PROTOCOL FRAME/EXPLANATION

Frame: 0xA4 0x-- 0x-- 0x-- 0x-- 0x-- 0x-- 0x-- 0x-- 0x-- 0x-- 0x-- … 0x--
Start Length Destination Protocol Response Command Channel Value B Value C Checksum

Start	Length	Destination	Protocol Version	Expected Response	Command Type	16-bits (2 bytes) Channels (LEDs)	(Color Sensors)	...	...	Checksum
0xA4	Length of the rest of the sequence in HEX. 0x00 to 0xFF. This does not include the Start and length bytes.	Two-bytes for 65,536 possible destinations.	MSb: 0 backend protocol flag	0x00: Quick Response	0x00: OK	0x3F: All_LEDS	0x01: RED_CS			Bit-wise xor check
				0x01: Wait Response	0x01: ERR	0x01: RED_LED	0x02: BLUE_CS			
			.	0x02: Repeated Response	0x02: END	0x02: GREEN_LED	0x04: GREEN_CS			
			.		0x03: PING	0x04: BLUE_LED	0x08: AMBER_CS			
			.		0x04: LOCALIZATION*	0x08: AMBER_LED	0x0F: ALL_CS			
					0x05: DEVS_STATUS*	0x10: COOLWHITE_LED	0x00: NO_CS			
					0x06: LOCATION_STATUS*	0x20: WARMWHITE_LED				
			MSb: 1 frontend protocol flag		0x07: HELLO	0x40: INFRARED				
					0x08: ASK_IF_READY+	0x00: NO_LEDS				
					0x09: SYSTEM_READY+					
					0x0A: SYSTEM_BUSY+					
					0x0B: DEVS_STATUS_ALL*					
						These bytes are determined by what the command type is selected. There could be more bytes or less. Commands with standard lengths do not have these bytes.				
			.		0xCA: SLEEP					
			.		0xCB: WAKE					
			.		0xCC: SAMPLE					
					0xCD: SET_DYNAMICS					
					0xCE: GET_GAIN					
					0xCF: GET_PRESCALAR					
					0xAA: SET					
					0xAB: INCREASE					
					0xAC: DECREASE					
					0xAD: DIM					
					0xAE: STATUS					

* Unique to frontend protocol, ^ Unique to backend protocol, +Unique to Smart Client

Standard Command Length is 9 bytes: Start Byte, Length Byte, Destination Bytes (2), Protocol Bytes (2), Response Byte, Command Byte, and Checksum Byte. Anything less is invalid. The first eight bytes are defined as the HEADER.

Start Byte – Every command starts with this byte.

Length Byte – The length byte contains the value of the length in HEX of the rest of the command. This length *does not* include the Start and Length Bytes. It *does* include the checksum byte. Max Length: 0xFF (255).

Destination Bytes – There are two bytes allocated for the Destination. This gives us a possibility of 65,536 different destinations. Destination is used primarily to indicate which TiLED (Arduino) device to contact.

Protocol Bytes – There are two bytes allocated for the Protocol. This gives us a possibility of 65,536 different protocols. An MSB of 1 will indicate frontend protocol and an MSB of 0 will indicate a backend protocol. Frontend is defined as interacting with the server side of the System Server (Ex: user interface, localization interface). Backend is defined as interacting with the client side of the System Server (Ex: Arduino, Optitrack).

Current Protocols:
0xF000: Protocol Front – Used to communicate with a regular client.
0xF001: Protocol Front User – Used to communicate with a smart client.
0x0000: Protocol Back – Used to communicate with the Arduino.

Protocol Tracker has yet to be assigned a value. Currently, Protocol Tracker is using the same identifier as Protocol Front.

Response Byte – The response byte is used to alert the receiving end of the command of what kind of response to expect.

0x00: QUICK RESPONSE – no additional responses after the current.
0x01: WAIT RESPONSE – the receiving end should expect and wait for another command.
0x02: REPEATED RESPONSE – the receiving end should expect multiple commands.

Command Byte – This is the main byte and contains the type of command being sent. Depending on this byte, the rest of the frame and length is determined.

General Commands: (These have Standard Lengths unless noted otherwise by an asterisk)
0x00: OK – General okay command.
0x01: ERR – General error command.
0x02: END – Sent to indicate all data has been sent and to end communication.
0x03: PING – Pings for the status of a specific Arduino.

0x04: LOCALIAZATION – deals with localization*.
0x05: DEVS_STATUS – deals with the online/offline statuses of the all the Arduinos*.
0x06: LOCATION_STATUS – deals with the location of the markers from the Optitrack*.
0x07: HELLO – Hello command, this is the first command sent, usually.
0x08: ASK_IF_READY – Asks if the server is ready. (Smart Client)
0x09: SYSTEM_READY – Server is ready. (Smart Client)
0x0A: SYSTEM_BUSY – Server is not ready, i.e. pinging. (Smart Client)
0x0B: DEVS_STATUS_ALL – deals with the online/offline and location status of all the Arduinos*.

Color Sensor Commands: (None of these have been implemented yet.)
0xCA: SLEEP – Sleeps the color sensor.
0xCB: WAKE – Wake the color sensor.
0xCC: SAMPLE – Sample from the color sensor.
0xCD: SET_DYNAMICS – set the dynamics value of the color sensor.
0xCE: GET_GAIN – get the gain from the color sensor
0xCF: GET_PRESCALER – get the pre-scalar from the color sensor.

LED Controls Commands: (Non-standard Lengths; More info below)
0xAA: SET – Set LED(s) to specified intensities.
0xAB: INCREASE – Increase the LED(s) by specified intensities, with a time duration option.
0xAC: DECREASE – Decrease the LED(s) by specified intensities, with a time duration option.
0xAD: DIM – Dim the LED(s) to specified intensities, with a time duration option.
0xAE: STATUS – deals with the LED intensity status.

Commands with non-standard lengths:

0x04: LOCALIAZATION – Frame: [HEADER][ON/OFF][CHECKSUM]
The LOCALIZATION command is used to turn on and off localization. The Command Type byte is followed by either by ON (0xFF) or OFF (0x00).

0x05: DEVS_STATUS – There are two instances of the DEVS_STATUS command: a query instance and an update instance. For the query instance, the command takes the standard length and format: [HEADER][CHECKSUM]. This instance is usually sent from the user interface to the System Server to ask of the status of all the Arduino Devices. The update instance is sent from the System Server to the user interface with the statuses of the Arduinos. The frame for the update instance is as follows: [HEADER][ARD0 ID][ARD0 STATUS][ARD1 ID][ARD1 STATUS] ... [ARDN ID][ARDN STATUS][CHECKSUM]. The length field will depend on how many Arduinos are associated with the System Server.

0x06: LOCATION_STATUS – This command deals with the Location status of a particular marker. There are two instances of the LOCATION_STATUS command: a query instance and an update instance. The query instance is used to ask for the location of a specific marker. The frame for that is as follows: [HEADER][MARKER ID] [CHECKSUM]. The update instance is used to provide the location of the specified marker. That frame is as follows: [HEADER][MARKER ID][X STATUS][Y STATUS][Z STATUS][CHECKSUM]. The x, y, and z coordinates are given in double precision, 8 bytes each. These commands are used between the System Server and a user interface or localization application.

0x0B: DEVS_STATUS_ALL – There are two instances of DEVS_STATUS_ALL: a query instance and an update instance. Similar to DEVS_STATUS, the query instance takes the standard length and format: [HEADER][CHECKSUM]. This instance is used to ask for more detail information for all the Arduino Devices. The update instance is sent from the System Server to the user interface with the statuses and region locations associated with each Arduino Device linked with the System Server. Locations are double precision and 8 bytes for each coordinate. There are two coordinates: x and y; and a min and max for each coordinate. The frame for the update instance is as follows: [HEADER][Number of Devices][ARD0 ID][ARD0 STATUS][ARD0 X_min][ARD0 X_max][ARD0 Y_min][ARD0 Y_max] ... [ARDN ID][ARDN STATUS][ARDN X][ARDN Y][CHECKSUM]. The length field will depend on how many Arduinos are associated with the System Server.

0xAA: SET – This command is used to turn on LEDs immediately to specified intensities. The intensity ranges from 0 (off) to 255 (max brightness). Each LED color has its own bit within the LED color byte (Explained further in Arduino Documentation). Frame: [HEADER][LED_COLORS][COLOR_SENSOR][R_INTENTSITY][G_INTENSITY][B_INTENSITY][A_INTENSITY][CW_INTENSITY][WW_INTENSITY][CHECKSUM].

0xAB: INCREASE/0xAC: DECREASE – These two functions increase the brightness of the LEDs by specified intensities. There's also an option to set the time duration byte so that the LEDs gradually increase/decrease to their new brightness level. Frame: [HEADER][LED][COLOR_SENSOR][R_INTENTSITY][G_INTENSITY][B_INTENSITY][A_INTENSITY][CW_INTENSITY][WW_INTENSITY][TIME_DURATION][CHECKSUM]

0xAD: DIM – This command dims to specified intensities with time duration; differs from SET in that it has a time duration option. [HEADER][LED][COLOR_SENSOR][R_INTENTSITY][G_INTENSITY][B_INTENSITY][A_INTENSITY][CW_INTENSITY][WW_INTENSITY][CHECKSUM]

0xAE: STATUS – This command has two instances: a query instance and an update instance. The query instance is of standard length and is used to ask for the LED brightness status. The update instance contains the intensities of each LED. Frame: [HEADER] [R_INTENSITY][G_INTENSITY][B_INTENSITY][A_INTENSITY][CW_INTENSITY][WW_INTENSITY][CHECKSUM]

Checksum Byte – This is always the last byte in a command and where the checksum value resides. For our checksum, we decided to use a bitwise xor of all the bytes in the command *excluding* the checksum byte. The checksum is used to determine whether or not data has been corrupted during transfer over the Ethernet.