Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

Parallax Laser Range Finder (#28044)

Designed in conjunction with Grand Idea Studio (www.grandideastudio.com), the Parallax Laser Range Finder (LRF) Module is a distance-measuring instrument that uses laser technology to calculate the distance to a targeted object. The design uses a Propeller processor, CMOS camera, and laser diode to create a low-cost laser range finder. Distance to a targeted object is calculated by optical triangulation using simple trigonometry between the centroid of laser light, camera, and object.

Features

- Optimal measurement range of 6–48 inches (15–122 cm) with an accuracy error <5%, average 3%
- Maximum object detection distance of ~8 feet (2.4 meters)
- Compact module with integrated CMOS camera and laser system
- Single row, 4-pin, 0.1" header for easy connection to a host system
- All engineering materials released as open source under a Creative Commons license; see page 11

Key Specifications

- Power Requirements: 5 VDC @ 150 mA
- Communication: Asynchronous serial 300–115,200 baud with automatic baud rate detection
- Operating temperature: 32 to 122 °F (0 to 50 °C)
- Dimensions: 3.95" W x 1.55" H x 0.67" D (10.05 W x 3.95 H x 1.7 D cm)

Application Ideas

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

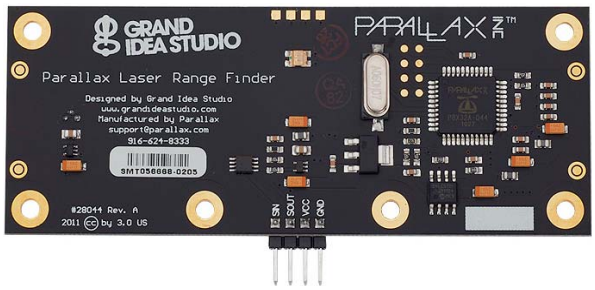


Table of Contents

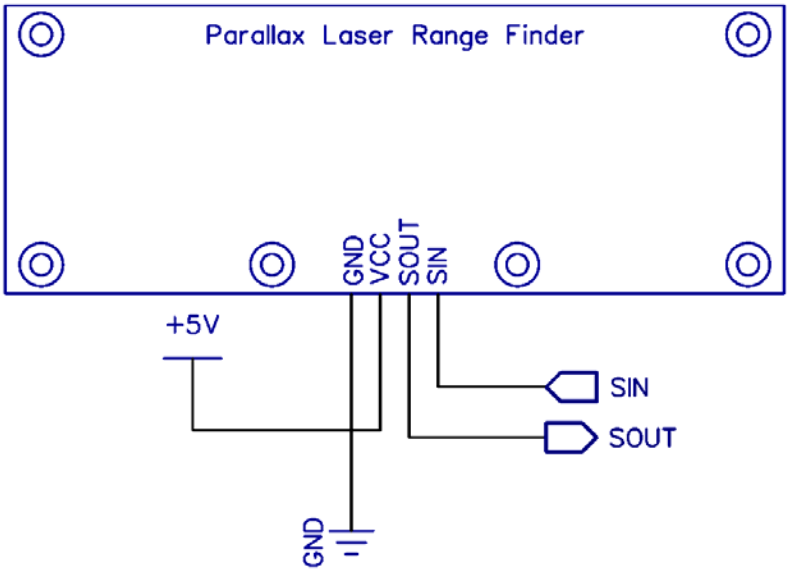
Connections

The LRF Module interfaces to any host microcontroller or computer system using only four connections (GND, VCC, SOUT, SIN).

Pin	Name	Type	Function
1	GND	G	System ground. Connect to power supply's ground (GND) terminal.
2	VCC	P	System power, 5 VDC input.
3	SOUT	O	Serial output ***to*** host. 5 V TTL-level interface, non-inverted, 8 data bits, no parity, 1 stop bit, baud rate matched to host. Note: The LRF Module hardware design provides bidirectional communication capability for the SOUT pin. This allows the use of a standard three-pin servo extension cable (Parallax #805-00001) for connecting the LRF Module to the host. However, bidirectional communication requires changes to the LRF Module firmware, not currently supported by Parallax.
4	SIN	I	Serial input ***from*** host. 3.3 V to 5 V TTL-level interface, non-inverted, 8 data bits, no parity, 1 stop bit, baud rate matched to host.

Type: I = Input, O = Output, P = Power, G = Ground

Use the following example circuit for connecting the Parallax Laser Range Finder Module:



Usage

The LRF Module is controlled by the host via a serial communications interface. To use, simply align the LRF Module towards the target object and send the desired command.

The serial interface is configured for 8 data bits, no parity, 1 stop bit (8N1). Automatic baud rate detection occurs on initial power-up of the LRF Module. The Module waits for a "U" character to be sent

by the host and will set its baud rate to match that of the host. Supported baud rates include 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200.

When the LRF is ready to receive commands, it will send a ":" to the host. The LRF waits in an idle state until it receives a valid command, at which time it performs the command and returns command-specific data (if any). The LRF will return a "?" upon receiving an invalid command.

The camera system used in the LRF Module has automatic white balance, automatic exposure, and automatic gain control enabled by default, and will automatically adjust its image to account for sudden changes in lighting conditions. However, the LRF works best in a controlled environment, such as indoors with minimal changes in brightness across the frame. The LRF is also less reliable when the laser is shining onto a bright object (for example, a white piece of paper), since the background subtraction done during image processing could potentially "subtract" the bright laser from the already bright frame. Giving the camera time for its automatic white balance and automatic exposure to settle helps a bit to make the laser spot stand out. Using a red filter over the camera will also help the red laser spot become more visible to the camera in certain situations. See the "E" command in the Command Set section below, and Range and Accuracy, page 8, for more details.

Status Indicator

A visual indication of the LRF Module's operating state is given with the on-board LED (Light-Emitting Diode). The LED is located on the back side of the LRF near the center of the board. The LED denotes four states of the Module:

1. **Green: Idle state**. Waiting for a valid command to be sent by the host.
2. **Red: Active state**. For example, performing a range calculation or capturing an image with the camera.
3. **Orange (Solid)**: Baud rate detection state. The LRF is waiting for a "U" character to be sent by the host in order to automatically set the communications baud rate. Occurs on LRF power-up only. See, Usage, page 2, for more details.
4. **Orange (Blinking)**: Error state. The LRF has malfunctioned. A message identifying the failed operation will be transmitted on the SOUT (Serial Out) pin. See Error Modes, page 9, for more details.

If the LED is OFF, the LRF may not be receiving power.

Command Set

All commands are single-byte, ASCII printable values and are not case-sensitive (upper case and lower case will both work). The basic and advanced command sets are described below.

Basic Commands	
R	Single range measurement (returns a 4-digit decimal value in millimeters)
B	Single range measurement (returns a 4-byte binary value in millimeters)
L	Repeated range measurement (any subsequent byte will stop the loop)
E	Adjust camera for current lighting conditions
S	Reset camera to initial settings
V	Print version information
H	Print list of available commands

Advanced Commands	
O	Display coordinate, mass, and centroid information for all detected blobs
X	Calibrate camera system for range finding
G	Capture & send single frame (8 bits/pixel grayscale @ 160x128)
C	Capture & send single frame (16 bits/pixel YUV422 color @ 640x16) w/ laser enabled
P	Capture & send image processed frame (16 bits/pixel YUV422 color @ 640x16) w/ background subtraction

Command Details

R: Single range measurement (decimal)

Takes a single range finding measurement and returns the distance to the target object as a printable ASCII string ("D = ") with a *4-digit decimal* value in millimeters.

The LRF Module is most accurate within its optimal measurement range of 6–48 inches (15–122 cm). In ideal conditions, a distance measurement of up to ~8 feet (2.4 meters) may be possible. While the accuracy will be hindered at distances outside of the optimal range, the LRF could still be used for gross distance measurements or simple object detection. See Range and Accuracy, page 8, for more details.

Example:

```
:R
D = 0288 mm
:
```

B: Single range measurement (binary)

Takes a single range finding measurement and returns the distance to the target object as a *4-byte binary* value in millimeters. Data is sent MSB first.

The LRF Module is most accurate within its optimal measurement range of 6–48 inches (15–122 cm). In ideal conditions, a distance measurement of up to ~8 feet (2.4 meters) may be possible. While the accuracy will be hindered at distances outside of the optimal range, the LRF could still be used for gross distance measurements or simple object detection. See Range and Accuracy, page 8, for more details.

Example:

```
:B
<binary data>
:
```

L: Repeated range measurement

Continuously calls the Single range measurement (decimal) command ("R"). Once the loop has started, any byte sent to the LRF Module will stop the command.

Example:

```
:L
D = 0288 mm
D = 0289 mm
D = 0289 mm
D = 0289 mm
D = 0289 mm
D = 0289 mm
:
```

E: Adjust camera for current lighting conditions

Calibrates the LRF Module's camera for the current lighting conditions. This may aid the camera in successfully detecting the laser spot within the frame, which increases accuracy of the range finding functionality.

The command first ensures that automatic white balance (AWB), automatic exposure control (AEC), and automatic gain control (AGC) are enabled. Then, after a 10 second delay for the camera image to settle, AWB, AEC, and AGC are disabled. The EV/exposure level is also reduced to a minimum value. This will cause the entire camera frame, including any previously bright areas, to appear dark. A bright laser spot may now be more easily identifiable within the frame.

Example:

```
:E
<short delay>
:
```

The camera's settings can be reset using the "S" command.

S: Reset camera to initial settings

Resets and initializes the LRF Module's camera to its default, power-up configuration, including enabling automatic white balance (AWB), automatic exposure control (AEC), and automatic gain control (AGC). It may take up to 10 seconds for the camera image to settle after reset.

This command is particularly helpful when the "E" command has been used to adjust the camera for current lighting conditions and the user wishes to reset the camera settings without power cycling the LRF Module.

Example:

```
:S
:
```

V: Print version information

Lists version and calibration information for the LRF Module. This data is useful for troubleshooting and debugging.

- FW: Firmware revision (major.minor)
- MFG and PID: Manufacturer ID and Product ID of the LRF's on-board camera
- SLOPE, INTERCEPT, PFC_MIN: Device-specific values calculated during the calibration process ("X") and used for range finding (if the LRF Module is uncalibrated, the values will all be 0xFFFFFFFF)

Example:

```
:V
Parallax Laser Range Finder
Designed by Grand Idea Studio [www.grandideastudio.com]
Manufactured and distributed by Parallax [support@parallax.com]

FW = 1.0
MFG = 7FA2
PID = 7691
SLOPE = +0.001413373 (3AB940EC)
INT = -0.01235497 (BC4A6C80)
PFC_MIN = 30
:
```

H: Print list of available commands

Lists all available commands that the LRF Module supports.

Example:

```
:H
Basic Commands:
R   Single range measurement
B   Single range measurement (binary response, 4 bytes)
< more commands listed, but not shown in this manual >
:
```

O: Display coordinate, mass, and centroid information for all detected blobs

Displays coordinate, mass, and centroid (center of mass) information for up to 6 detected blobs within the camera's field-of-view. This information can be used for custom image processing or object detection outside of the standard LRF Module functionality. The blob detection begins on the left side of the frame and "scans" to the right. See Image Processing and Blob Detection, page 18, for details.

- L: X coordinate of the beginning (left side) of the detected blob
- R: X coordinate of the end (right side) of the detected blob
- M: Mass of blob (sum of all valid pixels within the blob)
- C: Centroid (center of mass) of blob

In some cases, only a single blob is detected within the frame and is typically the laser spot. In other cases, there may be reflections of the laser light or other spots that are not related to the laser. Generally, the blob with the largest mass within the frame can be considered the actual laser spot.

Example:

```
:O
0: L = 81 R = 88 M = 38 C = 84
1: L = 137 R = 232 M = 917 C = 181
2: L = 235 R = 254 M = 170 C = 244
:
```

X: Calibrate camera system for range finding

To account for manufacturing and assembly variances, particularly related to the camera and laser diode alignments, each LRF Module must be calibrated. This occurs during production, but the LRF can be recalibrated by the user at a later date if desired.

The calibration routine requires the user to place the LRF Module at 6 fixed distances (from 20 cm to 70 cm at 10 cm intervals). The LRF takes measurements at each distance and calculates the SLOPE, INTERCEPT, and PFC_MIN values (the routine is based lightly on the one found at www.eng.umd.edu/~nsw/ench250/slope.htm). The values are then stored in an unused portion of the non-volatile boot Serial EEPROM. They will remain intact even if the LRF firmware is re-programmed onto the Module.

The SLOPE and INTERCEPT are used to convert the pixel offset to angle using a best-fit slope-intercept linear equation. The PFC_MIN value is used to set the maximum allowable distance of the LRF Module, which is represented by a minimum pixels from center value. See Optical Triangulation, page 15, for more details.

A video demonstrating the calibration process can be found on YouTube:
www.youtube.com/watch?v=1gk_tRbJO84

Example:

```
:X
Are you sure you want to calibrate (Y/N)?Y
Set LRF to D = 20 cm and press spacebar (any other key to abort)
pfc: 246 angle: 0.3718561
pfc: 245 angle: 0.3718561
pfc: 244 angle: 0.3718561
pfc: 244 angle: 0.3718561

< more steps listed, but not shown in this manual >

Set LRF to D = 70 cm and press spacebar (any other key to abort)
pfc: 66 angle: 0.1109708
pfc: 65 angle: 0.1109708
pfc: 65 angle: 0.1109708
pfc: 67 angle: 0.1109708

SLOPE = +0.001450448 (3ABE1CF8)
INT = +0.01748975 (3C8F46A8)
PFC_MIN = 9

Write new values (Y/N)?Y
:
```

G: Capture & send single frame (8 bits/pixel grayscale @ 160x128)

Capture a 160x128 resolution grayscale image with the LRF Module's camera and return the data in a binary format.

The data is sent MSB first, one byte per pixel, starting at the upper-left pixel location (0,0), moving left to right, top to bottom across the frame, and ending at the lower-right pixel location (160,128). Each byte corresponds to the brightness value of a single pixel where 0x00 is black and 0xFF is white.

A total of 20,480 bytes of binary data is sent. An ASCII footer ("END") is then attached to the end of the binary data stream to assist in identifying the end of frame.

This command can be used from within the LRF Image Viewer tool to easily view the image.

Example:

```
:G
<binary data>END
:
```

C: Capture & send single frame (16 bits/pixel YUV422 color @ 640x16)

Capture a 640x16 resolution image with the LRF Module's camera and return the data in a binary format. The laser diode is enabled during the frame grab for testing and alignment purposes.

The data is sent MSB first, two bytes per pixel, starting at the upper-left pixel location (0,0), moving left to right, top to bottom across the frame, and ending at the lower-right pixel location (640,16). Every two bytes corresponds to a single pixel. The pixel format is YUY2 (www.fourcc.org/yuv.php), a subset of YUV422 formatting in which each 16-bit pixel is given an 8-bit Y component (luma/brightness) and alternating 8-bit U or 8-bit V component (chroma). See Camera Interface, page 17, for more details.

A total of 20,480 bytes of binary data is sent. An ASCII footer ("END") is then attached to the end of the binary data stream to assist in identifying the end of frame.

This command can be used from within the LRF Image Viewer tool to easily view the image. YUVTools (www.sunrayimage.com) can also be used to convert the raw binary data into a YUV422-decoded color image.

Example:

```
:C
<binary data>END
:
```

P: Capture & send image processed frame (16 bits/pixel YUV422 color @ 640x16)

Capture a 640x16 resolution "processed" image with the LRF Module's camera and return the data in a binary format. This mode is specific for range finding functionality and consists of a background subtracted image where one frame is taken with the laser diode off, one frame taken with laser diode on, and the data subtracted to help isolate the laser spot from the rest of the image frame. See Image Processing and Blob Detection, page 18, for more details.

The data is sent MSB first, two bytes per pixel, starting at the upper-left pixel location (0,0), moving left to right, top to bottom across the frame, and ending at the lower-right pixel location (640,16). Every two bytes corresponds to a single pixel. The pixel format is YUY2 (www.fourcc.org/yuv.php), a subset of YUV422 formatting in which each 16-bit pixel is given an 8-bit Y component (luma/brightness) and alternating 8-bit U or 8-bit V component (chroma). See Camera Interface, page 17, for more details.

A total of 20,480 bytes of binary data is sent. An ASCII footer ("END") is then attached to the end of the binary data stream to assist in identifying the end of frame.

This command can be used from within the LRF Image Viewer tool to easily view the image. YUVTools (www.sunrayimage.com) can also be used to convert the raw binary data into a YUV422-decoded color image.

Example:

```
:P
<binary data>END
:
```

Range and Accuracy

The LRF Module is most accurate within its optimal measurement range of 6–48 inches (15–122 cm). Within this range, error can span from zero (no difference between actual distance and the distance calculated by the LRF) to 5%. On average, the error is approximately 3%.

Within the optimal measurement range, the horizontal position of the laser spot within the camera's frame (which is used to determine distance to the target object) changes noticeably. At longer distances, although the camera can still "see" the laser spot, the horizontal position does not change as much, causing a significant reduction in accuracy. Accuracy also varies with lighting conditions and material of the target object, which affect the LRF Module's capability to determine the laser spot within the camera's frame.

The LRF Module intentionally limits the maximum detectable distance to ~8 feet (100 inches). At distances less than 6 inches, the laser spot is out of the camera's field-of-view, so no range calculation can occur. See Theory of Operation, page 14, for range finding and image processing details.

The following chart shows example measurements taken with a LRF Module. The Difference (Δ) and % Error will vary per unit.

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

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

Error Modes

In the event that the LRF enters the error state (signified by a blinking orange LED status indicator), a message identifying the failed operation will be transmitted on the SOUT (Serial Out) pin.

Messages include:

- **ERR: cam.start**. Error initializing the camera on LRF power-up. This may be caused by a communication error between the Propeller and camera.
- **ERR: cam.init**. Error performing the "S" command (Reset camera to initial settings). This may be caused by a communication error between the Propeller and camera.
- **ERR: cam.calibrate**. Error performing the "E" command (Adjust camera for current lighting conditions). This may be caused by a communication error between the Propeller and camera.
- **ERR: cam.setRes**. Error setting the camera's resolution. This may be caused by a communication error between the Propeller and camera.
- **ERR: cam.getID**. Error retrieving the camera's manufacturer and product IDs. This may be caused by a communication error between the Propeller and camera.
- **ERR: eeprom.ReadLong**. Error reading calibration data from the external Serial EEPROM. This may be caused by a communication error between the Propeller and EEPROM or an incorrect type of EEPROM device.
- **ERR: eeprom.WriteLong**. Error writing calibration data to the external Serial EEPROM. This may be caused by a communication error between the Propeller and EEPROM or an incorrect type of EEPROM device.

If the LRF is in an error state, but no error message is transmitted, then there is a failure either with starting the JDCogSerial serial communication cog or the auto baud detection cog.

For further assistance, please contact Parallax technical support.

Electrical Characteristics

At V_{CC} = +5.0 V and T_A = 25 ºC unless otherwise noted.

Parameter	Symbol	Specification			Unit
		Min.	Typical	Max.	
Supply Voltage	V_{CC}	4.5	5.0	5.5	V
Supply Current, Idle	I_{IDLE}	---	82	---	mA
Supply Current, Active	I_{CC}	---	126	150	mA

Absolute Maximum Ratings

Condition	Value
Operating Temperature	0 ºC to +50 ºC (+32 ºF to +122 ºF)
Storage Temperature	-40 ºC to +125 ºC (-40 ºF to +257 ºF)
Supply Voltage (V_{CC})	+4.5 V to +5.5 V
Ground Voltage (V_{SS})	0 V
Voltage on any pin with respect to V_{SS}	-0.3 V to +7.0 V

NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

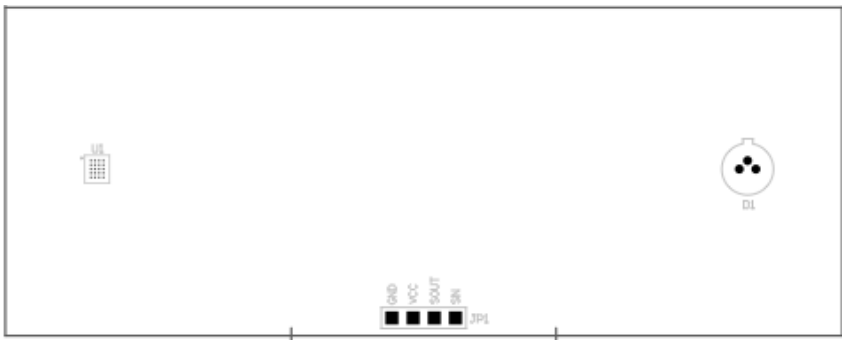
Hardware Design

The LRF Module's major hardware components include:

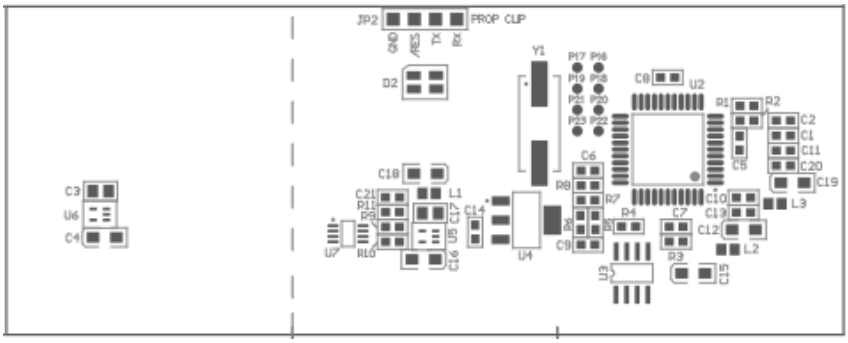
- Parallax Propeller P8X32A-Q44 (www.parallax.com/propeller)
- OmniVision OVM7690 640x480 CMOS CameraCube (www.ovt.com/products/sensor.php?id=45)
- Arima APCD-635-02-C3-A Laser Diode (www.arimalasers.com/en/products02.aspx)

Assembly Drawing

All components are mounted on the back side of the board with the exception of the camera and laser diode. The center points of the camera and laser diode are 78 mm apart, as shown below.



Referring to the back side of the LRF Module below, the Parallax Propeller (U2), its supporting electronics, and the camera interface circuitry are located on the right and the laser diode control circuitry is on the left. A dotted line (not printed on the actual LRF Module) indicates the recommended cutting point if the user wishes to separate the laser diode control circuitry from the rest of the LRF for custom modifications or projects.



A Prop Clip (Parallax #32200) or Prop Plug (Parallax #32201) can attach to the four pads (JP2) on the back of the LRF for reprogramming of the Propeller's firmware.

Eight unused GPIO pins (P16-P23) are available via surface mount pads near the Propeller processor. The pins are spaced to support a dual-row, 0.1" male header.

Open Source Files

All engineering materials are released as open source under a Creative Commons Attribution 3.0 United States license (http://creativecommons.org/licenses/by/3.0/us/), allowing free distribution and reuse. The following items are available from the 28044 product page at www.parallax.com:

- Schematic
- Firmware for Propeller processor
- Bill-of-Materials
- PCB Layout/Gerber Plots
- Mechanical Drawing
- Example Code for BASIC Stamp 2
- Example Code for Propeller P8X32A
- LRF Image Viewer Software for Windows

Additional Resources

- Grand Idea Studio's *Laser Range Finder* web page: www.grandideastudio.com/portfolio/laser-range-finder/
- Demonstration videos of the LRF Module: www.youtube.com/watch?v=MqYRM6calOI
- Sam's Laser FAQ: www.repairfaq.org/sam/lasersam.htm

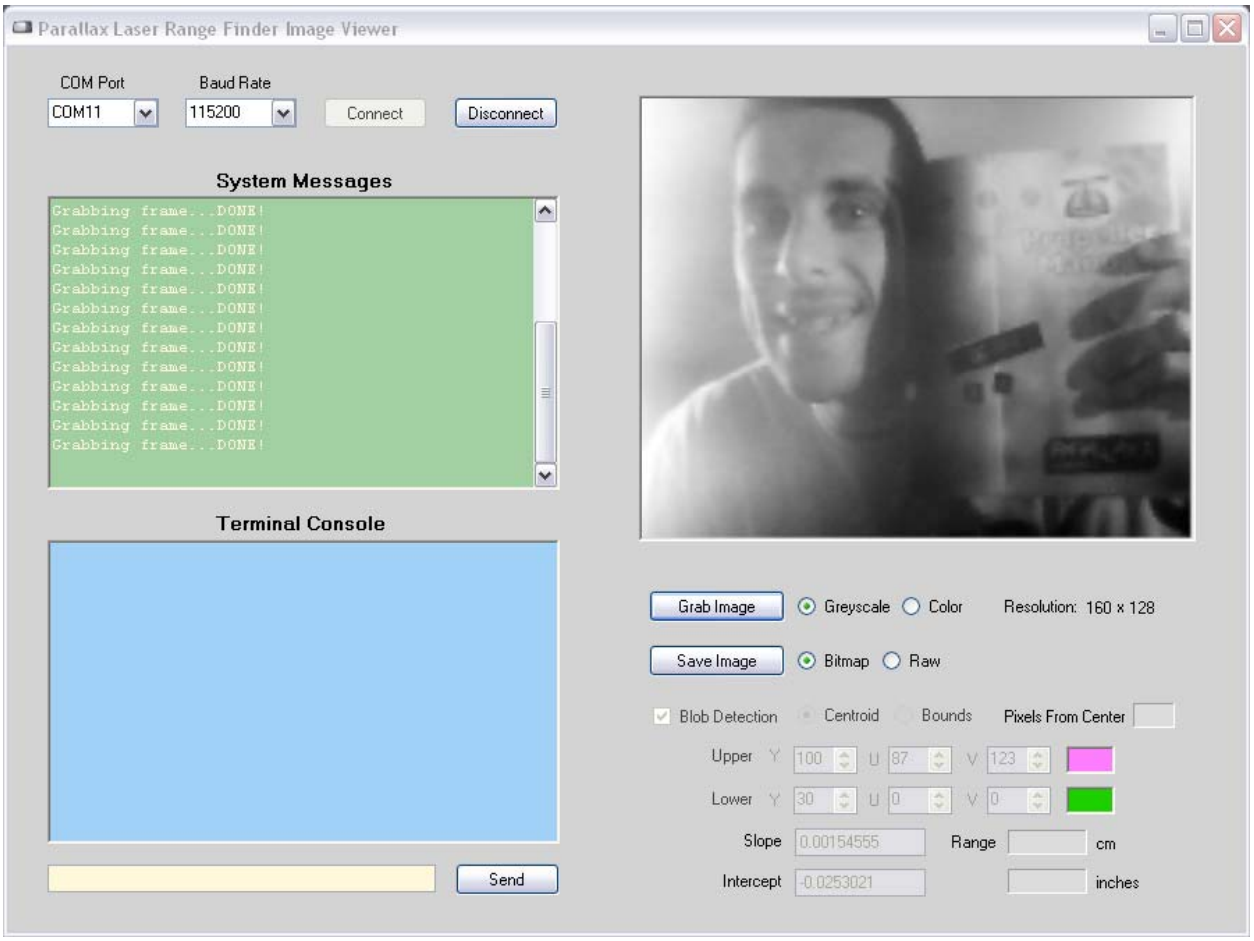
LRF Image Viewer

The LRF Image Viewer is an easy-to-use, PC-based graphical interface that allows direct and simple control of the LRF Module. The primary features include:

- Read system/debug messages sent from the LRF
- Send commands to the LRF
- Capture/display/save images from the LRF's camera (grayscale or color, bitmap or raw binary format)
- Enable PC-side image processing functionality (blob detection and identification, range/distance calculations)

The LRF Image Viewer application was designed in Microsoft Visual Basic .NET. It requires Microsoft's .NET Framework Version 2.0 Redistributable Package to be installed on the host PC: www.microsoft.com/download/en/details.aspx?displaylang=en&id=19

The LRF Module connects to the host PC via its serial interface through a USB-to-Serial adapter. A demonstration of the LRF Image Viewer can be found on YouTube: www.youtube.com/watch?v=iHvMl2scUdA



LRF Image Viewer Interface

The LRF Image Viewer has a variety of windows and controls:

COM Port menu: Provides a selection of COM ports available on the host PC (includes any virtual serial ports provided by common USB-to-Serial interfaces)

Baud Rate menu: Provides a list of standard baud rates (300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200).

Connect/Disconnect buttons: Opens/closes a serial connection on the specified COM port at the selected baud rate. After opening a serial connection with the Connect button, the LRF Image Viewer will send the required "U" character to allow the LRF Module to automatically detect the host PC's baud rate.

System Messages window: Displays operational or debug messages sent by the LRF Image Viewer software.

Terminal Console window: Displays the data sent to and received from the LRF Module via its serial interface.

Text Entry box: This yellow text box beneath the Terminal Console allows the user to type commands and send them directly to the LRF Module. The text is only transmitted when the Send button (or Enter key) is pressed. Any data returned by the LRF will be displayed in the Terminal Console.

Image window: Displays an image captured by the LRF Module.

Grab Image button: Capture an image with the LRF Module's camera. The image will be displayed in the Image Window. The adjacent radio buttons are used to select a Grayscale (160x128 resolution) or Color (640x16) image. The button sends a "G" or "C" command, respectively. If the Blob Detection checkbox is checked, a "P" command is sent and the image will contain Centroid or Bounds markings.

Save Image button: Saves the image currently displayed in the Image Window. The adjacent radio buttons are used to select a file type of standard bitmap (.BMP) or raw binary. The raw binary format is for advanced users who wish to pipe the data into another program for post-processing or who don't want any bitmap headers or down-sampled color (which occurs when the camera's YUV422 image format is converted to the bitmap's RGB color space).

Blob Detection functionality: Enables the LRF Image Viewer to use its internal image processing and blob detection routines (instead of the functionality on-board the LRF Module) to identify pixels within a captured frame that match a set of criteria and to determine range to the target object. See Theory of Operation, page 14, for more details.

The Blob Detection checkbox is only available when the Color radio button is selected (located next to the Grab Image button). When the checkbox is selected, the adjacent controls become available.

The Upper and Lower controls are used to set the color bounds that a pixel must fall between in order for it to be displayed. The up/down arrows are used to adjust the individual Y, U, and V color components and the resulting color is displayed next to the controls. The default values will identify any pixel with a brightness (Y) between 0.3 and 1 of any color (U/V).

The Centroid and Bounds radio buttons are used to choose how to mark the primary blob if one is detected. The Centroid selection draws a red vertical line through the centroid (center of mass) of the blob. The Bounds selection draws red vertical lines at the beginning and end locations (left and right) of the blob.

The Pixels from Center text box displays the number of pixels the identified blob is away from the center point of the frame. This value is used for range finding calculations.

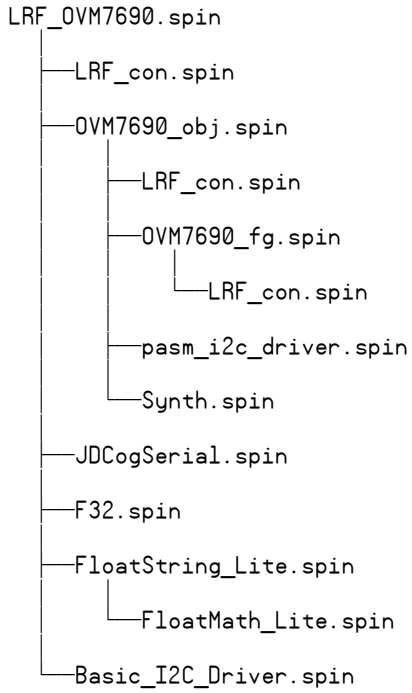
The Slope and Intercept text boxes allow the user to enter specific values to affect range finding calculations. The values will need to match those of the connected LRF Module in order for the LRF Image Viewer to provide accurate range finding results. The "V" command can be used to obtain the calibrated Slope and Intercept values of the LRF Module. The distance to the target object is displayed in centimeters and inches in the Range text boxes.

Theory of Operation

This section describes advanced details of the LRF Module operation.

Program Structure

The source tree for the LRF Module consists of 4 custom designed objects and 7 objects either included with the Parallax Propeller development environment or written by others and posted to the Parallax Object Exchange (http://obex.parallax.com):



LRF_OVM7690.spin is the top object file. This object handles the user interface, command processing, and laser range finding mathematics.

LRF_con.spin provides the global constants used throughout the program, including camera resolution, blob detection, range finding, and calibration settings.

OVM7690_obj.spin provides the low-level communication interface for the Omnivision OVM7690 CMOS CameraCube module.

pasm_i2c_driver.spin provides the I2C protocol interface for communication with the OVM7690. Written by Dave Hein and modified by Joe Grand for specific OVM7690 support. Original version available from: http://obex.parallax.com.

Synth.spin is a frequency synthesizer used to generate the input clock signal for the OVM7690. Included with the Parallax Propeller Tool.

OVM7690_fg.spin is the frame grabber object. This function is extremely timing sensitive, so it is written in Propeller Assembly (PASM). The object grabs a frame from the OVM7690 and stores it in the hub RAM frame buffer.

JDCogSerial.spin provides full-duplex serial communication. Written by Carl Jacobs and modified by Joe Grand for LRF-specific functionality. Original version available from: http://obex.parallax.com.

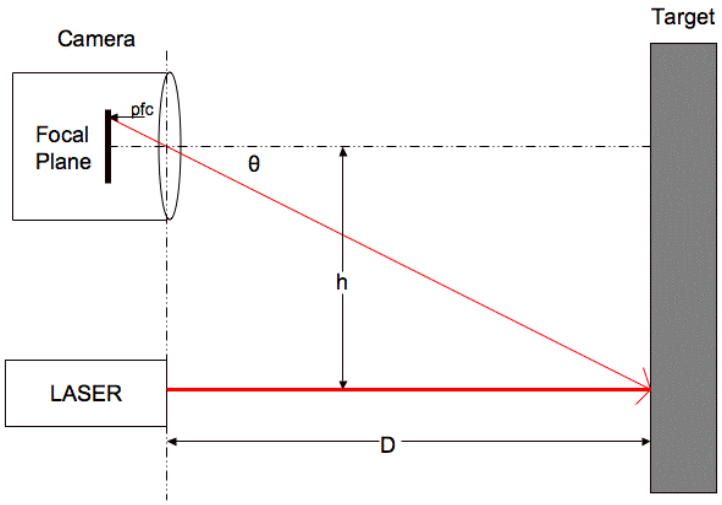
F32.spin provides IEEE 754 compliant 32-bit floating point math routines used for the laser range finding and calibration functionality. Written by Jonathan "lonesock" Dummer and available from: http://obex.parallax.com.

FloatString_Lite.spin and **FloatMath_Lite.spin** are used to provide IEEE 754 compliant 32-bit floating point-to-ASCII string conversion routines. These objects are included with the Parallax Propeller Tool, but have been modified by Joe Grand to remove code not used by the LRF Module.

Basic_I2C_Driver.spin provides the I2C protocol interface for boot EEPROM communication. The LRF Module has a 64KB boot EEPROM. Only 32KB is required for program storage, so the remaining 32KB is available for data storage and used to store the calibration values specific to each LRF Module. Written by Michael Green and available from: http://obex.parallax.com.

Optical Triangulation

The LRF Module uses optical triangulation for range finding, where the distance to the target object is calculated with simple trigonometry between the center points of laser light, camera, and object. The design of the LRF Module is based, in theory, on the implementation of Todd Danko's Webcam Based DIY Laser Rangefinder (https://sites.google.com/site/todddanko/home/webcam_laser_ranger).



Referring to the figure, a laser diode module shines a laser spot onto the target object. The value h is a fixed, known distance between the center points of the laser diode and the camera (78 mm for the LRF Module). When the distance to the target object D changes, so do both the angle θ and the value pfc (pixels from center), which is the number of pixels the centroid of the primary blob (laser spot) is away from the camera's center point.

As the object gets closer, the value of pfc (and angle θ) increases. As the object gets farther away, pfc (and angle θ) approaches zero. A short video made by Parallax's Beau Schwabe demonstrates this phenomenon; see the 28044 product page's Downloads and Resources section for a link.

If the angle θ is known, then basic trigonometry can be used to calculate the distance value D:

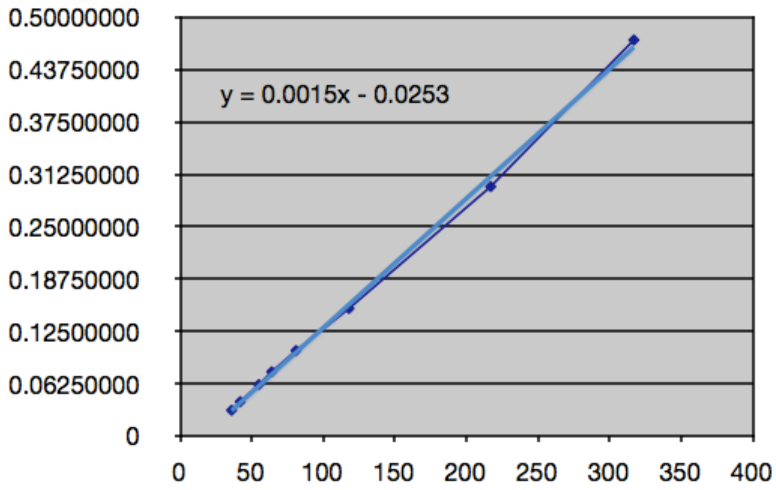
$$\tan \theta = h / D$$

Solving for D:

$$D = h / \tan \theta$$

Since the LRF Module's blob detection routine returns a pfc value, an intermediate step is required to correlate that value with an actual angle θ. The relationship between pfc and angle can be described with a slope-intercept linear equation (www.math.com/school/subject2/lessons/S2U4L2GL.html).

The following chart shows measurements taken with a LRF Module. pfc is on the X-axis and angle is on the Y-axis. The dark blue diamonds show the actual measurements and the resulting best-fit linear equation is printed on the chart and denoted with a light blue line:



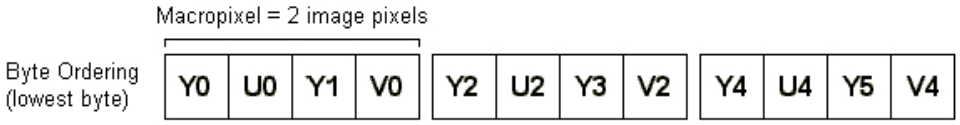
So, once the laser is shined onto the target object and the pfc value of the laser spot is received, an angle θ can be calculated using the slope-intercept equation and passed to the trigonometric function to determine the actual distance the range finder is from the target object.

Camera Interface

The OmniVision OVM7690 640x480 CMOS CameraCube provides a digital interface:

- **DVP[7:0] (Digital video port)**: 8-bit wide output bus corresponding to pixel information sent in the selected output format from the OVM7690 (RAW RGB, RGB565, CCIR656, or YUV422/YCbCr422).

  The LRF Module is configured for YUV422 output (http://en.wikipedia.org/wiki/YUV and http://en.wikipedia.org/wiki/YCbCr). Y is the luma component - brightness in grayscale - and U and V are chroma components - color differences of blue and red, respectively. The particular format of YUV422 used by the OVM7690 is known as YUY2 (www.fourcc.org/yuv.php), in which each 16-bit pixel is given an 8-bit Y component and alternating 8-bit U or 8-bit V component:



  Y0U0 corresponds to a single pixel starting from the left, Y1V0 is the 2nd pixel, etc. Every pixel has Y data, and U and V are every other pixel.

- **VSYNC (Vertical sync)**: Indicates the beginning of a new frame by pulsing high.
- **HREF (Horizontal reference)**: Indicates the start of the next row of pixels by pulsing high. By keeping count of the number of HREF pulses received since the last VSYNC, we can determine which horizontal line of the video frame we are currently on.
- **PCLK (Pixel clock)**: Asserted when valid pixel data is available on the DVP bus. For a 640 pixel line in YUV422 format (16 bits/pixel), 10,240 pixel clock cycles will occur after each HREF pulse.

There are nearly one hundred 8-bit registers within the OVM7690 device that require configuration, including, but not limited to, general settings, output format selection, resolution, frames per second, automatic white balance, and gain control. Due to confidentiality concerns, OmniVision does not allow explicit references or detailed explanations of camera configuration registers. Explanations of group settings are allowed and provided within the LRF Module source code to give the user an overview of camera operation. To obtain full OVM7690 product specifications, a non-disclosure agreement (NDA) must be executed with OmniVision Technologies, Inc. (www.ovt.com).

Frame Grabber

The frame grabber cog (OVM7690_fg.spin) only runs when started by a calling object. It grabs an image frame from the camera and stores it in the frame buffer. 5,280 longs (~20.6KB) are allocated in hub RAM for the frame buffer, leaving just under 4KB remaining for stack and variables.

Grabbing a frame consists of waiting for VSYNC to go high, which signals the start of a new frame, and then waiting for HREF to go high, which signals the start of a new line. Then, pixel data (alternating Y and U/V bytes) is captured from DVP[7:0] every time PCLK goes high and is stored in the frame buffer. After the complete frame is stored in the buffer, the cog sets a flag in hub RAM to a non-zero state so the calling object knows that the frame grab is done. The cog then stops itself.

The frame grabber supports three modes:

1. **Single Frame, Greyscale**: 8 bits/pixel greyscale image (just the Y component of the YUV422 data is captured) at 160x128 resolution.

2. **Single Frame, Color**: 16 bits/pixel YUV422 image at 640x16 resolution. The laser diode is enabled during the frame grab for testing and alignment purposes.

3. **Processed Frame, Color**: 16 bits/pixel YUV422 image at 640x16 resolution. This mode is specific for range finding functionality and consists of a "background subtracted" image where one frame is taken with the laser diode off, one frame taken with laser diode on, and the data subtracted to help isolate the laser spot from the rest of the image frame. See Image Processing and Blob Detection, below.

Image Processing and Blob Detection

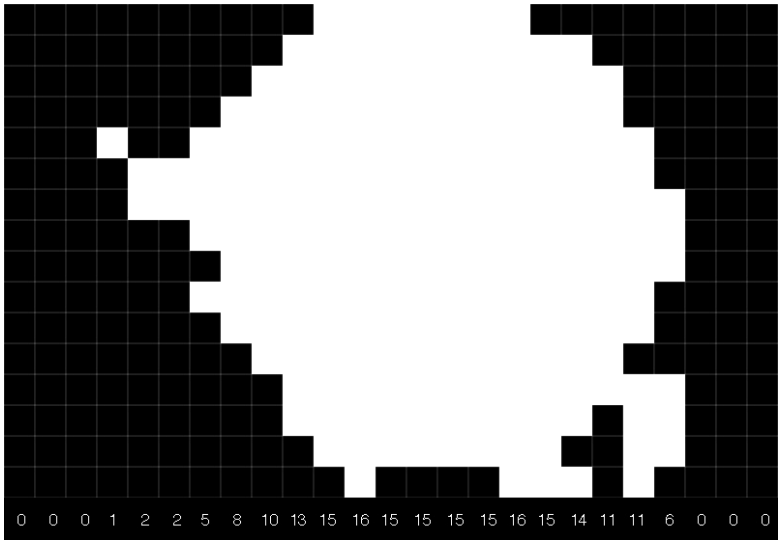
The primary function of the LRF Module is to capture an image with the camera and determine the location of the laser spot (blob) within the frame. Once the blob is detected, its centroid (center of mass) and resulting pfc value (pixels from center) are determined and used in the function that calculates the distance from the LRF to the target object.

The image processing and blob detection routines function as follows:

1. **Background Subtraction.** Grab two consecutive frames—one with the laser diode off and one with the laser diode on. Each pixel's Y/luma component from the first frame is subtracted from the same pixel's Y/luma component from the second frame (and absolute valued), leaving only the pixels that have changed in brightness between the two frames. All other background details (anything that has stayed the same between the two frames) disappear. The U/V color components are grabbed only on the first of the two frames and not modified. Details of pixel/background subtraction can be found at:
   http://homepages.inf.ed.ac.uk/rbf/HIPR2/pixsub.htm

2. **Thresholding.** Look at each pixel within the frame and determine if it is above the defined brightness threshold (currently, any pixel with a Y value greater than 0.3). If so, the pixel is set to '1' (white). If not, the pixel is set to '0' (black):

3. **Column Sum**. Count the number of '1' pixels within each vertical column. This results in a one-dimensional array containing the number of "valid" pixels per column. Summing the valid pixels makes it easier to quickly search the frame to locate any blobs. The following image shows the zoomed-in blob with the column's sum printed at the bottom of each column:



4. **Blob Detection.** Traverse the one-dimensional array of column sums looking for any sums above a defined threshold (currently, a column sum needs to be greater than 2 in order to be considered part of the blob). For example, in the image from Step 3, the blob would start at column 7 (which has a sum of 5) and end at column 22 (which has a sum of 6). This is repeated across the entire frame from left to right until all blobs have been detected.

5. **Mass/Centroid Calculation.** Calculate the total mass and centroid for the detected blob(s) in the frame. The *mass* is simply the number of valid '1' pixels within the total blob. The *centroid* of a blob is its center of mass and is calculated by weighting every valid pixel with where it is in the blob and averaging by the total mass:

   For column 1..n of the blob

   sum = 1 * s1 + (2 * s2) + ... + (n * sn)

   Where sn = column sum for column n

   Then, centroid = sum / mass

   Performing the weighted average (as opposed to simply setting the centroid location as the center point of the blob) gives a more accurate center of mass result regardless of blob shape.

   Here's an example of determining the centroid using the blob from Step 3:

   sum = (1 * 5) + (2 * 8) + (3 * 10) + ... + (15 * 11) + (16 * 6) = 1737

   centroid = sum / mass = 1737 / 200 = ~8.7

   The blob with the largest mass is then chosen as the primary blob (which is assumed to be the actual laser spot) and will be used for the subsequent range finding calculation. If there are multiple blobs with the same mass, the first occurrence remains the primary.

Care and Handling

Alignment of the laser diode and camera are critical to the proper operation of the LRF Module. Both components are aligned during production and affixed to the printed circuit board with cyanoacrylate adhesive. Care should be taken to protect the LRF Module from sudden shock, excessive vibration, or blunt force.

To verify proper alignment of the LRF Module, use the LRF Image Viewer tool. Set the LRF Module 10-30 inches away from a target object (preferably a grey or other non-white surface). Ensure the Color radio button is selected, uncheck Blob Detection, and press the Grab Image button. A color image with the red laser spot should be displayed in the image box. The laser spot should be centered vertically within the frame:

If the laser spot is out of position, the laser diode will need to be manually re-adjusted and the LRF Module will need to be re-calibrated. See the "X" command in the Command Details section, page 4, for calibration instructions.

Safety

The LRF Module uses an Arima APCD-635-02-C3-A Laser Diode that contains integrated automatic power control (APC) circuitry and glass collimating lens. The laser diode is a Class IIIa laser device with a maximum power output of <= (less than or equal to) 3 mW @ 635 nm. The laser diode is only enabled during a single frame capture, which takes ~400 ms.

To prevent eye damage, do not stare into the laser diode output on the front of the LRF Module. Many documented cases of eye damage with Class IIIa devices (which include, for example, most run-of-the-mill red laser pointers, laser levels, and laser-based thermometers), were caused by prolonged exposure of the direct laser output: http://en.wikipedia.org/wiki/Laser_safety#Laser_pointers

Example Code

BASIC Stamp® 2 Microcontroller

The following code example demonstrates features of the Laser Range Finder. This code is available for download from the Parallax Laser Range Finder Module product page; search "28044" at www.parallax.com.

This program uses the Debug Terminal, which is built into the BASIC Stamp Editor software. The software is a free download from www.parallax.com/basicstampsoftware.

```
' =================================================================================
'
'   File....... LRF Basic.bs2
'   Purpose.... Demonstrates features of the Parallax Laser Range Finder Module
'   Author..... Joe Grand, Grand Idea Studio, Inc. [www.grandideastudio.com]
'   E-mail..... support@parallax.com
'   Updated.... 30 JUN 2011 (v1.1)
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =================================================================================

' -----[ Program Description ]---------------------------------------------
'
' This program demonstrates the Parallax Laser Range Finder (LRF) Module.
'
' When the pushbutton switch is pressed, the Laser Range Finder calculates
' the distance between itself and the target object. The result is
' displayed on a Parallax Serial LCD Module (#27977 backlight or #27976
' non-backlight).
'
' When the measurement is out-of-range (outside of the defined minimum or
' maximum distance bounds of the Laser Range Finder), the piezo buzzer
' will sound a warning tone.
'

' -----[ Revision History ]------------------------------------------------
'
' 1.0: Initial release
'

' -----[ I/O Definitions ]-------------------------------------------------

Enable  PIN           0             ' Pushbutton switch (active LOW)

Piezo   PIN           12            ' Piezo element (for alarm sound)

LCD TX  PIN           13            ' Serial output to LCD

LRF TX  PIN           8             ' Serial output to LRF (connects to SIN)
LRF_RX  PIN           9             ' Serial input from LRF (connects to SOUT)


' -----[ Constants ]-------------------------------------------------------

#SELECT $STAMP
  #CASE BS2, BS2E, BS2PE
    T1200       CON     813
    T2400       CON     396
    T4800       CON     188
    T9600       CON     84
    T19K2       CON     32
    T38K4       CON     6
  #CASE BS2SX, BS2P
    T1200       CON     2063
    T2400       CON     1021
```

```
    T4800       CON     500
    T9600       CON     240
    T19K2       CON     110
    T38K4       CON     45
#ENDSELECT

'---- [LCD Constants] ----

LcdBaud         CON     T19K2

LcdBkSpc        CON     $08             ' move cursor left
LcdRt           CON     $09             ' move cursor right
LcdLF           CON     $0A             ' move cursor down 1 line
LcdCls          CON     $0C             ' clear LCD (use PAUSE 5 after)
LcdCR           CON     $0D             ' move pos 0 of next line
LcdBLon         CON     $11             ' backlight on
LcdBLoff        CON     $12             ' backlight off
LcdOff          CON     $15             ' LCD off
LcdOn1          CON     $16             ' LCD on; cursor off, blink off
LcdOn2          CON     $17             ' LCD on; cursor off, blink on
LcdOn3          CON     $18             ' LCD on; cursor on, blink off
LcdOn4          CON     $19             ' LCD on; cursor on, blink on
LcdLine1        CON     $80             ' move to line 1, column 0
LcdLine2        CON     $94             ' move to line 2, column 0

'---- [LRF Constants] ----

LrfBaud         CON     T2400

' -----[ Variables ]-------------------------------------------------------

range           VAR     WORD            ' Range (in mm) returned from the LRF

' -----[ Initialization ]--------------------------------------------------

Init:
  DEBUG CLS, "Parallax Laser Range Finder", CR, CR

  INPUT Enable                                  ' configure the Enable pin as an input

  HIGH LCD_TX                                   ' setup serial output pin
  PAUSE 100                                     ' allow LCD to initialize
  SEROUT LCD_TX, LcdBaud, [LcdOn1, LcdCls]
  SEROUT LCD_TX, LcdBaud, [LcdBLon]             ' turn backlight on (for backlit serial LCD module)
  PAUSE 250
  SEROUT LCD_TX, LcdBaud, [LcdLine1, "LRF Module Demo"]

  ' when the LRF powers on, it launches an auto-baud routine to determine the
  ' host's baud rate. it will stay in this state until a "U" ($55) character
  ' is sent by the host.
  DEBUG "Waiting for LRF module..."
  PAUSE 2000                                    ' delay to let LRF module start up
  SEROUT LRF_TX, LrfBaud, ["U"]                 ' send character
  ' when the LRF has initialized and is ready to go, it will send a single ":" character
  SERIN LRF_RX, LrfBaud, [WAIT(":")]            ' wait until the module is ready (at baud rates
                                                ' higher than 2400, the BS2 misses LRF's response)
  DEBUG "Ready!", CR, CR
  SEROUT LCD_TX, LcdBaud, [LcdLine2, "READY!"]
  PAUSE 250

' -----[ Program Code ]----------------------------------------------------

Main:
  IF Enable THEN Main                           ' wait for a button press

  ' When a single range (R) command is sent, the LRF returns the distance to the target
  ' object in ASCII in millimeters. For example:
  '
  ' D = 0123
  '
  SEROUT LRF_TX, LrfBaud, ["R"]                                         ' send command
```

```
  SERIN  LRF_RX, LrfBaud, 3000, No_Response, [WAIT("D = "), DEC4 range]    ' wait for data to be
                                                                          ' received and store it
  ' With the data in hand, let's display it...
  SEROUT LCD_TX, LcdBaud, [LcdCls]                                        ' erase contents of LCD
  PAUSE 5

  DEBUG "D = ", DEC (range / 10), ".", DEC1 (range // 10), " cm, "  '...in centimeters
  SEROUT LCD_TX, LcdBaud, [LcdLine1, DEC (range / 10), ".", DEC1 (range // 10), " cm"]

  range = (range * 10) ** 2580              ' convert mm to 100ths of an in (2580=1/25.4*$FFFF)
  DEBUG DEC (range / 10), ".", DEC1 (range // 10), " in", CR  ' ..and in inches
  SEROUT LCD_TX, LcdBaud, [LcdLine2, DEC (range / 10), ".", DEC1 (range // 10), " inches"]

  IF (range = 0) THEN FREQOUT Piezo, 250, 4000 ' if no distance/out-of-range, sound alarm (4kHz)
                                               ' for 1 second (timing for BS2)

  GOTO Main   ' Do it all over again!

  END

' -----[ Subroutines ]----------------------------------------------------

No_Response:
  DEBUG CR, LF, "Error: No response from LRF Module"
  SEROUT LCD_TX, LcdBaud, [LcdLine2, "ERROR!"]
  PAUSE 5000
  GOTO Init

' -----[ End of File ]----------------------------------------------------
```

Propeller™ P8X32A Example Code

The following code example demonstrates features of the Laser Range Finder. This code is available for download from the Parallax Laser Range Finder Module product page; search "28044" at www.parallax.com.

Note: This application uses additional objects, which are included in the example code .ZIP file on the product page. It also uses the Parallax Serial Terminal to display the device output. The Parallax Serial Terminal is included with the Propeller Tool v1.2.7 or higher, which is available from the Downloads link at www.parallax.com/Propeller.

```
{{
┌──────────────────────────────────────────┐
│ Laser Range Finder: Basic Demonstration  │
│                                          │
│ Author: Joe Grand [www.grandideastudio.com]  │
│ Contact: support@parallax.com            │
│                                          │
│ See end of file for terms of use.        │
└──────────────────────────────────────────┘

Program Description:

This program demonstrates the Parallax Laser Range Finder (LRF) module.
The distance to the target object is displayed in the Parallax Serial Terminal.

Revisions:

1.1 (July 2011): Initial release

}}


CON
  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

  LRF_TX        = 6                         ' Serial output to LRF (connects to SIN)
  LRF_RX        = 7                         ' Serial input from LRF (connects to SOUT)
```

```
VAR
  long range                                    ' Distance from LRF module to target object (mm)

OBJ
  pst       : "Parallax Serial Terminal"        ' Debug Terminal
  serial    : "Extended_FDSerial"               ' Extended Full Duplex Serial by Martin Hebel

PUB main
  pst.Start(115_200)                            ' Set Parallax Serial Terminal to 115.2kbps
  pst.Str(@InitHeader)                          ' Print header; uses string in DAT section.

  ' Set-up serial port for communication with the LRF module
  serial.Start(LRF_RX, LRF_TX, %0000, 115_200)      ' Start serial port, normal mode, 115.2kbps

 {{
   When the LRF powers on, it launches an auto-baud routine to determine the
   host's baud rate. It will stay in this state until a "U" ($55) character
   is sent by the host.
 }}
  pst.Str(String("Waiting for LRF module..."))
  waitcnt(clkfreq << 1 + cnt)                   ' Delay for 2 seconds to let the LRF module start-up
  serial.Tx("U")                                ' Send character
  repeat until serial.RxCheck == ":"            ' When the LRF has initialized and is ready to go, it will
                                                ' send a single ':' character, so wait here until we receive it
  pst.Str(String("Ready!", pst#NL, pst#NL))     ' Ready to go!

  repeat
    {{
      When a single range (R) command is sent, the LRF returns the distance to the target
      object in ASCII in millimeters. For example:

      D = 0123
    }}

    serial.Tx("R")                              ' Send command
    repeat until serial.RxCheck == "D"          ' Wait for the header to be sent...
    repeat until serial.RxCheck == "="          '
    repeat until serial.RxCheck == " "          '
    range := serial.RxDec                       ' ...then grab the value

    ' With the data in hand, let's display it...
    pst.Str(String("Distance = "))
    pst.Dec(range / 10)
    pst.Char(".")
    pst.Dec(range // 10)
    pst.Str(String(" cm, "))                    ' ... in centimeters

    range := (range * 10) ** 169_093_200        ' convert mm to hundredths of an inch
                                                '(169_093_200 = 1 / 25.4 * $FFFFFFFF)
    pst.Dec(range / 10)
    pst.Char(".")
    pst.Dec(range // 10)
    pst.Str(String(34, pst#CE, pst#NL, pst#MU))     ' ...and in inches

DAT
InitHeader    byte "Parallax Laser Range Finder", pst#NL
              byte "Basic Range Demonstration", pst#NL, pst#NL, 0

{{
```



```
}}
```