← Часы реального времени DS1302 и микроконтроллер AVR

Порты ввода-вывода микроконтроллеров AVR →

Таймеры счетчики микроконтроллеров AVR

21 февраля 2011 г. Просмотров: 168455 Комментарии: 8

Электроника. Схемотехника AVR.Начинающим

AVR Таймер/Счетчик

В микроконтроллерах AVR может быть до 4х таймеров/счетчиков (ТС). Разрядность этих таймеров 8 или 16 бит (т.е. они могут считать до 2^8=256 или до 2^{16}=65536). Обычно их используют для точного формирования временных интервалов, подсчета импульсов на выводах микроконтроллера, формирования последовательности импульсов. Таймеры способны вырабатывать запросы на прерывания, при этом освобождая процессор от необходимости опроса состояния таймеров. В данной статье мы разберем работу таймеров и прерываний, которые они могут вырабатывать, на примере 16-ти битного таймера/счетчика1 (ТС1) микроконтроллера ATmega8. Всего у этого МК три таймера - два 8ми битных (ТС0, ТС2) и один 16 битный (ТС1). Взглянем на таблицу векторов прерываний МК mega8 - 7 прерываний связаны с таймерами микроконтроллера, из них 4 связаны с таймером/счетчиком1 (ТС1). Давайте разберемся, что же может этот таймер, какие регистры им управляют и что в них нужно записать, чтобы настроить таймер как нам нужно. Для начала рассмотрим все регистры ТС1 и за что какой бит отвечает, а потом рассмотрим какие-нибудь простые примеры по настройке таймера. Начнем с регистров управления таймером.

TCCR1A - регистр управления А

Bit	7	6	5	4	3	2	1	0	
	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	W	W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Биты 7:6 - COM1A 1:0: контролируют поведение выхода **OC1A** (см. Таблицу 1).
Биты 5:4 - COM1B 1:0: контролируют поведение выхода **OC1B** (см. Таблицу 1).

Таблица 1 Состояние выходов OC1A и OC1B по событию сравнения по каналу A или B

COM1A1/ COM1B1	COM1A0/ COM1B0	Описание состояния выхода
0	0	Выключен
0	1	Переключение уровня на противоположный
1	0	Переключение в "0"
1	1	Переключение в "1"

Биты 3:2 - FOC1A, FOC1B: служат для принудительного изменения состояния выходов OC1A и OC1B.
Биты 1:0 - WGM11, WGM10: служат для настройки ТС1 для работы в качестве широтно-импульсного модулятора (ШИМ). В режиме ШИМ состояние выходов OC1A и OC1B будет отличаться. В этой статье не буду ничего писать про режим ШИМа, а то будет куча информации, которую трудно переварить. Что такое ШИМ опишу как-нибудь в другой раз.

TCCR1B - регистр управления В

Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Бит 7 - ICNC1: подавление дребезга на входе **ICP1**. Если бит установлен, то определение события на входе ICP1 происходит с задержкой в 4 машинных цикла (см. дальше по тексту).
Бит 6 - ICES1: выбор фронта срабатывания прерывания по захвату. Если установлен - на растущем фронте, если сброшен - на падающем фронте.
Бит 5 - не используется
Биты 4:3 - WGM1 3:2: для настройки ШИМа.
Биты 2:0 - CS1 2:0: выбор тактирования TC1 (Таблица 2).

Таблица 2 Выбор тактирования таймера 1

CS12	CS11	CS10	Описание
0	0	0	Нет источника (таймер выключен)
0	0	1	CLK
0	1	0	CLK/8
0	1	1	CLK/64
1	0	0	CLK/256
1	0	1	CLK/1024
1	1	0	Внешний источник тактирования на T1. Падающий фронт
1	1	1	Внешний источник тактирования на T1. Растущий фронт

Ну, настроили мы таймер и что? А дальше с частотой, которую мы выбрали в регистре TCCR1B счетчик таймера начинает считать и записывать значение счетчика в регистры **TCNT1H** и **TCNT1L** - старший и младший байт счетного регистра. При достижении **TCNT1** значения 2^{16} счетчик переполняется и сбрасывается, и начинает счет заново. В этот регистр мы также можем записать какое-нибудь значение, с которого мы хотим, чтобы наш счетчик стартовал. Для 16-битной операции записи, **старший байт** должен быть записан **первым**. **Младший** - **вторым**. Для операции 16-битного чтения, младший байт должен быть прочитан первым, а содержимое старшего байта считывается вторым. Если мы настроили изменение состояния входа OC1A или OC1B, тогда значение счетного регистра сравнивается каждый раз со значением регистров **OCR1A** и **OCR1B** - регистры сравнения. Каждый из этих регистров состоит из двух байт (например, **OCR1AH** и **OCR1AL**). Мы можем записать в эти регистры нужное нам значение и по совпадению значения регистра счетчика с регистром сравнения будет происходить нужное нам изменение на выходах OC1A и OC1B. Также есть регистр захвата входа - **ICR1** (**ICR1H** и **ICR1L**). Значение TCNT1 в этот регистр записывается при наступлении события на входе ICP1.
Счет и изменение состояния ножек МК это уже хорошо, но также этот таймер может при определенных событиях генерить прерывания. Как мы уже видели из таблицы векторов прерываний у TC1 есть 4 вектора прерывания - прерывание по захвату, прерывание по совпадению A, прерывание по совпадению B, прерывание по переполнению (указаны в порядке уменьшения приоритета). Рассмотрим регистры, настраивая которые можно управлять прерываниями TC1.

TIMSK - регистр маски прерываний таймеров/счетчиков

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	–	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Бит 7 - OCIE2: прерывание по совпадению TC2
Бит 6 - TOIE2: прерывание по переполнению TC2
Бит 5 - TICIE1: прерывание по захвату TC1
Бит 4 - OCIE1A: прерывание по совпадению A TC1
Бит 3 - OCIE1B: прерывание по совпадению B TC1
Бит 2 - TOIE1: прерывание по переполнению TC1
Бит 1 - не используется
Бит 0 - TOIE0: прерывание по переполнению TC0
Если соответствующий бит установлен в "1" и бит **I** (7-й бит) регистра состояний **SREG** установлен в "1", тогда соответствующее прерывание будет срабатывать.

TIFR - регистр флагов прерываний таймеров/счетчиков

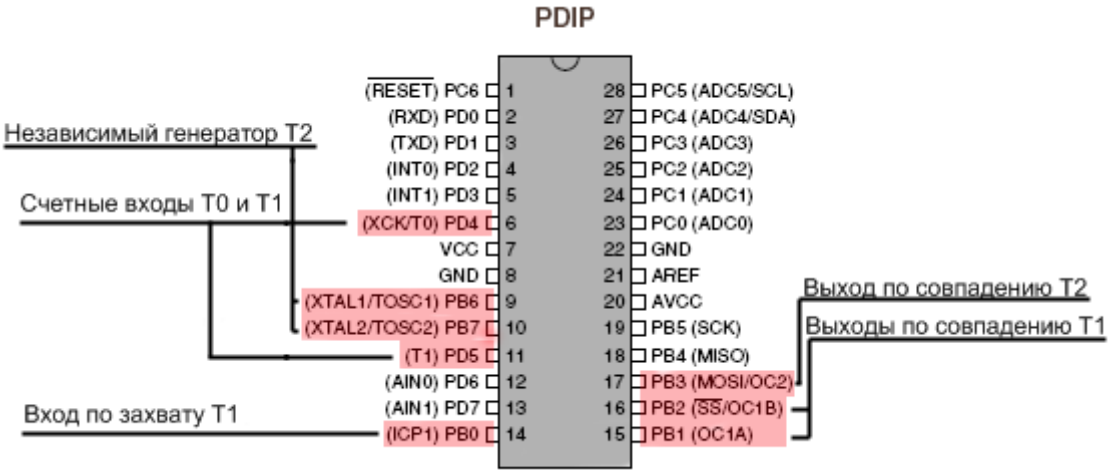
Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	–	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Флаги соответствуют прерываниям в регистре TIMSK. Устанавливаются в "1" при выполнении условий соответствующего прерывания.

Вот и вся теория связанная с TC1. Остальные таймеры настраиваются аналогичным образом. Теперь мы можем обобщить и записать какими возможностями обладает этот таймер:

- Счетчик 16-ти битный (16 битный ШИМ)
- Два независимых выхода, срабатывающих по совпадению
- Один вход по захвату события (растущий или падающий фронт) и подавление дребезга на этом входе
- Тактирование от встроенного тактового генератора или внешний источник тактирования таймера (T1)
- 4 независимых прерывания

Есть и другие возможности, но о них не в этот раз. На рисунке отметил ножки связанные с таймерами/счетчиками МК.



Теперь давайте придумаем себе какие-нибудь условия работы TC1 и настроим его соответствующим образом, записав нужные значения в регистры таймера.
Итак, наш МК тактируется от внешнего кварца с частотой 2.048МГц, мы хотим, чтобы на выходе OC1A был прямоугольный сигнал с периодом 1сек. Как мы дальше думаем? Нам нужно чтобы на выходе OC1A при достижении какого-то значения в счетчике менялся уровень на противоположный каждые 0.5сек. Чтобы уровень менялся на противоположный в регистр TCCR1A нужно записать 0x40 (см. таблицу 1). Теперь посчитаем до какого значения должен считать счетчик. Т.к. время 1сек довольно большая то выберем делитель тактовой частоты 1024, и запишем соответственно в регистр TCCR1B значение 0x05 (см. таблицу 2). Теперь наш таймер считает с частотой 2048000/1024=2000Гц, т.е. за 1сек TC1 досчитает до 2000, а нам нужно значение до которого он досчитает за 0.5сек. Это значение равно 2000/2=1000 (0x03E8). Именно это значение запишем в регистр OCR1A.Теперь TC1 досчитает до 1000, его значение совпадет с OCR1A и он изменит состояние выхода OC1A на противоположное и пойдет считать дальше, что нам совсем не нужно. Чтобы этого не произошло запустим прерывание по совпадению A (TIMSK=0x10) и в обработчике будем сбрасывать регистр TCNT1. Вот собственно и все настройки. Эта программа в WinAVR будет выглядеть вот так

```
#include <iom8.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#define F_CPU 2048000
ISR(TIMER1_COMPA_vect)//обработчик прерывания по совпадению A
{
TCNT1H=0;//обнуляем регистр TCNT1
TCNT1L=0;
}
int main()
{
DDRB=0x02;//настраиваем OC1A как выход
PORTB=0x00;
TCCR1A=0x40;//при совпадении уровень OC1A меняется на противоположный
TCCR1B=0x05;//CLK/1024
OCR1AH=0x03;//записываем в регистр OCR1A 1000
OCR1AL=0xE8;
TIMSK = 0x10;//разрешаем прерывание по совпадению
sei();//разрешаем прерывания глобально
while(1);
```

```
}
```

Обновлено 27.03.2011 - ШИМ микроконтроллеров AVR

ЕЩЕ ЗАПИСИ ПО ТЕМЕ

- Аналоговый компаратор микроконтроллеров AVR
- Широтно-Импульсная Модуляция (ШИМ, PWM) микроконтроллеров AVR
- SFIOR - регистр специальных функций ввода вывода МК AVR