EECS 192: Mechatronics Design Lab

Discussion 1: Introduction

GSI: Andrew Barkan

January 20, 2021 (Week 1)

- Administrivia
- Project Detail
- Huzzah32 Intro
- LED PWM
- There's More!

Welcome

Welcome to EE192!

Introductions!

Let's begin with some introductions!

- Name, year/major, preferred pronouns
- What do you hope to get out of the course?
- ... and something about yourself!

About Me

- GSI: Andrew Barkan, PhD Candidate, ME
  - email: andrew_barkan@berkeley.edu
  - OH: (tentative) Mon. 2:00pm-3:00pm, Tues. 2:00pm-3:00pm
- Preferred pronouns: he, him, his
- Help each student feel more comfortable with embedded systems (at least a little)
- Love playing PC games

Project

- Project: build an autonomous track-following racecar given a stock RC car and microcontroller dev kit
- Teams should be 2 students (3 with permission)
  - Combined skillset should include mechanical hardware experience, electronics, programming
  - Controls experience helpful
- Teams formed by checkoff Friday

Checkoffs

- One-hour time slot on Friday TBD to demonstrate that your project is where it should be
- At least one team member needs to show up to run your hardware
- These are graded, half credit if late

- First checkoff this Friday
  - Form project teams
  - Make sure you have ordered equipment
  - Get private course GitHub repository
  - More details to come

Do you have your cars?

The Project in more detail

Planning & Reliability

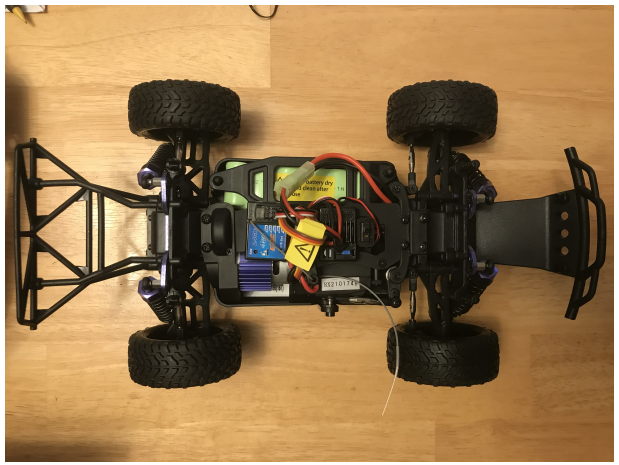
- Get started early thinking about how to approach the project
- ~~Measure once, cut twice, then hammer~~
- Measure twice, cut once
- Start thinking about high-level project plan
  - Plan ahead and examine feasibility
  - Get feedback on ideas
- Reliability first, THEN performance
  - "Better is the enemy of the good enough"
  - Very fast car going into a wall (and breaking) gets you few points
  - Fast enough car hitting all of the checklist items gets you all the points

One weird trick to flunk ee192!

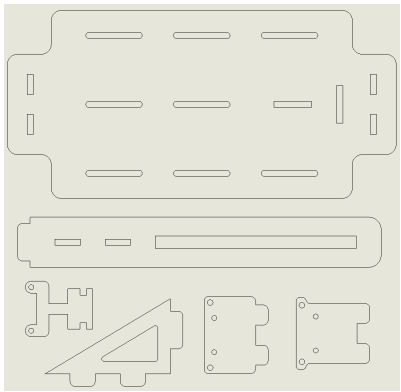
image from LOLCaption.com

Mechanical

- You should have your RC car! Either of these:
  - Desert Short Course Truck
  - Magnet EP Electric RTR Off Road Truck
- Will be making use of stock mechanical parts:
  - DC motor
  - Gearbox and transmission
  - Steering linkage and servo
  - Shocks
- Adding our own chassis adapter for mounting



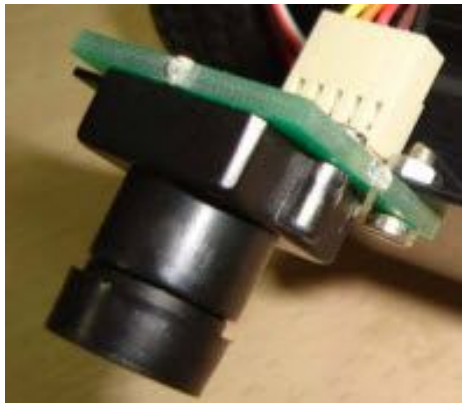
Exposed chassis of RC car! (Desert)



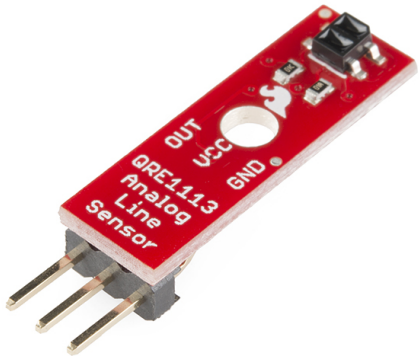
Chassis adapter layout

Electrical

- Again, making use of some stock components
  - Electronic speed controller (ESC)
  - NiMH battery
- With some additions of our own!
  - Adafruit Huzzah32 Feather ESP32 Board
  - USB battery
  - Line scan camera
  - Encoder?



Line scan camera



Line sensor board

Car comments

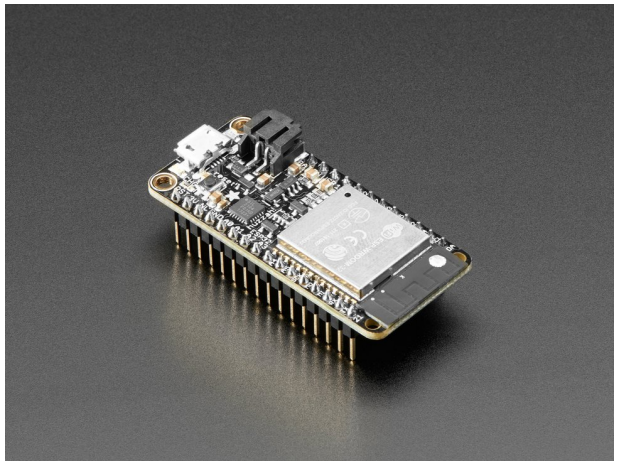
- Goal: don't reinvent the wheel
  - Assist in understanding all of the pieces that go into the project
  - Take design cues from those who came before you - recognize and use good ideas
  - Conversely, learn from others' mistakes, so you don't have to repeat them
- Some design points to consider:
  - Robustness
  - Maintainability
  - Design for Test
  - Graceful error handling
  - Anything else you want to add?

It's been done before (don't repeat it!)

©Fox

Hardware

- Adafruit Huzzah32 Feather ESP32 Board
- Espress-If ESP32 SOC
  - 240 MHz dual core Tensilica LX6 microcontroller
  - 4MB flash
  - 520KB SRAM
- Programmable using micro USB
- I/O headers including
  - GPIO
  - 12-bit ADC, 8-bit DAC
  - Built-in WiFi, Bluetooth
  - PWM, I^2C, SPI, I^2S, and UART modules
- On-board RGB LED and power management



Huzzah32 Board

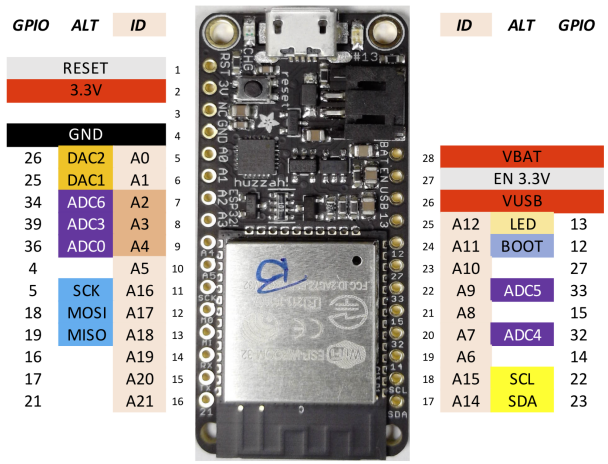
image from Adafruit

IO Refresher

- GPIO (general purpose input/output) pins

  `https://microkit.berkeley.edu/gpio-basics/`

  - As an output: sets voltage on pin from software, either GND (0) or Vdd (1)
  - As an input: samples voltage on the pin, returning either 0 (LOW) or 1 (HIGH)
- PWM (pulse-width modulation) module
  - Every *period*, the pin is high based on the *duty cycle*, then low for the remainder
  - Can digitally approximate analog outputs
- Analog Inputs (ADC)
  - Converts a continuous analog voltage (0-3.3v) to a 12-bit (0-4095) quantity



Huzzah32 Pinout

image from microkit.berkeley.edu

Getting Started!

- We will be using Microsoft Visual Studio Code IDE and a special plugin called Platform IO.
  - `https://code.visualstudio.com/`
  - `https://platformio.org/platformio-ide`
- Allows you to program and flash your microcontroller!
- Will be communicating over UART and WiFi
  - Do you have a preferred terminal (e.g. PuTTY)?
  - Python3 distribution

Getting Started!

- Platform IO will install the necessary μC backend
  - ESP-IDF framework that includes source code and firmware for ESP32 functionalities
  - Example code to run!
- Convenient way to write, organize, build, flash your code
- Easily installable/configurable through VS Code plugins
- Follow the "Getting Started" instructions on SkeletonHuzzah32 repo!
  - `https://github.com/ucb-ee192/SkeletonHuzzah32`

"Hello, World!" Code

```
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "esp_system.h"
#include "esp_spi_flash.h"


void app_main()
{
    printf("Hello world!\n");

    /* Print chip information */
    esp_chip_info_t chip_info;
    esp_chip_info(&chip_info);
    printf("This is ESP32 chip with %d CPU cores,
         WiFi%s%s, ",
            chip_info.cores,
            (chip_info.features & CHIP_FEATURE_BT  }
                 ) ? "/BT" : "",
            (chip_info.features &
                 CHIP_FEATURE_BLE) ? "/BLE" : ""
                 );

    printf("silicon revision %d, ", chip_info.
         revision);
```

```
    printf("%dMB %s flash\n",
         spi_flash_get_chip_size() /
         (1024 * 1024),
            (chip_info.features &
                 CHIP_FEATURE_EMB_FLASH)
                  ? "embedded" : "
                 external");

    for (int i = 10; i >= 0; i--) {
        printf("Restarting in %d seconds
             ...\n", i);
        vTaskDelay(1000 /
             portTICK_PERIOD_MS);
    }
    printf("Restarting now.\n");
    fflush(stdout);
    esp_restart();
```

Hello, World! Demo

Live Demo!

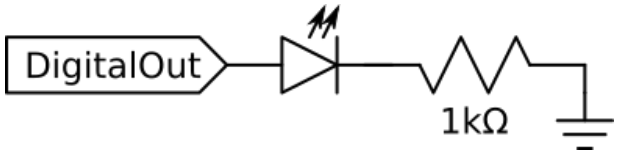
This is essentially the procedure demonstrated in the Getting Started section of the SkeletonHuzzah32 page

... and hopefully goes Murphy-free ...

`https://github.com/ucb-ee192/SkeletonHuzzah32`

Exercise: Hardware from Software

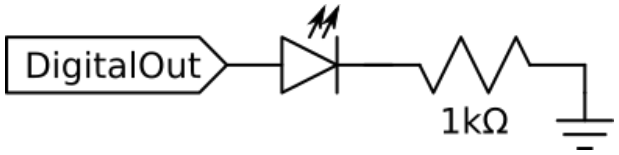
- What would happen if the GPIO was 1?



"Analyze" this simple circuit

Exercise: Hardware from Software

- What would happen if the GPIO was 1?
  - The GPIO output would be 3.3v, and the LED lights up
- What if the GPIO was 0?

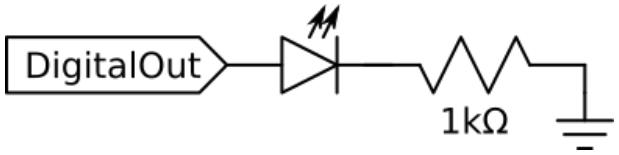


"Analyze" this simple circuit

Exercise: Hardware from Software

- What would happen if the GPIO was 1?
  - The GPIO output would be 3.3v, and the LED lights up
- What if the GPIO was 0?
  - Nothing: the GPIO would be 0v, and no current flows across the LED
- What if the GPIO was PWMed?

  Pulse Width Modulation (PWM) toggles an output between 0 and 1 "really fast", controlling the on and off ratio

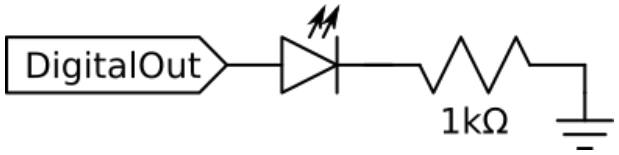


"Analyze" this simple circuit

Exercise: Hardware from Software

- What would happen if the GPIO was 1?
  - The GPIO output would be 3.3v, and the LED lights up
- What if the GPIO was 0?
  - Nothing: the GPIO would be 0v, and no current flows across the LED
- What if the GPIO was PWMed?

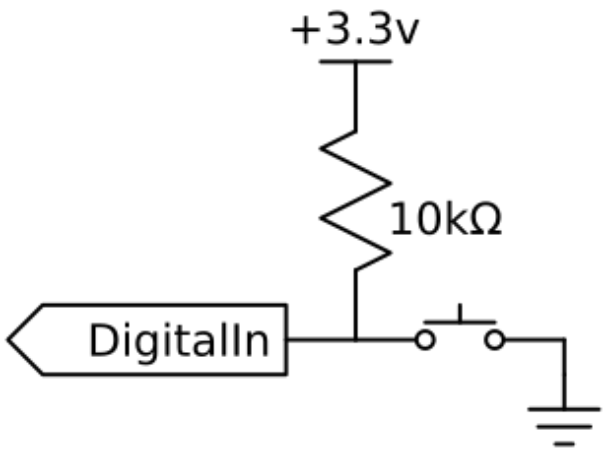
  Pulse Width Modulation (PWM) toggles an output between 0 and 1 "really fast", controlling the on and off ratio
  - The LED would light at half intensity, but it may be (perceived as brighter)



"Analyze" this simple circuit

Exercise: Hardware from Software

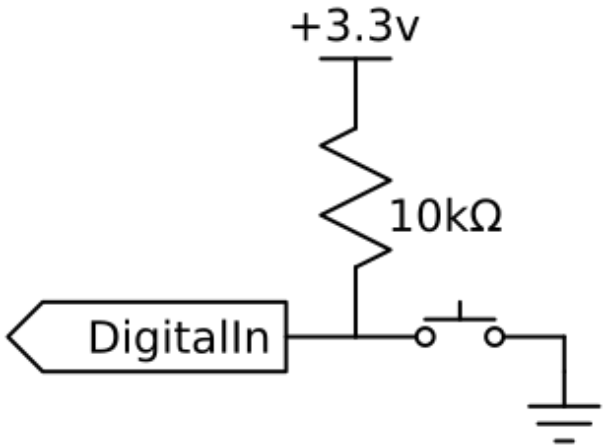
- What would the GPIO read if the switch was pressed?
  (shorted)



"Analyze" this simple circuit

Exercise: Hardware from Software

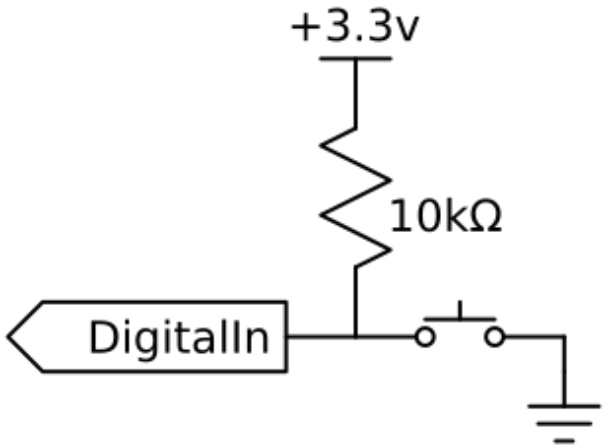
- What would the GPIO read if the switch was pressed?
  (shorted)
  - The GPIO would read 0, because of the 0v at the pin
- What if the switch was not pressed?



"Analyze" this simple circuit

Exercise: Hardware from Software

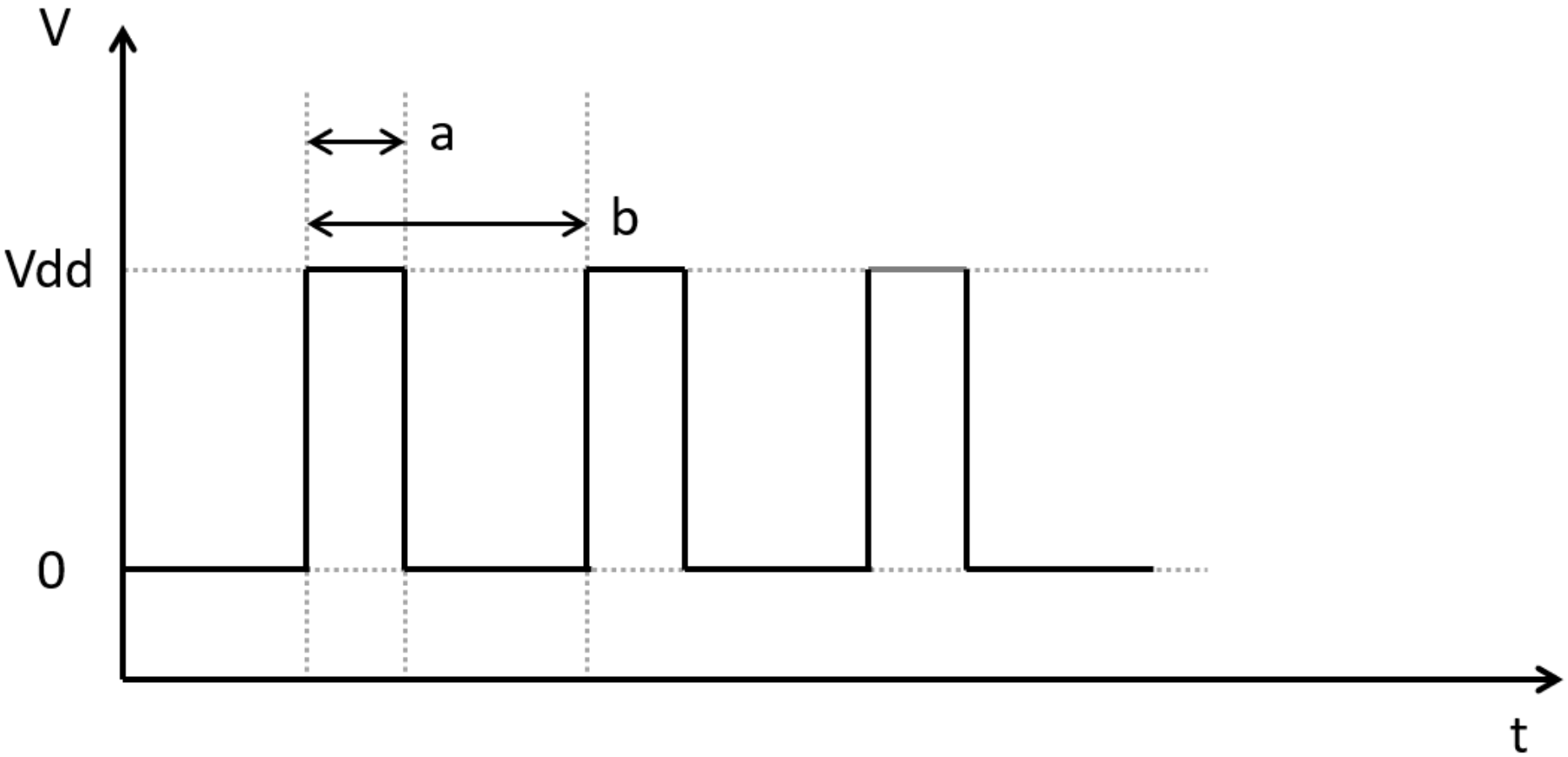
- What would the GPIO read if the switch was pressed?
  (shorted)
  - The GPIO would read 0, because of the 0v at the pin
- What if the switch was not pressed?
  - The GPIO would read 1, because of the 3.3v at the pin.



"Analyze" this simple circuit

PWM Review!

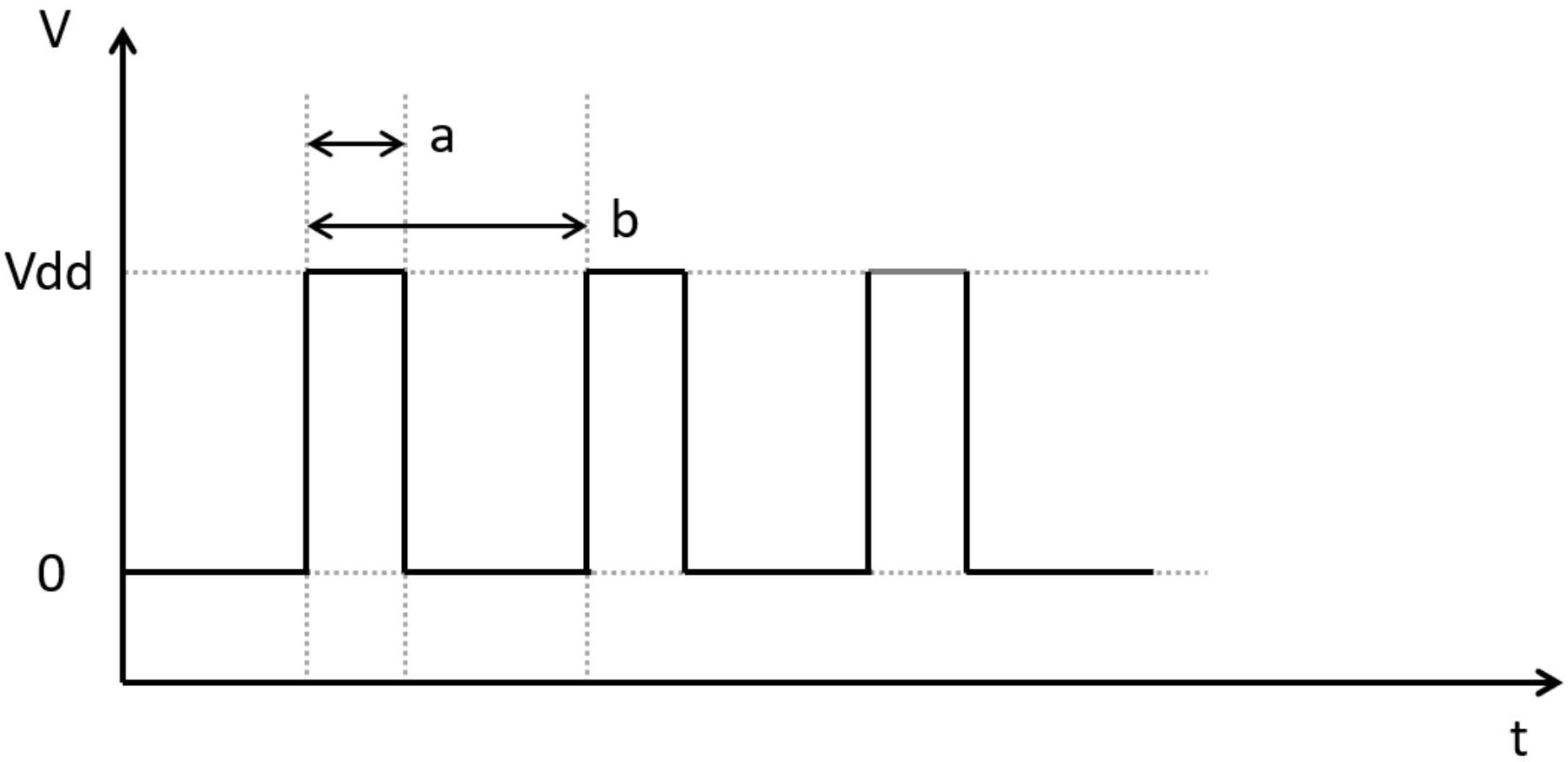
- Let's review some PWM basics...
- Can you label the figure?



A standard PWM signal

PWM Review!

- Period: $T = b$, Frequency: $f = 1/b = 1/T$, Duty cycle: a/b

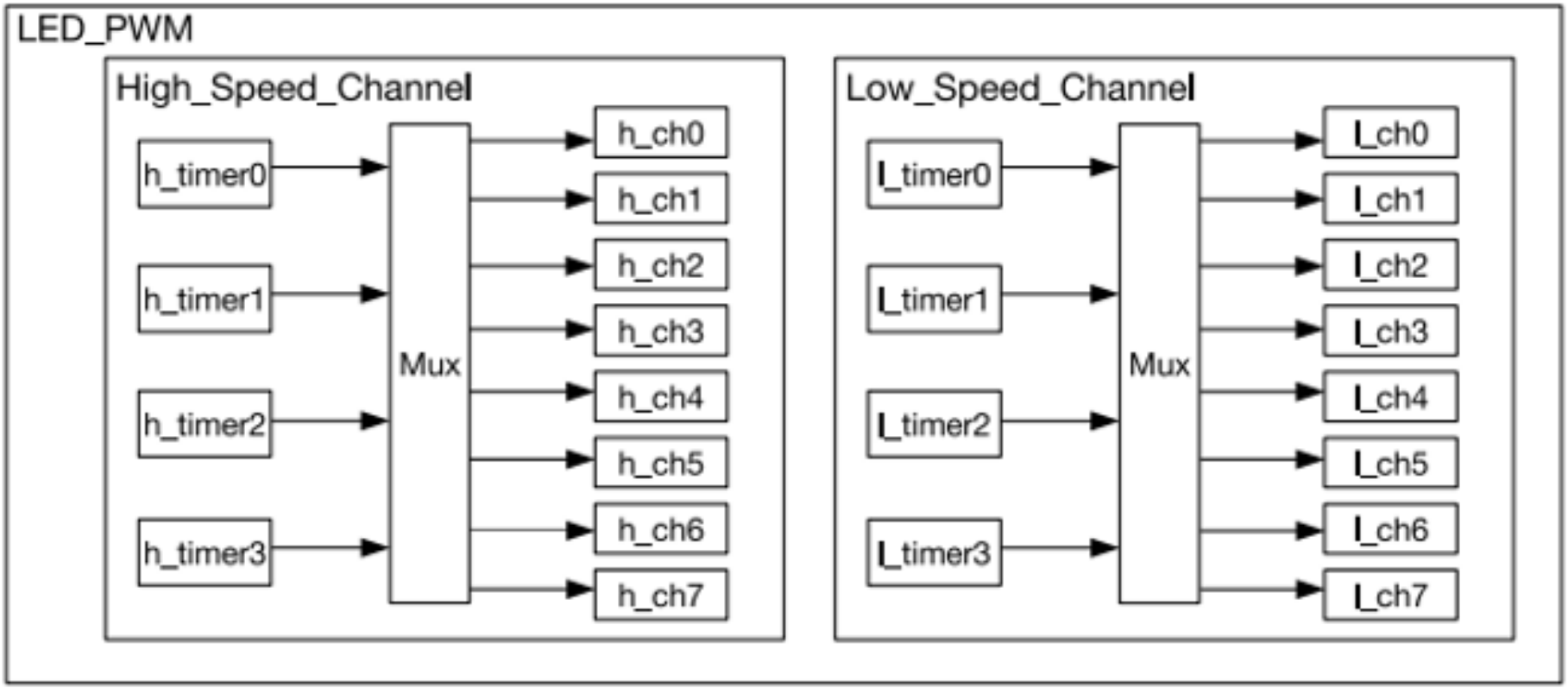


A standard PWM signal

LED PWM Controller

- The ESP32 has several PWM controllers!
  - LED PWM controller
  - MCPWM (motor control pwm) controller
- Different functionalities for different use cases
- Starting with LED PWM, MCPWM later in lecture

LED PWM


LED_PWM
High_Speed_Channel
h_timer0
h_timer1
h_timer2
h_timer3
Mux
h_ch0
h_ch1
h_ch2
h_ch3
h_ch4
h_ch5
h_ch6
h_ch7
Low_Speed_Channel
L_timer0
L_timer1
L_timer2
L_timer3
Mux
L_ch0
L_ch1
L_ch2
L_ch3
L_ch4
L_ch5
L_ch6
L_ch7

LED PWM

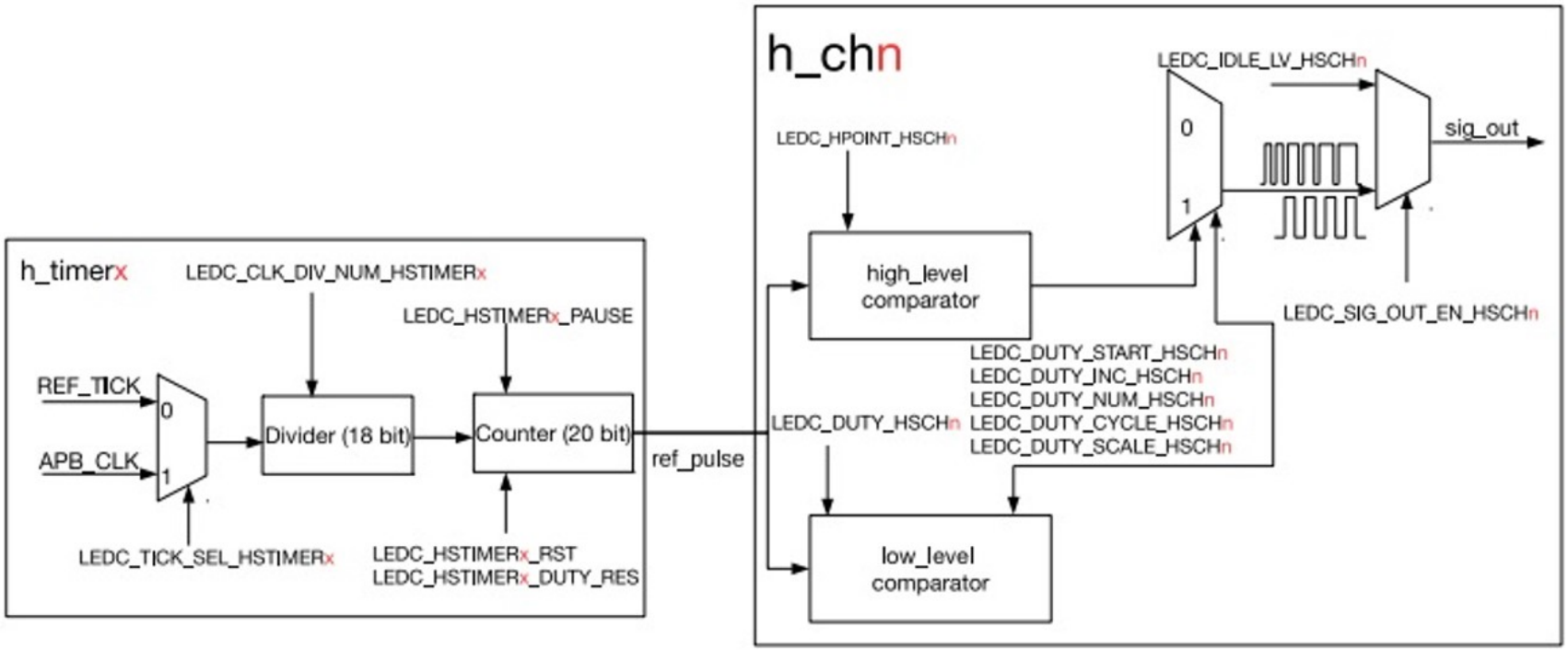


Figure 86: LED_PWM High-speed Channel Diagram

LED PWM

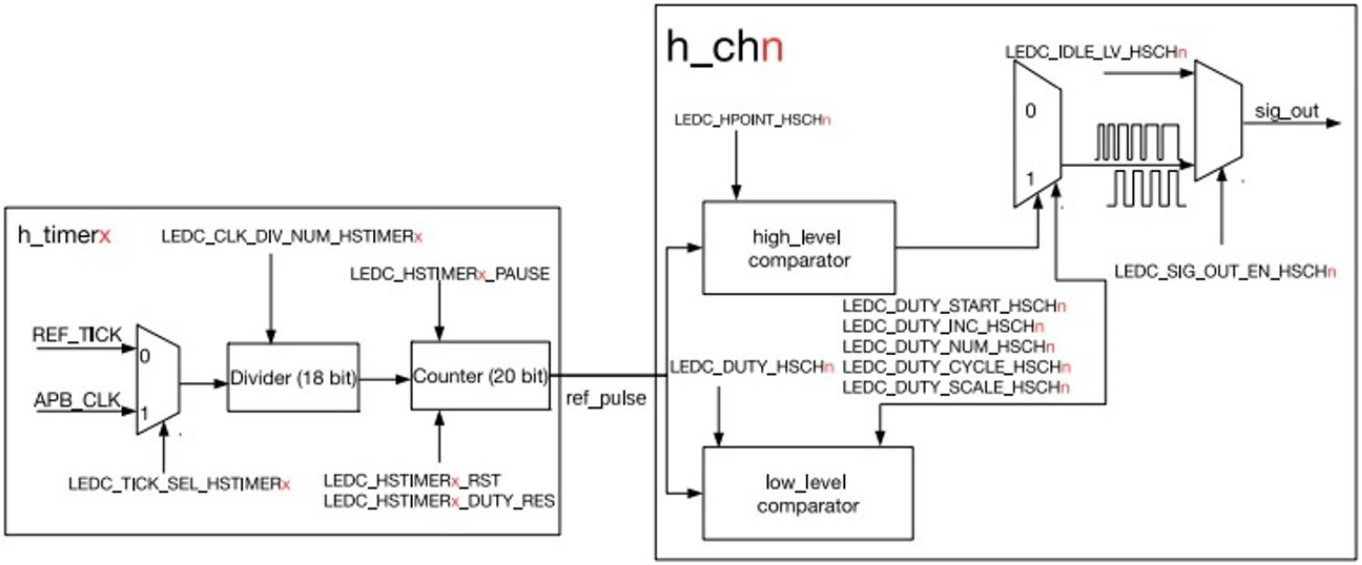
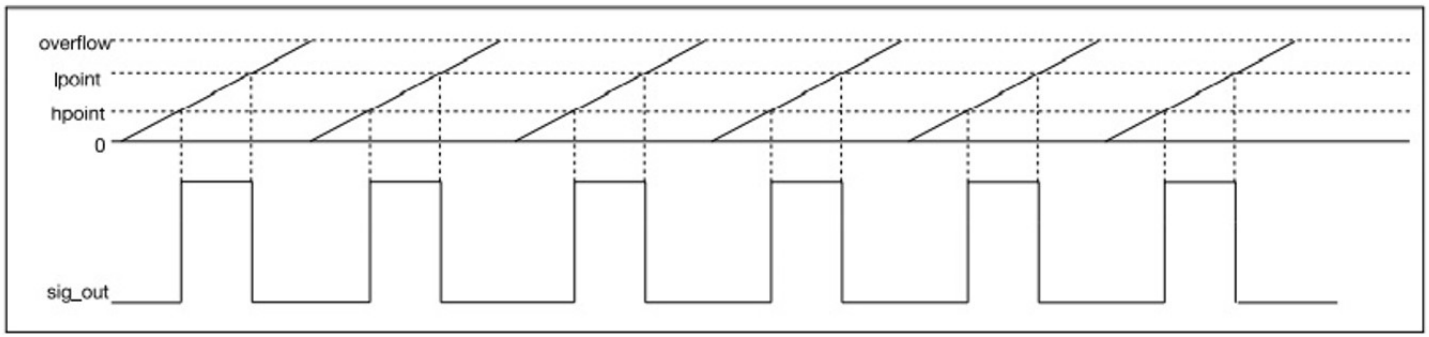


Figure 86: LED_PWM High-speed Channel Diagram



Detour: How to not kill your Huzzah32

- The supply of replacement boards is limited...
- Only power board from USB
- Do NOT use the LiPo connector unless explicitly permitted with your own LiPo battery
  - And ONLY when not using onboard power to energize peripheral hardware

Don't let this happen to you

Detour: How to not kill your Huzzah32

- Your body builds up static charge
  - ... just by walking, especially when dry
  - ... and up to several kV
    but under ∼2kV is imperceptible
- Chips are sensitive to high voltages:
  may cause permanent damage
  - read: board stops working
    "for no reason"
- Remember to ground (discharge) yourself before handling sensitive electronics
  - Touch a grounded surface
  - You can use an ESD wriststrap if you have one
  - Avoid touching traces on boards

Don't let this happen to you

So, how are you going to manage your code?

- ~~main.cpp~~
  ~~main_1.cpp~~
  ~~main_final.cpp~~
  ~~main_really_final.cpp~~
  what a disorganized mess
- ~~on a single team member's laptop~~
  what if their hard drive fails?
  or they're out sick during checkoff day?
- ~~by email~~
  another disorganized mess
- ~~by email, *with code in .doc files*~~
  I don't even...

Don't let this be your code. ©Fox

Use Git!

- Git: distributed version control software
  - Each commit: like complete snapshot
    - Full version history:
      you might not realize it now,
      but you'll be glad you had it
  - Distributed: everyone has compete copy
    - Most operations local, periodically sync
  - Allows *branching* for concurrent work,
    which can be *merged*
- Best Practices
  - Small, logical, frequent commits
  - Write good commit messages
  - Keep master clean

git logo, by Jason Long, CC BY 3.0

Learn git here:
`try.github.io`

GitHub Desktop Demo

Live Demo!

we wrote some code, we're now going to commit it!

We recommend GitHub Desktop GUI for those new to Git.