Лабораторная работа №2

Порты ввода/вывода. Вывод сигналов

1. Цель работы

Целью работы получение практических навыков настройки портов ввода/вывода микроконтроллера STM8S207.

2. Теоретические сведения по портам ввода/вывода

Порты ввода-вывода общего назначения используются для обмена данными между микроконтроллером и внешними периферийными устройствами. Каждый порт микроконтроллера STM8 может содержать до восьми выводов. Каждый вывод можно индивидуально запрограммировать как цифровой вход или цифровой выход. Кроме того, некоторые выводы могут иметь альтернативные функции, такие как аналоговые входы, внешние прерывания, вход-выход для периферийных устройств.

Настройка и работа с портом осуществляется с помощью соответствующих ему регистров: регистр выходных данных, регистр входных данных, регистр направления данных и два регистра конфигурации.

2.1 Основные характеристики GPIO

Порты ввода/вывода микроконтроллера STM8S207 имеют следующие характеристики:

- каждый бит порта можно настроить индивидуально;
- возможные режимы порта, настроенного на вход: *floating input* (высокоимпедансный вход или вход в *Z*-состоянии) или *input with pull-up* (вход с подтягивающим резистором);
- возможные режимы порта, настроенного на выход: p*ush-pull output* (двухтактный выход) или *pseudo-open-drain output* (выход с псевдо-открытым стоком);
- отдельные регистры для ввода и вывода данных;
- внешние прерывания можно включать и отключать индивидуально;
- контроль крутизны выходного сигнала для баланса ЭМС/быстродействие;
- альтернативные функции портов ввода-вывода для периферийных устройств;

- входной триггер Шмитта можно отключить на аналоговых входах для снижения энергопотребления;
- входы толерантные к напряжению 5 В;
- гарантированное состояние ввода-вывода в диапазоне напряжений от 1,6 В до $V_{DDIOmax}$.

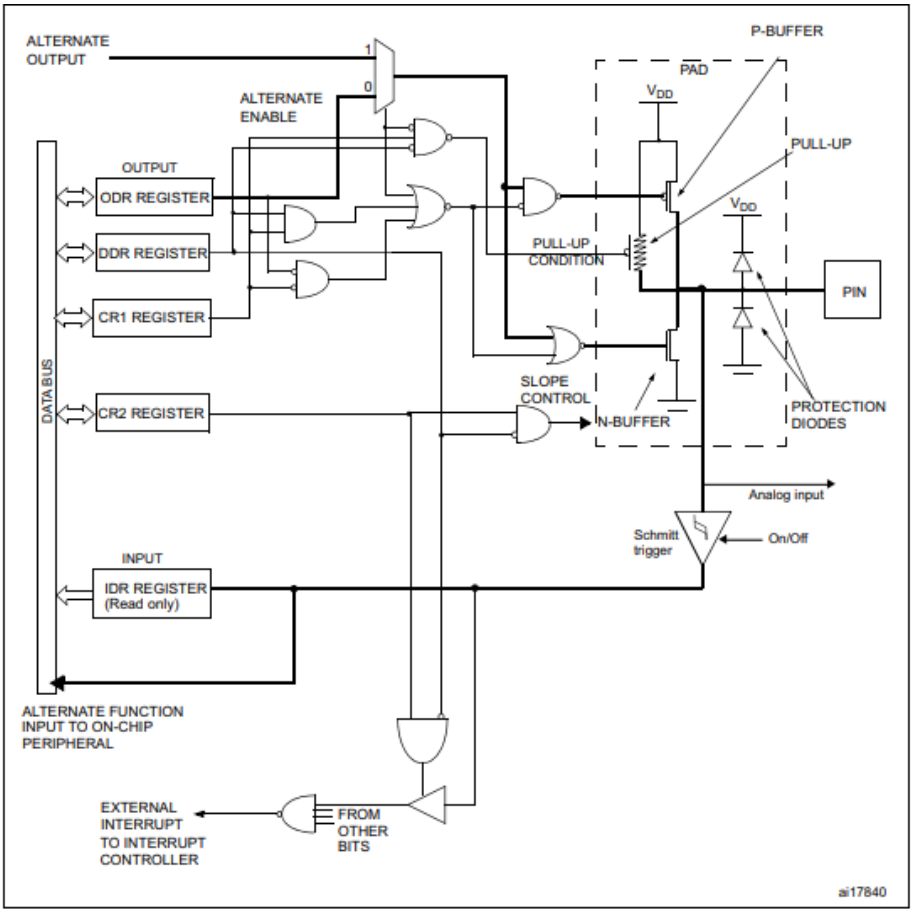

Рис.1. Блок-схема портов ввода-вывода общего назначения (GPIO)

2.2 Конфигурация и использование порта

Регистр выходных данных (*ODR*), входных данных (*IDR*), регистр направления данных (*DDR*) всегда привязаны к каждому порту.

С помощью регистра управления 1 (*CR1*) и регистра управления 2 (*CR2*) осуществляется дополнительная настройка ввода/вывода. Каждый вывод порта настраивается с помощью соответствующих ему битов в регистрах *DDR*, *ODR*, *CR1* и *CR2*.

Бит *n* в регистрах соответствует выводу *n* порта. Различные конфигурации приведены в Таблице 1.

Таблица 1

Конфигурации портов GPIO

Режим	DDR бит	CR1 бит	CR2 бит	Функция	Подтяжка	P-буфер	Диоды	
							к V_{dd}	к V_{ss}
Вход	0	0	0	Высокоимпедансный вход, без прерываний	Выкл.	Выкл.	Вкл.	Вкл.
	0	1	0	С подтяжкой без прерываний	Вкл.			
	0	0	1	Высокоимпедансный вход, с прерыванием	Выкл.			
	0	1	1	С подтяжкой с прерыванием	Вкл.			
Выход	1	0	0	С псевдо-открытым стоком, 2 МГц	Выкл.	Выкл.		
	1	1	0	Двухтактный, 2 МГц		Вкл.		
	1	0	1	С псевдо-открытым стоком, 10 МГц		Выкл.		
	1	1	1	Двухтактный, 10 МГц	Выкл.	Вкл.		
	1	x	x	С открытым стоком	Не реализ.		Не реализ.	

2.3 Режимы ввода

Настройка вывода порта на работу в режиме ввода осуществляется очисткой соответствующего бита в регистре *DDRx*. В этом режиме чтение бита *IDR* возвращает цифровое значение соответствующего порта ввода/вывода.

Как показано в Таблице 1, теоретически могут быть реализованы четыре различных режима ввода: высокоимпедансный вход без прерывания, высокоимпедансный вход с прерыванием, вход с подтяжкой к питанию без прерывания или вход с подтяжкой к питанию с прерыванием. Однако на практике не все порты имеют возможность внешнего прерывания или подтяжки к питанию. Для получения подробной информации о фактических аппаратных возможностях каждого порта следует обратиться к описанию выводов в технической документации.

2.4 Режимы вывода

С помощью установки бита *DDRx* выбирается режим вывода. В этом режиме запись в биты *ODR* применяет цифровое значение к соответствующим выводам через защелку. Чтение бита IDR возвращает цифровое значение с соответствующего контакта. Используя регистры *CR1, CR2* можно программно настроить режимы вывода двухтактный

выход или выход с псевдо-открытым стоком, а также крутизну фронта сигнала на выходе.

2.5 Настройки по умолчанию

В основном, все контакты ввода/вывода являются высокоимпедансными входами во время и после процедуры сброса. Однако некоторые контакты могут вести себя иначе. Подробную информацию см. в описании выводов в технической документации.

2.6 Неиспользуемые контакты ввода/вывода

Неиспользуемые контакты ввода/вывода не рекомендуется оставлять в режиме высокоимпедансного входа, чтобы избежать дополнительного потребления тока. Их следует перевести в одну из следующих конфигураций:

- подключить внешний резистор, подтягивающий к линии питания или земли и сохранить настройки высокоимпедансного входа;
- настроить, как вход с внутренним подтягивающим к питанию или земле резистором,
- установить, как двунаправленный выход с низким уровнем.

Порты ввода-вывода, отсутствующие в корпусах меньшего размера, автоматически настраиваются заводскими настройками (если иное не указано в документации). Как следствие, эти порты ввода-вывода не требуют настройки. Биты, соответствующие этим портам, в регистры конфигурации *Px_ODR*, *PxDD*R, *PxCR1* и *PxCR2* могут быть записаны, но это не будет иметь никакого эффекта. Значение, считываемое в соответствующих битах регистра *PxIDR*, будет равно «0».

2.7 Режимы низкого энергопотребления

В таблице 2 показано влияние режимов пониженного энергопотребления на порты *GPIO*.

Таблица 2

Влияние режимов низкого энергопотребления на порты GPIO

Режим	Описание
Ожидание	Не влияет на порты ввода/вывода. Вывод из режима осуществляется внешним прерыванием или сбросом.
Остановка	Не влияет на порты ввода/вывода. Вывод из режима осуществляется внешним прерыванием или сбросом.

Примечание: если контакты *PA1/PA2* используются для подключения внешнего генератора, чтобы обеспечить наименьшее

энергопотребление в режиме остановки, *PA1* и *PA2* должны быть настроены как вход *pull-up*.

2.8 Альтернативная функция ввода

Некоторые порты ввода-вывода могут использоваться для реализации альтернативной функции. Например, порт может использоваться как вход для таймера. **Входы альтернативных функций не выбираются автоматически, их необходимо выбрать, записывая управляющий бит в регистры соответствующего периферийного устройства.**

Для входа с альтернативной функцией необходимо выбрать конфигурацию входа в *floating* или *pull-up* режиме в регистрах *DDR* и *CR1*.

2.9 Возможность прерывания

Каждый порт ввода-вывода может быть настроен как вход с возможностью прерывания с помощью установки бита *CR2x*, когда порт ввода-вывода находится в режиме ввода. В этом режиме сигнал по фронту или уровню генерирует запрос прерывания.

Чувствительность к переднему или заднему фронту программируется независимо для каждого вектора прерывания в регистрах *EXTI_CR [2:1]*.

Возможность внешнего прерывания доступна, только если порт настроен в режиме ввода.

Прерывания можно разрешить/запретить отдельно, запрограммировав соответствующий бит в регистре конфигурации (*Px_CR2*). В состоянии после сброса прерывания отключены.

Если альтернативной функцией вывода является *TLI* (top level interrupt – прерывание высшего уровня), используйте бит *Px_CR2* для включения/выключения прерывания *TLI*. Прерывание *TLI* связано с выделенным вектором прерывания.

2.10 Аналоговые каналы

Аналоговые каналы портов ввода-вывода могут быть выбраны периферийным устройством АЦП. В этом случае, соответствующие входные и выходные каскады автоматически отключаются. Входной триггер Шмитта должен быть отключен в регистре *ADC_TDR* при использовании аналоговых каналов.

Таблица 3

Рекомендованные и не рекомендованные конфигурации аналоговых входов

DDR	CR1	CR2	ADC_TDR	Конфигурация	Комментарий
0	0	0	1	Вход с отключенными подтягивающими резисторами. Триггер Шмитта должен быть отключен	Рекомендована конфигурация аналогового входа.
0	1	x	x	Вход с подтяжкой	Не рекомендуется в качестве аналогового входа, когда присутствует аналоговое напряжение, поскольку эти конфигурации вызывают избыточный ток на входном контакте. Входной и выходной каскады отключены на выбранном канале АЦП.
1	0	x	x	Выход	
1	1	x	x	Выход	

2.11 Триггер Шмитта

На всех портах ввода-вывода с аналоговым входом можно отключить триггер Шмитта, даже если соответствующий канал АЦП не включен. Два регистра *ADC_TDRH* и *ADC_TDRL* позволяют отключить триггер Шмитта. Установка одного бита в эти регистры приводит к отключению соответствующего входного буфера триггера Шмитта.

В случае если порт ввода-вывода используется в качестве аналогового входа и соответствующий канал АЦП включен (биты *CH [3:0]* в регистре *ADC_CSR*), триггер Шмитта отключен, независимо от состояния соответствующего бита в регистрах *ADC_TDRH* или *ADC_TDRL*.

2.12 Аналоговая функция

Выбранные порты ввода-вывода могут использоваться для передачи аналогового сигнала периферии: АЦП, компараторов или ЦАП. Вывод порта должен быть настроен как высокоимпедансный вход без прерывания (состояние по умолчанию), чтобы использовать его как аналоговый. Потребление тока аналоговым портом можно уменьшить, отключив неиспользуемый триггер Шмитта с помощью регистра *ADC_TRIGRx* в интерфейсе АЦП, либо путем включения

соответствующего аналогового переключателя в *RI*, установив соответствующий бит *CHxE* в *RI_IOSRx* (для семейства STM8L). Информацию о выводах с аналоговыми функциями см. в техническом описании.

2.13 Альтернативная функция вывода

Альтернативные функции обеспечивают прямой путь от периферийного устройства к выходу или к портам ввода-вывода, имея приоритет над битом порта в регистре защелки вывода данных (*Px_ODR*) и устанавливая соответствующий бит *Px_DDR* в единицу.

Альтернативный выход функции может быть *push-pull* или *pseudo-open-drain* в зависимости от периферийного устройства и регистра управления 1 (*Px_CR1*), а крутизной фронта можно управлять в зависимости от значений регистра управления 2 (*Px_CR2*).

2.14 Контроль крутизны фронта

Максимальная частота, которая может быть применена к вводу/выводу, может контролироваться программно с помощью бита *CR2*. При сбросе выбирается низкочастотный режим с улучшенными характеристиками ЭМС. При необходимости можно выбрать более высокую частоту (до 10 МГц). Эта функция может применяться как в режиме выхода *open drain*, так и в режиме выхода *push-pull*.

2.15 Регистры GPIO

Каждый отдельный бит регистра порта управляет соответствующим выводом порта!!!

2.15.1 Регистр выходных данных порта *x* (Px_ODR)

Адрес со смещением: 0x00

Значение сброса: 0x00

7	6	5	4	3	2	1	0
ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rw	rw	rw	rw	rw	rw	rw	rw

Биты 7:0 ***ODR [7:0]***: биты регистра выходных данных.

Запись в регистр *ODR* в режиме выхода применяет цифровое значение к вводу/выводу через защелку.

Чтение *ODR* возвращает ранее зафиксированное значение в регистре.

В режиме входа запись в регистр *ODR* фиксирует значение в регистре, но не меняет состояние вывода. Регистр *ODR* всегда очищается после сброса. Команды чтения-записи битов (*BSET*, *BRST*) могут использоваться в регистре *DR* для управления отдельным выводом, не затрагивая другие.

2.15.2 Регистр входных данных порта x (Px_IDR)

Адрес со смещением: 0x01

Значение сброса: 0xXX

7	6	5	4	3	2	1	0
IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
rw	rw	rw	rw	rw	rw	rw	rw

Биты 7:0 **IDR [7:0]**: входные значения.

Регистр вывода может использоваться для чтения значения вывода независимо от того, находится ли порт в режиме ввода или вывода. Этот регистр доступен только для чтения.

0: низкий логический уровень

1: высокий логический уровень

Примечание: значение сброса *Px_IDR* зависит от внешней схемы.

2.15.3 Регистр направления данных порта x (Px_DDR)

Адрес со смещением: 0x02

Значение сброса: 0x00

7	6	5	4	3	2	1	0
DDR7	DDR6	DDR5	DDR4	DDR3	DDR2	DDR1	DDR0
rw	rw	rw	rw	rw	rw	rw	rw

Биты 7: 0 **DDR [7:0]**: биты направления данных

Эти биты устанавливаются и сбрасываются программно для выбора режима ввода или вывода для определенного контакта порта.

0: режим ввода

1: режим вывода

2.15.4 Управляющий регистр 1 порта x (Px_CR1)

Адрес со смещением: 0x03

Значение сброса: 0x00, за исключением *PD_CR1*, значение сброса которого равно 0x02.

7	6	5	4	3	2	1	0
C17	C16	C15	C14	C13	C12	C11	C10
rw	rw	rw	rw	rw	rw	rw	rw

Биты 7:0 ***C1[7:0]:*** управляющие биты

Эти биты устанавливаются и сбрасываются программно. Они выбирают различные функции в режиме ввода и вывода.

- В режиме ввода (*DDR* = 0):

0: высокоимпедансный вход

1: вход с подтяжкой к питанию

- В режиме вывода (*DDR* = 1):

0: выход с псевдо открытым стоком

1: двухтактный выход, управление крутизной для выхода зависит от соответствующего бита *CR2*

Примечание: этот бит не влияет на порты с истинным открытым стоком (см. вывод с пометкой «T» в таблице описания портов).

2.15.5 Управляющий регистр 2 порта x (Px_CR2)

Адрес со смещением: 0x04

Значение сброса: 0x00

7	6	5	4	3	2	1	0
C27	C26	C25	C24	C23	C22	C21	C20
rw	rw	rw	rw	rw	rw	rw	rw

Биты 7: 0 **C2[7: 0]**: управляющие биты

Эти биты устанавливаются и сбрасываются программно. Они выбирают разные функции в режиме ввода и вывода. В режиме ввода бит CR2 включает возможность прерывания, если она доступна. Если ввод/вывод не имеет возможности прерывания, установка бита CR2 не имеет никакого эффекта. В режиме вывода установка бита увеличивает скорость ввода/вывода. Это относится к портам с типами выходов O3 и O4 (см. таблицу описания портов).

- В режиме ввода (DDR = 0):

0: внешнее прерывание отключено

1: внешнее прерывание разрешено

- В режиме вывода (DDR = 1):

0: выходная скорость до 2 МГц

1: выходная скорость до 10 МГц

2.15.6 Таблица регистров GPIO и значений сброса

Каждый порт GPIO имеет пять регистров, отображаемых, как показано в Таблице 4. В таблице регистров в соответствующей документации содержатся базовые адреса для каждого порта.

Таблица 4

Регистры GPIO

Смещение адреса	Наименование регистра	7	6	5	4	3	2	1	0
0x00	Px_ODR	ODR7 0	ODR6 0	ODR5 0	ODR4 0	ODR3 0	ODR2 0	ODR1 0	ODR0 0
0x01	Px_IDR	IDR7 x	IDR6 x	IDR5 x	IDR4 x	IDR3 x	IDR2 x	IDR1 x	IDR0 x
0x02	Px_DDR	DDR7 0	DDR 6 0	DDR 5 0	DDR 4 0	DDR 3 0	DDR 2 0	DDR 1 0	DDR 0 0
0x03	Px_CR1	C17 0	C16 0	C15 0	C14 0	C13 0	C12 0	C11 0	C10 0
0x04	Px_CR2	C27 0	C26 0	C25 0	C24 0	C23 0	C22 0	C21 0	C20 0

Примечание: в состоянии сброса все порты являются высокоимпедансными входами. Исключения указаны в таблице описания контактов соответствующей документации.

Обозначения в таблице 6 документации на микроконтроллер STM8S207 (Table 6. Pin description).

Таблица 5

Обозначение в таблице описания выводов микроконтроллера

Тип	I = Вход, O = Выход, S = Источник питания	
Уровень	Вход	CM = CMOS
	Выход	HS = High sink
Скорость выхода	O1 = Низкая (до 2 МГц) O2 = Высокая (до 10 МГц) O3 = Быстрое/медленное программирование с медленным по умолчанию после сброса O4 = Быстрое/медленное программирование с быстрым как состояние по умолчанию после сброса	
Настройки порта	Вход	float = floating, wpu = weak pull-up
	Выход	T = True open drain, OD = Open drain, PP = Push pull
Состояние при сбросе	Обозначено знаком **X** (состояние вывода после внутреннего сброса) Если не указано иное, состояние вывода одинаково во время фазы сброса и после сброса.	

3. Программа работы

1. Ознакомьтесь с составом отчета по лабораторной работе (раздел 4). Это необходимо для фиксации необходимых программных кодов и результатов их выполнения.
2. Открыть программу *ST Visual Develop*. Создать проект или продолжить работу в Вашем ранее созданном проекте.
3. Написать программу настройки двух линий любого порта в режим вывода информации. Осуществить вывод на одну линию сигнала логической 1, а на вторую линию – логического 0. Вывод данных в порт необходимо осуществить в цикле *main* до бесконечного цикла.
4. Подключить отладочный макет и подключить настроенные выводы к светодиодам.

Рис.2. Светодиоды на отладочной плате

5. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Проверить правильность выполнения кода программы. Сделать выводы.
6. Написать программу, которая осуществляет включение и выключение светодиода (любого), подключенного к порту А (в дальнейшем, **все задания в лабораторной выполнять, используя порт А**), в бесконечном цикле без временной задержки. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Пронаблюдать работу программы и сделать выводы.
7. Подключить пробник осциллографа к соответствующему выводу микроконтроллера (предварительно отключив его от светодиода) и пронаблюдать сигнал на выводе микроконтроллера. Зафиксировать полученную осциллограмму и сделать выводы.
8. В предыдущую программу добавить временную задержки после включения и выключения светодиода с использованием цикла for. Например:

for (i=0; i<1000; i++);

При этом необходимо не забыть предварительно объявить переменную *i*. Подключить светодиод к соответствующему выводу микроконтроллера. Выполнить компиляцию программы, сборку и ее

загрузку в микроконтроллер. Подключить пробник осциллографа к соответствующему выводу микроконтроллера, пронаблюдать сигнал на выводе микроконтроллера и сравнить с результатами работы предыдущей программы. Зафиксировать полученную осциллограмму и сделать выводы.

9. Организовать в программе временную задержку в соответствии с вариантом задания (таблица 5). При этом объявить переменную, используемую в цикле задержки, типа *int*.

Таблица 5

Варианты заданий

Вариант	**Значение**
1	100
2	110
3	120
4	130
5	140
6	150
7	160
8	170
9	180
10	190
11	200
12	210
13	220
14	230
15	240

10. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Подключить пробник осциллографа к соответствующему выводу микроконтроллера и пронаблюдать сигнал на выводе микроконтроллера. Зафиксировать осциллограмму и значение длительности импульса сигнала на выводе микроконтроллера.

11. Организовать в программе временную задержку в соответствии с вариантом задания (таблица 5). При этом объявить переменную, используемую в цикле задержки, типа *char*.

12. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Подключить пробник осциллографа к соответствующему выводу микроконтроллера и пронаблюдать сигнал на выводе микроконтроллера. Зафиксировать осциллограмму и значение длительности импульса сигнала на выводе микроконтроллера и сравнить

его с предыдущим вариантом программы. Сделать соответствующие выводы.

13. Организовать в программе временную задержку в соответствии с вариантом задания (таблица 6). При этом объявить переменную типа *int*. Подключить к выводу микроконтроллера светодиод.

Таблица 6

Варианты заданий

Вариант	**Задержка**
1	10000
2	11000
3	12000
4	13000
5	14000
6	15000
7	10500
8	11500
9	12500
10	13500
11	14500
12	15500
13	8500
14	9000
15	9500

14. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Подключить светодиод к линии порта. Пронаблюдать процесс выполнение программы. Сделать выводы.

15. Увеличить задержку в два раза (относительно значения в таблице 6). Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Подключить светодиод к линии порта. Пронаблюдать процесс выполнение мерцания светодиода и сделать выводы.

16. В предыдущей программе установить точку останова на следующей операции после любого из циклов *for*. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Посмотреть, что после запуска программы осуществляется остановка на указанной операции. Также посмотреть и зафиксировать значение переменных и используемых регистров.

17. Увеличить задержку в четыре раза (относительно значения в таблице 6). Удалить точку останова. Выполнить компиляцию

программы, сборку и ее загрузку в микроконтроллер. Запустить программу и пронаблюдать мерцание светодиода. В случае отсутствия мерцания светодиода сделать выводы.

18. Остановив процесс выполнения программы установите точку останова на следующей операции после первого цикла *for*. Программа должна выдать Вам ошибку. Зафиксировать ошибку и сделать выводы о причине.

19. Повторить пункт 17 изменив тип переменной в цикле задержки на *long int*. Сделать выводы.

20. Написать программу мерцания светодиода *LED1* с временными задержкой между изменениями сигналов на линии порта соответствующими выполнению пустого цикла *for* 1 и 499 раз:

Включение светодиода

for (i=0; i<1; i++) { }

Выключение светодиода

for (i=0; i<499; i++) { }

21. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Запустить программу и пронаблюдать мерцание светодиода. Светодиод при этом будет гореть постоянно, так как глаз не воспринимает его мерцание на данной частоте.

22. В предыдущей программе изменить задержки на 50 и 450, соответственно. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Запустить программу и пронаблюдать мерцание светодиода. Сравнить с предыдущим результатом, сделать выводы.

23. В предыдущей программе изменить задержки на 400 и 100, соответственно. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Запустить программу и пронаблюдать мерцание светодиода. Сравнить с предыдущими результатами, сделать выводы.

24. Реализовать программу мерцания светодиода с использованием временной задержки, реализованной в функции. Временная задержка должна быть одинаковая для обоих состояний линии порта (0 и 1) и не превышать 500 циклов *for*. **Если функция временной задержки будет определена после цикла *main*, то необходимо ее объявить в начале программы.** Пример объявления функции с именем *delay*:

void delay(void);

25. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Запустить программу и пронаблюдать яркость свечения светодиода.

26. В предыдущей программе изменить функцию так, чтобы можно было в нее передавать числа типа *int*. Передаваемые значения должны определять временную задержку. Пример объявления функции с именем *delay*:

void delay(int a);

27. Используя данную функцию получить яркое и тусклое свечение светодиода.

28. В предыдущей программе изменить способ переключения состояния линии порта с использованием операций *XOR*. Пример (операция *XOR* нулевого бита порта А с «1»):

PA_ODR ^= 0x01;

29. В предыдущей программе изменить способ переключения состояния линии порта с использованием операций инверсии порта. Пример:

PA_ODR = ~PA_ODR;

30. Реализовать программу плавного изменения яркости светодиодов, согласно варианту в таблице 7. После достижений максимальных или минимальных значений яркости процесс должен повторяться. Время t_0 = 3-5 сек.

Таблица 7

Варианты заданий

Вариант	**Задержка**
1	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно нарастает за время $2t_0$;
2	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно нарастает за время $3t_0$;
3	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно нарастает за время $t_0/2$;
4	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно нарастает за время $t_0/3$;
5	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно спадает за время t_0;
6	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно спадает за время $2t_0$;
7	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно спадает за время $t_0/2$;

8	LED1- яркость плавно нарастает за время t_0; LED2 - яркость плавно спадает за время $t_0/3$;
9	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно нарастает за время $2t_0$;
10	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно нарастает за время $3t_0$;
11	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно нарастает за время $t_0/2$;
12	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно нарастает за время $t_0/3$;
13	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно спадает за время $2t_0$;
14	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно спадает за время $3t_0$;
15	LED1- яркость плавно спадает за время t_0; LED2 - яркость плавно спадает за время $t_0/2$;

31. Выполнить компиляцию программы, сборку и ее загрузку в микроконтроллер. Проверить правильность работы программы.

4. Отчет

1. Титульный лист/
2. Коды программ всех заданий/
3. Результаты выполнения кода программы (скриншоты, фотографии и т.д.). Результаты представляются для тех заданий, где это возможно.
4. Выводы по пунктам заданий.
5. Для пункта 30 необходимо представить алгоритм программы с подробными комментариями.