Interface Description Document
for the
MultiSense-SL

Preliminary Communications Protocol Description

Contact:

David LaRose
Director of Software Engineering
412-327-3970
dlr@carnegierobotics.com
Carnegie Robotics LLC
Ten 40th Street
Pittsburgh, PA 15201

Revision History

Rev.	Page Nos.	Description	Date	Name
1.0	Entire	Initial Pre-Release	3 Oct, 2012	S. Turner
2.0	Entire	Update to ToC, Reworked Fig 2 and revamped detailed message section	4 Oct, 2010	S. Turner
3.0	Entire	Created Table of Figures and reformatted	4 Oct, 2012	S. Turner
4.0	Entire	Updates to all messages and added messages from source code.	9 Oct, 2012	S. Turner
5.0	Entire	Reformatted document	18 Oct, 2012	S. Turner
6.0	Entire	Reworked Messaging sections to provide more data	18 Oct, 2012	S. Turner
7.0	1	Updated contact information	20 Oct, 2012	D. LaRose

Table of Contents

1.OVERVIEW **6**
2. MESSAGING **7**
2.1. Message Basics 7
2.2. Message Types 7
2.3. General Messages 7
2.4.Stereo Camera Messages 8
2.5. LIDAR Messages 9
2.6. Lighting Messages 9
2.7.Message Structure 10
2.8.Additional Datatypes 11
3.MESSAGE DETAILS **12**
3.1. Overview 12
3.2. Detailed General Messages 12
3.3.Detailed Stereo Camera Messages 12
3.4.Detailed LIDAR Messages 13
3.5.Detailed Lighting Messages 14

Table of Figures

1. OVERVIEW

Data transfer between the MultiSense-SL and an external user CPU is accomplished via a gigabit Ethernet connection. This connection transfers information bi-directionally between the devices, allowing the user CPU to send queries and commands, while the MultiSense-SL sends configuration, version, and gathered data back to the user CPU.

Figure : Communication Flow

The MultiSense-SL data stream includes information from the stereo camera pair, such as rectified calibration information, images, and disparities. The stream also includes information from the rotating LIDAR, including the LIDAR configuration information, scan range data, motor speed, and rotation angle of the LIDAR assembly.

2. MESSAGING

2.1. MESSAGE BASICS

Two types of messaging sequences are used to facilitate communication between the devices. The first is a query/response message, used in retrieving calibration information, system settings, and status. In this method the user CPU initiates a message, and the MultiSense-SL responds with an acknowledgement message.

The second type is a query/stream message, used in retrieving data gathered by the MultiSense-SL. In this method the user CPU initiates the request with a single message. The MultiSense-SL responds by sending an acknowledgement message, and then begins sending a continuous data stream. The user CPU can then send additional commands to modify the data stream without having to interrupt the stream, e.g. changes in exposure and gain. The data stream will continue until the Multisense-SL receives a single message instructing it to stop streaming, which elicits another acknowledgement response.

In both messaging formats the same basic header and data structures are used. All multi-byte numeric data types are little-endian.

2.2. MESSAGE TYPES

Each type of message has a corresponding 8-bit identifier, represented as a uint8_t (unsigned integer that is eight bits in length). The fourth through sixth bits indicate the specific hardware component -- broken down into: general, stereo cameras, LIDAR sensor and motor, lighting, and the -inertial measuring unit. The seventh bit indicates the source of the message, whether the user CPU or MultiSense. Symbolic names for these 8-bit identifiers are prefixed with two abbreviations, one to designate the source of the message, and a second to indicate the hardware component when applicable.

Device	Prefix
User CPU	CPU
MultiSense	MS

Figure : Pods and their prefixes used in message type formatting.

Component	Prefix
Stereo Camera	CAM
Lidar	LIDAR
Inertial Measuring Unit	IMU
Lighting	LED

Figure : Hardware components and their prefixes used in message type formatting.

2.3. GENERAL MESSAGES

The general messages provide information regarding the version and statuses of the MultiSense-SL. Detailed information regarding the general messages can be found in section 3.2.

Type	Description	ID
CPU_VERSION_REQUEST	Retrieve the version information	0x00
CPU_STATUS_REQUEST	Retrieve status information	0x01
MS_VERSION_RESPONSE	Version string information	0x80
MS_STATUS_RESPONSE	Current status	0x81

Figure : General Messages

2.4. STEREO CAMERA MESSAGES

Stereo camera messages provide configuration and calibration information, as well as the commands to start and stop data streams, and send various acknowledgements. Detailed information regarding the stereo camera messages can be found in section 3.3.

Type	Description	ID
CPU_CAM_GET_CONFIG	Get configuration/calibration information	0x10
CPU_CAM_START_STREAM	Start streaming data	0x11
CPU_CAM_STOP_STREAM	Stop streaming data	0x12
CPU_CAM_CONTROL	Dynamically set exposure etc. without interrupting stream	0x13
CPU_CAM_GET_HISTORY	Request a history of recently transmitted frame number from the MultiSense-SL	0x14
CPU_CAM_START_IMAGE_STREAM	Start image stream	0x15
CPU_CAM_STOP_IMAGE_STREAM	Stop image stream	0x16
MS_CAM_CONFIG	Camera rectified calibration information	0x90
MS_CAM_START_STREAM_ACK	Acknowledgement	0x91
MS_CAM_STOP_STREAM_ACK	Acknowledgement	0x92
MS_CAM_CONTROL_ACK	Acknowledgement	0x93
MS_CAM_HISTORY	Provides a list of recently-sent frame numbers	0x94
MS_CAM_START_IMAGE_STREAM_ACK	Acknowledgement	0x95
MS_CAM_STOP_IMAGE_STREAM_ACK	Acknowledgement	0x96
MS_CAM_DATA	Provides left camera image and disparity map	0x98
MS_CAM_IMAGE_DATA	Provides left and right camera images	0x99
MS_CAM_SET_HDR_ACK	Acknowledgement	0x9A

Figure : Stereo camera messages.

2.5. LIDAR MESSAGES

Much like the stereo camera, the LIDAR messages relay information about unit configuration, start and stop commands, as well as acknowledgements but also include commands to control the LIDAR’s motor. Detailed information regarding the LIDAR messages can be found in section 3.4.

Type	Description	ID
CPU_LIDAR_GET_CONFIG	Get configuration information	0x20
CPU_LIDAR_START_SCAN	Start streaming scan data	0x21
CPU_LIDAR_STOP_SCAN	Stop streaming scan data	0x22
CPU_LIDAR_SET_MOTOR	Set motor speed, acceleration	0x23
MS_LIDAR_START_SCAN_ACK	Acknowledgement	0xA1
MS_LIDAR_STOP_SCAN_ACK	Acknowledgement	0xA2
MS_LIDAR_MOTOR_ACK	Acknowledgement	0xA3
MS_LIDAR_DATA	LIDAR scan data and motor encoder position, speed, and acceleration	0xA8

Figure : LIDAR messages

2.6. LIGHTING MESSAGES

The messages for the lighting system provide information regarding the status of the LEDs and well as send the commands to set the power level of the lights and provide acknowledgements. Detailed information regarding the LED messages can be found in section 3.5.

Type	Description	ID
CPU_LED_GET_STATUS	Get current LED information	0x30
CPU_LED_SET	Set the LED lighting power	0x31
MS_LED_STATUS	Current LED lighting power	0xB0
MS_LED_SET_ACK	Acknowledgement	0XB1

Figure : Lighting messages

2.7. MESSAGE STRUCTURE

As referenced earlier, the structure of the transmitted messages remains the same regardless of which message format is being sent. The transmitted data starts with a message header, which includes information such as the type, length and sequence of the packet, ensuring that the system is receiving only packets that belong to it and that it does so in the right order.

Header code	Description
Uint8_t magic = 0xDA	Ensures that the system only accepts packets starting with 0xDA, so that foreign packets are ignored.
Uint8_t type	Ensures that the system only accepts packets of a specific type.
Uint16_t sequence id	Provides a unique sequence ID for the packet. Each packet as a unique ID based on the order they were generated.
Unit16_t packet length	The length in bytes of the data segment in this packet.
Uint16_t number packets	*The total number of packets in this message.
Uint16_t packet id	*Provides the ID for the specific packet

Descriptions starting with * only apply to multi-length packets.

Figure : Message headers

Message Component	Description
Header	The first portion of the message structure, generally provides control information. See Figure 8 for more detailed information.
Payload/Data	The body of the message packet that carries any data being sent by either the user's computer or the Multisense
Footer	Message footer is not utilized in this communication system.

Figure : Message Components

2.8. ADDITIONAL DATATYPES

In addition to basic types, the data portion of a message may contain the following datatypes..

Structure	Format	Description
Timestamp	Unit16_t secs Unit16_t nsecs	Provides a time stamp signature in seconds and/or nanoseconds as compared to another point of reference (such as time of last system boot).
General Arrays	Uint32_t size Type data[size]	The structure for a general array that is not an image.
Ack Type	Uint16_t status	Values of “0” indicate a successful status check. Any other number indicates an error.

Figure : Other structures and their formats

3. MESSAGE DETAILS

3.1. OVERVIEW

The tables in the following sections give more detail regarding the messages from the previous tables.

3.2. DETAILED GENERAL MESSAGES

Type	Content	Comments
CPU_ VERSION_REQUEST	None	Header only.
CPU_ STATUS_REQUEST	None	Header only.
MS_VERSION_RESPONSE	Uint16_t version id	
MS_STATUS_RESPONSE	TimeStamp uptime Float temperature0 Float temperature1 Uint32_t status General ok Laser ok Laser motor ok Cameras ok	Temperature0 refers to the FPGA and Temperature1 refers to the Imager, both report temperature in degrees C.

Figure : Detailed general messages

3.3. DETAILED STEREO CAMERA MESSAGES

Type	Content	Comments
CPU_CAM_GET_CONFIG	None	Header only.
CPU_CAM_START_STREAM	None	Header only
CPU_CAM_STOP_STREAM	None	Header only
CPU_CAM_CONTROL	Float exposure (-1: auto) Float gain (-1: auto) Uint32_t framesPerSecond	
CPU_CAM_GET_HISTORY	None	Header only
CPU_CAM_START_IMAGE_STREAM	Bool sendRectifiedImages	
CPU_CAM_STOP_IMAGE_STREAM	None	Header only

Type	Name/Content	Comments
MS_CAM_CONFIG	Uint16_t width Uint16_t heigh Float fx, fy Float cx, cy Float tx, ty, tz Float roll, pitch, yaw	Calibration parameters correspond to rectified images returned by the MultiSense-SL
MS_CAM_START_STREAM_ACK	Ack type Uint16_t status	
MS_CAM_STOP_STREAM_ACK	Ack type Uint16_t status	
MS_CAM_CONTROL_ACK	Ack type Uint16_t status	
MS_CAM_HISTORY	Int32_t history	When the Cam_Data and Cam_Image_Data are both enabled frame ID's will be shared between the stream and images.
MS_CAM_START_IMAGE_STREAM_ACK	Ack type Uint16_t status	
MS_CAM_STOP_IMAGE_STREAM_ACK	Ack type Uint16_t status	
MS_CAM_DATA	Float framesPerSecond Float gain Uint32_t exposureTime Uint32_T frameID Timestamp timestamp; Uint32_t angleOfSpindle Uint16_t width Uint16_t height Uint8_t grayScaleImage[width*height] Uint16_t disparityImage[width*height]	Image arranged in row major order. Default Frames Per Second= 10 Default exposure = 10,000 microseconds Default Gain = 1.0
MS_CAM_IMAGE_DATA	Float framesPerSecond Float gain Uint32_t exposureTime Uint32_t frameID Crl::TimeStamp timestamp Uint32_t angleOfSpindle Uint16_t width Uint16_t height Uint8_t leftImage[width*height] Uint8_t rightImage[width*height]	angleOfSpindle refers to the angle of the laser spindle at the time the photographic data is taken.

Figure : Detailed stereo camera messages

3.4. DETAILED LIDAR MESSAGES

Type	Name/Content	Comment
CPU_LIDAR-GET_CONFIG	None	Header only
CPU_LIDAR_START_SCAN	None	Header only
CPU_LIDAR_STOP_SCAN	None	Header only

CPU_LIDAR_SET_MOTOR	Float rpm	Default RPM is 15
MS_LIDAR_START_SCAN_ACK	Ack type Uint16_t status	
MS_LIDAR_STOP_SCAN_ACK	Ack type Uint16_t status	
MS_LIDAR_MOTOR_ACK	Ack type Uint16_t status	
MS_LIDAR_DATA	Static const uint32_t HOKUYO_MAX_POINTS Static const uint32_t HOKUYO_ECHO_MARKER Uint32_t scanCount TimeStamp timeStart TimeStamp timeEnd Int32_t angleStart Int32_t angleEnd Uint32_t distance[HOKUYO_MAX_POINTS] Uint32_t intensity[HOKUYO_MAX_POINTS] Uint32_t points	Hokuyo Max Points = 1081*4 Hokuyo Echo Marker = (1<<31)

Figure : Detailed LIDAR messages

3.5. DETAILED LIGHTING MESSAGES

Type	Name/Content	Comments
CPU_LED_GET_STATUS	None	Header only
CPU_LED_SET	Static const int MAX_LEDS=4 Uint8_t mask Uint8_t intensity[MAX_LEDS] Uint8_t flash	Allows for up to 4 controllable dimmer LEDs. Intensity is 0-255, with 255=100% power on. The Uint8_t mask selects which LED's are modified. For the Uint8_t flash 0 is the off command and 1 is the on command for the flash.
MS_LED_STATUS	Static const int MAX_LEDS=4 Uint8_t available Uint8_t intensity[MAX_LEDS] Uint8_t flash	Default intensity = 13 Default flash = off
MS_LED_SET_ACK	Ack type Uint16_t status	

Figure : Detailed Lighting messages