# T08 - Practicum Project
# 360 Proximity Detector

Presentation by Matt Whiteside, Jared Rue

Naser Alshami and Timothy Nelson

# Outline

1. Problem, Motivation, Objective - Naser
2. Alternatives - Naser
3. Requirements - Naser
4. Schedule - Naser
5. Approach – Tim
6. Device Design – Tim
7. BOM Overview – Tim
8. State Diagram – Matt
9. Program Algorithm – Jared
10. Module Decomposition – Jared
11. Schematic and Layout – Tim
12. Firmware – Jared
13. IP and Licensing – Jared
14. Testing – Matt
15. Results – Matt
16. Learning - All

# Problem

- RC cars are difficult to drive without colliding into objects
- Limited visibility of RC car, especially at a distance, contributes to this problem
- RC cars might damage objects be damaged by collisions

# Motivation

- Giving feedback to user can prevent collisions
- Reduced collisions can prevent damage to RC car or other objects
- By preventing collisions, user will have a more enjoyable experience

# Objective

- Create a prototype that can detect objects near the RC car and alert the user so that they can correct their driving.

# Alternatives

- No commercial product satisfies this problem
- There are many DIY projects that are similar but serve a different purpose

# Requirements

- Detect objects up to 80cm away
- When object is detected, will notify user through visual and audio stimulus
- Must detect 360 degrees around RC car
- Link to PDS for complete document

# Schedule

- Never fell behind schedule
- Link to task view of schedule

# Approach

- Targeted Inexpensive modules
- The simpler the better
- Used iterative approach to implementing features in order of importance

# Device Design

- Decided on a single range sensor spinning on a servo
  - Simple
  - Cheap
  - More effective at detection in every direction
- Alternative would be multiple fixed sensors set radially around RC car
  - Multiple IR sensors would cause project to be too expensive
  - Would possibly be faster to detect objects

# Microcontroller

- Decided on Atmega328
  - Supported by class
  - Cheap
  - Plenty of GPIOs, including analog

# Rotation Actuator

- Decided on parallax servo
  - Easy PWM control
  - 180 RPM
  - Self contained package
- Alternative would be Motor
  - Additional circuitry required for direction control

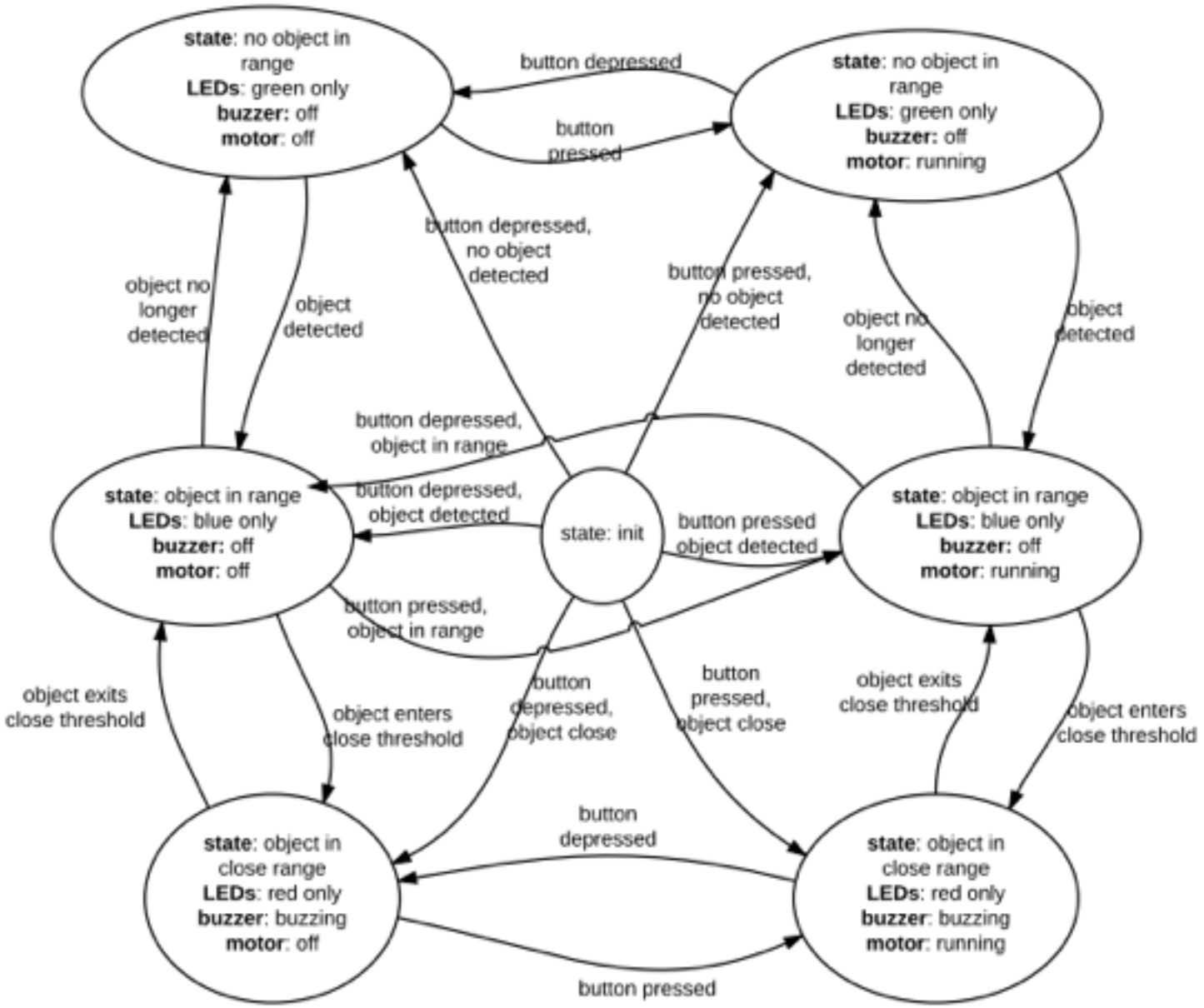
# Audio Alert

- Decided on CX-0905C Buzzer
  - Powered by DC signal
  - High dB for small package
  - No PWM or additional components required
- Alternative was speaker speaking words
  - Complex
  - Unnecessary for purpose of alerting user

# LCD

- Decided on NHD-C0220BiZ-FSW-FBW-3V3M
  - Experience with use
  - Compact design
- Drawbacks
  - Required logic level shifting circuitry
  - Difficult to program

# Software State Diagram



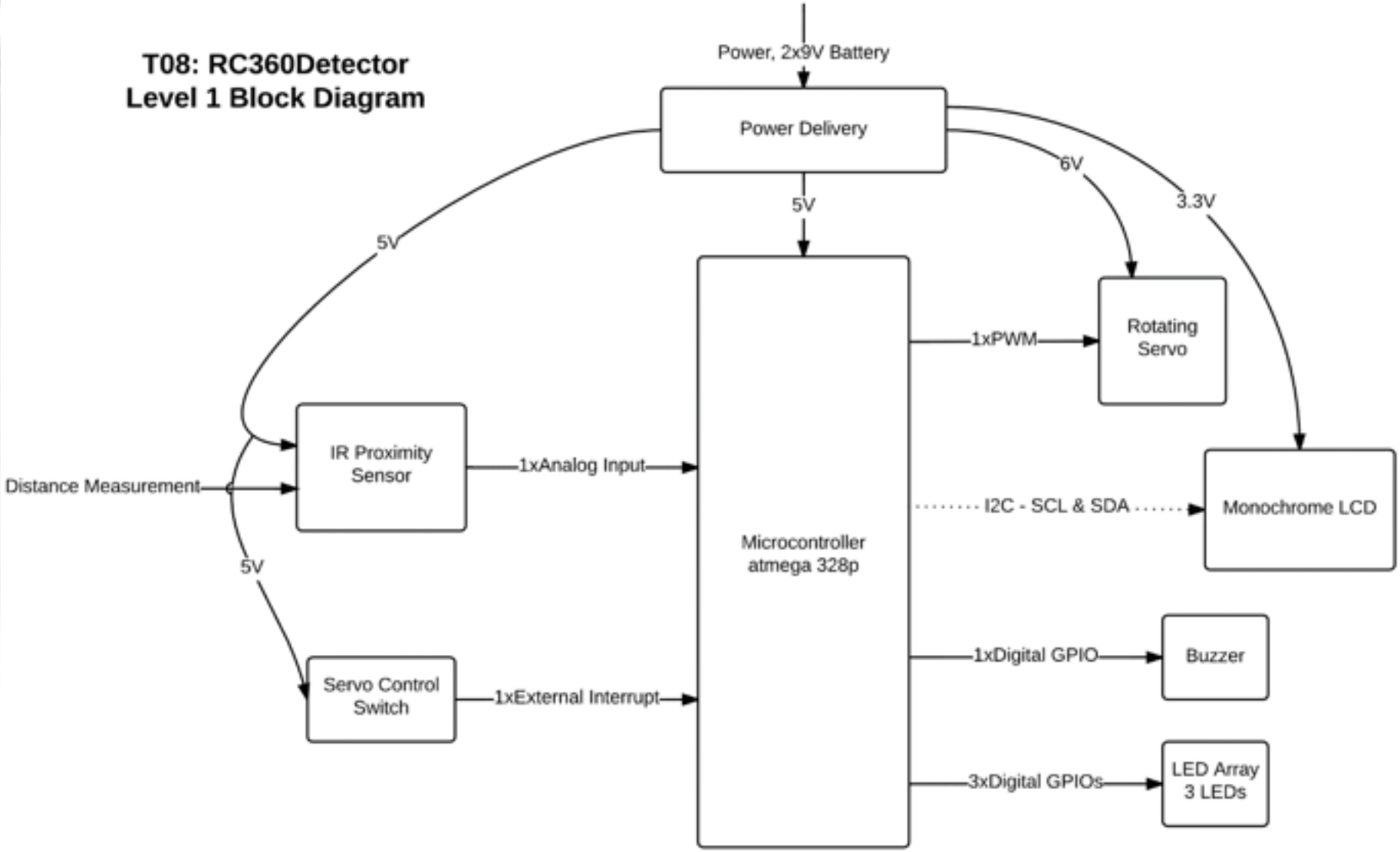
# Algorithm

```
begin
        initialize ADC
        initialize PWM
        initialize LEDs
        initialize interrupt

        interrupt checking for button press
                if button press and PWM off
                        turn on rotating servo
                if button press and PWM on
                        turn off rotating servo

        loop
                read analog input from IR proximity sensor
                        if input > near
                                light red LED and buzzer
                        else if input < near and input > far
                                light blue LED
                        else input < far
                                light green LED
        loop forever
end
```

# Module Decomposition



# Implementation

- Schematic
- PCB
- Main.c

# IP and Prior Work

- Started software off example code for programming ATmega328
  - LED Example: http://www.micahcarrick.com/tutorials/avr-microcontroller-tutorial/getting-started.html
  - PWM example: https://gist.github.com/Wollw/2425784
- Current software under GPL v3 license

# Testing

- Motor start/stop button: Here
- PCB: Here
- Motor spinning performance/endurance: Here
- Object detection: Here

# Results

- LCD program was too complex to implement in time
- Button issues for servo control. Edge trigger vs state hold.
- Product works and meets most specifications

# Contributions

- Timothy Nelson: Sch, PCB, box, solder assembly, assignments
- Naser Alshami: Schedule, RC car mount, assignments
- Matt Whiteside: Coding, assignments
- Jared Rue: Sch, PCB, solder, coding, assignments

# Lessons Learned

- Learning to program the ATmega was probably the most substantive takeaway
- A good team makes everything easier
- The devil is in the details