Development Diary: The Parallax Laser Range Finder

by Joe Grand (@joegrand)
Grand Idea Studio

About

electrical engineer.

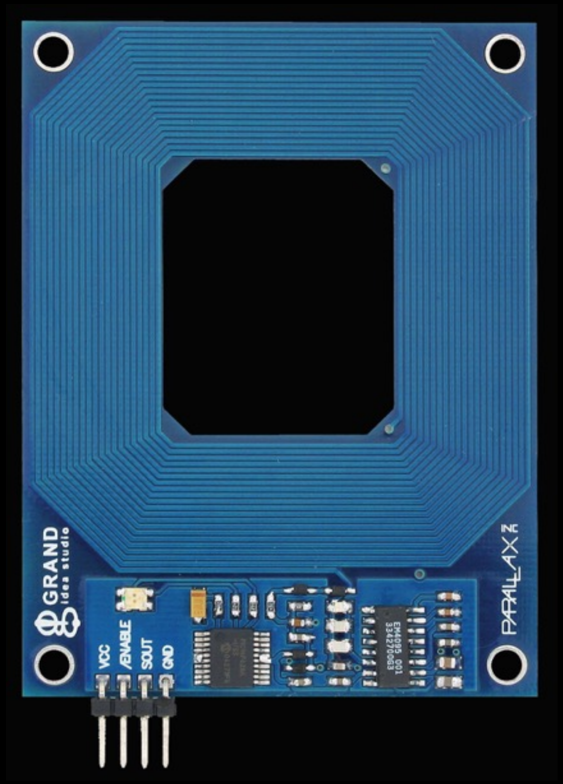
hardware hacker.

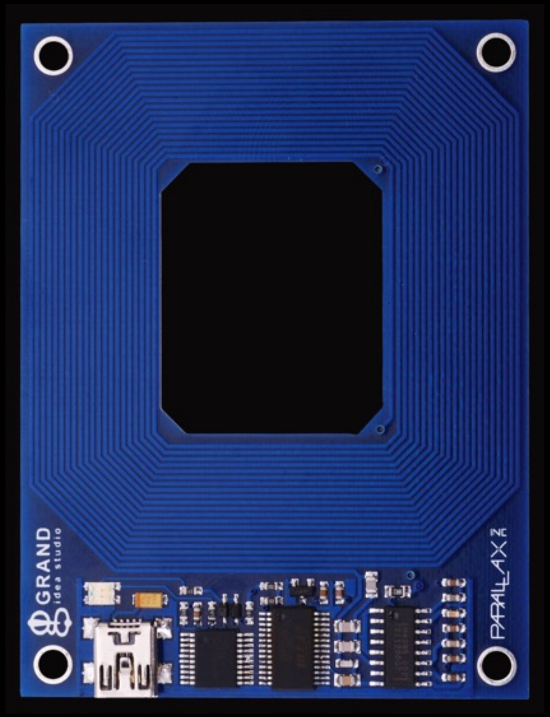
daddy.

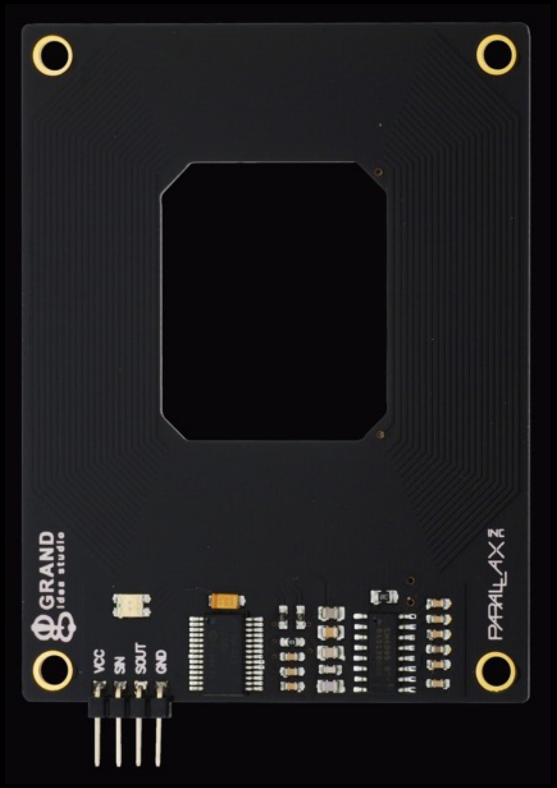
runner.

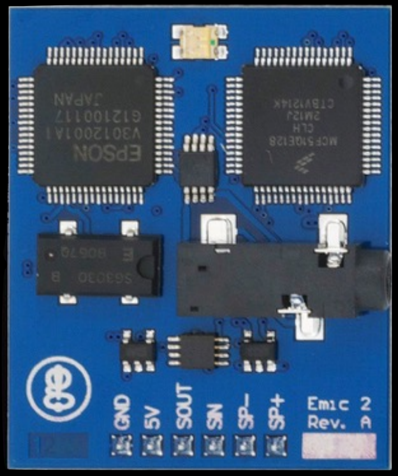
(former) tv host.

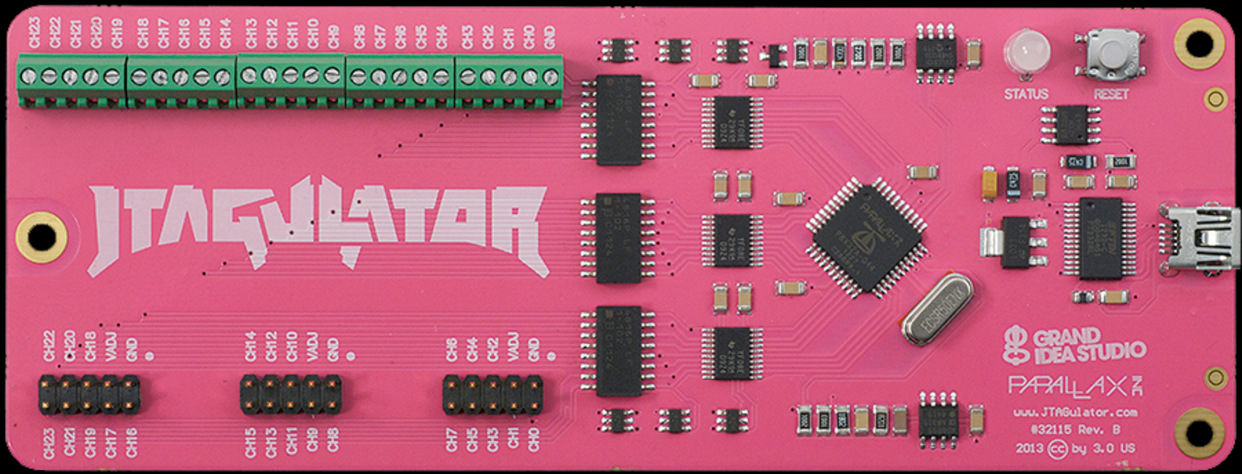
Designer of Parallax Things











Agenda

- Introduction
- Triangulation Theory
- Early Attempts & Development
- Camera/Image Processing
- Demonstrations

Design Goals

- Low cost
- Small footprint
- Easy to use
- Simple serial interface
- Open source/customizable

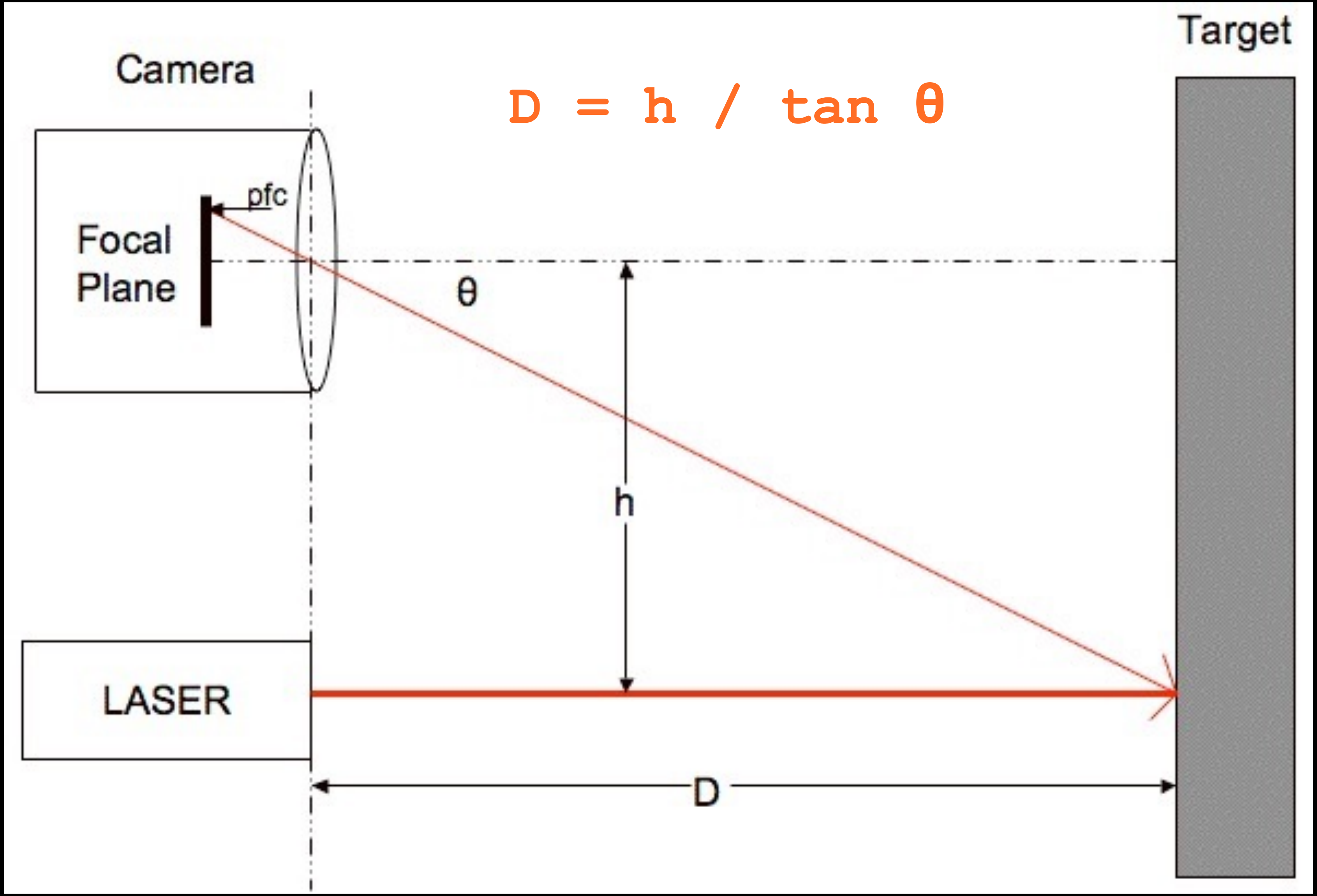
Application Ideas

- Distance or liquid level measurements
- Object detection and/or avoidance
- Item counting

Typical Range Finding Methods

- Time-of-Flight
- Phase Measurement
- Optical Triangulation

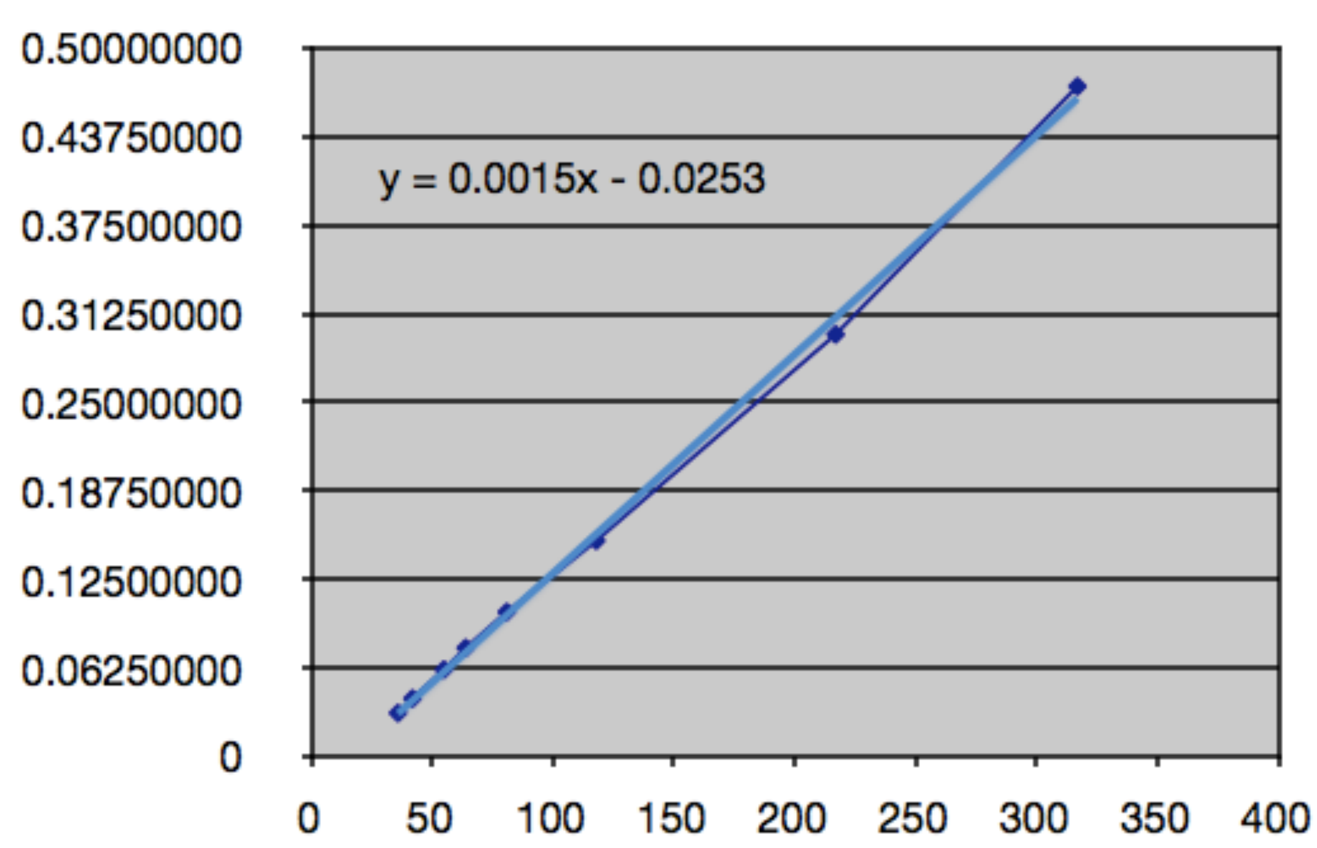
Optical Triangulation



http://sites.google.com/site/todddanko/home/webcam_laser_ranger/laser_ranger_drawing.gif

Optical Triangulation 2

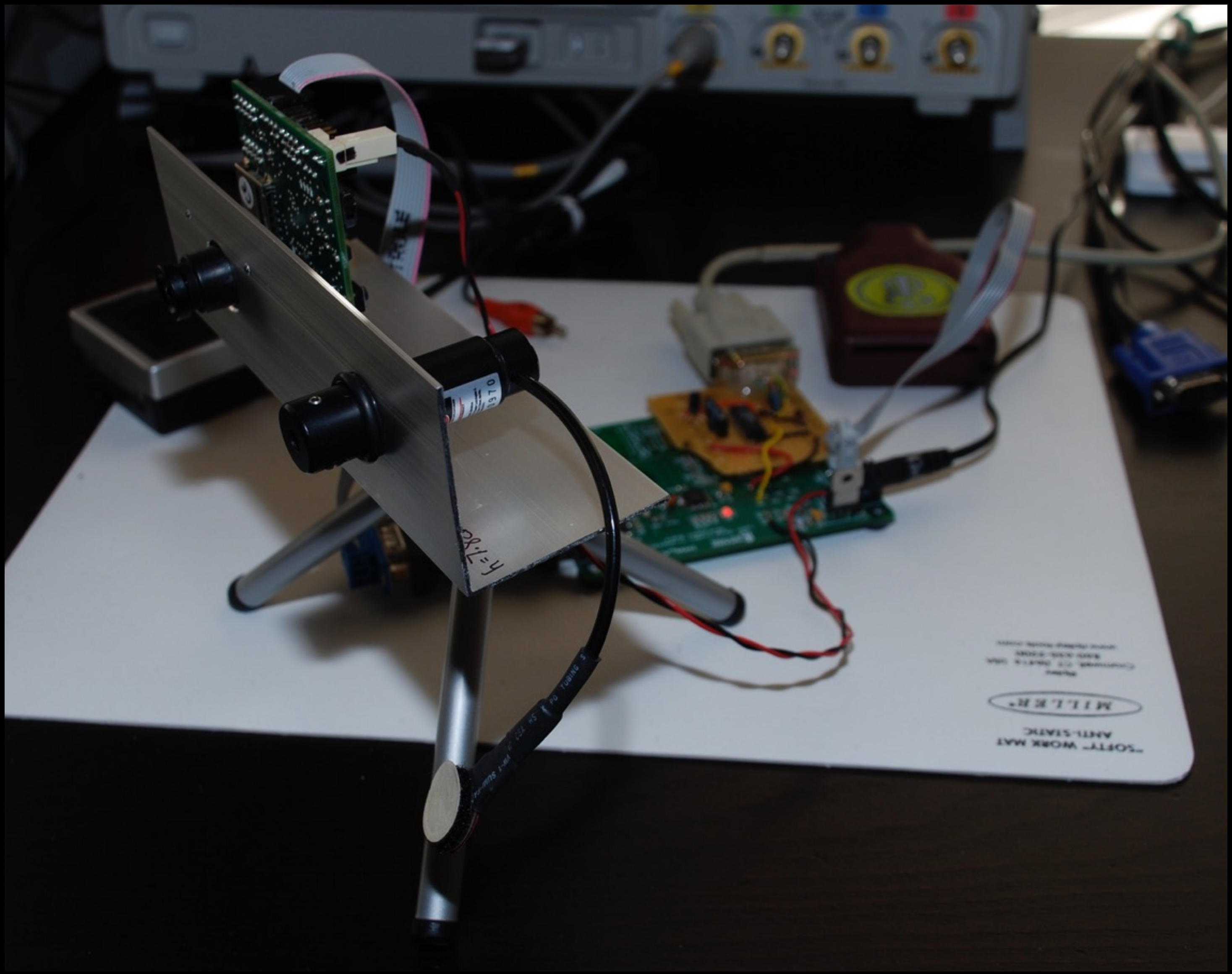
- Relationship between *pfc* and θ is a SLOPE-INTERCEPT linear equation
  - **http://www.math.com/school/subject2/lessons/S2U4L2GL.html**



Early attempt 1

Recreation of Todd Danko's Webcam Based DIY Laser Rangefinder
Not very accurate, but a good starting point to prove the concept
http://sites.google.com/site/todddanko/home/webcam_laser_ranger

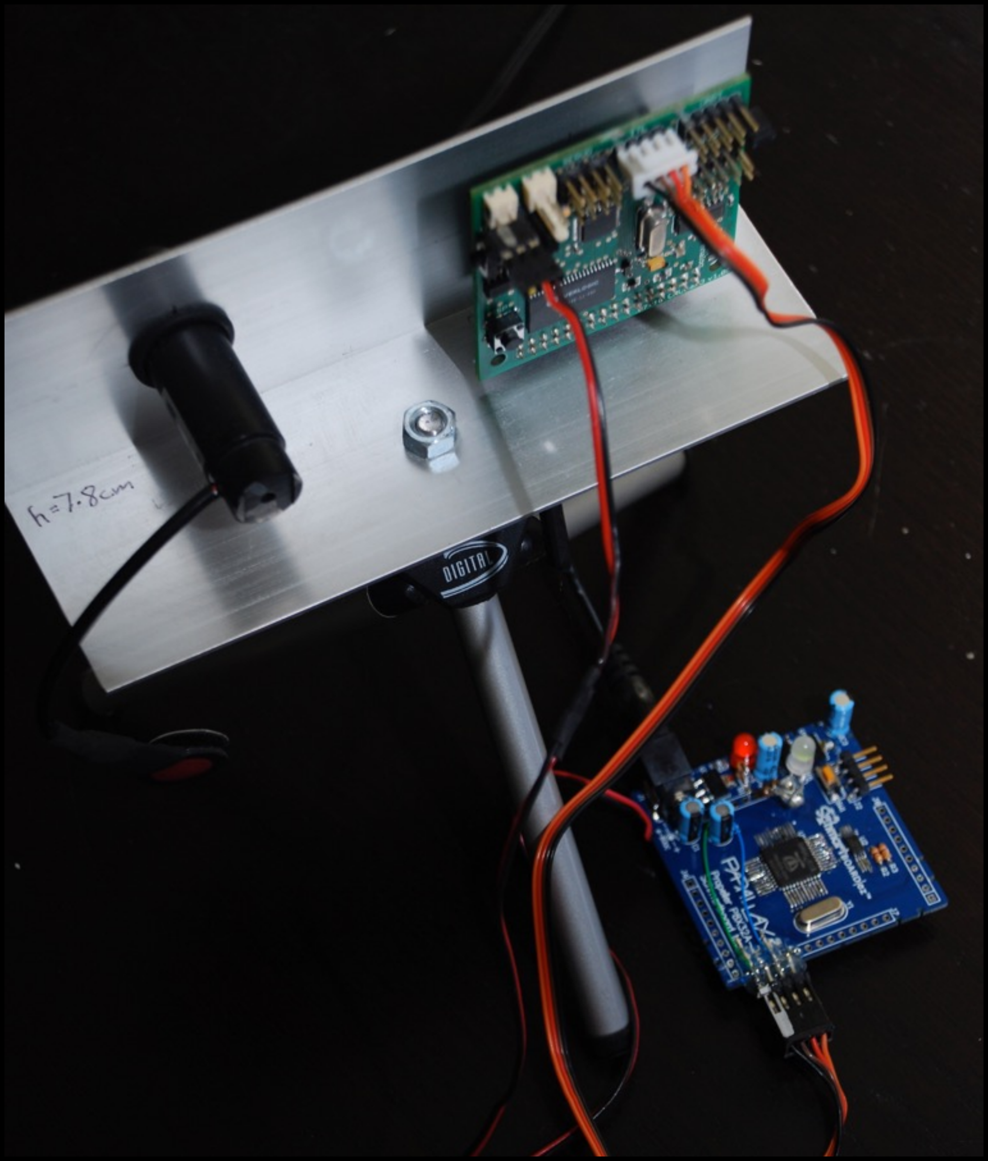
Early Attempt 2



CMUcam2 + Freescale MC9S08QG8

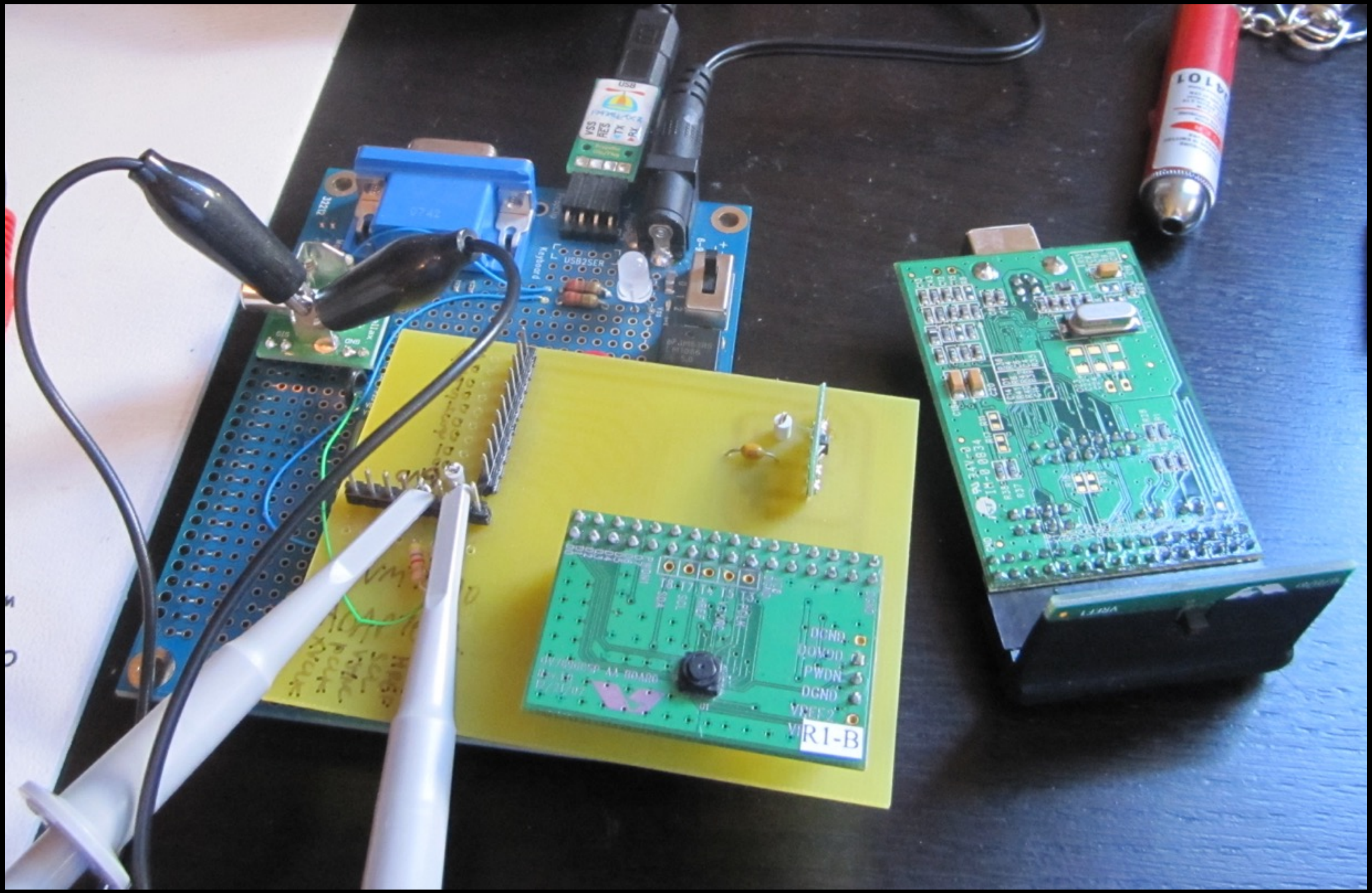
Resolution 176x255, Accuracy ~1/4", Range 7-40"

Early Attempt 3



CMUcam2 + Propeller
Resolution 176x255, Accuracy ~1/4", Range 7-40"

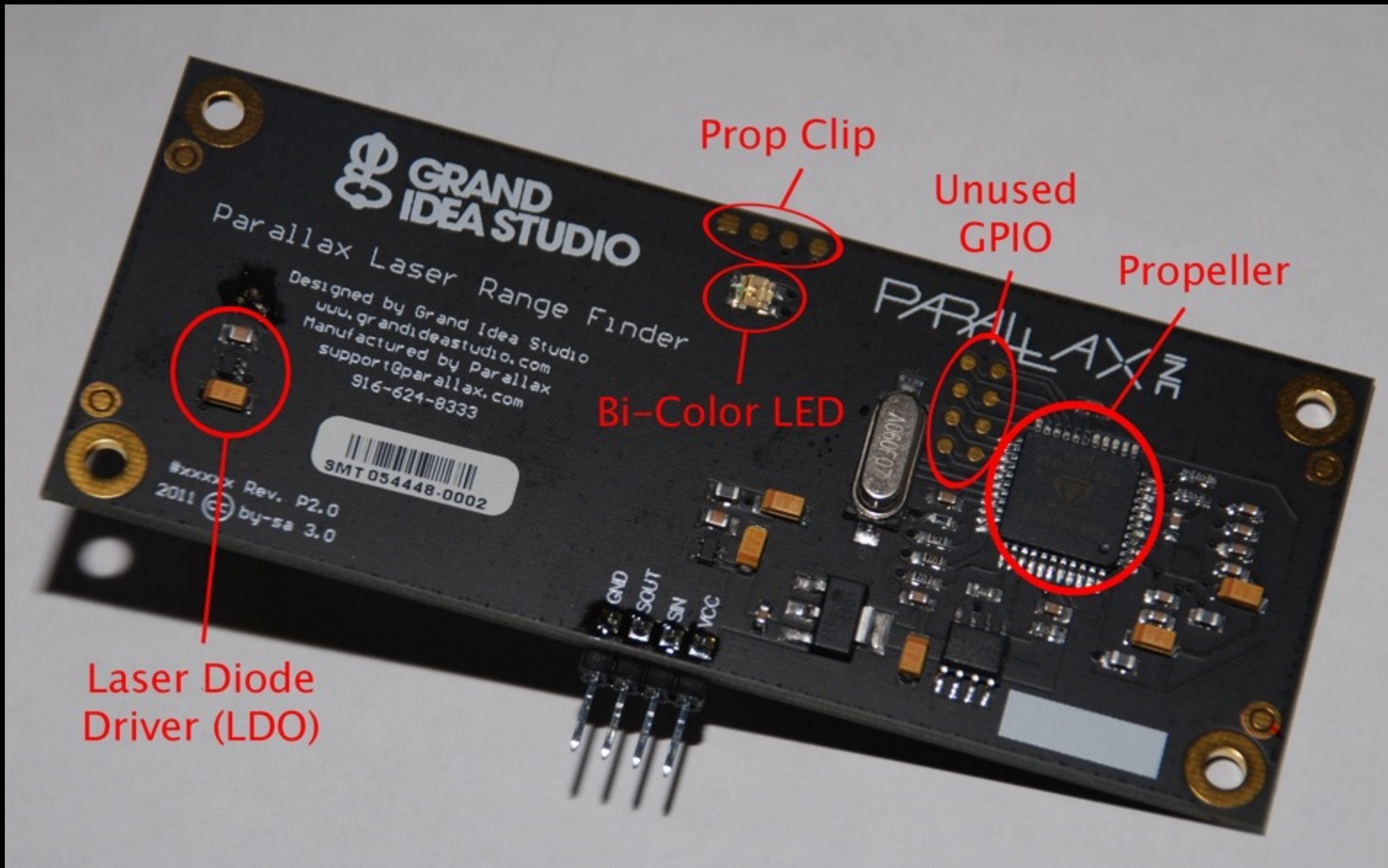
Development Hardware



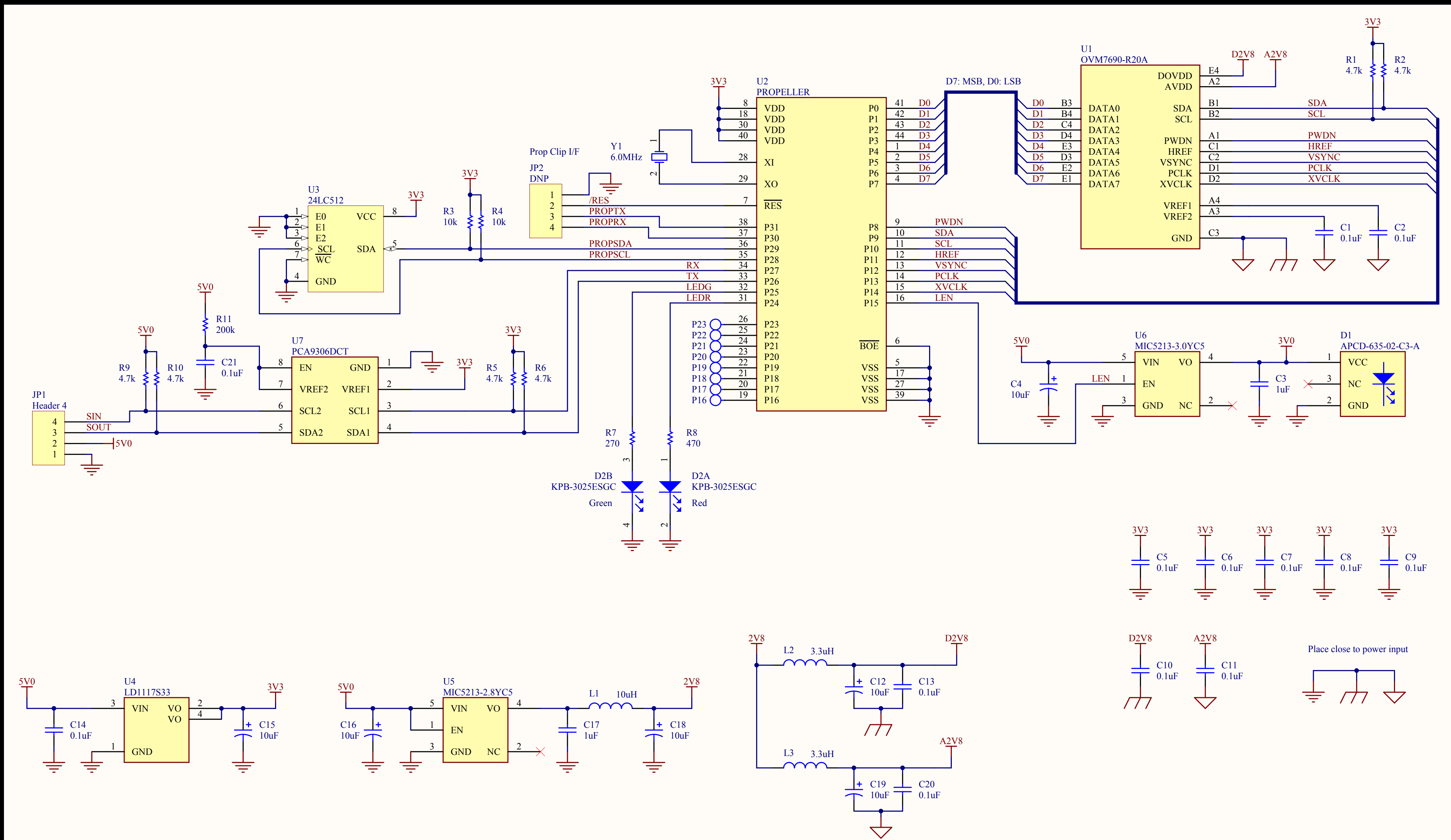
Propeller Proto Board + OVM7690 Eval. Board + Custom PCB

LRF Module: Front

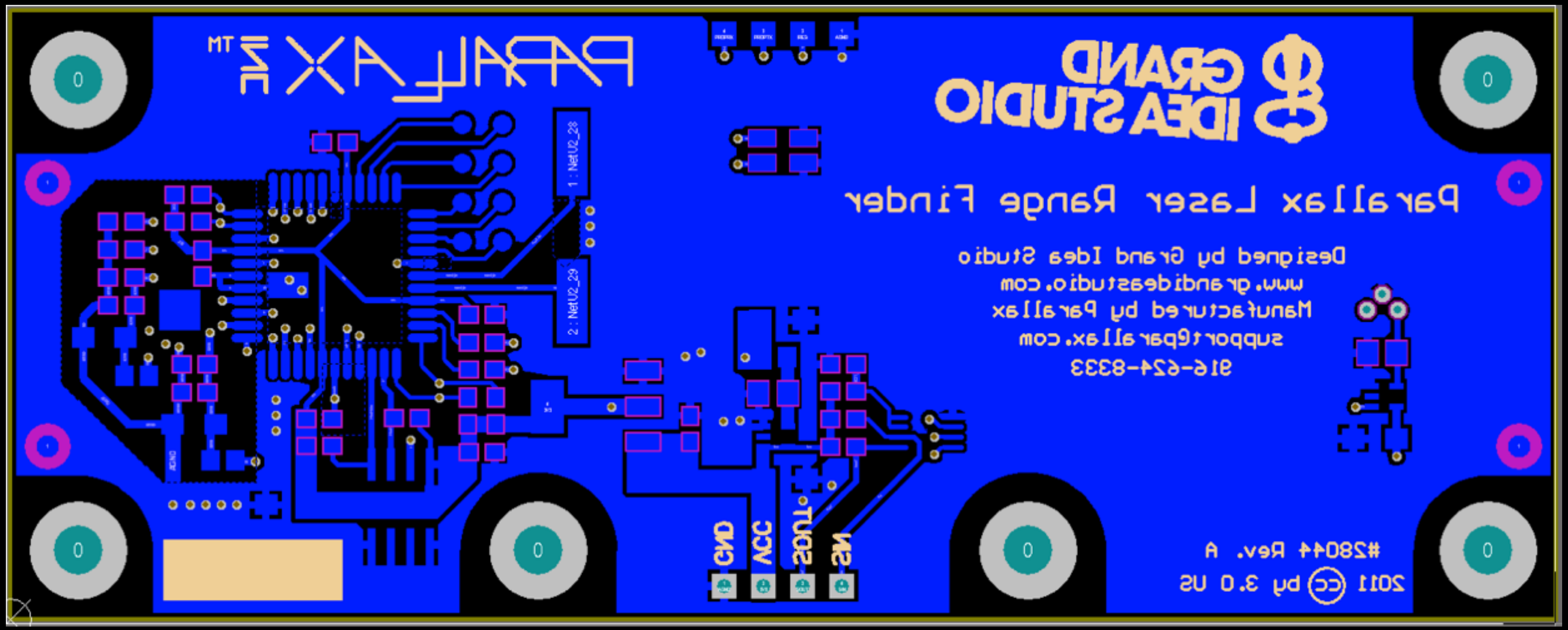
LRF Module: Back



LRF Module: Schematic

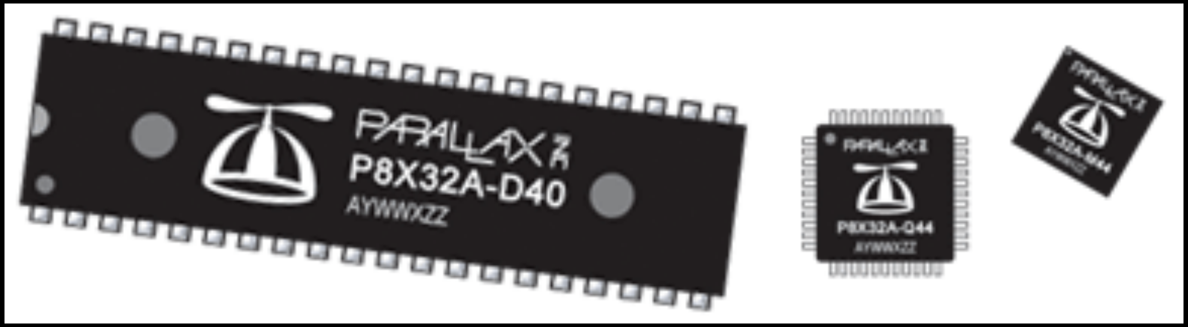


LRF Module: PCB Layout



Propeller

- Completely custom, ground up, open source
- Multicore: 8 parallel 32-bit processors (cogs)
- Code in Spin, ASM, or C



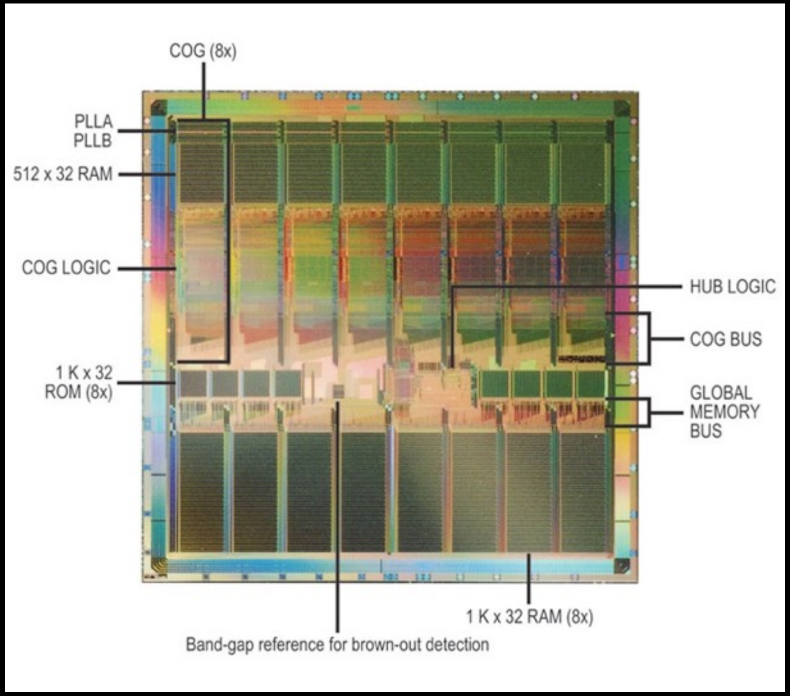
*** INFORMATION: **`www.parallax.com/propeller/`**

*** DISCUSSION FORUMS: **`http://forums.parallax.com`**

*** OBJECT EXCHANGE: **`http://obex.parallax.com`**

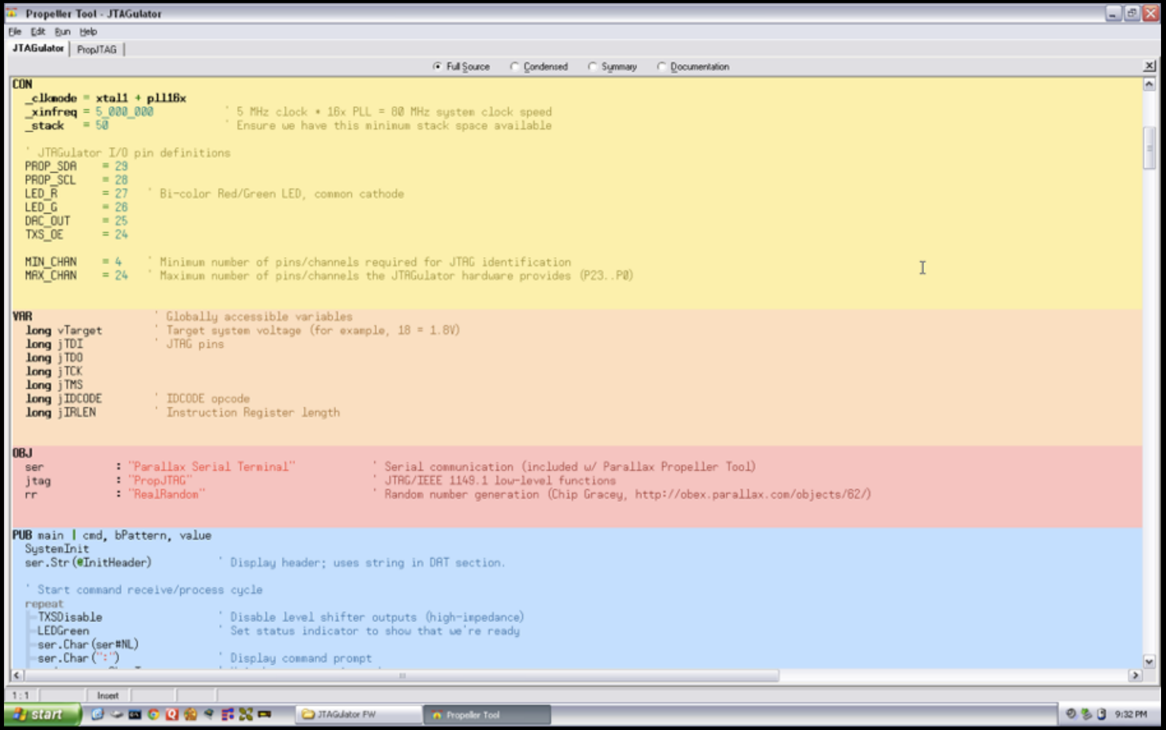
Propeller 2

- Clock: DC to 128MHz (80MHz recommended)
- Global (hub) memory: 32KB RAM, 32KB ROM
- Cog memory: 2KB RAM each
- GPIO: 32 @ 40mA sink/source per pin
- Program code loaded from external EEPROM on power-up



Propeller 3

- Standard development using Propeller Tool & Parallax Serial Terminal (Windows)
  - **`www.parallax.com/downloads/propeller-p8x32a-software`**
- Programmable via serial interface

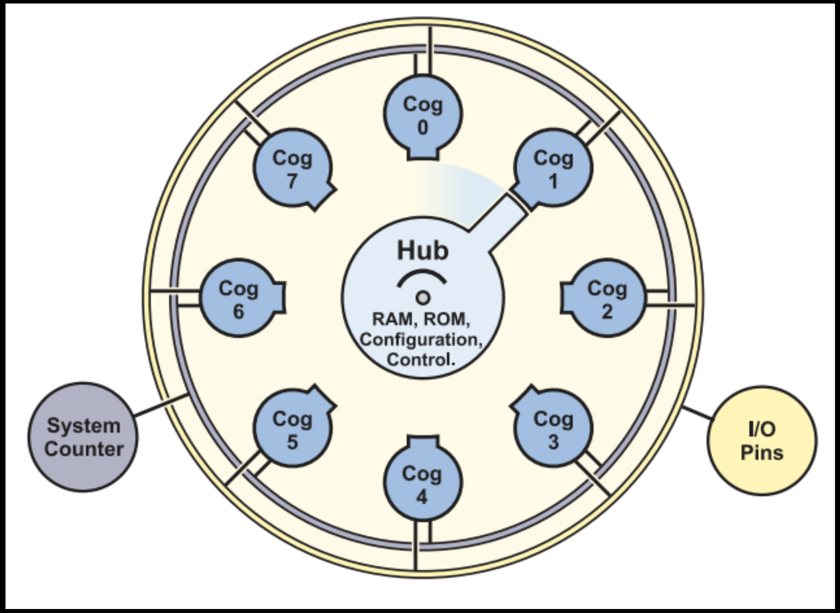


Source Tree

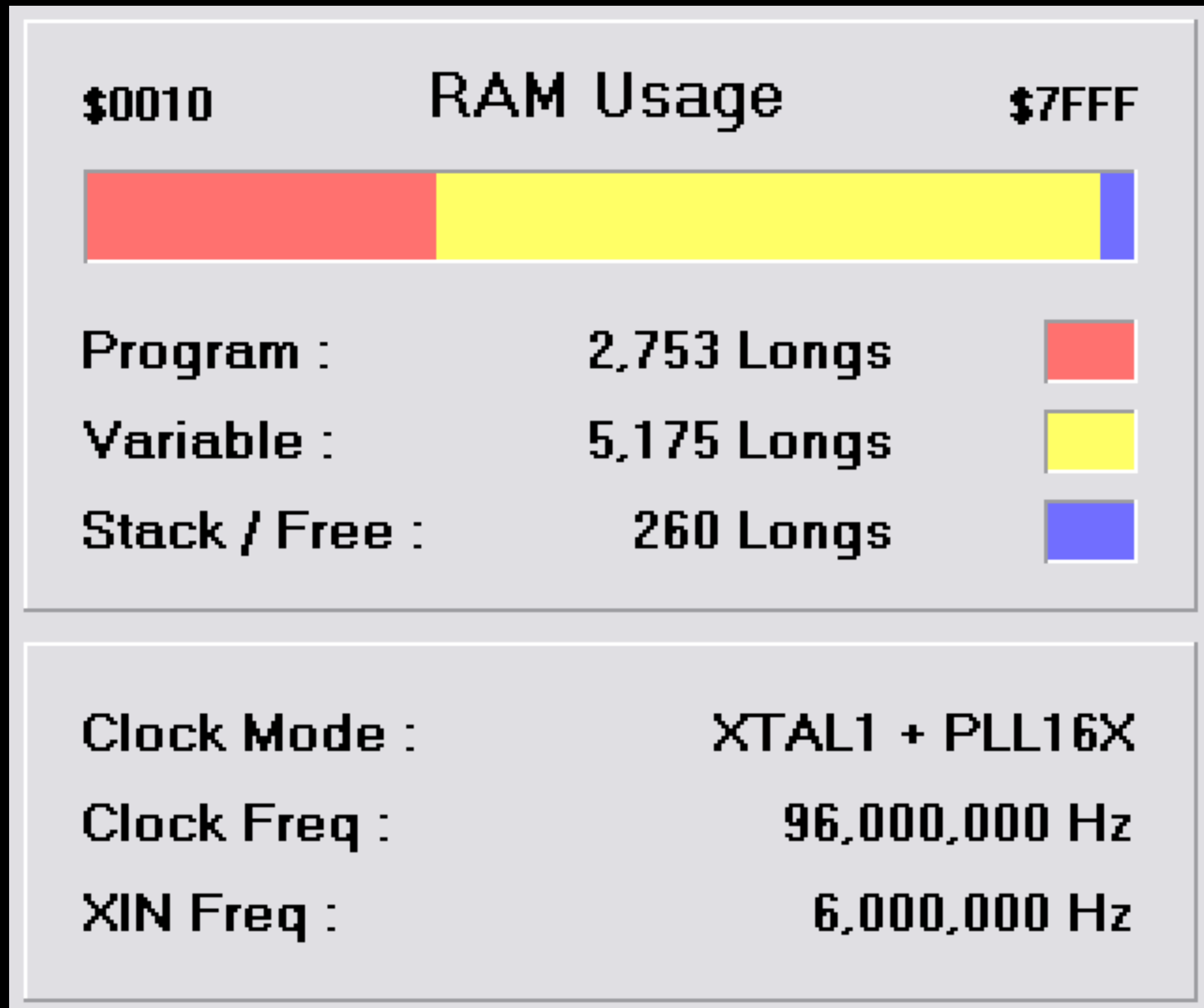
```
LRF_OVM7690.spin                         Top Object
├─LRF_con.spin                           Global Constants
├─OVM7690_obj.spin                       OVM7690 Interface
│ ├─LRF_con.spin
│ ├─OVM7690_fg.spin                      Frame Grabber (160x128)
│ │ └─LRF_con.spin
│ ├─OVM7690_fg_roi.spin                  Frame Grabber (320x16)
│ │ └─LRF_con.spin
│ ├─pasm_i2c_driver_Lite.spin            I2C (SCCB)
│ └─Synth.spin                           Frequency Synthesizer
├─JDCogSerial_Lite.spin                  Async. Serial Interface
├─F32_Lite.spin                          Floating Point Math
├─FloatString_Lite.spin                  Floating Point Display
│ └─FloatMath_Lite.spin
└─Basic_I2C_Driver_Lite.spin             I2C (EEPROM)
```

Cogs

- Spin Interpreter (Cog 0)
- Auto-Baud Detection (start-up only)
- Full-Duplex Serial (JDCogSerial)
- Floating Point (F32)
- I2C for OVM7690 SCCB interface (pasm_i2c_driver)
- OVM7690 Frame Grabbers (on request)



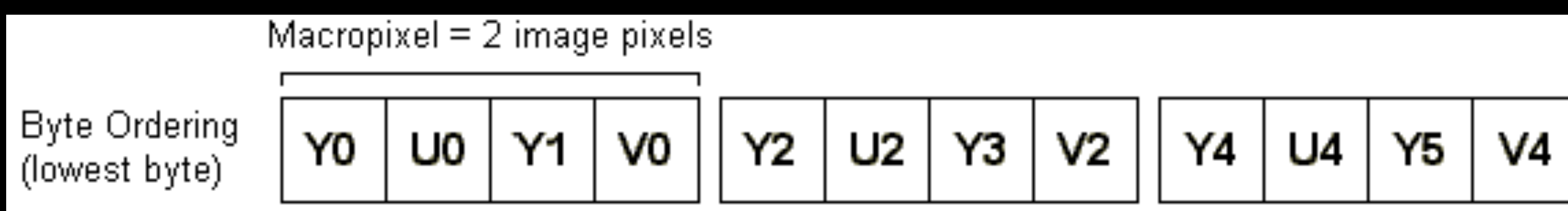
Propeller Resources



OVM7690 Camera Interface

- DVP[7:0] (Digital video port)
- VSYNC (Vertical Sync)
- HREF (Horizontal Reference)
- PCLK (Pixel Clock)

YUV422 color space @ 16 bits/pixel



www.fourcc.org/yuv.php#YUY2

OVM7690 Frame Grabber

- Custom frame grabbers written in PASM
- Launched on demand depending on command
- Used PASD to help debug
  - `www.insonix.ch/propeller/prop_pasd.html`

OVM7690 Frame Grabber 2

1. Full (ovm7690_fg)
    - 160 x 128 x 8bpp (greyscale)
    - Useful for testing, taking low-res photos
2. ROI (Region of Interest, ovm7690_fg_roi)
    - 320 x 16 x 8bpp (greyscale)
    - Better resolution for actual range finding
    - Handles preliminary image processing (on request)
        - Double frame grab w/ laser off/on
        - Background subtraction, thresholding, column sum

OVM7690 Frame Grabber 3

1. Start cog
2. Grab frame
   - 8 bits at a time
3. Preliminary image processing (if requested)
4. When done, set flag in hub RAM to non-zero
5. Cog self-destruct

OVM7690 Frame Grabber 4

- Extremely timing sensitive
  - Propeller overclocked to 96MHz
  - Only had 24 cycles to grab/store each byte
    - 6 instructions @ 4 cycles each!

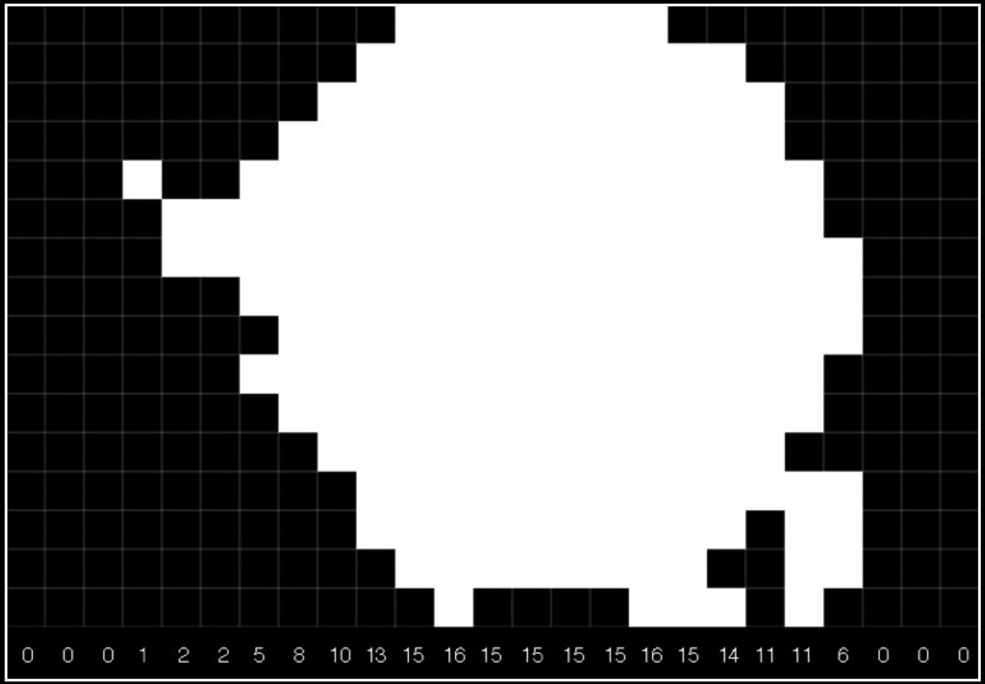
```
640x480 (VGA) @ 10fps (8MHz PCLK)
----------------------------------                          @96MHz
VSYNC width                                    = 782.5uS  = 75095 cycles
Time from VSYNC low to HREF high               = 3.9325mS = 377399 cycles
Time in between lines/HREF                     = 35uS     = 3358 cycles
Time from last HREF in frame to next VSYNC     = 1.555mS  = 149232 cycles
Pixel clock (PCLK)                             = 0.125uS  = 12 cycles/bit
                       (must grab data within 6 cycles of PCLK going high)
```

```
  Timing diagram @ 96MHz Propeller
         12 cycles/bit
  Data valid when PCLK is HIGH

     Y       U/V       Y   ...
   _/‾‾‾‾\___/‾‾‾‾\___/‾‾‾‾\___
   ↑    ↑    ↑    ↑    ↑
  t=0   6    12   18   24
 cycles
```

Image Processing

1. Background Subtraction
2. Thresholding
3. Column Sum
4. Blob Detection
5. Mass/Centroid Calculation(s)
6. Select Primary Blob



Command Interface

- TTL-level serial interface
- ASCII commands/responses
- Auto-baud rate detection (300-115.2kbps)
- Four physical connections:
  1. GND
  2. VCC
  3. SOUT (Serial Out)
  4. SIN (Serial In)

Basic Commands (FW 2.0)

- Single range measurement (in mm, decimal)
- Single range measurement (in mm, binary)
- Repeated range measurement
- Adjust camera for current lighting conditions
- Reset camera to initial settings
- Toggle laser on/off
- Display version information
- Display available commands

Advanced Commands (FW 2.0)

- Display coordinate, mass, and centroid for all blobs
- Calibrate camera system for range finding
- Adjust blob detection parameters
- Capture & send single frame (160x128)
- Capture & send single frame (320x16) w/ laser enabled
- Capture & send processed frame (320x16) w/ background subtraction

Calibration

- Required during production to account for manufacturing variances (camera and laser diode alignment)
- Required after major firmware update
- Done "automatically" using 'X' command
  1. Take a number of measurements from known distances
  2. Record *pfc* value and actual angle at each distance
  3. Calculate slope & intercept values
  4. Store calibration data in unused portion of boot Serial EEPROM

Measurement Results (cm)

Actual Distance to Target (cm)	Calculated Distance (cm)	Difference (Δ)	% Error
20	19.9	0.1	-0.50
30	29.7	0.3	-1.00
40	40.1	-0.1	0.25
50	50.3	-0.3	0.60
60	60.2	-0.2	0.33
70	70.8	-0.8	1.14

Average % Error

0.64

Prototype unit, serial #0

Measurement Results (in)

Actual Distance to Target (in)	Calculated Distance (in)	Difference (Δ)	% Error
10	9.9	0.1	-1.00
20	20.1	-0.1	0.50
30	30.7	-0.7	2.33
40	40.3	-0.3	0.75
50	48.8	1.2	-2.40
75	70.3	4.7	-6.27

Average % Error

2.21

Prototype unit, serial #0

Key Specifications

- Optimal measurement range: 6-48 in. (4 ft.)
- Accuracy error: < 5% (typically much better)
- Sample rate: 5Hz
- Power: 5V @ 150mA
- Operating temperature: 32-122 °F (0-50°C)
- Dimensions: 3.95" W x 1.55" H x 0.67" D

Limitations

- Range
    - Longer distances will result in a noticeable reduction in accuracy due to very slight changes in angle
    - Firmware limits maximum distance to 100"
- Environment
    - Works best in a controlled environment w/ minimal changes in brightness (e.g., indoors)
    - Not reliable against bright targets, as background subtraction will remove the bright spot from the frame (including the laser)

Demonstrations

- Terminal Program
- LRF Image Viewer (VB.Net)
- BASIC Stamp II
- Arduino
- FSLBOT (MCF52259)
- LRF + Nintendo Game Boy Printer

Get It

- parallax.com/product/28044

  *** Assembled units, example code, documentation

- grandideastudio.com/portfolio/laser-range-finder

  *** Schematics, BOM, videos, other documentation

- github.com/grandideastudio/laser-range-finder

  *** Source code


DANGER
The End.