Лабораторная работа № 13

Тема: Использование таймеров в микроконтроллерах.

Цель работы: Изучение возможностей и применение таймеров микроконтроллера для создания временных задержек.

Таймер микроконтроллера – это цифровой счётчик, осуществляющий подсчёт количества подаваемых на него импульсов.
Источником импульсов для таймера-счётчика могут служить: как тактовые импульсы от внутреннего генератора МК, так и импульсы, подаваемые непосредственно на вход таймера с внешнего источника.

В МК ATmega8 (Atmega16) есть три таймера: два 8-битных – T0 и T2, и один 16-битный – T1.
Счёт ведётся до **255** тактовых импульсов для 8-битных счётчиков и до **65535** импульсов – для 16-битного.
Далее, если не выполняется никаких программных действий, то происходит переполнение счётчика, он сбрасывается в 0 и всё повторяется бесконечное количество раз.

Для каждого таймера можно настроить делитель частоты тактовых импульсов, и, таким образом, заставить его тактироваться не только на основной частоте МК, но и на частотах, находящихся в пропорциях от 1:8 до 1:1024 по отношению к основной. Пропорция эта называется "prescaler" и прописывается в регистре **TCCRx** (Timer/Counter Control Register) посредством установки значений соответствующих битов.
конфигурационные регистры таймеров/счетчиков:
TCCR0 – 0-й таймер/счетчик
TCCR1B – 1-й таймер/счетчик
TCCR2 – 2-й таймер/счетчик

Конфигурационный регистр TCCR

Timer/Counter Control Register - TCCR2

Номер бита	7	6	5	4	3	2	1	0
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20
Read/Write	W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Биты **CS22, CS21, CS20** (Clock Select) – задают для таймера T2 коэффициент предделителя.
Все возможные комбинации состояний этих битов описаны в таблице ниже:

CS22	CS21	CS20	Описание
0	0	0	Источника тактирования нет, таймер остановлен
0	0	1	Тактовая частота МК
0	1	0	Тактовая частота МК/8
0	1	1	Тактовая частота МК/64
1	0	0	Тактовая частота МК/256
1	0	1	Тактовая частота МК/1024
1	1	0	Внешний источник тактовых импульсов

1	1	1	Внешний источник тактовых импульсов

Биты **WGM21, WGM20** (Wave Generator Mode) – определяют режим работы таймера-счетчика T2. Всего их может быть четыре: нормальный режим (normal), сброс таймера при совпадении значения счётного регистра с содержимым регистра сравнения (CTC), два режима широтно-импульсной модуляции (FastPWM и Phase Correct PWM). Все возможные значения описаны в таблице ниже:

WGM21	WGM20	Режим работы таймера/счётчика
0	0	Нормальный режим счётчика (normal)
1	0	Сброс таймера при совпадении регистров OCR2 и TCNT2 (CTC)
0	1	ШИМ с коррекцией фазы (Phase Correct PWM)
1	1	Быстрая ШИМ (Fast PWM)

Биты **COM21, COM20** (Compare Match Output Mode) – определяют поведение вывода OC2. Если хоть один из этих битов установлен в 1, то вывод OC2 перестаёт функционировать как обычный вывод общего назначения и подключается к схеме сравнения таймера счётчика T2. При этом его необходимо настроить как выход. Рассмотрим различные комбинации этих битов:

COM21	COM20	Режим работы вывода OC2
0	0	Вывод OC2 отключён от таймера/счётчика
0	1	Состояние вывода меняется на противоположное при совпадении TCNT2 и OCR2 (только в режимах Normal и CTC)
1	0	На OC2 устанавливается "0" при совпадении TCNT2 и OCR2, и устанавливается "1" при сбросе счётчика
1	1	На OC2 устанавливается "1" при совпадении TCNT2 и OCR2, и устанавливается "0" при сбросе счётчика

Пример:

```
TCCR2=0b01101001; //  режим быстрой ШИМ, тактовая частота равна частоте МК
```

Бит регистра TCCR2 – **FOC2** (Force Output Compare) предназначен для принудительной установки логического уровня на выходе OC2. Он работает только для режимов Normal и CTC. При установке бита FOC2 в единицу состояние выхода меняется в соответствии со значениями битов COM21 и COM20.

Конфигурацию регистра TCCR удобно производить в двоичном коде, т. к. каждый разряд этого кода равен соответствующему разряду регистра. Например, запись:

```
TCCR2 = 0b00011101;
```

означает, что у счётчика выбран режим CTC со сбросом таймера при совпадении регистров OCR2 и TCNT2. Тактовая частота T2 – это рабочая частота МК, делённая на 1024. Состояние вывода OC2 при совпадении меняется на противоположное.
А запись:

```
TCCR2=0b01101001;
```

означает, что счётчик установлен в режим Fast PWM (быстрая ШИМ). Делитель частоты отключён – таймер тикает с тактовой частотой МК. Выход OC2 установлен в состояние логического 0.

Счётный регистр TCNT2

TCNT2 – это такой же 8-битный регистр, как и TCCR2, только все разряды в нём отведены для числа, соответствующего количеству импульсов, посчитанному счётчиком. Когда таймер-счётчик работает, то по каждому импульсу тактового сигнала значение TCNT2 изменяется на единицу. В зависимости от режима работы таймера, счётный регистр может или увеличиваться, или уменьшаться.
Содержимое регистра TCNT2 можно как читать, так изменять посредством записи в него. Запись в регистр используется при необходимости задать его начальное значение.
Когда таймер работает, изменять его содержимое TCNT2 не рекомендуется, так как это блокирует схему сравнения на один такт.

Регистр сравнения OCR2

OCR2 – это также 8-ми разрядный регистр. Его значение в каждом цикле сравнивается со значением счётного регистра TCNT2, и в случае совпадения, заставляет таймер выполнять какие-либо действия, как то: вызывать прерывание, менять состояние вывода OC2 и т. д. в полном соответствии с командами программного кода прошивки.
Значение OCR2 можно как читать, так и записывать.

Регистр флагов разрешения прерываний TIMSK

TIMSK – это общий регистр для всех трёх таймеров Atmega8, 16.

TIMSK (Timer/Counter Interrupt Mask Register)

Номер бита	7	6	5	4	3	2	1	0
	OSIE2	TIOE2	TISIE1	OSIE1A	OSIE1B	TOIE1	OSIE0	TIOE0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Таймер-счётчик T2 может вызывать прерывания при следующих условиях:
1. при переполнении счётного регистра TCNT2,
2. при совпадении значения счётного регистра со значением регистра сравнения OCR2.
При этом в регистре TIMSK для таймера T2 зарезервированы два бита: TOIE2 и OCIE2. Все остальные биты относятся к другим таймерам.

TOIE2	OCIE2	Разрешение прерываний
0	0	Все прерывания запрещены
0	1	Разрешает прерывание по событию совпадение
1	0	Разрешает прерывание по событию переполнение
1	1	Разрешает прерывания по обоим событиям

Необходимо отметить, что прерывания будут работать только тогда, когда в регистре состояния SREG разрешены общие прерывания. Это делается командой в начале программы:

```
sei(); //разрешение глобальных прерываний
```

Причём, в случае наступления прерывания, флаг глобального разрешения прерываний автоматически сбрасывается в 0, запрещая все прерывания, пока не произойдёт выход из обработчика прерывания.

Регистр флагов прерывания таймеров/счётчиков TIFR

TIFR также является общим регистром для всех трех таймеров-счётчиков Atmega8, 16.

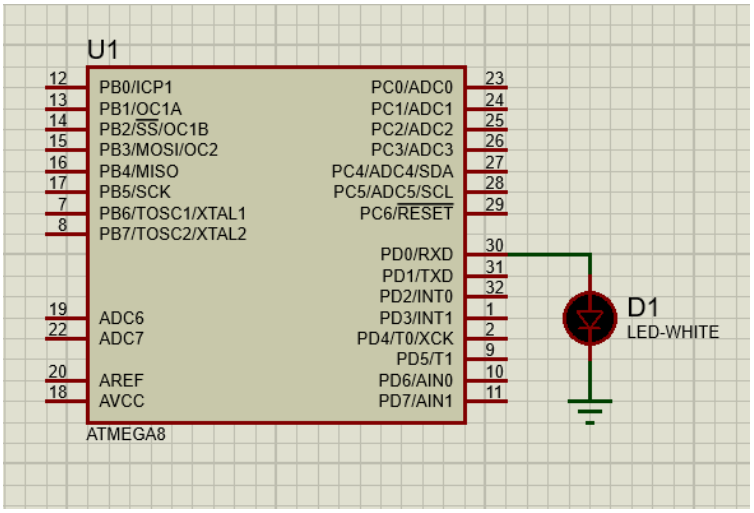
TIFR (Timer/Counter0 Interrupt Flag Register)

Номер бита	7	6	5	4	3	2	1	0
	OSF2	TOV2	ISF1	OSF1A	OSF1B	TOV1	OSF0	TOV0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

TIFR – это регистр флагов. Когда срабатывает какое-то прерывание, то выскакивает статусный флаг, сигнализирующий о том, что произошло то или иное событие. Для таймера T2 – этими событиями являются: переполнение счётного регистра TCNT2 или совпадение счётного регистра с регистром сравнения OCR2.
В эти моменты в регистре устанавливаются следующие флаги:
TOV2 – записывается 1 при переполнении счётного регистра,
OCF2 – записывается 1 при совпадении счётного регистра с регистром сравнения.
Если в эти моменты в регистре TIMSK разрешены прерывания, то микроконтроллер вызовет соответствующий обработчик.
Если прерывания запрещены, то флаг так и будет стоять до тех пор, пока программа не разрешит данный тип прерываний.
При входе в подпрограмму обработки прерывания, соответствующий прерыванию флаг регистра TIFR автоматически сбрасывается в состояние лог. 0.

Практика.

Соберите схему:



Код:

```c
#define F_CPU 2000000
#include <avr/io.h>

int main(void) {
    // Устанавливаем PD0 (пин 0 порта D) как выход
    DDRD |= (1 << PD0);


    TCCR1B = 0b00000101; //1024

    while (1) {
        PORTD ^= (1 << PD0); // Инвертируем состояние пина
        TCNT1 = 0; // Сброс значения таймера
        while (TCNT1 < 1000) {
        // пуза, пока счетчик не дойдет до 1000
        // 1 такт = 1024/2000000  0.0005 сек
        // 1000 тактов = 0.5 сек
        }
    }

}
```

Частота процессора установлена 204800 Гц:
#define F_CPU 2048000

Конфигурация счетчика определяется содержимым регистра **TCCR1:**
TCCR1B = 0b00000101;

CS22, CS21, CS20 – 101, коэффициент предделителя = 1024
WGM21, WGM20 – 00, Нормальный режим счётчика
COM21, COM20 – 00, Вывод OC2 отключён от таймера/счётчика
FOC2 – 0, состояние выхода не меняется

Следующая строка каждый раз после завершения цикла WHILE, меняет состояние пина на противоположное:

PORTD ^= (1 << PD0);

Благодаря использованию оператора XOR с присваиванием. Пример:
00000001 - текущее состояние пина
00000001 - маска
00000000 - результат

00000000 - текущее состояние пина
00000001 - маска
00000001 - результат

XOR gate truth table

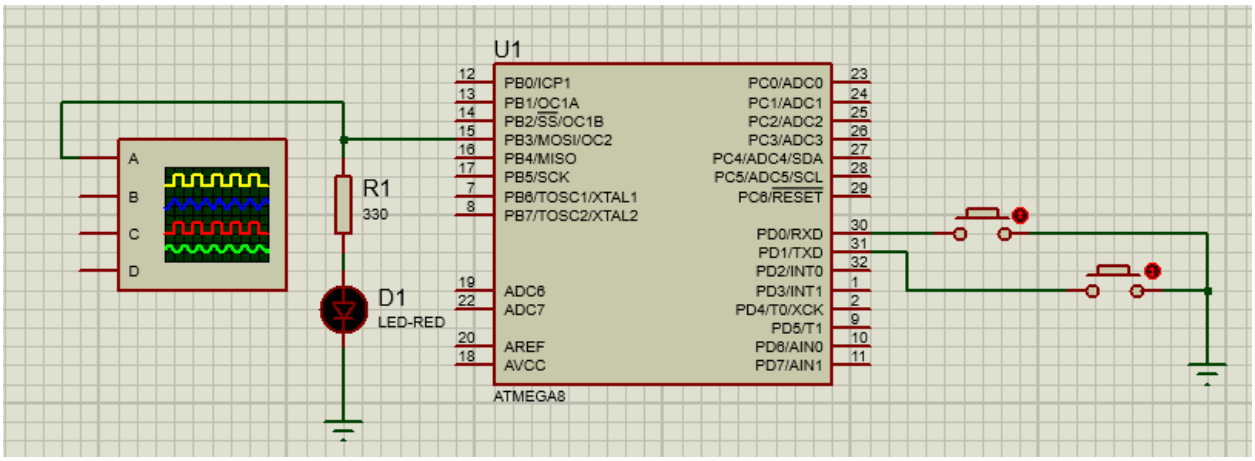
INPUT		OUTPUT
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Задание 1.

Соберите схему в Proteus и напишите код для микроконтроллера. Светодиод должен мигать с частотой примерно 1Гц. Использовать таймер и др. параметры согласно своему варианту:

Вариант	F_CPU, Мц	Таймер/Счетчик
1 (9)	1	T0
2 (10)	2	T2
3 (11)	4	T0
4 (12)	8	T2
5 (13)	1	T0
6 (14)	2	T2
7 (15)	4	T0
8 (16)	8	T2

Задание 2. Соберите схему в Proteus:



Скомпилируйте код для ШИМ:

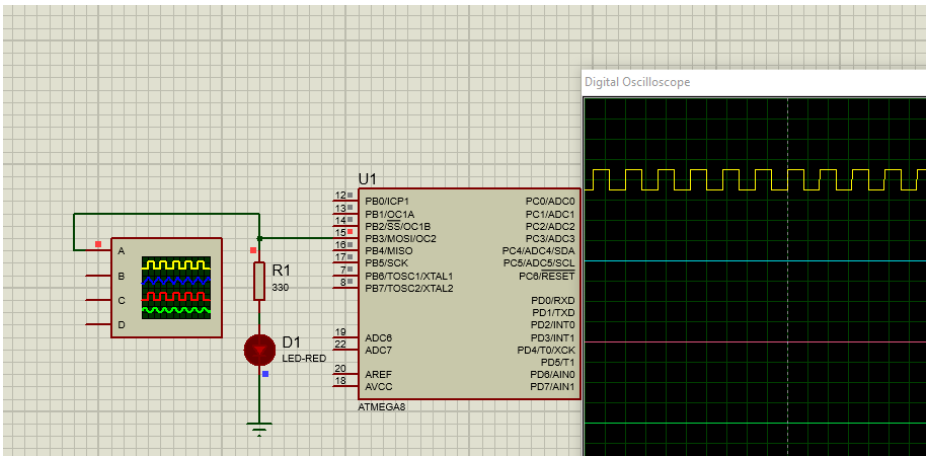
```c
#include <avr/io.h>
#define F_CPU 8000000UL   // частота МК 8 МГц

int main(void)
{   // Начало основной программы
    DDRB |= ( 1 << 3 );   // Конфигурируем вывод порта PB3 как выход (OC2)
    PORTB &= ~(1 << PB3);   // Устанавливаем 0 на его выходе

    TCCR2=0b01101001;
/*   1. Режим быстрой ШИМ (Fast PWM) (6 и 3 разряды TCCR2 - 1,1).
     2. Регистр асинхронного состояния настроен на работу от внутреннего тактового
     генератора, делитель частоты отключён - таймер тикает с тактовой частотой
     (2,1,0 разряды TCCR2 - 0,0,1). Это означает, что частота ШИМ равна 8000000/256 =
     31250Гц.
     3. На выходе PB3 устанавливается "0" при совпадении TCNT2 и OCR2, и "1" при сбросе
     счётчика (5,4 разряды TCCR2 - 1,0)   */
/*   Регистр сравнения. Зададим его начальное значение равным 0. Когда счётчик
     досчитает до значения OCR2, напряжение на выходе PB3(OC2) изменится с 1 на 0   */

OCR2=128; // ширина импульса 50%
while (1)
    {    // начало цикла
   }     // конец цикла
}    // конец программы
```

Запустите симуляцию, убедитесь, что на выходе последовательность импульсов шириной 50%:



Доработайте код так, чтобы ширину импульса (и яркость светодиода) можно было менять кнопками «+» и «-».

Отчет должен содержать:
- скриншоты рабочего поля Proteus с собранными схемами;
- скриншоты осциллограмм;
- листинги исходного кода;
- выводы.