Embedded Systems: Timers

Professor: M Campisi

TAs: Abhimanyu (Abhi) Ghosh

Sravani Bhuma Koppula

Overview

- Admin
  - Github, Wiki
  - Office hours
  - Announcements
  - Brief intro to Git
  - Coding “Best Practices” & use in industry
- Wrap-up of C programming concepts
- Embedded systems and time
  - What is “Real-time” and when do we care?
- The humble timer
  - Applications, et. al.
- Clocks
- Embedded “Hello World” Revisited
  - Can we do this differently? :)
- PWM
  - Hardware to the rescue!
- PWM and relation to Analog
- Putting it together: Analog I/O, Hardware-In-The-Loop testing
  - Sensors everywhere
- Q&A

Github, Wiki

- Links posted to NYU Classes
- We have a GitHub page!
  - https://github.com/Aghosh993/EmbeddedSystems/
  - Code samples, snippets
  - Recitation presentations (including this one!)
  - Useful as a reference later on
- There is a Wiki!
  - https://github.com/Aghosh993/EmbeddedSystems/wiki
  - Basic course info
  - Office hours, availability, TA calendar links
  - Shall be appended with further info, articles, et. al. as semester progresses, so check back frequently!

Office Hours

- Abhi: (Almost) always at LC 029 in Dibner Building
  - Technically accessible 24/7/365
  - Take stairs down to basement, make left, right, right
    - Or follow the left wall for a while...
  - Should end up at a lab full of robots
  - Highly prefer email or chat prior to meeting
    - Contact info on Wiki
- Sravani: Email to set up appointment and figure out location
  - Contact info & calendar also on Wiki

Announcements

Brief Intro to Git

- What is Git?
  - Trivia: Similar systems?
  - Trivia: Creator?
- Rationale for use
  - Software dev happens in teams
  - Asynchronous process
  - Need to keep different “branches” sep. and mark/tag versions
  - Rapidly view changes between versions as diffs
  - Merge codebase from different teams when ready to ship

Brief Intro to Git (Cont’d)

- Usage
  - Linux: git is typically in package repository
    - Ubuntu: “sudo apt-get install git”
    - Arch: “pacman -Syy && pacman -S git”
    - Others: Read the man pages :)
  - Windows:
    - TortoiseGit is an excellent GUI front-end with Windows Explorer integration
    - Plenty of online installation guides
    - Git shell available for power users who want more “linux-y” experience
- To “clone” repo:
  - “git clone <repo>”
    - “git clone https://github.com/Aghosh993/EmbeddedSystems.git”

Coding “Best Practices”

- 70’s and earlier
  - Hardware WAS expensive, software was... unaccounted for
    - Exhibit A: Software delays in Apollo almost cost the US the race to the Moon
- Now...
  - Hardware is cheap
  - Software is the new “top expense” in projects
  - SoftEng a whole new field
- Think carefully before making extremely machine-dependent optimizations
  - Machine time = <<<$ in extra electricity or cost of small HW upgrade
  - Programmer time = >>>$$$ in misery, salary and gray hair spent debugging/porting
- Macros, #defines important
- Structuring of code into modules of source and header files very important in large projects
  - Structure code into portions that are HW-dependent and those that are HW-independent
  - HW-dependent code frequently called the HAL – Eases porting process to diff. platform
- Pick the right toolchain, optimize b/w setup time and time saved in easier usage or some other metric
- Don’t listen to the bean-counters

Coding “Best Practices” (Cont’d)

- “Everyone knows that debugging is twice as hard as writing a program in the first place. So if you’re as clever you can be when you write it, how will you ever debug it?”
  - Brian W. Kernighan (Co-author, *The C Programming Language*)
- Most mere mortals aren’t very clever...
- Comments and “good code”
  - Comments should COMPLEMENT, not REPLACE understandable code
  - “int magic, tmp, temp, foo; //These variables hold values... “
    - No...
    - Surefire way to get poor performance reviews and eventually be fired
- Unit-testing, HITL
- More advanced: Doxygen
  - Auto-generate HTML, PDF docs of software based on structured comments
- Even More Advanced(TM): Coding standards
  - MISRA, DO-178b, et. al.
  - If you’re coding to these standards, your mistakes will follow your clients to their graves :)

Wrap-up of C programming concepts

- Bitwise math
  - AND, OR, XOR
- Two’s complement, conversions of types
  - Trivia: Why do we care?
- Enums, unions re-visited
- Q&A

Embedded systems and time

- Time is important
  - Especially so in Embedded Systems
  - Frequently a variable that actions are based on
    - Ex: Measure a sensor every X seconds and update some actuator value based on a set of calculations
- Real-time:
  - “Hard”-RT: Airliner controls, missile guidance system, car airbag deployment system
  - “Soft”-RT: Infotainment system, average user desktop
  - Everything is ultimately “hard”-RT in the sense that eventually the response characteristic will be deemed unacceptable by the user (i.e. people killed in a crash, user throwing out computer)

The Humble Timer

- Variety of ways to create delays, keep track of time
  - Why not just a busy loop?
  - External time reference? Trivia!
- Timers to the rescue!
  - Processors already use some kind of timebase, known as crystal or oscillator
  - Timer uses the transitions in the waveform of oscillator/other clock source to increment a counter
  - Since timebase (crystal, oscillator, et. al.) has a well-defined period, the counter will also count up/down at a very well-defined rate
    - We can use this in our app!
  - Many HW methods to take various actions based on counter and a few other regs
    - Toggle a pin
    - Take some user-defined action
    - Sample a pin and use it to trigger reset/stop of counter

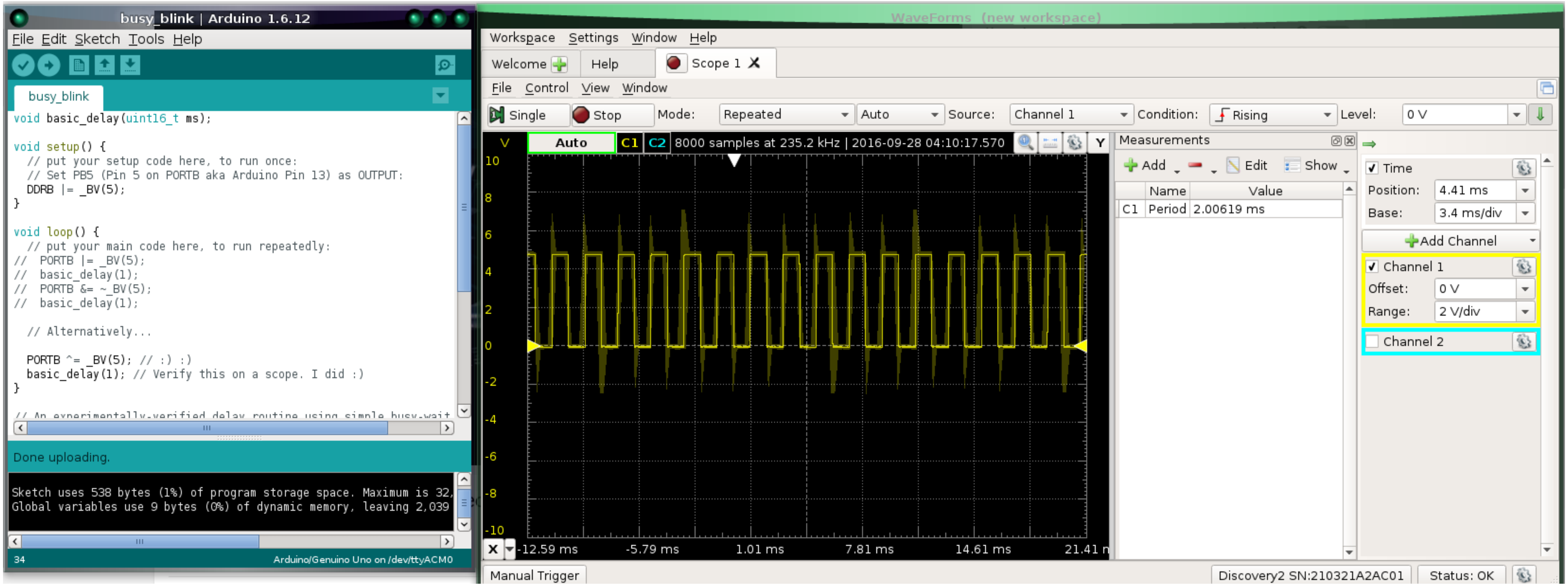
Clocks

- Commonly some kind of quartz crystal that oscillates at a VERY consistent frequency
  - Consistency/accuracy often given by the manufacturer as PPM (parts per million) or max % deviation
  - Often temperature-sensitive
    - Is this an issue in some apps? Trivia!
  - Might need to consider crystal drift in some apps...
    - Trivia: Where might there be an immediate issue?
- Clock source can also be some other logic circuit
  - Free lunch/dinner for any ideas/circuit names that do this :)
    - Cannot use crystal/anything besides transistor-based ICs, resistors, capacitors

Clocks (Cont’d)

- Problem: We don’t just want to count at 1/CLK
  - Prescalers
  - What else can be done? Trivia!

Embedded Hello World Revisited



```
void basic_delay(uint16_t ms);

void setup() {
  // put your setup code here, to run once:
  // Set PB5 (Pin 5 on PORTB aka Arduino Pin 13) as OUTPUT:
  DDRB |= _BV(5);
}

void loop() {
  // put your main code here, to run repeatedly:
//  PORTB |= _BV(5);
//  basic_delay(1);
//  PORTB &= ~_BV(5);
//  basic_delay(1);

  // Alternatively...

  PORTB ^= _BV(5); // :) :)
  basic_delay(1); // Verify this on a scope. I did :)
}

// An experimentally-verified delay routine using simple busy-wait
```

Done uploading.

```
Sketch uses 538 bytes (1%) of program storage space. Maximum is 32,
Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039
```

34 Arduino/Genuino Uno on /dev/ttyACM0

Mode: Repeated Auto Source: Channel 1 Condition: Rising Level: 0 V

8000 samples at 235.2 kHz | 2016-09-28 04:10:17.570

	Name	Value
C1	Period	2.00619 ms

☑ Time
Position: 4.41 ms
Base: 3.4 ms/div

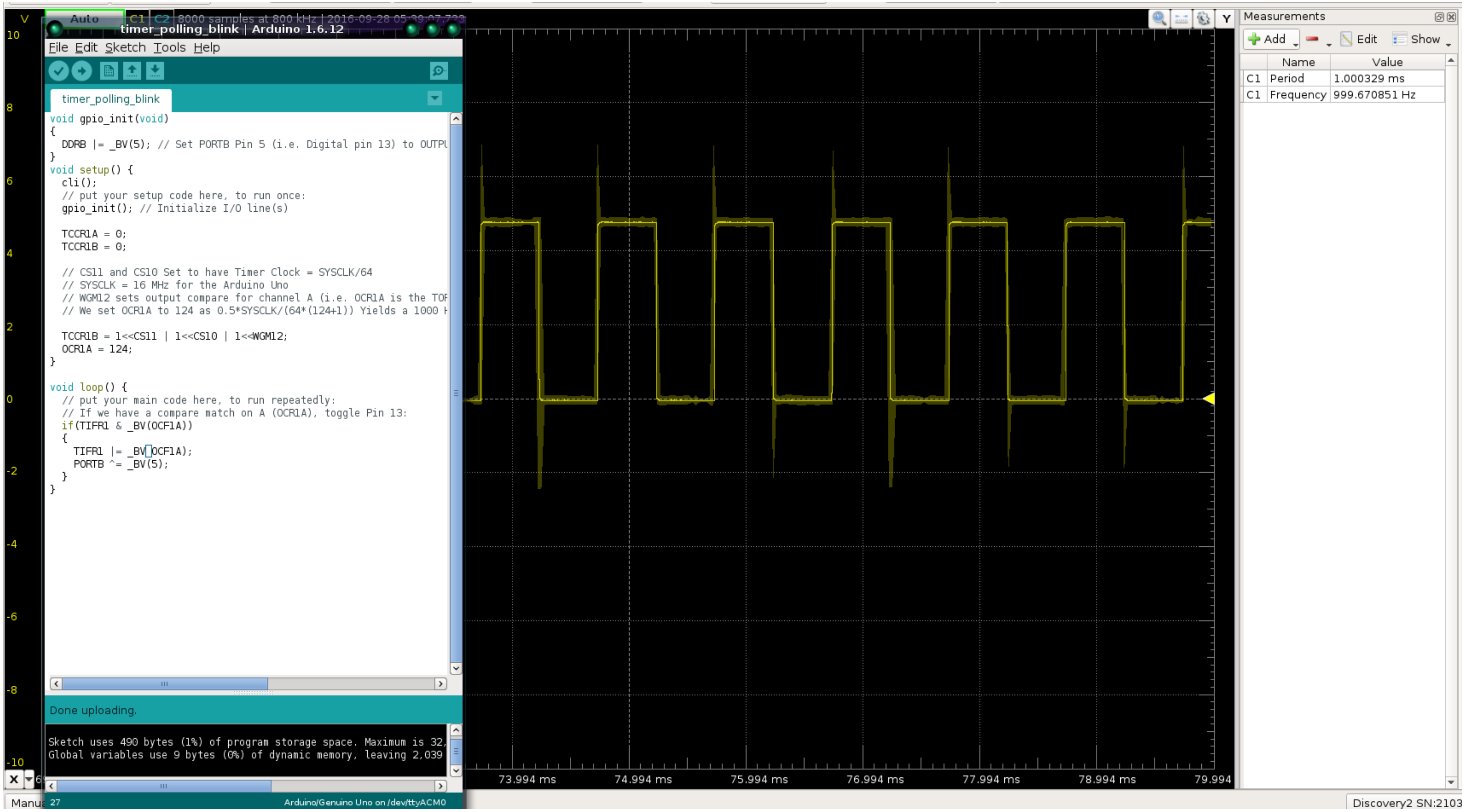
☑ Channel 1
Offset: 0 V
Range: 2 V/div

□ Channel 2

Discovery2 SN:210321A2AC01 Status: OK

Trivia: Any idea why the scope shot has artifacts?

Can we do better?



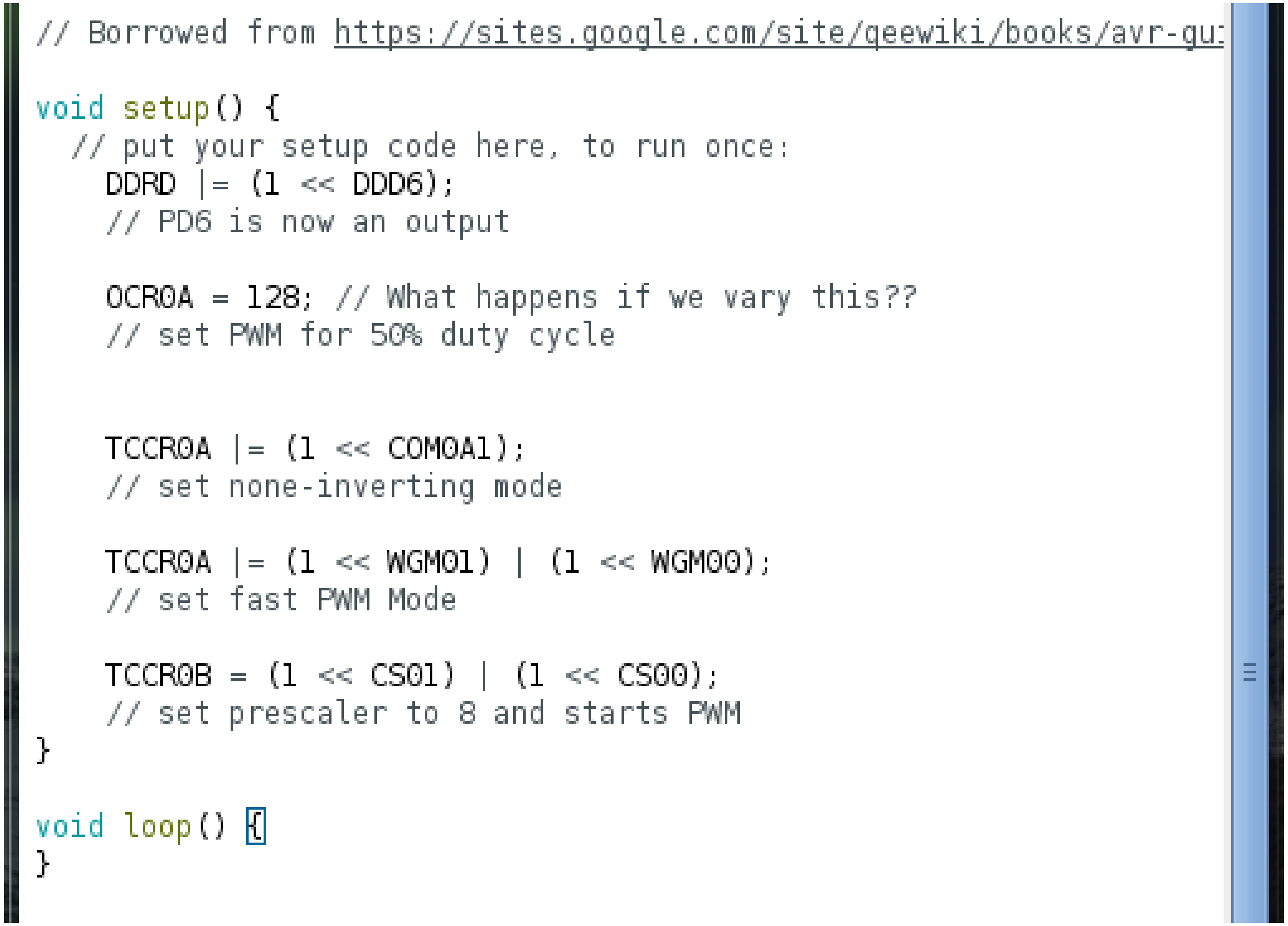
```
void gpio_init(void)
{
  DDRB |= _BV(5); // Set PORTB Pin 5 (i.e. Digital pin 13) to OUTPU
}
void setup() {
  cli();
  // put your setup code here, to run once:
  gpio_init(); // Initialize I/O line(s)

  TCCR1A = 0;
  TCCR1B = 0;

  // CS11 and CS10 Set to have Timer Clock = SYSCLK/64
  // SYSCLK = 16 MHz for the Arduino Uno
  // WGM12 sets output compare for channel A (i.e. OCR1A is the TOF
  // We set OCR1A to 124 as 0.5*SYSCLK/(64*(124+1)) Yields a 1000 H
  TCCR1B = 1<<CS11 | 1<<CS10 | 1<<WGM12;
  OCR1A = 124;
}

void loop() {
  // put your main code here, to run repeatedly:
  // If we have a compare match on A (OCR1A), toggle Pin 13:
  if(TIFR1 & _BV(OCF1A))
  {
    TIFR1 |= _BV(OCF1A);
    PORTB ^= _BV(5);
  }
}
```

Done uploading.

```
Sketch uses 490 bytes (1%) of program storage space. Maximum is 32,
Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039
```

	Name	Value
C1	Period	1.000329 ms
C1	Frequency	999.670851 Hz

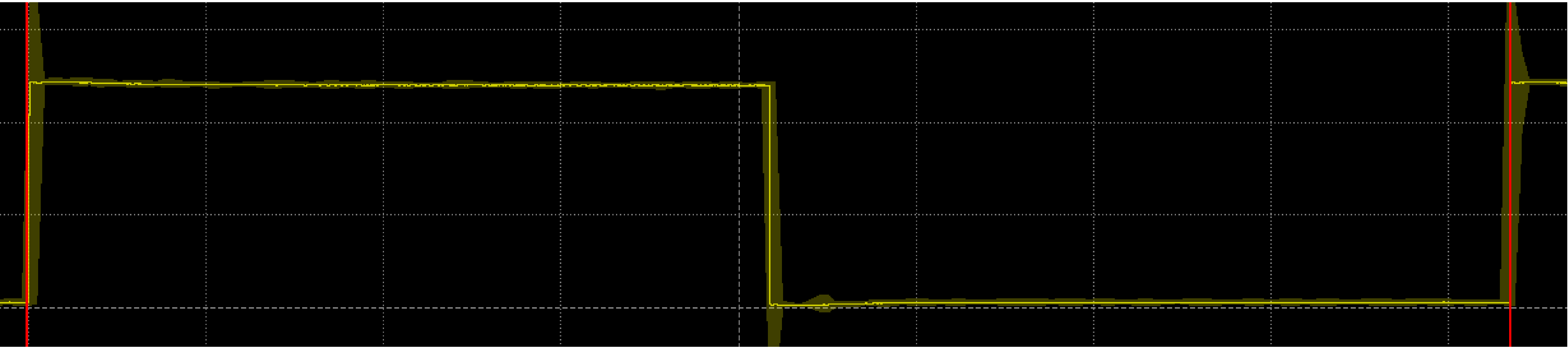
PWM and Hardware to the Rescue!



```
// Borrowed from https://sites.google.com/site/qeewiki/books/avr-gu

void setup() {
  // put your setup code here, to run once:
    DDRD |= (1 << DDD6);
    // PD6 is now an output

    OCR0A = 128; // What happens if we vary this??
    // set PWM for 50% duty cycle


    TCCR0A |= (1 << COM0A1);
    // set none-inverting mode

    TCCR0A |= (1 << WGM01) | (1 << WGM00);
    // set fast PWM Mode

    TCCR0B = (1 << CS01) | (1 << CS00);
    // set prescaler to 8 and starts PWM
}

void loop() {
}
```

PWM and Relation to Analog

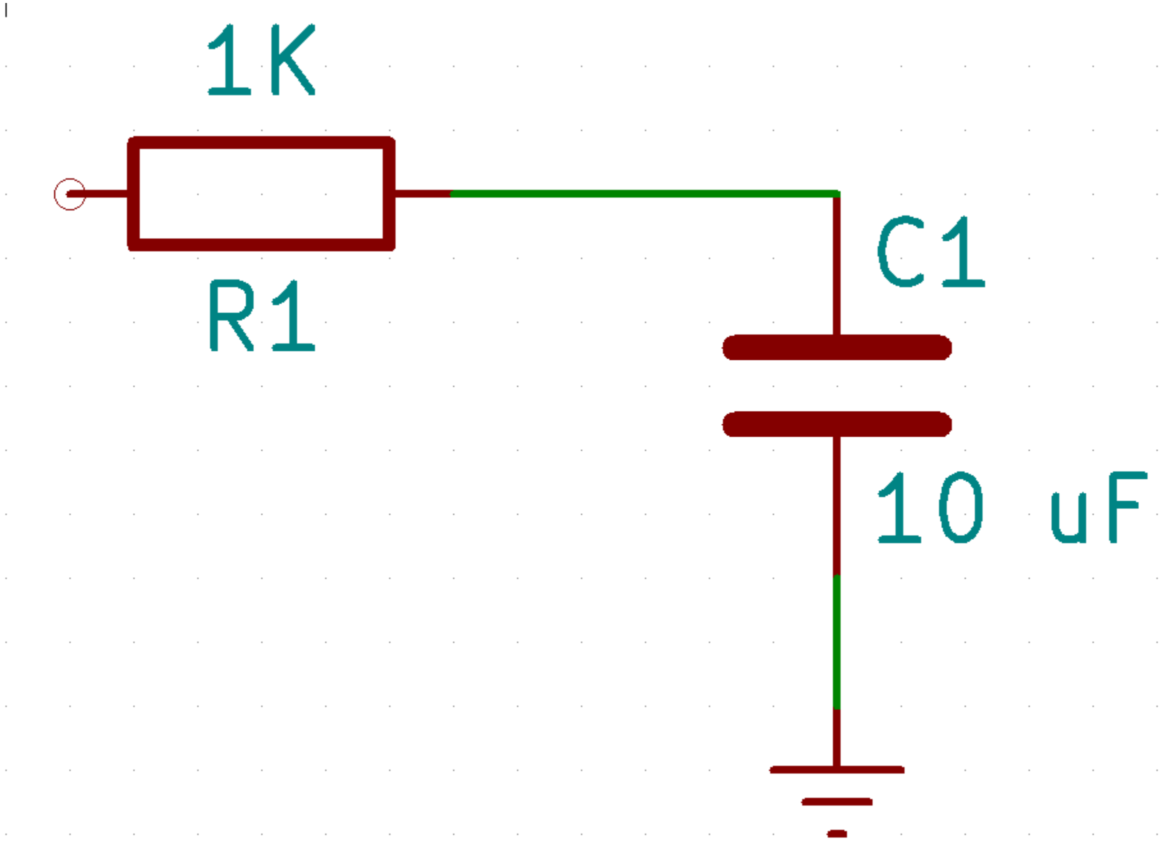
- PWM a method of conveying information/controlling systems
- Square wave with a constant period
  - Vary DUTY cycle i.e. time “on” as % of total period
  - Period measured between rising edges



Above: 1 kHz Square Wave. Period of 1 ms is measured between red cursors

PWM: More than on/off

- PWM normally a DIGITAL signal that can only be high or low
- Lots of things in the real world don’t like this…
- Duty cycle of PWM a measure of RMS power
- If only there was a way to do this in hardware...



Does this look familiar?

Trivia: What’s the cutoff frequency of this filter?

Putting it together: Analog I/O, HITL, sensors