержания для PD0 (0 - первый этап, 1 - второй этап)

.equ HoldStagePD1 = 0x006C ; ; Флаг этапа удержания для PD1 (0 - первый этап, 1 - второй этап)

.equ AutoIncrementActive = 0x006D ; Флаг активности автоинкремента (PD0)

.equ AutoDecrementActive = 0x006E ; Флаг активности автодекремента (PD1)

; Определение регистров

.def CurrentDisplay = R18 ; Текущий разряд

.def Digit0 = R19

.def Digit1 = R20

.def Digit2 = R21

.def Digit3 = R22

.def PinD\_ = R23

.def Button0Pressed = R24

.def Button1Pressed = R25

.def Blink = R26

.def PinDigit0 = R27

.def PinDigit1 = R28

.def PinDigit2 = R29

.def PinDigit3 = R30

.def IncorrectAttempts = R31

; Векторная таблица прерываний

.org $000

JMP RESET

.org $002

JMP INT0\_HANDLER

.org $004

JMP INT1\_HANDLER

.org $01A

JMP TIMER1\_COMPA

; Начало программы

.org $02A

RESET:

; Инициализация стека

LDI R16, HIGH(RAMEND)

OUT SPH, R16

LDI R16, LOW(RAMEND)

OUT SPL, R16

; Инициализация портов

SER R16

OUT DDRC, R16

CLR R16

OUT DDRD, R16

LDI R16, 0xFF

OUT PORTD, R16

CLR R16

OUT PORTA, R16

OUT PORTC, R16

LDI R16, 0xCF ; 0b11001111

OUT DDRA, R16 ; PA0-PA3 — выходы (выбор разряда индикаторов), PA6-PA7 — выходы (светодиоды)

; Инициализация начального состояния

CLR CurrentDisplay

CLR Digit0

CLR Digit1

CLR Digit2

CLR Digit3

CLR Button0Pressed

CLR Button1Pressed

LDI Blink, 0b00000001

CLR PinDigit0

CLR PinDigit1

CLR PinDigit2

CLR PinDigit3

CLR IncorrectAttempts

; Считывание PIN-кода из EEPROM

LDI R16, 0x00

OUT EEARH, R16 ; EEARH - старший байт адреса EEPROM (только 2 младших бита используются)

LDI R16, 0x00

OUT EEARL, R16 ; EEARL - младший байт адреса EEPROM

SBI EECR, EERE ; EECR - управляет операциями, EERE - бит чтения в EECR

IN PinDigit0, EEDR ; EEDR - содержит прочитанные данные

LDI R16, 0x01

OUT EEARL, R16

SBI EECR, EERE

IN PinDigit1, EEDR

LDI R16, 0x02

OUT EEARL, R16

SBI EECR, EERE

IN PinDigit2, EEDR

LDI R16, 0x03

OUT EEARL, R16

SBI EECR, EERE

IN PinDigit3, EEDR

; Инициализация переменных в SRAM

CLR R16

STS CurrentDigitDisplay, R16

STS BlinkCounter, R16

STS LockState, R16

STS LockCounter, R16

STS LockCounter+1, R16

STS ButtonHoldCounterPD0, R16

STS ButtonHoldCounterPD1, R16

STS HoldStagePD0, R16

STS HoldStagePD1, R16

STS AutoIncrementActive, R16

STS AutoDecrementActive, R16

; Настройка Timer1 (5 мс, 200 Гц)

CLR R16

OUT TCCR1A, R16

OUT TCCR1B, R16

LDI R16, HIGH(156)

OUT OCR1AH, R16

LDI R16, LOW(156)

OUT OCR1AL, R16

LDI R16, 0b00001100 ; Режим CTC, предделитель 256

OUT TCCR1B, R16

LDI R16, 0b00001000 ; Разрешение прерывания по совпадению OCR1A Timer1

OUT TIMSK, R16

;LDI R16, 0b01000000 ; Сброс флага прерывания

;OUT TIFR, R16

; Настройка прерываний INT0 и INT1

LDI R16, 0x0A ; 0b00001010

OUT MCUCR, R16 ; INT0 и INT1 по спаду фронта

LDI R16, 0xC0 ; 0b11000000

OUT GICR, R16 ; Разрешить INT0 и INT1

LDI R16, 0xC0

OUT GIFR, R16 ; Сбросить флаги INTF0 и INTF1

SEI ; Глобальное разрешение прерываний

; Основной цикл

main\_loop:

RJMP main\_loop

; Подпрограмма преобразования значения цифры

show\_digit:

CPI R16, 0

BREQ digit\_zero

CPI R16, 1

BREQ digit\_one

CPI R16, 2

BREQ digit\_two

CPI R16, 3

BREQ digit\_three

CPI R16, 4

BREQ digit\_four

CPI R16, 5

BREQ digit\_five

CPI R16, 6

BREQ digit\_six

CPI R16, 7

BREQ digit\_seven

CPI R16, 8

BREQ digit\_eight

CPI R16, 9

BREQ digit\_nine

RET

digit\_zero: LDI R16, zero RET

digit\_one: LDI R16, one RET

digit\_two: LDI R16, two RET

digit\_three: LDI R16, three RET

digit\_four: LDI R16, four RET

digit\_five: LDI R16, five RET

digit\_six: LDI R16, six RET

digit\_seven: LDI R16, seven RET

digit\_eight: LDI R16, eight RET

digit\_nine: LDI R16, nine RET

; Подпрограммы инкремента и декремента

increment\_digit:

CPI CurrentDisplay, 0

BREQ inc\_digit0

CPI CurrentDisplay, 1

BREQ inc\_digit1

CPI CurrentDisplay, 2

BREQ inc\_digit2

CPI CurrentDisplay, 3

BREQ inc\_digit3

RET

inc\_digit0: INC Digit0 CPI Digit0, 10 BRNE exit\_increment CLR Digit0 RJMP exit\_increment

inc\_digit1: INC Digit1 CPI Digit1, 10 BRNE exit\_increment CLR Digit1 RJMP exit\_increment

inc\_digit2: INC Digit2 CPI Digit2, 10 BRNE exit\_increment CLR Digit2 RJMP exit\_increment

inc\_digit3: INC Digit3 CPI Digit3, 10 BRNE exit\_increment CLR Digit3

exit\_increment:

RET

decrement\_digit:

CPI CurrentDisplay, 0

BREQ dec\_digit0

CPI CurrentDisplay, 1

BREQ dec\_digit1

CPI CurrentDisplay, 2

BREQ dec\_digit2

CPI CurrentDisplay, 3

BREQ dec\_digit3

RET

dec\_digit0: DEC Digit0 CPI Digit0, 0xFF BRNE exit\_decrement LDI Digit0, 9 RJMP exit\_decrement

dec\_digit1: DEC Digit1 CPI Digit1, 0xFF BRNE exit\_decrement LDI Digit1, 9 RJMP exit\_decrement

dec\_digit2: DEC Digit2 CPI Digit2, 0xFF BRNE exit\_decrement LDI Digit2, 9 RJMP exit\_decrement

dec\_digit3: DEC Digit3 CPI Digit3, 0xFF BRNE exit\_decrement LDI Digit3, 9

exit\_decrement:

RET

; Обработчик INT0

INT0\_HANDLER:

PUSH R16

IN R16, SREG

PUSH R16

LDS R16, LockState

CPI R16, 1

BREQ exit\_int0

CPI CurrentDisplay, 3

BREQ check\_pin\_call

INC CurrentDisplay

CPI CurrentDisplay, 4

BRNE exit\_int0

CLR CurrentDisplay

RJMP exit\_int0

check\_pin\_call:

RCALL check\_pin

exit\_int0:

POP R16

OUT SREG, R16

POP R16

RETI

; Обработчик INT1

INT1\_HANDLER:

PUSH R16

IN R16, SREG

PUSH R16

LDS R16, LockState

CPI R16, 1

BREQ exit\_int1

DEC CurrentDisplay

CPI CurrentDisplay, 0xFF

BRNE exit\_int1

LDI CurrentDisplay, 3

exit\_int1:

POP R16

OUT SREG, R16

POP R16

RETI

TIMER1\_COMPA:

PUSH R16

IN R16, SREG

PUSH R16

PUSH R17

PUSH R2 ; Используется для LockCounter

PUSH R3 ; Используется для LockCounter

; Очистка дисплея, сохраняя PA6 и PA7

IN R16, PORTA

ANDI R16, 0b11000000 ; ANDI - логическое И с константой, оставляем только биты PA6 и PA7

OUT PORTA, R16

CLR R16

OUT PORTC, R16

; Проверка состояния блокировки

LDS R16, LockState

CPI R16, 1

BRNE not\_locked

RJMP locked\_display

not\_locked:

; Считываем состояние PIND

IN PinD\_, PIND

; Проверка PD0 (кнопка инкремента)

SBIC PIND, 0

RJMP pd0\_released ; Если PD0 = 1 (отпущена), переходим к pd0\_released

; PD0 = 0 (кнопка нажата)

SBR Button0Pressed, 1 ; Устанавливаем флаг нажатия кнопки PD0

; Увеличиваем счётчик удержания для PD0

LDS R16, ButtonHoldCounterPD0

INC R16

STS ButtonHoldCounterPD0, R16

; Проверяем этап удержания

LDS R17, HoldStagePD0

CPI R17, 0

BREQ pd0\_first\_stage

; Второй этап (ещё 1 секунда)

CPI R16, 200

BRLO check\_pd1

; 2 секунды прошло, активируем автоинкремент

LDI R16, 1

STS AutoIncrementActive, R16

; Сбрасываем счётчик для следующего цикла

CLR R16

STS ButtonHoldCounterPD0, R16

RJMP check\_pd1

pd0\_first\_stage:

; Первый этап (1 секунда)

CPI R16, 200

BRLO check\_pd1

; 1 секунда прошла, переходим на второй этап

LDI R17, 1

STS HoldStagePD0, R17

CLR R16

STS ButtonHoldCounterPD0, R16

RJMP check\_pd1

pd0\_released:

SBRS Button0Pressed, 0

RJMP pd0\_not\_held

; Кнопка отпущена и флаг был установлен

CBR Button0Pressed, 1

RCALL increment\_digit

pd0\_not\_held:

; Сбрасываем счётчик, этап удержания и флаг автоинкремента для PD0

CLR R16

STS ButtonHoldCounterPD0, R16

STS HoldStagePD0, R16

STS AutoIncrementActive, R16

check\_pd1:

; Проверка PD1 (кнопка декремента)

SBIC PIND, 1

RJMP pd1\_released

; PD1 = 0 (кнопка нажата)

SBR Button1Pressed, 1

; Увеличиваем счётчик удержания для PD1

LDS R16, ButtonHoldCounterPD1

INC R16

STS ButtonHoldCounterPD1, R16

; Проверяем этап удержания

LDS R17, HoldStagePD1

CPI R17, 0

BREQ pd1\_first\_stage

; Второй этап (ещё 1 секунда)

CPI R16, 200

BRLO skip\_hold\_action

; 2 секунды прошло, включаем PA7 и активируем автодекремент

LDI R16, 1

STS AutoDecrementActive, R16

; Сбрасываем счётчик для следующего цикла

CLR R16

STS ButtonHoldCounterPD1, R16

RJMP skip\_hold\_action

pd1\_first\_stage:

; Первый этап (1 секунда)

CPI R16, 200

BRLO skip\_hold\_action

; 1 секунда прошла, переходим на второй этап

LDI R17, 1

STS HoldStagePD1, R17

CLR R16

STS ButtonHoldCounterPD1, R16

RJMP skip\_hold\_action

pd1\_released:

SBRS Button1Pressed, 0

RJMP pd1\_not\_held

; Кнопка отпущена и флаг был установлен

CBR Button1Pressed, 1

RCALL decrement\_digit

pd1\_not\_held:

; Сбрасываем счётчик, этап удержания и флаг автодекремента для PD1

CLR R16

STS ButtonHoldCounterPD1, R16

STS HoldStagePD1, R16

STS AutoDecrementActive, R16

skip\_hold\_action:

; Выбор текущего индикатора

LDS R17, CurrentDigitDisplay

LDI R16, 0b00000001 ; Начальная маска для PA0 (первый индикатор)

CPI R17, 0 ; Если CurrentDigitDisplay = 0, пропускаем сдвиг

BREQ skip\_shift

shift\_loop:

LSL R16 ; Сдвигаем бит влево для выбора следующего индикатора (PA1, PA2, PA3)

DEC R17 ; Уменьшаем счётчик

BRNE shift\_loop ; Повторяем, пока не выберем нужный индикатор

skip\_shift:

IN R17, PORTA

ANDI R17, 0b11000000 ; Сохраняем состояния PA6 и PA7 (индикаторы ошибок)

OR R16, R17 ; Объединяем маску индикатора с сохранёнными битами PA6 и PA7

OUT PORTA, R16 ; Выводим на PORTA для активации текущего индикатора

; Загрузка и вывод цифры с учётом мигания

LDS R17, CurrentDigitDisplay

CPI R17, 0

BREQ load\_digit0

CPI R17, 1

BREQ load\_digit1

CPI R17, 2

BREQ load\_digit2

MOV R16, Digit3

RCALL show\_digit

RJMP display\_digit

load\_digit0:

MOV R16, Digit0

RCALL show\_digit

RJMP display\_digit

load\_digit1:

MOV R16, Digit1

RCALL show\_digit

RJMP display\_digit

load\_digit2:

MOV R16, Digit2

RCALL show\_digit

display\_digit:

CP R17, R18 ; Сравниваем CurrentDigitDisplay с CurrentDisplay (текущий редактируемый разряд)

BRNE static\_digit ; Если не совпадают, выводим статично

SBRC Blink, 0 ; Если совпадают и бит мигания = 1, выводим цифру

OUT PORTC, R16

RJMP update\_digit

static\_digit:

OUT PORTC, R16

RJMP update\_digit

locked\_display:

IN R16, PORTA

ANDI R16, 0b10000000

ORI R16, 0b01000000

OUT PORTA, R16

update\_digit:

; Увеличение CurrentDigitDisplay

LDS R17, CurrentDigitDisplay

INC R17

CPI R17, 4

BREQ met

RJMP store\_digit

met:

CLR R17

; Управление миганием и автоинкремент/автодекремент

LDS R16, LockState

CPI R16, 1

BREQ skip\_blink

LDS R16, BlinkCounter

INC R16

STS BlinkCounter, R16

CPI R16, 25

BRNE no\_blink\_toggle

; Полный цикл мигания (каждые 125 мс)

LDI R16, 0b00000001 ; Маска для переключения бита

EOR Blink, R16 ; Инвертируем бит мигания (EOR = XOR)

CLR R16

STS BlinkCounter, R16

; Проверяем автоинкремент (PD0)

LDS R16, AutoIncrementActive

CPI R16, 1

BRNE check\_auto\_decrement

; Выполняем автоинкремент, только если Blink = 1 (каждые 250 мс)

SBRS Blink, 0

RJMP check\_auto\_decrement

RCALL increment\_digit

check\_auto\_decrement:

; Проверяем автодекремент (PD1)

LDS R16, AutoDecrementActive

CPI R16, 1

BRNE no\_blink\_toggle

; Выполняем автодекремент, только если Blink = 1 (каждые 250 мс)

SBRS Blink, 0

RJMP no\_blink\_toggle

RCALL decrement\_digit

no\_blink\_toggle:

STS BlinkCounter, R16 ; Сохраняем значение счётчика (пока не достиг 25)

skip\_blink:

; Управление задержкой

LDS R16, LockState

CPI R16, 1

BRNE store\_digit

; Увеличение LockCounter

LDS R2, LockCounter

LDS R3, LockCounter+1

LDI R16, 1

ADD R2, R16

CLR R16

ADC R3, R16

STS LockCounter, R2

STS LockCounter+1, R3

; Проверка на 1000 (20 секунд)

LDI R16, LOW(1000)

CP R2, R16

BRNE skip\_reset

LDI R16, HIGH(1000)

CP R3, R16

BRNE skip\_reset

; Сброс состояния

CLR R16

STS LockState, R16

STS LockCounter, R16

STS LockCounter+1, R16

CBI PORTA, 6

RCALL reset\_display

skip\_reset:

store\_digit:

STS CurrentDigitDisplay, R17

POP R3

POP R2

POP R17

POP R16

OUT SREG, R16

POP R16

RETI

; Проверка PIN-кода

check\_pin:

CP Digit0, PinDigit0

BRNE pin\_incorrect

CP Digit1, PinDigit1

BRNE pin\_incorrect

CP Digit2, PinDigit2

BRNE pin\_incorrect

CP Digit3, PinDigit3

BRNE pin\_incorrect

RCALL pin\_correct

RET

pin\_incorrect:

INC IncorrectAttempts

CPI IncorrectAttempts, 3

BRGE pin\_locked

LDI R16, 1

STS LockState, R16

CLR R16

STS LockCounter, R16

STS LockCounter+1, R16

SBI PORTA, 6

CBI PORTA, 0

CBI PORTA, 1

CBI PORTA, 2

CBI PORTA, 3

RET

pin\_correct:

SBI PORTA, 7

CBI PORTA, 0

CBI PORTA, 1

CBI PORTA, 2

CBI PORTA, 3

LDI R16, 0xC0

IN R17, GICR

AND R17, R16

OUT GICR, R17

CLI

lock\_loops:

RJMP lock\_loops

pin\_locked:

LDI R19, 3

blink\_loop:

CBI PORTA, 0

CBI PORTA, 1

CBI PORTA, 2

CBI PORTA, 3

CLR R16

OUT PORTC, R16

SBI PORTA, 7

SBI PORTA, 6

RCALL blink\_locked

CBI PORTA, 7

CBI PORTA, 6

RCALL blink\_locked

DEC R19

BRNE blink\_loop

CLR Blink

CLI

lock\_loop:

RJMP lock\_loop

; Сброс дисплея

reset\_display:

CLR Digit0

CLR Digit1

CLR Digit2

CLR Digit3

CLR CurrentDisplay

RET

blink\_locked:

LDI R20, 100

outers:

LDI R21, 80

middles:

LDI R22, 100

inners:

DEC R22

BRNE inners

DEC R21

BRNE middles

DEC R20

BRNE outers

RET