:: Timer1 ::

This library is a collection of routines for configuring the 16 bit hardware timer called Timer1 on the ATmega168/328. There are 3 hardware timers available on the chip, and they can be configured in a variety of ways to achieve different functionality. The development of this library began with the need for a way to quickly and easily set the PWM period or frequency, but has grown to include timer overflow interrupt handling and other features. It could easily be expanded upon or ported to work with the other timers.

The accuracy of the timer depends on your processor speed and the frequency. Timer1's clock speed is defined by setting the prescaler, or divisor. This prescale can be set to 1, 8, 64, 256 or 1024.

For 16MHz:

Prescale	Time per counter tick	Max Period
1	0.0625 uS	8.192 mS
8	0.5 uS	65.536 mS
64	4 uS	524.288 mS
256	16 uS	2097.152 mS
1024	64uS	8388.608mS

In general:

- Max Period = (Prescale)*(1/Frequency)*(2^17)
- Time per Tick = (Prescale)*(1/Frequency)

License: GPLv2.0

Download -> TimerOne Google Code download (http://code.google.com/p/arduino-timerone/downloads/list)

To install, simply unzip and put the files in Arduino/hardware/libraries/Timer1/

A separately maintained and updated copy of TimerOne (https://github.com/PaulStoffregen/TimerOne) is also available, supporting more hardware and with optimizations for more efficient code.

:: Timer3 ::

Note that timer1 can be used on a Mega but does not support all three output pins OCR1A, OCR1B & OCR1C. Only A & B are supported. OCR1A is connected to pin 11 of the Mega and OCR1B to pin 12. Using one of the three calls that specify a pin, 1 will map to pin 11 on the Mega and 2 will map to pin 12. Timer3 has only been tested on the Mega.

License: Creative Commons

Download -> TimerThree.zip (http://playground.arduino.cc/uploads/Code/TimerThree.zip)

To install, simply unzip and put the files in Arduino/hardware/libraries/Timer3/

Revisions:

- Modified June 2011 adding read method and starting google code group - Lex Talionis
- Modified October 2009 adding TimerThree.zip for ATMega1280 / Arduino Mega support - keshka
- Modified June 2009 to fix a bug in setPeriod() that caused the timer to stop - thanks Michael Polli
- Modified March 2009 for ATmega 328 support - thanks Jérôme Despatis
- Created June 2008 - Jesse Tane

Documentation - Simplified descriptions of what the most important routines do

---

initialize(period)
You must call this method first to use any of the other methods. You can optionally specify the timer's period here (in microseconds), by default it is set at 1 second. Note that this breaks analogWrite() for digital pins 9 and 10 on Arduino.

setPeriod(period)
Sets the period in microseconds. The minimum period or highest frequency this library supports is 1 microsecond or 1 MHz. The maximum period is 8388480 microseconds or about 8.3 seconds. Note that setting the period will change the attached interrupt and both pwm outputs' frequencies and duty cycles simultaneously.

pwm(pin, duty, period)
Generates a PWM waveform on the specified pin. Output pins for Timer1 are PORTB pins 1 and 2, so you have to choose between these two, anything else is ignored. On Arduino, these are digital pins 9 and 10, so those aliases also work. Output pins for Timer3 are from PORTE and correspond to 2,3 & 5 on the Arduino Mega. The duty cycle is specified as a 10 bit value, so anything between 0 and 1023. Note that you can optionally set the period with this function if you include a value in microseconds as the last parameter when you call it.

attachInterrupt(function, period)
Calls a function at the specified interval in microseconds. Be careful about trying to execute too complicated of an interrupt at too high of a frequency, or the CPU may never enter the main loop and your program will 'lock up'. Note that you can optionally set the period with this function if you include a value in microseconds as the last parameter when you call it.

setPwmDuty(pin, duty)
A fast shortcut for setting the pwm duty for a given pin if you have already set it up by calling pwm() earlier. This avoids the overhead of enabling pwm mode for the pin, setting the data direction register, checking for optional period adjustments etc. that are mandatory when you call pwm().

detachInterrupt()
Disables the attached interrupt.

disablePwm(pin)
Turns PWM off for the specified pin so you can use that pin for something else.

read()
Reads the time since last rollover in microseconds.

Method Detail - calling conventions for all public methods

```
void initialize(long microseconds=1000000);
void start();
void stop();
void restart();
unsigned long read();
void setPeriod(long microseconds);
```

```
void pwm(char pin, int duty, long microseconds=-1);
void setPwmDuty(char pin, int duty);
void disablePwm(char pin);
void attachInterrupt(void (*isr)(), long microseconds=-1);
void detachInterrupt();
```

Example - Sets up PWM output on pin 9 with a 50% duty cycle, and attaches an interrupt that toggles digital pin 10 every half second.

```
1.  /*
2.   *  Timer1 library example
3.   *  June 2008 | jesse dot tane at gmail dot com
4.   */
5.
6.  #include "TimerOne.h"
7.
8.  void setup()
9.  {
10.   pinMode(10, OUTPUT);
11.   Timer1.initialize(500000);        // initialize timer1, and set a 1/2 second period
12.   Timer1.pwm(9, 512);               // setup pwm on pin 9, 50% duty cycle
13.   Timer1.attachInterrupt(callback);  // attaches callback() as a timer overflow interrupt
14. }
15.
16. void callback()
17. {
18.   digitalWrite(10, digitalRead(10) ^ 1);
19. }
20.
21. void loop()
22. {
23.   // your program here...
24. }
25.
```

[Get Code] (http://playground.arduino.cc/Code/Timer1?action=sourceblock&num=1)

Other sketches/snippets that use Timer1:

- DirectDriveLEDMatrix (http://playground.arduino.cc/Main/DirectDriveLEDMatrix#Timer1)
- DirectDrive88884Digit7SegmentDisplay (http://playground.arduino.cc/Main/DirectDrive88884Digit7SegmentDisplay)
- ReadReceiver (http://playground.arduino.cc/Code/ReadReceiver)
- ACPhaseControl (http://playground.arduino.cc/Code/ACPhaseControl)
- Suzuki V-Strom Motorcycle display (http://code.google.com/p/stromputer/)

Share

NEWSLETTER

Enter your email to sign up →

(https://twitter.com/arduino) (http://www.facebook.com/official.arduino)

(https://plus.google.com/+Arduino) (http://www.flickr.com/photos/arduino_cc)

(http://youtube.com/arduinoteam)