Parakeet ProE Ethernet API

Version 1.1

Version History:

Date	Version	Comments
2021/12/20	1.1	Added details about header and alarm packets.
2021/11/22	1.0	Initial Release.

Contents

Introduction

All communication to a Parakeet ProE is done via UDP. The IP Address, port, subnet mask, and gateway the device lives on is configurable. The IP Address and port which the device sends information to is also configurable. All binary data is sent in little-endian order. The sensor sends out a heartbeat message every second to inform the user about the device's status. Transmission to the device is done via a UDP Command Packet, and the device responds with a UDP Response Packet.

Network Defaults

Source IP	192.168.158.98
Source Port:	6543
Source subnet mask:	255.255.255.0
Source gateway:	192.168.158.1
Destination IP:	192.168.158.15
Destination Port:	6668

Command Set

Control Words

A set of strings which can be sent to the device to control it, a response is returned if the device successfully received the message. None represents no response from the sensor.

Start Sensor Spinning			
Tell the sensor to start spinning. This starts the transmission of lidar data packets.			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSTARH”
Device response value:	“LiDAR START”		

Stop Sensor Spinning			
Tell the sensor to stop spinning. This stops the transmission of lidar data packets but does not stop heartbeat packet transmission.			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSTOPH”
Device response value:	“LiDAR STOP”		

Reset and Reboot Sensor			
Tell the sensor to reboot and reset any non-persistent setting changes			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LRESTH”

Device response value:	None, as sensor instantly reboots upon the message received

Get Sensor Version			
Returns the version number of the sensor			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LVERSH”
Device response value:	“MCUVERSION”		

Reset / Restart Sensor			
Tell the sensor to turn off and come back on afterwards.			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LRESTH”
Device response value:	None		

Setting Words

A set of strings which can be sent to the device to customize the output, a response is returned if the device successfully received the message. These commands save their effect through power loss.

Modify Sensor Source IPv4 Settings			
Sets the IP Address, subnet mask, gateway, and service port number the device will be listed as on the network			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSUDP:”
2	IP Address	15	IP Address the sensor should be listed under on the network By default: 192.168.158.98
3	Subnet Mask	15	Subnet Mask the sensor should be listed under on the network By default: 255.255.255.0
4	Gateway	15	Gateway the sensor should be listed under on the network By default: 192.168.158.1
5	Service Port	5	Port the sensor will be listening on By default: 6543
6	End Identifier	1	Fixed as “H”
Example Usage:			“LSUDP:192.168.158.091 255.255.255.000 192.168.158.001 05000H”
Device response value:			NA
Extra:			Spaces are used to deliminate between fields 2 and 5

Modify Sensor Destination IPv4 Settings			
Sets the IP Address, port number the sensor will send packets to.			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSDST:”
2	Destination IP Address	15	IP Address the destination device is listed at on the network By default: 192.168.158.15

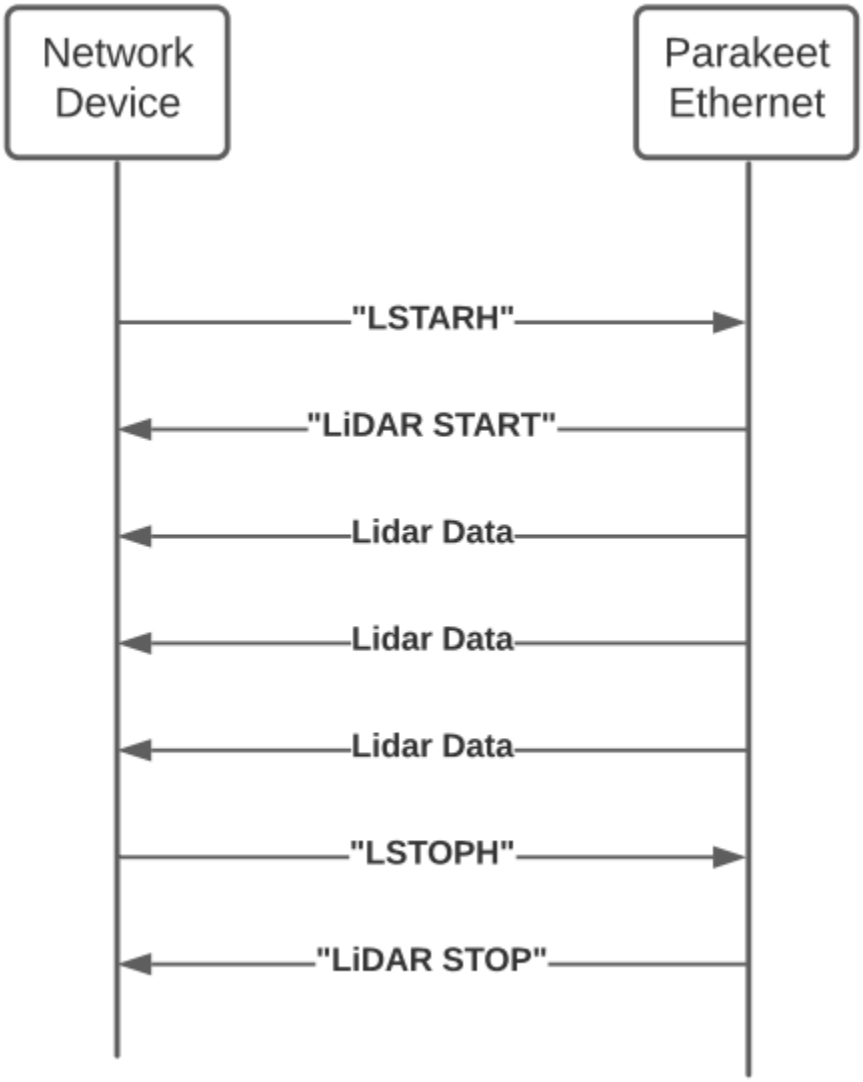
3	Destination Port	5	Port the destination device will be listening on By default: 6668
4	End Identifier	1	Fixed as “H”
Example Usage:			“LSDST:192.168.158.015 06668H”
Device response value:			NA
Extra:			A space is used to deliminate between fields 2 and 3

Set RPM			
Sets the RPM the device should spin at.			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSRPM:”
2	RPM	3	The RPM the device should spin at, bounded between 0 and 999 Supported values: 600 (10 Hz), 900 (15 Hz).
3	End Identifier	1	Fixed as “H”
Example Usage:			“LSRPM:600H”
Device response value:			"Set RPM: OK"

Set Data Smoothing			
Sets the state of the data smoothing filter			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSSMT:”
2	Enable	1	Boolean determining if data smoothing should be enabled
3	End Identifier	1	Fixed as “H”
Example Usage:		ON	“LSSMT:1H”
		OFF	“LSSMT:0H”
Device response value:		"LiDAR set smooth ok"	

Set Drag Point Removal			
Sets the state of the drag point removal filter			
Field #	**Name**	**Length**	**Interpretation**
1	Start Identifier	6	Fixed as “LSFLT:”
2	Enable	1	Boolean determining if drag point removal should be enabled
3	End Identifier	1	Fixed as “H”
Example Usage:		ON	“LSFLT:1H”
		OFF	“LSFLT:0H”
Device response value:		"LiDAR set filter ok"	

Nominal Message Sequence Diagram



Data Packets

All Parakeet network transmissions are done via UDP and are by default shipped to port 6668.

Command Packet			
A packet which is sent to the sensor in order to configure the device. The sensor responds with a response packet.			
Field #	**Name**	**Length**	**Interpretation**
1	signature	2	Fixed as “0x484C”
2	identifier	2	Tells the sensor which category of command this packet describes. Network commands use: 0x0053 The rest use: 0x0043
3	sequenceNumber	2	A unique identifier for this packet. The response packet which is returned from this packet, will have the same sequenceNumber
4	length	2	The number of characters in field 5
5	message	varies	A Control Word message or a Setting Word message. The Command Set section has information describing this
6	crc32	4	A CRC32 checksum of fields 1-5. Initial value: 0xFFFFFFFF Polynomial value: 0x04C11DB7 Code example maybe found here

Response Packet			
A packet which is sent to the host in response to the sensor receiving a command packet.			
Field #	**Name**	**Length**	**Interpretation**
1	signature	2	Fixed as “0x484C”
2	identifier	2	Fixed as “0xFFBC”
3	sequenceNumber	2	Contains the sequenceNumber matching the control packet the device is responding to
4	length	2	The number of characters in field 5
5	message	varies	The message response from the sensor. More information on this can be found in the Command Set section.

6	crc32	4	A CRC32 checksum of fields 1-5. Initial value: 0xFFFFFFFF Polynomial value: 0x04C11DB7 Code example maybe found here

Lidar Data packet			
Each scan revolution is made up of 10 sectors, and each sector is made up of subsectors. The number of subsectors varies depending on the number of points within the sector. A Lidar data packet maps to one subsector. Transmission of these packets starts when the sensor starts spinning. All subsector packets of the same sector are transmitted at the same time.			
Field #	**Name**	**Type**	**Interpretation**
1	lidarDataHeader	uint16	Fixed as 0xFAC7
2	numPoints	uint16	Number of points within this packet
3	totalPointsInSector	uint16	Total number of points within this sector of the lidar scan. E.g: a scan of 100 points may be broken up into two packets, where numPoints is 90 for one, and 10 for another. In both packets, this value will be set as 100
4	offset	uint16	Index of the first point of this packet within the sector.
5	startAngle	uint32	Starting angle of this sector, with 0.001° as the resolution. E.g: 180° will be stored as 180000
6	endAngle	uint32	Ending angle of this sector, with 0.001° as the resolution. E.g: 180° will be stored as 180000
7	flags	uint32	Bit 1 = Distance scale, 0 = mm, 1 = cm Bit 2 = Intensity data included, 0 = not included, 1 = included Bit 3 = Drag point removal filter, 0 = Off, 1 = On Bit 4 = Data smoothing filter, 0 = Off, 1 = On Bit 8 = Resample filter, 0 = Off, 1 = On
8	timestamp	uint32	0 - 25565, in milliseconds. Indicates the laser emission time of the first point in the sector, relative to when the sensor is turned on. The sensor timestamp defaults to 0 on startup
9	deviceNumber	uint32	Equipment number assigned to this device
10	distanceArray	uint16[numPoints]	Distance between the lidar device and where the lidar laser beam had come into contact. Stored as millimeters or centimeters, dependent on the value in flags
11	relativeAngleArray	uint16[numPoints]	Angle relative to startAngle, with 0.001° as the unit. E.g: 180° will be stored as 180000
12	intensityArray	uint8[numPoints] OR uint8[0]	Return strength of lidar laser beam. If the intensity is enabled in flags, intensityArray has size of numPoints, otherwise it has size of 0
13	checksum	uint16	Sum of fields 2 -> 12. More information in the Lidar Data Packet Checksum Calculation section

Byte	0	1	2	3	4	5	6	7
	headerCode		numPoints		totalPointsInSector		offset	
	startAngle				endAngle			
	flags				timestamp			
	deviceNumber				distanceArray			
	relativeAngleArray				intensityArray			
	checksum							

Example Lidar data packet (big-endian form)				
headerCode	numPoints	totalPointsInSector	offset	startAngle
C7FA	0004	00BC	00B8	0003D860
endAngle		flags		timestamp
00046500		0000000E		DDE60861
deviceNumber		distanceArray		
00000001		0000 1708 0000 1704		
relativeAngleArray				intensityArray
C689 848A 428B 058C				00 11 00 0F
checksum				
F27E				

Network Heartbeat Protocol Packet			
Packet containing the current state of the lidar device. Transmissions of these packets begins on sensor startup. This message is always broadcast to 255.255.255.255. A heartbeat packet is transmitted from the device every second.			
Field #	**Name**	**Type**	**Interpretation**
1	heartbeatHeader	uint8[4]	Fixed as “LiDA”
2	protocolVersion	uint32	Protocol version
3	timestamp	uint64	Timestamp
4	deviceSerialNumber	uint8[20]	Serial number of the device
5	deviceType	uint8[16]	Type of device
6	deviceVersion	uint32	Bottom control panel version
7	deviceID	uint32	Device number
8	sourceIPAddress	uint8[4]	Device address on the network
9	sourceSubnetMask	uint8[4]	Subnet mask of the device is listed on
10	sourceGateway	uint8[4]	Gateway the device is listed on
11	destinationIPAddress	uint8[4]	The IP Address of the destination device the sensor is currently communicating to
12	destinationPort	uint16	The port which the sensor is sending data to
13	sourcePort	uint16	The port which the sensor is listening on
14	reserved	uint16	NA
15	rpm	uint16	# of revolutions per minute. Multiplied by 10
16	frequency_Hz	uint16	# of revolutions per second. Multiplied by 100
17	rangeVersion	uint8[2]	Top ranging head program version
18	cpuTemp	uint16	Temperature of device CPU. measured in celsius. Multiplied by 10.
19	inputVoltage	uint16	Voltage through the device. Multiplied by 1000
20	deviceState	uint8[16]	Device state information, each byte is a boolean value matching the Device State table below.
21	crc32	uint32	A CRC32 checksum of fields 1-20. Code example maybe found here

Device State			
Index	**Name**	**Interpretation**	**Available for ProE**
0	LV_1	Viewing area	✗
1	LV_2	Warning area	✗
2	LV_3	Alarm area	✗
3	COVER	Shelter	✗
4	NO_DATA	No data from ranging head	✓
5	ZONE_ACTIVE	No defense zone setting	✗
6	SYS_ERR	System internal error	✓
7	RUN	Abnormal system operation	✓
8	NET_LINK	Network connection error	✓
9	UPDATING	Device update in progress	✓
10	ZERO_POS	0 output	✗
11-13	Reserved	reserved	✗
14	USB_LINK	USB Connect	✗
15	ZERO_DEFINED	Zero position could not be detected	✓
16-32	Reserved	reserved	✗

Alarm Packet			
Packet which describes an ongoing alarm			
Field #	**Name**	**Type**	**Interpretation**
1	alarmHeader	uint8[4]	Fixed as “LMSG”
2	protocolVersion	uint32	Protocol version
3	deviceSerialNumber	uint8[20]	Serial number of the device
4	deviceID	uint32	Device number
5	timestamp	uint32	Timestamp of when this message was captured
6	alarmType	uint32	0 = No Error 1 = Error occurred Other values = Reserved
7	alarmMessage	uint32	Bit 1 = Low Power Bit 2 = Motor Locked Rotation Bit 3 = Overheated Bit 4 = Network Error Bit 5 = No output from ranging module Bit 6-32 = Reserved
8	extra	uint16	Reserved

Checksum Calculation

Each packet the sensor sends out begins with a header and ends with a checksum. The value of this checksum can be calculated from summing all values in the packet excluding the header and checksum. Data types smaller than uint16 get expanded to uint16, ie: 0xF -> 0x0F. Data types larger than uint16 get separated into uint16, ie: 0xDDFF -> 0xDD 0xFF. Bytes will also need to be converted from little-endian to big-endian.

Checksum Example

This is an example of how to calculate a checksum for a Lidar data packet. Note that field 1 of a Lidar data packet is the header, and headers are not included in the checksum calculation. Also note that since field 13 is the checksum, it will also not be included in the calculation. For simplicity, mod 0x10000 will be done as the final step in this calculation to get it to fit into 16 bits. A code example may be located here.

Field #	Little Endian	Big Endian	Sum of components	Cumulative sum
2	0400	0004	0004	0004
3	BC00	00BC	00BC	00000C0
4	B800	00B8	00B8	0178
5	60D80300	0003D860	0003 + D860 = D863	D9DB
6	00650400	00046500	0004 + 6500 = 6504	13EDF
7	0E000000	0000000E	0000 + 000E = 000E	13EED
8	6108E6DD	DDE60861	DDE6 + 0861 = E647	22534
9	01000000	00000001	0000 + 0001 = 0001	22535
10	0000 0817 0000 0417	0000 1708 0000 1704	0000 + 1708 + 0000 + 1704 = 2E0C	25341
11	C689 848A 428B 058C	89C6 8A84 8B42 8C05	89C6 + 8A84 + 8B42 + 8C0 = 22B91	47ED2
12	00 11 00 0F	00 11 00 0F	0000 + 0011 + 0000 + 000F = 0020	47EF2 % 0x10000 = 7EF2