Лекция 4

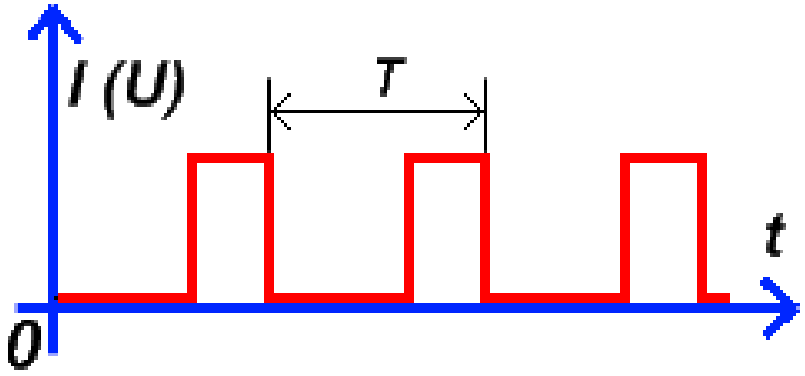
Восьмиразрядный таймер/счетчик. Часть 1.

Таймер/счётчик (восьмиразрядный) – это универсальный восьмиразрядный счётный модуль с двумя независимыми модулями совпадения. Он позволяет формировать заданные промежутки времени (для работы в режиме реального времени), а также может служить генератором периодических сигналов.

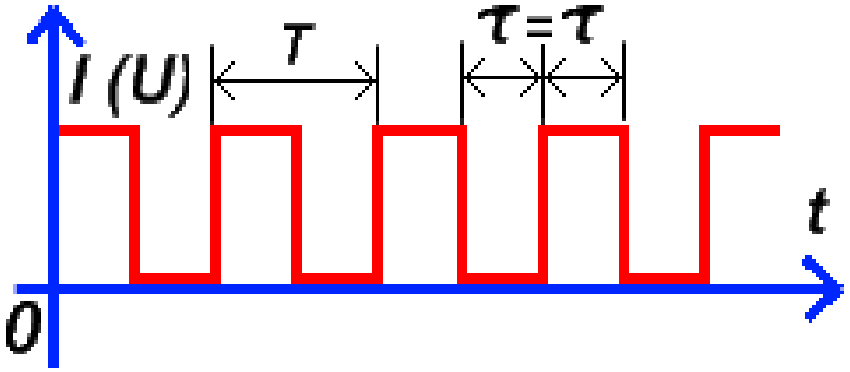
Основные особенности:

- два независимых модуля совпадения;
- сброс таймера при совпадении (авто перезагрузка);
- программно изменяемый период в режиме ШИМ;
- три независимых источника прерывания.

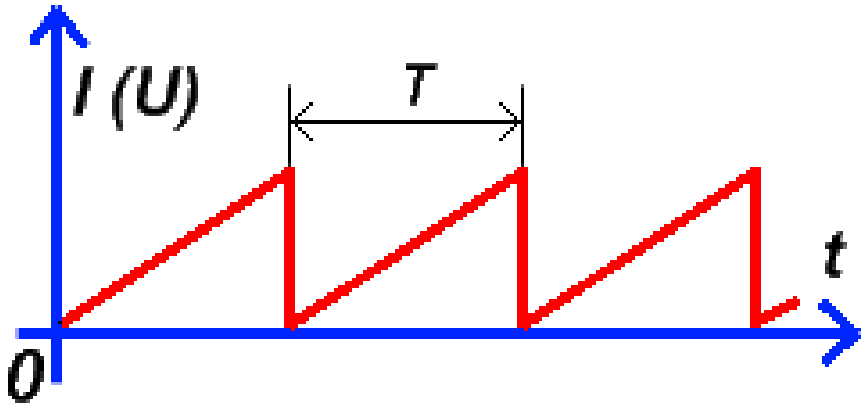
Виды простых периодических сигналов



Прямоугольный периодический сигнал



Меандр – периодический сигнал прямоугольной формы, длительность импульса и длительность паузы которого в периоде равны.



Пилообразный сигнал – это сигнал с постоянной амплитудой и частотой следования импульсов. Имеет пилообразную форму составляющих его импульсов.

Модуль счета

Основой восьмиразрядного таймера/счетчика является реверсивный счетный модуль.

В зависимости от режима работы, каждый импульс тактового сигнала может очищать, увеличивать или уменьшать значение счетчика.

Тактовые импульсы могут быть получены как от внешнего, так и от внутреннего источника, это определяется битами CS02:0 в регистре управления таймера/счетчика T0 – TCCR0B .

Последовательность счета T0 определяется установкой бит WGM02:0 из регистров TCCR0A и TCCR0B.

Счетный регистр таймера/счетчика T0 – TCNT0

Bit	7	6	5	4	3	2	1	0
	TCNT0[7:0]							
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

Когда таймер работает, по каждому импульсу тактового сигнала значение TCNT0 изменяется на единицу.

В зависимости от режима работы таймера счетный регистр может увеличиваться или уменьшаться.

Регистр TCNT0 можно как читать, так и записывать. Последнее можно использовать для задания начального значения.

Модуль совпадения

Основа модуля - восьмиразрядный цифровой компаратор, который постоянно сравнивает регистр TCNT0 с регистрами сравнения OCR0A и OCR0B. В случае равенства значения, находящегося в регистре TCNT0, с каким-нибудь из регистров совпадения (OCR0A или OCR0B) компаратор выдает сигнал о равенстве этих регистров.

Сигнал о равенстве регистров следующим тактом таймера устанавливает соответствующий ему Флаг равенства (сигнализирующий бит) OCF0A или OCF0B. Если установлен Флаг равенства и Прерывание по совпадению значений регистров разрешено, то будет вызвано Прерывание. Флаг равенства очищается автоматически после выхода из подпрограммы обработки прерывания.

Регистр сравнения таймера/счетчика T0 (канал A) – OCR0A

Bit	7	6	5	4	3	2	1	0
	OCR0A[7:0]							
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

OCR0A - восьмиразрядный регистр сравнения канала A.
Его значение постоянно сравнивается со счетным регистром TCNT0. В случае совпадения таймер может выполнять какие-то действия (например, вызывать прерывание).

Регистр сравнения таймера/счетчика T0 (канал B) – OCR0B

Bit	7	6	5	4	3	2	1	0
	OCR0B[7:0]							
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

OCR0B - восьмиразрядный регистр сравнения канала B.
Его значение постоянно сравнивается со счетным регистром TCNT0. В случае совпадения таймер может выполнять какие-то действия (например, вызывать прерывание).

Регистр маски таймера/счетчика T0 – TIMSK

Bit	7	6	5	4	3	2	1	0
	TOIE1	OCIE1A	OCIE1B	–	ICIE1	OCIE0B	TOIE0	OCIE0A
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Общий регистр для таймеров ATtiny2313, он содержит флаги разрешения прерываний.

Таймер T0 может вызывать прерывания при переполнении счетного регистра TCNT0 и при совпадении счетного регистра с регистрами сравнения OCR0A или OCR0B.

Для таймера T0 в регистре TIMSK зарезервированы три бита - это TOIE0, OCIE0A и OCIE0B. Остальные биты относятся к таймеру T1.

Бит 2 - OCIE0B : разрешение прерывания по совпадению в канале B.

Когда бит OCIE0B установлен в единицу, а флаг I регистра состояния также установлен, то прерывание по совпадению в канале B таймера T0 разрешено. Генерация запроса на прерывание происходит в момент совпадения, если бит OCF0B регистра TIFR установлен.

Бит 1 - TOIE0: разрешение прерывания по переполнению таймера/счетчика T0

Если бит TOIE0 установлен в единицу, а также установлен флаг I регистра состояния, то прерывание по переполнению таймера разрешено. Генерация запроса на прерывание происходит в случае переполнения таймера/счетчика T0, если бит TOV0 регистра TIFR установлен.

Бит 0 - OCIE0A : разрешение прерывания по совпадению в канале A.
Когда бит OCIE0A установлен в единицу, а флаг I регистра состояния также установлен, то прерывание по совпадению в канале A таймера T0 разрешено. Генерация запроса на прерывание происходит в момент совпадения, если бит OCF0A регистра TIFR установлен.

Регистр флагов таймера/счетчика T0 – TIFR

Bit	7	6	5	4	3	2	1	0
	TOV1	OCF1A	OCF1B	–	ICF1	OCF0B	TOV0	OCF0A
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Общий для таймеров-счетчиков регистр. Содержит статусные флаги, которые устанавливаются при возникновении событий.

Для таймера T0 - это переполнение счетного регистра TCNT0 и совпадение счетного регистра с регистрами сравнения OCR0A и OCR0B.

Если в регистре TIMSK разрешены прерывания и установлен бит I, то микроконтроллер вызовет соответствующий обработчик.

Флаги автоматически очищаются при запуске обработчика прерывания.

Бит 2 – OCF0B: флаг совпадения в канале B

Бит OCF0B устанавливается, когда возникает совпадение содержимого счетного регистра таймера/счетчика T0 и регистра совпадения OCR0B.

Бит OCF0B аппаратно сбрасывается в ноль в тот момент, когда начинается выполнение соответствующей процедуры обработки прерывания.

Прерывание выполняется, когда флаг I регистра SREG, бит OCIE0B и флаг OCF0B установлены.

Бит 0 – OCF0A: флаг совпадения в канале A

Бит OCF0A устанавливается, когда возникает совпадение содержимого счетного регистра таймера/счетчика T0 и регистра совпадения OCR0A.

Бит OCF0A аппаратно сбрасывается в ноль в тот момент, когда начинается выполнение соответствующей процедуры обработки прерывания.

Прерывание выполняется, когда флаг I регистра SREG, бит OCIE0A и флаг OCF0A установлены.

Бит 1 – TOV0: флаг переполнения таймера/счетчика T0

Бит TOV0 устанавливается в единицу, когда происходит переполнение таймера/счетчика T0.

Бит TOV0 сбрасывается в тот момент, когда начинается выполнение соответствующей процедуры обработки прерывания.

Прерывание выполняется, когда флаг I регистра SREG, бит TOIE0 и флаг TOV0 установлены.

Режимы работы таймера/счетчика T0

1. Режим «Normal».
2. Режим сброса при совпадении (Clear Timer on Compare - CTC).
3. Режим «Быстрый ШИМ» (Fast PWM).
4. Режим «ШИМ, корректный по фазе» (Phase Correct PWM).

Режим работы, т.е. поведение таймера/счетчика T0, определяется комбинацией битов WGM2:0 из регистров TCCR0A и TCCR0B.

Работа схемы вывода сигнала совпадения определяется битами COMx1:0 из регистра TCCR0A.

Режим Normal

В этом режиме направление счета всегда является инкрементирующим и очищение счетчика не выполняется. Счетчик просто переполняется, когда достигает максимального 8-битного значения (0xFF), а затем перезапускается заново со значения (0x00).

В режиме Normal Флаг переполнения TOV0 будет устанавливаться на том же цикле, на котором будет происходить обнуление регистра TCNT0.

Новое значение в Режиме Normal может быть записано в TCNT0 в любое время.

Режим CTC

В CTC-режиме значение регистра счетчика TCNT0 сбрасывается в нуль, если значение TCNT0 соответствует значению, записанному в регистр сравнения OCR0A. Для счетчика значение регистра сравнения OCR0A будет являться максимальным значением, что позволяет настраивать разрешение T0.

Прерывание по переполнению вызывается только если значение регистра сравнения OCR0A равно максимальному значению (0xFF).

Прерывание по совпадению в канале A вызывается одновременно с обнулением счетного регистра.

Прерывание по совпадению в канале B вызывается только в том случае, если значение OCR0B меньше или равно значению OCR0A.

Регистр A управления таймера/счетчика T0 – TCCR0A

Bit	7	6	5	4	3	2	1	0
	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Биты 7:6 – COM0A1:0 : Режимы работы схемы вывода сигнала совпадения (канал A)

Эти биты управляют режимом работы выхода сигнала совпадения (OC0A).

Для Не-ШИМ-режимов

COM0A1	COM0A0	Описание
0	0	Стандартный режим порта. Выход OC0A не подключен
0	1	Переключение OC0A на противоположное в момент совпадения
1	0	Сброс OC0A в момент совпадения
1	1	Установка OC0A в момент совпадения

Биты 5:4 – COM0B1:0 : Режимы работы схемы вывода сигнала совпадения (канал B)

Эти биты управляют режимом работы выхода сигнала совпадения (OC0B).

Для Не-ШИМ-режимов

COM0B1	COM0B0	Описание
0	0	Стандартный режим порта. Выход OC0B не подключен
0	1	Переключение OC0B на противоположное в момент совпадения
1	0	Сброс OC0B в момент совпаденияB
1	1	Установка OC0B в момент совпадения

Биты 1:0 – WGM01:0 : Выбор режима работы генератора сигналов

Совместно с битом WGM02 регистра TCCR0B биты управляют направления счета, выбором максимального значения для счетчика и видом генерируемого сигнала на выходе.

Режимы работы: Normal (счетный режим), режим сброса при совпадении (CTC) и два режима широтно-импульсной модуляции (PWM).

WGM02	WGM01	WGM00	Название режима	Верхний предел
0	0	0	Normal	0xFF
0	1	0	CTC	OCR0A

Регистр B управления таймера/счетчика T0 – TCCR0B

Bit	7	6	5	4	3	2	1	0
	FOC0A	**FOC0B**	**–**	**–**	**WGM02**	**CS02**	**CS01**	**CS00**
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Бит 3 – WGM02 : Выбор режима работы генератора сигналов

Биты 2:0 – CS02:0 : Выбор режима тактового генератора

Каждый таймер может использовать свой собственный коэффициент расчета предварительного делителя частоты.

Выбор источника тактового сигнала для T0

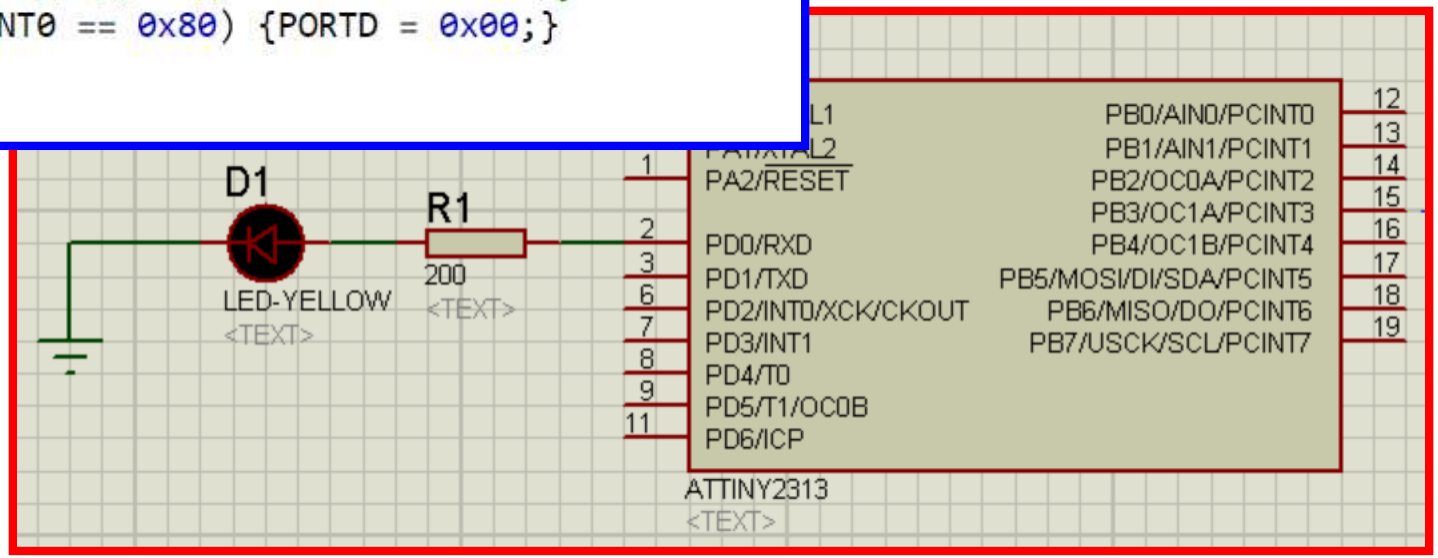
TCCR0B

CS02	CS01	CS00	Описание
0	0	0	Нет источника сигнала (таймер/счетчик остановлен)
0	0	1	ТЧ / 1 (нет предварительного делителя)
0	1	0	ТЧ / 8
0	1	1	ТЧ / 64
1	0	0	ТЧ / 256
1	0	1	ТЧ / 1024
1	1	0	Внешний источник сигнала, вход T0. Синхронизация по заднему фронту.
1	1	1	Внешний источник сигнала, вход T0. Синхронизация по переднему фронту.

ТЧ – тактовая частота микроконтроллера

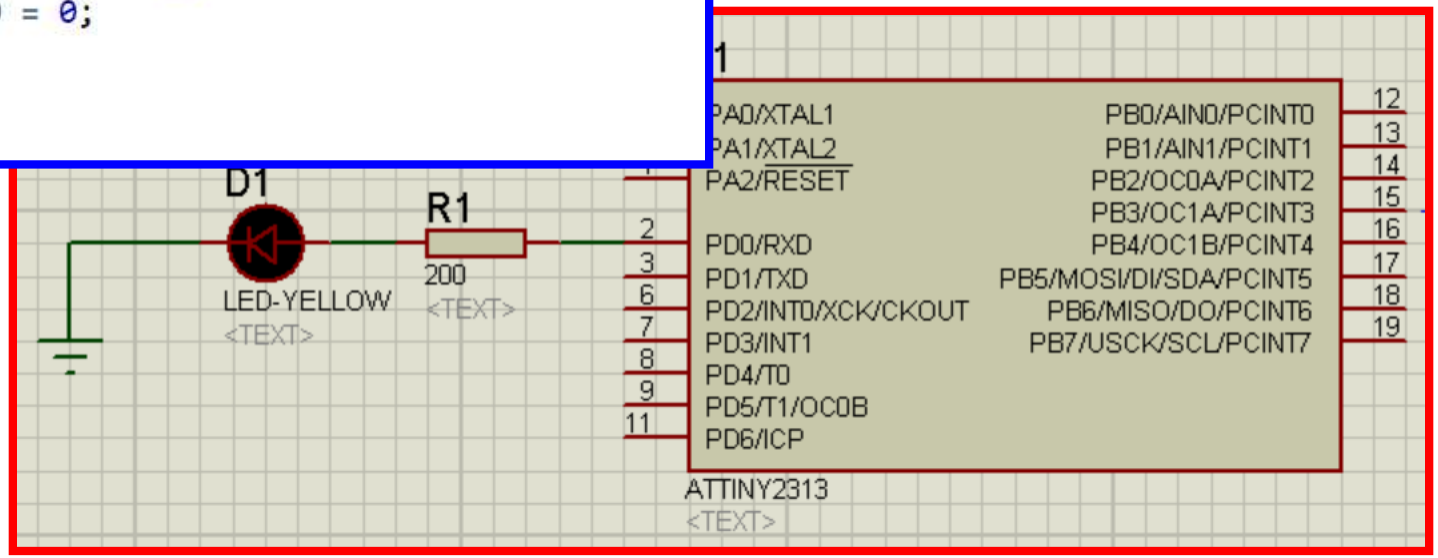
```c
#include <avr/io.h>

int main(void)
{
    // настройка вывода PD0 на вывод данных
    DDRD = 0x01;

    // запуск T/C T0 (предварительный делитель 1024)
    TCCR0B = 0x05;

    while(1)
    {
        // если значение CP TCNT0 = 0, то включаем
        // светодиод, подключенный к выводу PD0
        if (TCNT0 == 0x00) {PORTD = 0x01;}

        // если значение CP TCNT0 = 128, то выключаем
        // светодиод, подключенный к выводу PD0
        if (TCNT0 == 0x80) {PORTD = 0x00;}
    }
}
```

Пример программы, демонстрирующей работу T/C T0 в режиме «Normal»: переключение светодиода (вывод микроконтроллера PD0) через 128 отсчетов счетчика



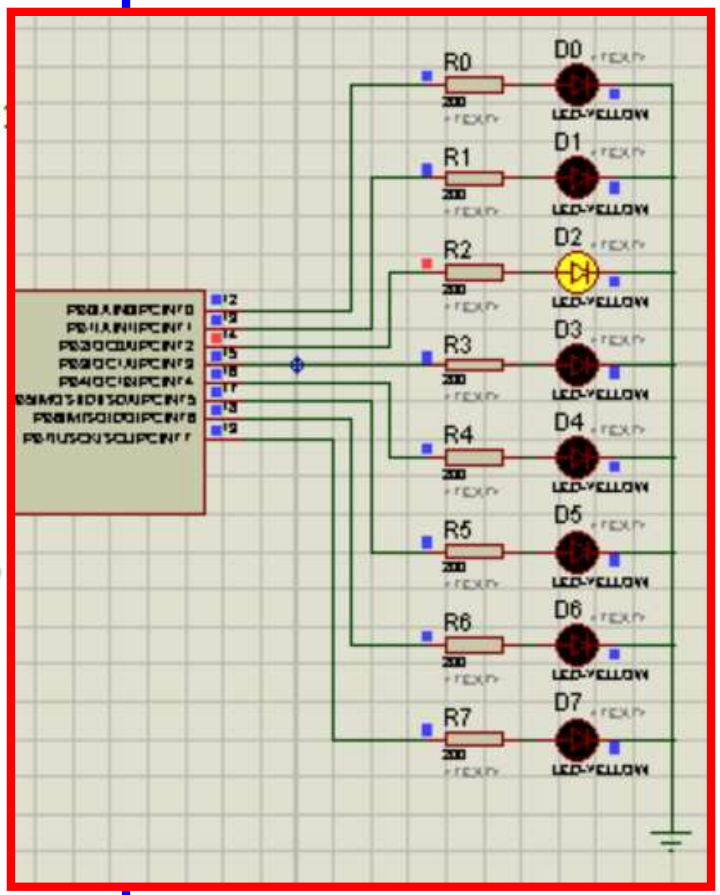
```c
#include <avr/io.h>

int main(void)
{
    // настройка вывода PD0 на вывод данных
    DDRD = 0x01;

    // запуск T/C T0 (предварительный делитель 1024)
    TCCR0B = 0x05;

    while(1)
    {
      // если значение CP TCNT0 = 220,
      // то изменяем состояние вывода PD0
      // (включаем или выключаем светодиод)
      // и обнуляем значение CP TCNT0
      if (TCNT0 == 220)
      {
          PORTD = PORTD^1;
          TCNT0 = 0;
      }
    }
}
```

Пример программы, демонстрирующей работу T/C T0 в режиме «Normal»: переключение светодиода (вывод микроконтроллера PD0) через 220 отсчетов счетчика



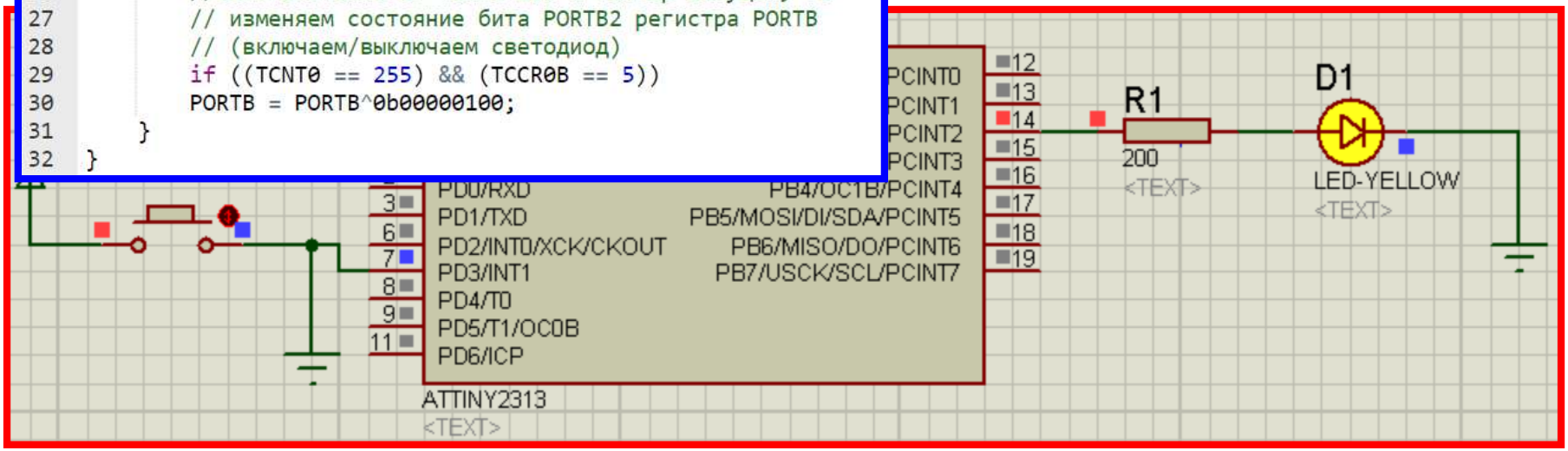
```c
#include <avr/io.h>

int main(void)
{
    // настройка выводов PB0 - PB7 на вывод данных
    DDRB = 0xFF;

    // определение вспомогательного счетчика i
    char i=0;

    // включение светодиода
    PORTB = 1<<i;

    // запуск T/C T0 (предварительный делитель
    TCCR0B = 0x05;

    while(1)
    {
        // если значение CP TCNT0 = 255, то
        if (TCNT0 == 255) {
            // инкрементируем  значение
            //вспомогательного счетчика i
            i++;
            // если значение счетчика i = 8, то
            // записываем в i значение «0»
            i=i%8;
            // переключаем светодиоды
            PORTB = 1<<i;
        }
    }
}
```

Светодиодная дорожка (режим «Normal»)



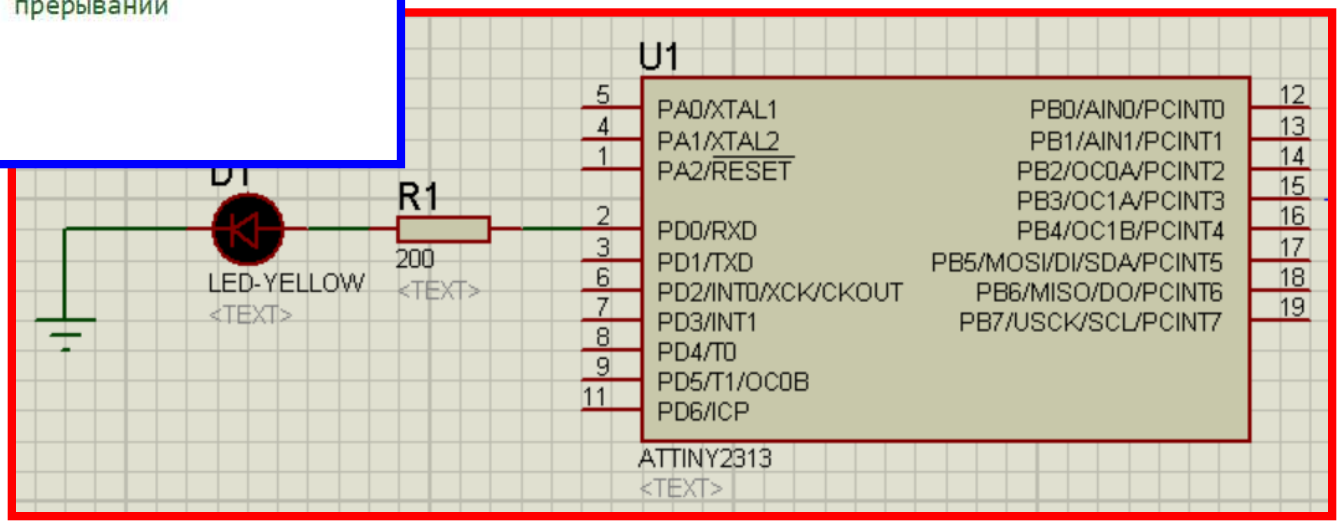
```c
#include <avr/io.h>
#include <avr/interrupt.h> //библиотека прерываний

// подпрограмма обработка прерывания INT0
ISR(INT1_vect) {
    //запуск/остановка таймера
    TCCR0B = TCCR0B^0b00000101;
}

int main(void)
{
    // настройка выводов порта B (PB0 - PB7) на вывод данных
    DDRB = 0xFF;
    // запуск T/C T0 (предварительный делитель 1024)
    TCCR0B = 0x05;
    // разрешение ВП INT1
    GIMSK=0b10000000;
    // настройка вызова прерывания INT1 (передний фронт)
    MCUCR=0b00001100;

    // глобальное разрешение прерываний
    sei();

    while(1)
    {
        //если значение СР TCNT0=255 и таймер запущен, то
        // изменяем состояние бита PORTB2 регистра PORTB
        // (включаем/выключаем светодиод)
        if ((TCNT0 == 255) && (TCCR0B == 5))
        PORTB = PORTB^0b00000100;
    }
}
```

Пример программы, демонстрирующей работу T/C T0 в режиме «Normal» и ВП INT1: включение/выключение мигающего светодиода (вывод микроконтроллера PB2) нажатием кнопки (вывод микроконтроллера INT1)



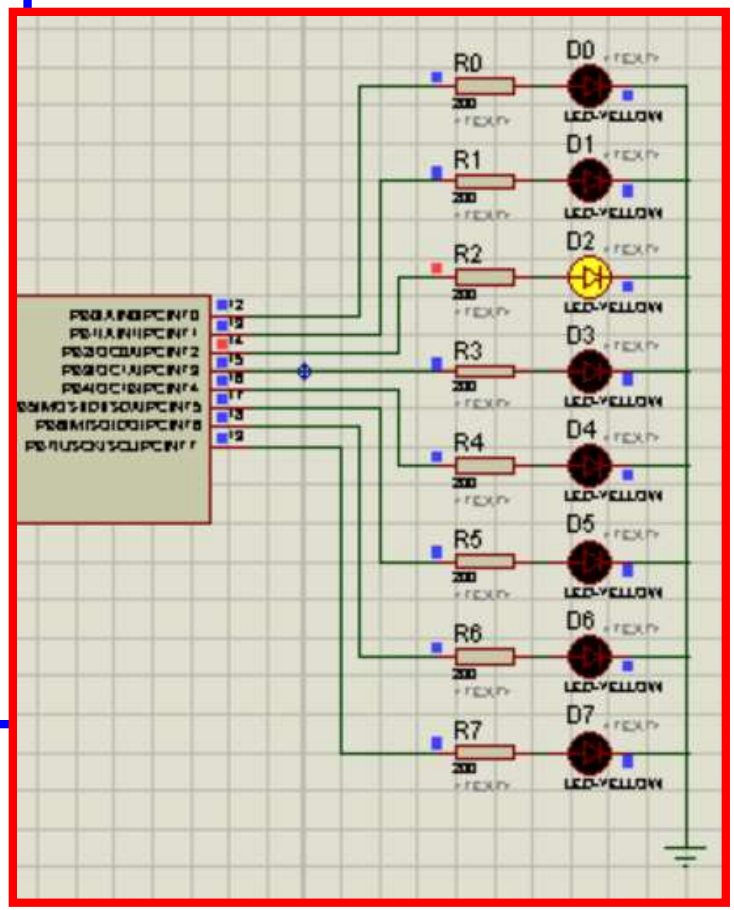
```c
#include <avr/io.h>
#include <avr/interrupt.h> //библиотека прерываний

// подпрограмма обработка прерывания
// по совпадению в канале А
// вызывается при TCNT0=250, поскольку OCR0A=250
ISR(TIMER0_COMPA_vect)
{
    //переключение состояния вывода PD0
    PORTD = PORTD^1;
    // обнуление значения СР TCNT0
    TCNT0=0;
}

int main(void)
{
    // настройка вывода PD0 на вывод данных
    DDRD = 0x01;
    // разрешение прерывания по совпадению в канале А
    TIMSK=0b00000001;
    // установка РС OCR0A для вызова прерывания
    // по совпадению в канале А
    OCR0A = 250;
    // запуск Т/С Т0 (предварительный делитель 1024)
    TCCR0B = 0x05;

    // глобальное разрешение прерываний
    sei();
    while(1)
    {  }
}
```

Пример программы, демонстрирующей работу Т/С Т0 в режиме «Normal» и прерывания по совпадению в канале А: переключение светодиода (вывод микроконтроллера PD0) через 250 отсчетов счетчика

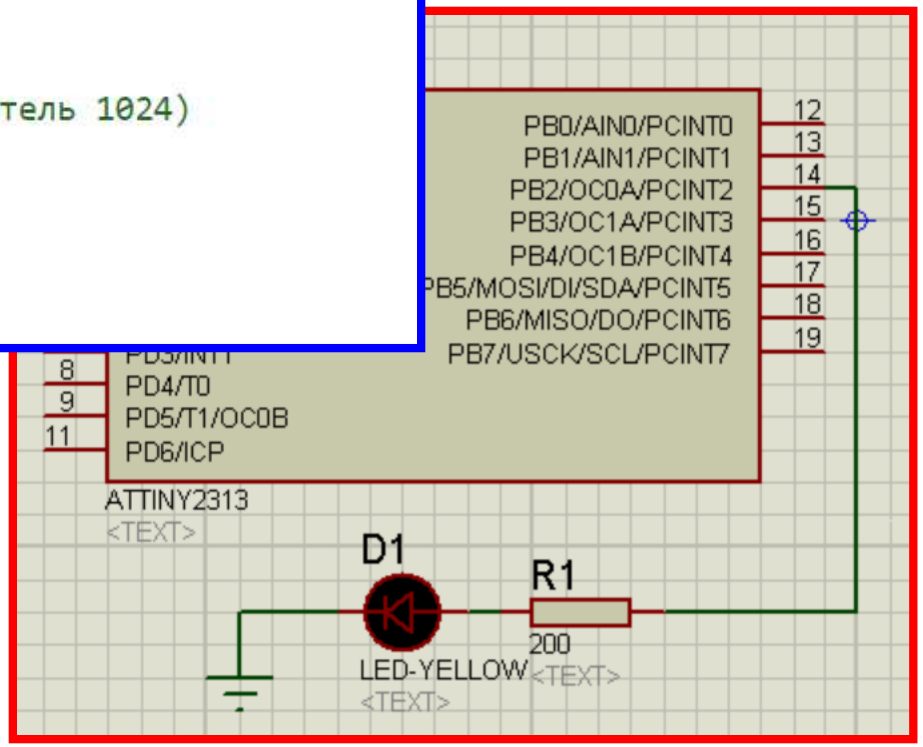


```c
#include <avr/io.h>
#include <avr/interrupt.h> //библиотека прерываний

// определение вспомогательного счетчика i
char i=0;

// подпрограмма обработка прерывания по переполнению
ISR(TIMER0_OVF_vect)
{
    // инкрементирование  значения вспомогательного счетчика i
    i++;
    i=i%8;
    // переключение светодиодов
    PORTB = 1<<i;
}

int main(void)
{
    // настройка выводов PB0 – PB7 (порт  B) на вывод данных
    DDRB = 0xFF;
    // включение светодиода
    PORTB = 1<<i;
    // разрешение прерывания по переполнению
    TIMSK=0b00000010;
    // запуск T/C T0 (предварительный делитель 1024)
    TCCR0B = 0x05;
    // глобальное разрешение прерываний
    sei();

    while(1)
    {  }
}
```

Пример программы, демонстрирующей работу T/C T0 в режиме «Normal» и прерывания по переполнению: светодиодная дорожка

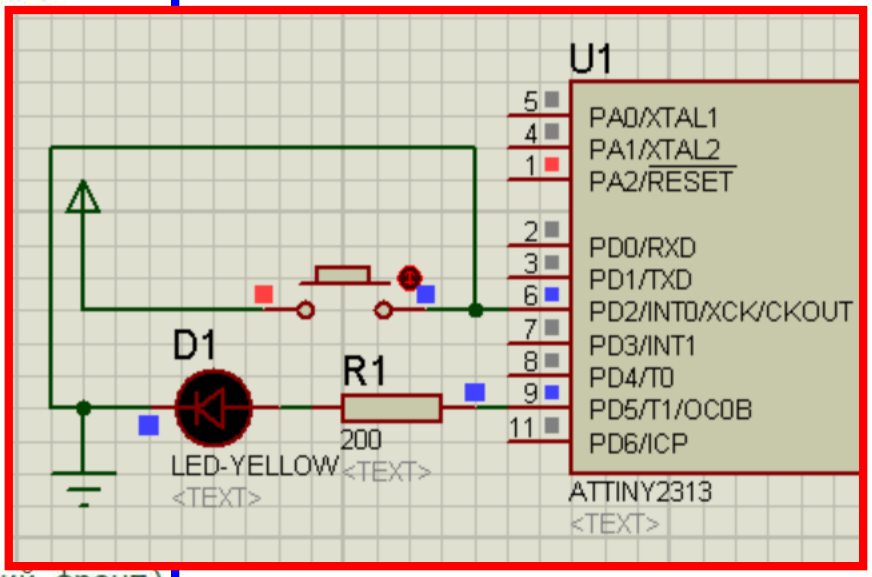


```c
#include <avr/io.h>

int main(void)

{
    // настройка вывода OC0A
    DDRB = 0x04;

    // установка РС OCR0A для схемы
    //вывода сигнала совпадения по каналу А
    OCR0A=100;

    // настройка схемы вывода сигнала совпадения
    // (переключение вывода OC0A  в момент совпадения значений
    // СР TCNT0 и РС OCR0A)
    TCCR0A=0x40;

    // запуск Т/С Т0 (предварительный делитель 1024)
    TCCR0B = 0x05;

    while(1)
    {  }
}
```



Пример программы, демонстрирующей работу схемы вывода сигнала совпадения (канал А) Т/С Т0 в режиме «Normal»: переключение светодиода

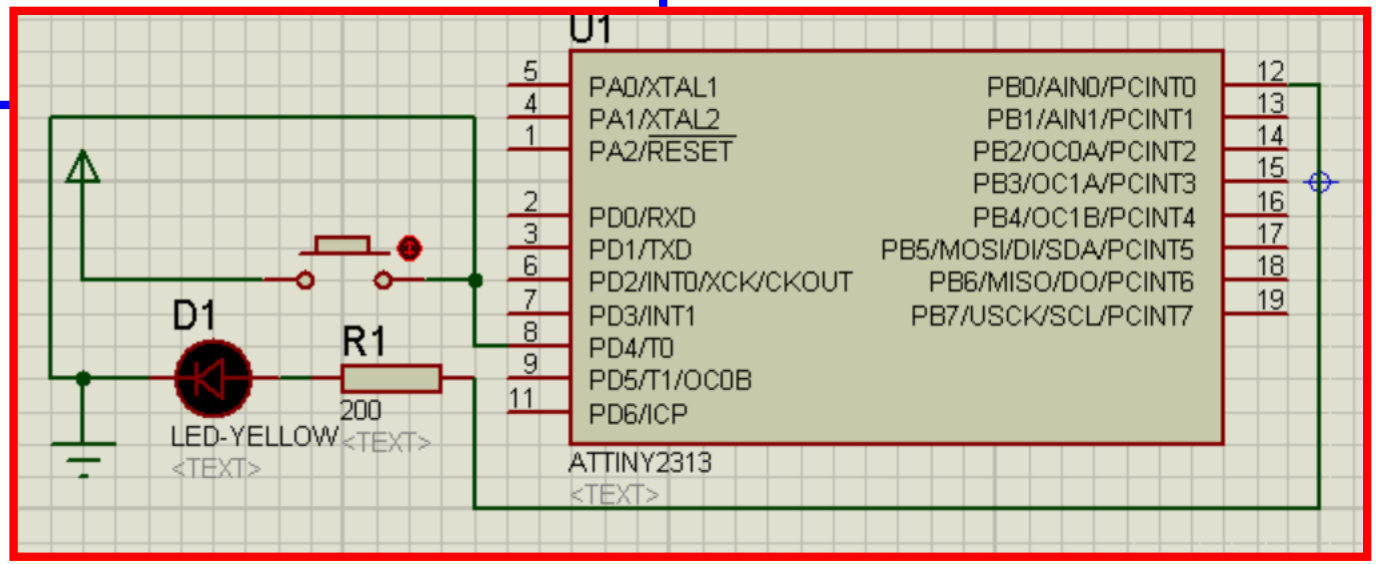
```c
#include <avr/io.h>
#include <avr/interrupt.h> //библиотека прерываний

// подпрограмма обработка прерывания INT0
ISR(INT0_vect)
{
    // запуск/остановка таймера
    TCCR0B = TCCR0B^0b00000101;
}

int main(void)
{
    // настройка вывода OC0B
    DDRD = 0x20;
    // разрешение ВП INT0
    GIMSK=0b01000000;
    // настройка вызова прерывания INT0 (передний фронт)
    MCUCR=0b00000011;
    // установка РС OCR0B для схемы
    //вывода сигнала совпадения по каналу B
    OCR0B=100;
    // настройка схемы вывода сигнала совпадения
    // (переключение OC0B в момент совпадения значений
    // СР TCNT0 и РС OCR0B)
    TCCR0A=0x10;
    //  Т/С Т0 остановлен
    TCCR0B = 0x00;
    // глобальное разрешение прерываний
    sei();
    while(1)
    { }
}
```



Пример программы, демонстрирующей работу схемы вывода сигнала совпадения (канал B) Т/С Т0 в режиме «Normal» и ВП INT0: включение/выключение мигающего светодиода

```c
#include <avr/io.h>

int main(void)
{
    // настройка вывода PB0 на вывод данных
    DDRB = 0x01;

    //внешний источник сигнала (передний фронт)
    TCCR0B = 0x07;

    while(1)
    {
        // если значение CP TCNT0=5 , то
        // изменяем состояние бита PORTB0 регистра PORTB
        // (включаем/выключаем светодиод)
        // и обнуляем значение CP TCNT0
        if (TCNT0 == 5) {
            PORTB = PORTB^1;
            TCNT0 = 0;
        }
    }
}
```

Пример программы, демонстрирующей работу Т/С Т0, настроенного на получение тактовых импульсов от внешнего источника, в режиме «Normal»: переключение состояния светодиода на каждое пятое нажатие кнопки



```c
#include <avr/io.h>

int main(void)
{
    // настройка вывода PD0 на вывод данных
    DDRD = 0x01;

    // максимальное значение СР TCNT0 в режиме СТС
    // определяется значением РС OCR0A
    OCR0A = 200;

    //настройка режима СТС
    TCCR0A = 0x02;

    // запуск Т/С Т0 (предварительный делитель 1024)
    TCCR0B = 0x05;

    while(1)
    {
        //если значение СР TCNT0=0 , то
        // изменяем состояние бита PORTD0 регистра PORTB
        // (включаем/выключаем светодиод)
        if (TCNT0 == 0) {
            PORTD = PORTD^1;

            //задержка
            while (TCNT0!=3) {}
        }
    }
}
```

Пример программы, демонстрирующей работу Т/С Т0 в режиме СТС: переключение светодиода (вывод микроконтроллера PD0) через 200 отсчетов счетчика