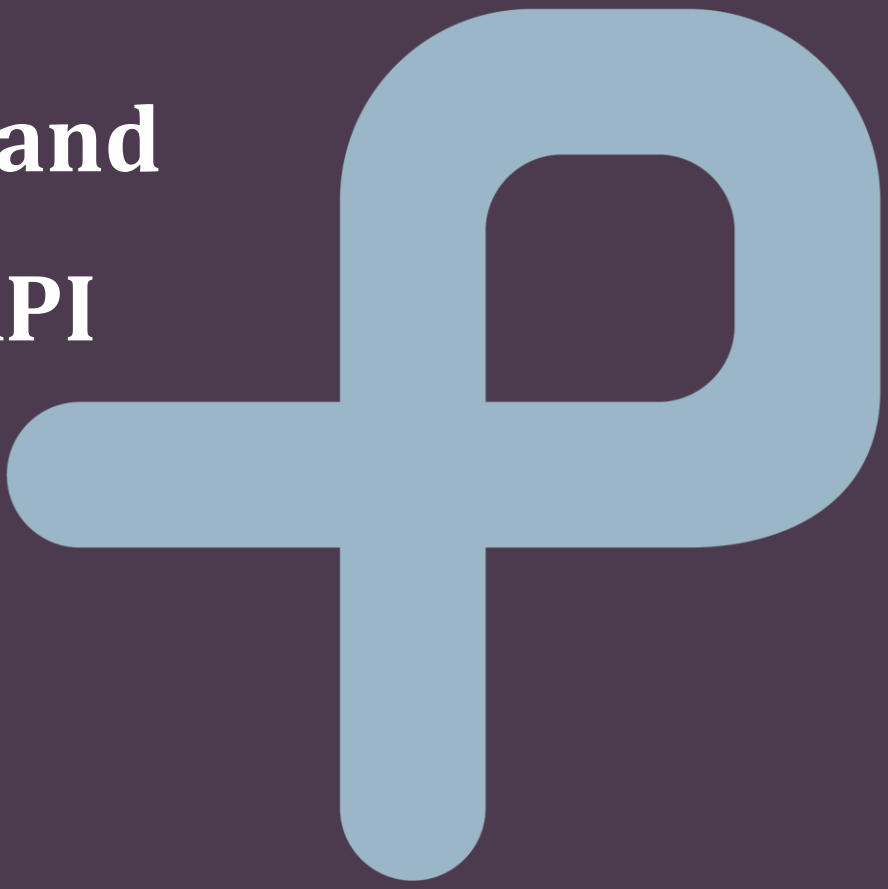
Capture Systems

Command and Control API



CAPTURE
precise tracking in its best

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shaham Petach Tikva 4900617 Israel

Capture Systems
Command and Control API

Capture Systems LTD.
6 Ravnitzky St.,
Petach-Tikva 4900617
Israel
Office: +972-(0)3-3038108
Fax: +972-(0)3-6200621
support@capture-sys.com
www.capture-sys.com

Revision: 2.4.14

Contact

For any questions, assistance and troubleshooting regarding this document please contact us at:
Web: capture-sys.com
Sales: sales@capture-sys.com
Support: support@capture-sys.com
Phone: +972-3-3038108

Copyright and Use Agreement

All data, specifications, and information contained in this publication are based on information that we believe is reliable at the time of printing. Capture Systems reserves the right to make changes without prior notice.

Alerts

The following notifications are used throughout the document to help identify important safety and setup information to the user:

IMPORTANT: identifies actions that may harm the system performance. If the instructions are not clear enough in the tagged section please contact Capture support team before performing any action.

End clause

We welcome all feedback from our customers and will appreciate any comments regarding this manual. If you find any errors while reading this manual or parts which are not clear please send us an email and we will fix the problem.

CAPTURE
precise tracking at its best

Table of contents

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

Revision History

Revision	Date	Changes
2.4.14	1/12/19	Data removed in MOT_Homing command.
2.4.13	27/10/19	Packet reply notes added. Video tracking working procedure added.
2.4.12	8/9/19	Preset structure changed. Video and Stabilization command notes added. Motor short path command added.
2.4.11	3/9/19	Typo in Error control section
2.4.10	10/7/19	Registers ICD merging
2.4.9	4/7/19	Added reply packet structure
2.4.8	1/7/19	Supported tracking modes changed, added acquisition assist, intelligent assist and detection support, set cross types
2.4.7	29/4/19	Added video tracking commands and target list format example, Preset commands added
2.4.6	4/19	Video tracking commands – initial release
2.4.5	5/8/18	Added communication commands
2.4.4	21/5/18	Document design changed, protocol description, scanning description, communication settings (chapter 6)
2.4.3	3/4/18	Typo fixes
2.4.0	13/2/18	Minor design changes, multiple scanning types added
2.3.1	1/1/18	Added GPS support, added motor packets examples
2.3	7/11/17	Packet structure description
1.0	5/6/17	Initial release

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

1 Introduction

The purpose of this document is to explain how to use the Capture systems protocol in the easiest way. The protocol has several types of commands and there are examples for each type of command.
Capture pedestals communication protocol describes the messages structure in order to communicate with the system's internal controller.
Our systems have three types of control levels: Manual, Stabilized and Tracker.
With this protocol the user is presented with a collection of commands which allow for an implementation of motion control applications over the pedestal system.
Communication with the system can be made by the following protocols:

- Ethernet – TCP (default)
- Serial RS-232
- Serial RS-422
- Serial RS-485

Remark: Your specific communication implementation should be mentioned in your order documents.

All packets in the protocol are the same for all communication types. The only difference is in the connection procedure and will be cleared further ahead.
Manual system will allow the user to control all axes in terms of motion profile (acceleration, speed, position, etc...) and retrieve data regarding the status of the motors (voltage, current motor position, etc…). Stabilized and Tracker systems will allow the user an extended control of the system and its components as described in the following table.

Feature	Manual	Stabilized	Tracker	Video
Speed mode motion profile	✓	✓	✓	✓
Position mode motion profile Absolute/Relative	✓	✓	✓	✓
Automatic scanning patterns	Dual axis system only	✓	✓	✓
Presets for desired fixed points of interest	Dual axis systems only	✓	✓	✓
Readings of Real-Time motion parameters	✓	✓	✓	✓
Orientation data about the system (Euler angles)		✓	✓	✓
Ability to stabilize the system on a certain position		✓	✓	✓
Movement under stabilization in all modes		✓	✓	✓
Self GPS position and heading (UTM or LLA)			✓	✓
Presets with datum-points of targets			✓	✓
Automatic aim the system on a GPS point			✓	✓
Video Tracking				✓

- Stabilized, Tracker and Video systems are not supported for single axis pedestals.

2 Protocol packet structure

Protocol packet structure was chosen to enable sending and receiving of message between the user and the system in the easiest way. There are three parts for each packet in the protocol: Header, Data and Check Sum.

Header:

Indicate the start of the packet with the destination of the packet and the OpCode of the command.

Data:

Data being sent in both directions, PC <=> Controller.
Protocol messages can be with or without data.
The data field can have the following data format:

- Double precision 64-bit (8 bytes).
- Floating point 32-bit (4 bytes).
- Unsinged Int 32-bit (UInt32) (4 bytes)
- Unsinged Int 16-bit (UInt16) (2 bytes)
- Unsinged/signed Int 8-bit (UInt8/Int8) (1 byte)
- ASCII string (Data length varies according to the string).

Check Sum:

Summation of all the bytes in the packet except the two Start Bytes and the Check Sum byte.

2.1 Packet Structure

Start Byte 1	Start Byte 2	Length	Group ID	Axis ID	OpCode High	OpCode Low	Data 1	Data 2	…	Data N	Check Sum

Table 1 - Capture Packet Structure

Start Bytes are fixed for every message, they indicate the start of the packet.

Start Byte 1: 0x50.

Start Byte 2: 0x54.

Length: This byte indicates the total number of bytes in the packet minus the Start Bytes, Length and Check Sum,
e.g. for messages without data Length is 0x04, for messages with 4 bytes of data Length is 0x08.

Group ID: Regarding system with multiple pedestal units only. Group ID specifies the pedestal unit which the current message is addressed to. For single pedestal system, this field should be set to 0x00.

Axis ID: This byte specifies which axis the information is sent to. 0x01 is for Yaw, 0x02 is for Pitch and 0x03 is for Roll.
All the commands regarding the motors require Axis ID in order to send the command to the correct destination.

OpCode High: In the Capture Protocol the OpCode is represented by 2 bytes (16-bit) as 4 Hex digits. The OpCode High byte is the first byte (or first two Hex digits).

OpCode Low: The OpCode Low byte is the last byte (or last two Hex digits).
Example:
The command MOT_SendPosition has the OpCode of 0x0132, so in this case the OpCode High will be 0x01 and the OpCode Low will be 0x32.

Data: These bytes are present only in case when data is included in the packet. The controller/user can send packet with or without data (depending on the command). Data type is specified in the protocol for each command.
Data 1 (first byte in order) is always the MSB (most significant byte).
Length byte is determined by the number of data bytes + 4.

Check Sum: This byte is used for error detection, it is the summation remainder (lowest byte in case of overflow) of all the bytes (Hex format) in the packet except the two Start Bytes and the Check Sum.
Check Sum Example:

Sending MOT_GetMotorVoltage message to Axis 1 (Yaw) with OpCode 0x0107:

Start Byte 1	Start Byte 2	Length	Group ID	Axis ID	OpCode High	OpCode Low	Check Sum
0x50	0x54	0x04	0x00	0x01	0x01	0x07	0x0D

Table 2 - Example for MOT_GetMotorVoltage message

The Check sum Byte is calculated in Hex as: 0x04+0x01+0x01+0x07 = 0x0D.
Pay attention to the structure of this message, the data bytes are missing. This is the correct way to send a packet without data. Length byte is 0x04 thus the controller knows that this message has no data.
When the Controller receives the packet, it checks if the Check Sum byte matches the content of the packet. If the checksum test fails, the controller will send Not ACK byte (0xF6) so that the user know there was something wrong while sending that packet.
When the check sum is correct, the controller will execute the command. After the controller is done, it will sent confirm that the message was received correctly. For commands that don't have return data from the controller, there will be an ACK byte sent from the controller – 1 byte of 0x06. In case there is return data the answer packet will be sent without ACK.

When the controller sends back a packet with data, the OpCode will be the same as the one being sent by the user.
For Example in the response packet for the MOT_GetMotorVoltage command the OpCode High is 0x01 and OpCode Low is 0x07.
Returning value of 24.12 for MOT_GetMotorVoltage at Yaw axis will be as follows:

Start Byte 1	Start Byte 2	Length	Group ID	Axis ID	OpCode High	OpCode Low	Data 1	Data 2	Data 3	Data 4	Check Sum
0x50	0x54	0x08	0x00	0x01	0x01	0x07	0x41	0xC0	0xF5	0xC3	0xCA

Table 3 - Example for MOT_GetMotorVoltage return message

2.2 Reply Packet Structure

The protocol has two types of reply packets.

- Ack / Nack reply.
- Data reply.

2.2.1 Ack / Nack Reply

The packet structure of Ack/Nack reply contains a single byte. This byte value indicates the last packet state.

Reply	Byte	Title	Description	Operation
Ack	0x06	Acknowledge	The command is valid and have current checksum	None
Nack	0x16	Pedestal Unavailable	The motor controllers are unavailable for some reason.	Power cycle. If the error remains, please contact Capture support team.
Nack	0x76	Video Tracker Unavailable	The video tracker is not available for some reason. *	Power cycle. If the error remains, please contact Capture support team.
Nack	0xA6	Invalid Command	The controller can't handle the command with the current configuration.	Verify that the current command is valid with your system configuration according to Capture API. e.g. you can't send stabilization commands for manual system and this behavior will cause sending reply of 0xA6
Nack	0xB6	Invalid Motor Checksum	There is an error in the communication line between the Cap-Track controller to the motor controllers.	Power cycle. If the error remains, please contact Capture support team.
Nack	0xE6	Execution Error	General executing error of the command.	Power cycle. If the error remains, please contact Capture support team.
Nack	0xF6	Wrong Checksum	The user sent a packet with wrong checksum.	Verify the checksum calculations of the packet.

- Relevant only for systems that includes video tracker.

2.2.2 Data Reply

Protocol reply packet structure was chosen to enable receiving data from the system in the easiest way. In this case, the reply packet structure is as same as the sending packet structure and also includes the asked data in the data part of the packet.

The fields that might change between the sending packets to the reply packet are the Length, Data and Checksum.

The following example will demonstrate the implementation of a single command that retrieve data from the controller:
User send: *IMU_GetRoll (message without data)*

0x50	0x54	0x04	0x00	0x00	0x06	0x02	0x0C

Controller answer: *return packet with data (value of 30.184 degrees, 0x41F178D5 in hex)*

0x50	0x54	0x08	0x00	0x00	0x06	0x02	0x41	0xF1	0x78	0xD5	0x8F

Marked in blue are the static fields of the packets.
Marked in orange are the changeable fields that changed according to the user request (in this case *IMU_GetRool*).

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

3 Commands Description

Capture commands are categorized into several groups, while each group defines a related set of commands. The groups are:

COM_commands:
Communication commands, for establishing communication with the system controller.

MOT_commands:
- Get Data - retrieve information about the motors current status.
- Set Data – send commands to determine the desired behavior of the motors.

Both Get Data and Set Data require specifying the axis destination.
Yaw is axis No. 1, Pitch is axis No. 2 and Roll is axis No. 3. 0x01, 0x02 and 0x03 in hex.

SCN_commands:
Commands for controlling the scanning modes, setting all the scanning parameters, starting and stopping the scan.

IMU_commands:
For stabilized and tracker systems only. Retrieve system orientation in Euler angles format.

STB_commands:
Activating and stopping the stabilization mechanism of the system as well as moving under stabilization mode.

GPS_commands:
For tracker systems only. Retrieve information about the system GPS position, send/receive targets and track targets in stabilization mode.

IP_commands:
Changing the unit IP address when needed. Take extreme precaution when using these commands and read the example before implementing the IP change. (Ethernet TCP communication only)

PRST_command:
Presets are predefined points that can be saved in the controller memory.

VDT_Commands:
For video tracking applications only, activate the video tracking mechanism and retrieve tracking information.

4 Commands

4.1 Get data commands

4.1.1 MOT_MerRegister

OpCode	0x0101
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Motion Error Register.

4.1.2 MOT_DerRegister

OpCode	0x0102
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Detailed Motion Error Register.

4.1.3 MOT_SrhRegister

OpCode	0x0103
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Motion Status Register High.

4.1.4 MOT_SrlRegister

OpCode	0x0104
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Motion Status Register Low.

4.1.5 MOT_MsrRegister

OpCode	0x0105
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Motion Status Register.

Remark: Information about the registers can be found in a separate file called 'Capture Control Panel Registers'.

4.1.6 MOT_GetMotorCurrent

OpCode	0x0106
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	milli-Amps
Description	Get the actual motor current.

4.1.7 MOT_GetMotorVoltage

OpCode	0x0107
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Voltage
Description	Get the actual motor voltage.

4.1.8 MOT_GetMotorPosition

OpCode	0x0108
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the actual motor position before Gear Ratio.

4.1.9 MOT_GetLoadPosition

OpCode	0x0109
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the actual load position after Gear Ratio.

4.1.10 MOT_GetMotorSpeed

OpCode	0x010A
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/Sec
Description	Get the actual load speed.

4.2 Set data Commands

4.2.1 MOT_SetAcceleration

OpCode	0x0130
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec^2
Return format	None
Data return unit	None
Description	Set the motor acceleration.

4.2.2 MOT_SetSpeed

OpCode	0x0131
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec
Return format	None
Data return unit	None
Description	Set the motor speed.

Remark: After sending Acceleration and Speed, the value remains in the system.
Another command is needed only if there is a need to change those values.

4.2.3 MOT_SendPosition

OpCode	0x0132
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Send the desired amount of movement to the motor.

Remark: Position command (with the last given speed and acceleration) will be executed only after a MOT_Update command.

4.2.4 MOT_Update

OpCode	0x0134
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Update (Execute) the current motion set.

Remark: After a MOT_Update command, the specified motor will execute the current motion profile using the last motion related commands.
These commands include:
MOT_SetAcceleration, MOT_SetSpeed, MOT_SendPosition , MotionComplete, etc…

4.2.5 MOT_Homing

OpCode	0x0135
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Activating Homing mode that is stored on each motor memory.

4.2.6 MOT_SetPositionRelative

OpCode	0x0138
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Setting the position commands to be relative.

Remark: In Relative Mode, the position commands will be added to the current position.

4.2.7 MOT_SetPositionAbsolute

OpCode	0x0139
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Setting the position commands to be absolute.

Remark: In Absolute Mode, the position commands will result an absolute movement of the motor. The motor will move until the position of the motor will be as mentioned in the data sent with the current command.

4.2.8 MOT_SetSpeedMode

OpCode	0x013A
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Setting motor to speed mode operation.

Remark: In Speed mode, the motor will rotate in a constant speed after receiving a MOT_SetSpeed command followed by a MOT_Update command. The motor will keep on rotating in the given speed until it receives different speed (can be zero speed in order to stop the motor) or until a MOT_SetPositionMode is received.

4.2.9 MOT_SetPositionMode

OpCode	0x013B
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Setting motor to position mode operation.

Remark: In position mode, in order to get the motor to rotate, a MOT_SendPosition command must be sent followed by a MOT_Update command.

4.2.10 MOT_AxisOn

OpCode	0x013C
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Turn on the motor.

Remark: Turn on the power supply to the motor.
If the controller buffer had a motion command when turning off the motor with MOT_AxisOff, the controller will return to the last command and execute it after the MOT_AxisOn command is received.

4.2.11 MOT_AxisOff

OpCode	0x013D
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Turn off the motor.

Remark: Turn off the power supply to the motor.
If the motor is currently in motion during the time MOT_AxisOff command is received the motion will stop, but the controller still remembers this last motion command and will continue from where it stopped when MOT_AxisOn is received.

4.2.12 MOT_AxisReset

OpCode	0x013E
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reset the current axis.

Remark: Resets the Axis and restore previous configurations. When resetting, the axis will perform its homing sequence.
After this command, halt all commands for the specific axis for at least 3 seconds in order to avoid communication errors.

4.2.13 MOT_SetTum

OpCode	0x013F
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Update the motion profile parameters.

Remark: Target Update Mode, build a new motion profile (with acceleration, speed and position) from scratch each time a MOT_Update command is sent. When not sending this command, the motion profile will use the last parameters. In some cases the motor will not move properly if this command is missing from the motion commands before sending MOT_Update command. It is advisable to send this command before each command session.

4.2.14 MOT_ResetFaults

OpCode	0x0143
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reset register faults of the desired axis

4.2.15 MOT_SetShortPath

OpCode	0x014E
Data send format	Uint-8
Data send unit	Number
Return format	None
Data return unit	None
Description	Sets the parameter of short path for the system motion.

Remark: This command sets a short path for motion in the horizontal axis. When the value is set to 1 the horizontal movement will be the smaller difference between the current location and the target location, when set to 0 the movement will be the larger difference between them.

4.2.16 MOT_GetShortPath

OpCode	0x014F
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Number
Description	Gets the parameter of short path for the system motion.

4.2.17 MOT_SetMotionComplete

OpCode	0x0144
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Puts the current command in Motion-Complete mode.

Remark: Sending this command will cause the motor driver to wait until the current position command has ended before executing a new one.
This command needs to be sent for each position command separately (before MOT_Update).

Example:
Sending two MOT_SendPosition with Motion complete, first with +20 degrees and second with -15 degrees will cause the motor to finish the +20 degrees and then move -15 degrees.

4.3 Scanning Commands

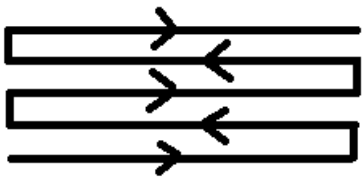
Capture's dual axis systems supports several automatic scanning modes. The scanning modes are:

- **Zig-Zag**

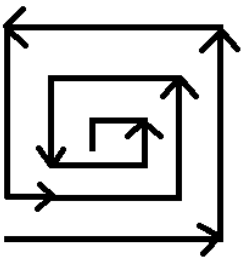
  While activating the Zig-Zag scanning mode the system will run the following sequence:
  - Set the system to the initial position that declared by YawMin and PitchMin variables.
  - Run the system in both axes to the (YawMax, StepHeight) position with the speed declared by SetScanSpeed.
  - Repeat this step according to the NumSteps declaration.
  - After the scan pattern is finished (reaches its higher position) the system will reverse the scan pattern and so on until the user will stop the scan manually.

- **Snake**

  While activating the Snake scanning mode the system will run the following sequence:
  - Set the system to the initial position that declared by YawMin and PitchMin variables.
  - Run the Yaw axis to the YawMax position.
  - Raise the Pitch axis with the StepHeigt value.
  - Run the Yaw axis back to the YawMin value.
  - Raise the Pitch axis with the StepHeigt value.
  - Repeat those steps until the system reach the (YawMax, NumSteps * StepHeight) position and then run back to the start position.

- **Square**

  While activating the Square scanning mode the system will run the following sequence:
  - Set the system to the initial position that declared by YawMin and PitchMin variables.
  - Run the Yaw axis to the YawMax position.
  - Run the Pitch axis to the max Pitch position that calculates by multiplying the SetpHeight and NumSteps variables.
  - Run the Yaw axis to the initial Yaw position.
  - Run the Pitch axis to the PitchMin minus StepHeight value.
  - Run the Yaw axis to the PitchMax minus StepHeight value.
  - Repeat those steps until the system reach the center of the square (according to the NumSteps value) and then reverse to the initial position.

All parameters for the scan area and speed must be sent before starting first scan.
Default values, for all of those parameters, when the program starts are 0.
All parameters must be set before each scan to make sure that the system will perform the desired scanning pattern. All scanning command needs to be send to axis number 0x00.
When the system is in scanning mode, the only motor commands that are allowed to be sent are: MOT_GetMotorSpeed and MOT_GetLoadPosition. Any other motor command will be blocked by the

controller, and the user will receive not valid ACK byte (0xA6).

4.3.1 SCN_SetYawMin

OpCode	0x0400
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the left corner of the scan area.

Remark: The value set is according to the position of the motor which is read by the MOT_GetMotorPosition command.
Example: If MOT_GetMotorPosition return 40 degrees, then setting SCN_YawMin to 50 degrees will result 10 degree rotation clockwise for the scan starting point.

4.3.2 SCN_SetYawMax

OpCode	0x0401
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the right corner of the scan area.

4.3.3 SCN_SetPitchMin

OpCode	0x0402
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the bottom side of the scan area.

Remark: The value set is according to the position of the motor which is read by the MOT_GetMotorPosition command.
Example: If MOT_GetMotorPosition return 0 degrees, then setting SCN_PitchMin to 5 degrees will result 5 degree rotation upwards for the scan starting point.

4.3.4 SCN_SetNumSteps

OpCode	0x0403
Data send format	Unsigned Integer (8-bit)
Data send unit	Integer Number
Return format	None
Data return unit	None
Description	Set the amount of steps from bottom to the top of the scan area.

Remark: One step is a movement from SCN_YawMin to SCN_YawMax with height in the Pitch axis set by the SCN_SetScanStep command. This command relevant for ZigZag and Snake scans only.

4.3.5 SCN_SetStepHeight

OpCode	0x0404
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the height of 1 step for each scan loop.

Remark: This parameter will determine the distance between each loop of the scan.
Relevant for all types of scans.

4.3.6 SCN_SetScanSpeed

OpCode	0x0405
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec
Return format	None
Data return unit	None
Description	Set the speed for the scan.

Remark: This parameter is relevant for all scans. Sets the speed for the horizontal (Yaw) axis for all scans. Vertical (Pitch) speed is calculated according to the scan type.

4.3.7 SCN_SetShortPath

OpCode	0x0406
Data send format	Single byte (value of 0 or 1)
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Set parameter of short path for the scanning.

Remark: This command sets a short path for scanning in the horizontal axis. When the value is set to 1 the horizontal movement will be the smaller difference between SCN_SetYawMax and SCN_SetYawMin, when set to 0 the movement will be the larger difference between them.

4.3.8 SCN_IsScanOn

OpCode	0x0407
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Boolean
Description	Ask the controller if it is in scanning mode.

Remark: Return value is 1 if the controller is in scan mode, and 0 if not.

4.3.9 SCN_StopScan

OpCode	0x0408
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Stop the motors immediately and exit scan mode.

Remark: This command stops both motors immediately and the controller exits scanning mode. Stop scan can take up to 500 millisecond before sending ACK and handle new incoming commands.

4.3.10 SCN_StartScanZigZag

OpCode	0x040C
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start scan in ZigZag pattern with the given parameters.

Remark: When SCN_StartScanZigZag is sent, the motors will go to the bottom left corner of the scan area and then start the scan in a ZigZag pattern.

4.3.11 SCN_StartScanSnake

OpCode	0x040D
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start scan in Snake pattern with the given parameters.

Remark: When SCN_StartScanSnake is sent, the motors will go to the bottom left corner of the scan area and then start the scan in a Snake pattern.

4.3.12 SCN_StartScanSquare

OpCode	0x040E
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start scan in square pattern with the given parameters.

Remark: When SCN_StartScanSquare is sent, the motors will go to the bottom left corner of the scan area and then start the scan in a Square pattern.

4.3.13 SCN Operation Notes

This section doesn't include an operation code of a specific command.
Here we explain some operation behaviours regarding the scanning mechanism.

4.3.13.1 While scanning is active (after activating opcode 0x040C-0x040E) the controller enters to scanning mode. During this mode the controller can receive only several 'Get' commands (that will be described in the next paragraph) and 'StopScan' command (opcode 0x0408). In any case that the user sends a command that is not described here the controller will return an invalid command error: **0xA6**.
The commands that the user can send during scanning are:

OpCode	Command Name
0x0801	STB_StabilizationOff
0x0101 – 0x010A	Chapter 4.1 'Get data commands'
0x0E01 – 0x0E0B	Chapter 4.10 'Error control commands'

4.4 GPS Commands

These commands retrieve data from the GPS unit (relevant to Tracker systems only).
In all of GPS commands, the axis byte in the packet is ignored (can be sent as 0x00).

4.4.1 GPS_GetHeading

OpCode	0x0502
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the current heading given by the GPS module.

Remark: The heading value indicates what is the angle between earth's North Pole and the azimuth (Yaw) pointing direction.

4.4.2 GPS_GetLatitude

OpCode	0x0503
Data send format	None
Data send unit	None
Return format	Double precision 64-bit
Data return unit	Degrees & Decimal minutes
Description	Get the current latitude given by the GPS module.

Remark: The value is positive for the upper side of the latitude lines (North) and negative for the lower side (South).

4.4.3 GPS_GetLongitude

OpCode	0x0504
Data send format	None
Data send unit	None
Return format	Double precision 64-bit
Data return unit	Degrees & Decimal minutes
Description	Get the current longitude given by the GPS module.

Remark: The value is positive for the East side of the latitude lines and negative for the West side.

Both Latitude and Longitude in the format of DDMM.mmmm
DD: degrees, MM.mmmm: decimal minutes.

4.4.4 GPS_GetAltitude

OpCode	0x0505
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Meters
Description	Get the current altitude given by the GPS module.

Remark: The given value is the height of the system above sea level in Meters.

4.4.5 GPS_GetNorthing

OpCode	0x0506
Data send format	None
Data send unit	None
Return format	Double precision 64-bit
Data return unit	Meters
Description	Get the current UTM Northing value given by the GPS module.

4.4.6 GPS_GetEasting

OpCode	0x0507
Data send format	None
Data send unit	None
Return format	Double precision 64-bit
Data return unit	Meters
Description	Get the current UTM Easting value given by the GPS module.

4.4.7 GPS_GetZone

OpCode	0x0508
Data send format	None
Data send unit	None
Return format	Int 8-bit.
Data return unit	Meters
Description	Get the current UTM Zone value given by the GPS module.

Remark: UTM Zone given by the GPS module parsed as follows:
Positive value for Northern Hemisphere, Negative value for Southern Hemisphere.

4.4.8 GPS_PositionReady

OpCode	0x050A
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Boolean
Description	Check if there is a valid position lock from GPS

Remark: this command return 1 if the values for position are valid or 0 if they are not valid.

4.4.9 GPS_HeadingReady

OpCode	0x050B
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Boolean
Description	Check if there is a valid heading from GPS

Remark: this command return 1 if the values for heading are valid or 0 if they are not valid.

4.4.10 GPS_isConnectedGPS

OpCode	0x050C
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Boolean
Description	Check if the system had communication since boot.

Remark: this command return 1 if there was a signal from the GPS since the system has started and 0 if there was no communication at all since system startup.

4.4.11 GPS_GetTargetLLA

OpCode	0x050F
Data send format	None
Data send unit	None
Return format	TargetLLA format (See section 5.1.3)
Data return unit	GPS Target
Description	Read target in LLA format. (target number between 1-15)

Remark: the target number is set in the Axis byte of the packet. The controller can hold up to 15 targets.

4.4.12 GPS_SetTargetLLA

OpCode	0x0510
Data send format	None
Data send unit	TargetLLA format (See section 5.1.3)
Return format	None
Data return unit	None
Description	Send target in LLA format. (targets number between 1-15)

Remark: the target number is set in the Axis byte of the packet. The controller can hold up to 15 targets.

4.4.13 GPS_GetTargetUTM

OpCode	0x0511
Data send format	None
Data send unit	None
Return format	TargetUTM format (See section 5.1.4)
Data return unit	GPS Target
Description	Read target in UTM format. (target number between 1-15)

Remark: the target number is set in the Axis byte of the packet. The controller can hold up to 15 targets.

4.4.14 GPS_SetTargetUTM

OpCode	0x0512
Data send format	TargetUTM format (See section 5.1.4)
Data send unit	GPS Target
Return format	None
Data return unit	None
Description	Read target in UTM format. (target number between 1-15)

Remark: the target number is set in the Axis byte of the packet. The controller can hold up to 15 targets.

4.4.15 GPS_GoToTarget

OpCode	0x0513
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Send the system load to look and lock on the target.

Remark: the target number is set in the Axis byte of the packet. The controller can hold up to 15 targets.

4.4.16 GPS_ClearAllTargets

OpCode	0x0514
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Clear all 15 targets of the controller.

4.5 IMU Commands

These commands retrieve data from the IMU unit.
In all of IMU commands, the axis byte in the packet is ignored (can be sent as 0x00).

4.5.1 IMU_IsReadyImu

OpCode	0x0601
Data send format	None
Data send unit	None
Return format	Unsigned integer (8-bit)
Data return unit	Boolean
Description	Check if the IMU is on and ready to be read.

Remark: This command returns 1 if the IMU unit is connected and communicating or 0 if not.

4.5.2 IMU_GetRoll

OpCode	0x0602
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the current Roll Value read by the IMU.

Remark: Roll can receive values between -180 to 180 degrees.

4.5.3 IMU_GetPitch

OpCode	0x0603
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the current Pitch Value read by the IMU.

Remark: Pitch can receive values between -180 to 180 degrees.

4.5.4 IMU_GetYaw

OpCode	0x0604
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the current Yaw Value read by the IMU.

Remark: Yaw can receive values between 0 to 360 degrees.

4.6 Communication Commands

These messages are responsible on the communication between the PC (client's application) and the Controller.
In all of COM commands, the axis byte in the packet is ignored (can be sent as 0x00).

4.6.1 COM_Reboot

OpCode	0x0700
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reboot the internal managing controller.

Remark: Allow the unit to reboot for about 20 seconds before trying to reconnect.

4.6.2 COM_Connect

OpCode	0x0702
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Starts Conversation with the internal controller.

Remark: This command activates both axes and allows the protocol commands to be processed. Movement mode is set to position mode, relative with speed set to 0 as default.

4.6.3 COM_Disconnect

OpCode	0x0703
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Terminate connection with the controller.

Remark: This command will stop internal controller and turn off the motors power supply. Movement mode is set to position mode, relative.

4.6.4 COM_SetComType

OpCode	0x0719
Data send format	Unsigned Integer (8-bit)
Data send unit	Communication type
Return format	None
Data return unit	None
Description	Change the communication type with the controller.

Remark: The communication types are:
- Ethernet – 0x01
- RS-232 – 0x02
- RS-422 – 0x03
- RS-485 – 0x04

The new communication type will be available after system reboot.
Make sure that your system supports the new communication type according to your order specification.

4.6.5 COM_GetPn

OpCode	0x0C2D
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the Part Number of the system.

4.6.6 COM_GetSn

OpCode	0x0C2F
Data send format	None
Data send unit	None
Return format	Unsinged Int 32-bit
Data return unit	Number
Description	Get the Serial Number of the system.

4.6.7 COM_GetFw

OpCode	0x0C4A
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the Firmware revision of the controller.

4.6.8 COM_GetHw

OpCode	0x0C4C
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	None
Description	Get the Hardware revision of the controller.

4.7 IP Change Commands

The following section is relevant for Ethernet TCP communication type only.
Read all of the following commands carefully! Wrong use of these commands might result irreversible damage to the system. It is advisable to write down all the values before and after the change in case the system won't connect properly.
In all of IP commands, the axis byte in the packet is ignored (can be sent as 0x00).
Please be advised that certain commands change the IP address for the controller unit.
If you intentionally or by mistake use these commands, and hence changing the IP configuration, our guaranty will automatically expire, for issues related to IP change.

In this case you will be able to still send us the system for re-configuration, but this service is not covered under the guaranty, and you will be charged for the re-configuration and shipping costs.

By performing those actions you agree to these terms.

4.7.1 IP_SetControllerIP

OpCode	0x070A
Data send format	4 byte hex value
Data send unit	IP Address
Return format	None
Data return unit	None
Description	Send new IP address for the controller.

Remark: The data in this message is the new IP address for the controller in 4 bytes format.
Example: The address 192.168.10.13 will be sent as: 0xC0, 0xA8, 0x0A, 0x0D.
The new IP address will be activated after reboot.

4.7.2 IP_GetControllerIP

OpCode	0x070D
Data send format	None
Data send unit	None
Return format	4 byte hex value
Data return unit	IP Address
Description	Get the current IP value of the controller.

Remark: The IP address will return as a set of 4 bytes.
Example: The data 0xC0, 0xA8, 0x0A, 0x0D represents the address 192.168.10.13.

4.7.3 IP_SetControllerPort

OpCode	0x070B
Data send format	Unsigned Integer (16 bit).
Data send unit	Port number
Return format	None
Data return unit	None
Description	Send new port for the controller.

Remark: The new port value will be activated after reboot.

4.7.4 IP_GetControllerPort

OpCode	0x070E
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	None
Description	Get port value of the controller.

4.7.5 IP_SetControllerSubnetMask

OpCode	0x071A
Data send format	4 byte hex value
Data send unit	IP Address
Return format	None
Data return unit	None
Description	Send new subnet mask for the controller.

Remark: The data in this message is the new subnet mask address for the controller in 4 bytes format.
Example: The address 192.168.10.13 will be sent as: 0xC0, 0xA8, 0x0A, 0x0D.
The new subnet mask will be activated after reboot.

4.7.6 IP_GetControllerSubnetMask

OpCode	0x070B
Data send format	None
Data send unit	None
Return format	4 byte hex value
Data return unit	IP Address
Description	Get the current subnet value of the controller.

Remark: The subnet mask will return as a set of 4 bytes.
Example: The data 0xC0, 0xA8, 0x0A, 0x0D represents the subnet 192.168.10.13.

4.7.7 IP_SaveIP

OpCode	0x0710
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Save the current IP and port to controller memory.

Remark: After setting the desired values for the Controller IP and Port, this command will save the values to the memory of the controller so that the system will boot with these values each time. (Until changed and saved again).
There is no need to set all values at the same time, each value can be set separately.
It is advisable to read (get) all the values before sending the 'save' command in order to check everything got through to the controller.
After saving all values, a system reboot is needed for the changes to be updated.
IP change is permanent and does not need to be made each time you power on the system.

For any questions or misunderstanding regarding the IP address changing of the system, please contact Capture support team.

4.8 Stabilization commands

All STB commands except STB_StabilizationOn and STB_StabilizationOff require axis specification. When the stabilization mode is active, all Motor-Set data commands are prohibited. The controller will block these commands, and send invalid ACK byte (0xA6).

4.8.1 STB_StabilizationOn

OpCode	0x0800
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Turn stabilization mechanism on.

Remark: When the stabilization mechanism is turned on, the system will point on its current position. A position sample is taken automatically when the command is sent. In other words, the stabilized position is the current position according to the IMU reading.

4.8.2 STB_StabilizationOff

OpCode	0x0801
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Turn stabilization mechanism off.

Remark: After STB_StabilizationOff command, the pedestal speed, acceleration and position is set to 0. Movement mode is position mode, relative.

4.8.3 STB_StabMoveRel

OpCode	0x0802
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Send relative movement to the motor while the stabilization mechanism is active.

4.8.4 STB_StabMoveAbs

OpCode	0x0803
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Send an absolute movement to the motor while the stabilization mechanism is active.

Remark: The angle is calculated according to the IMU reading (Yaw/Pitch).

4.8.5 STB_SetStabSpeed

OpCode	0x0804
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec
Return format	None
Data return unit	None
Description	Set the speed of position movement under stabilization.

Remark: Sets the speed for position mode movement under stabilization. This command completes the position profile building along with the STB_StabMoveAbs or STB_StabMoveRel commands.
The movement will be performed with the last value that was sent in STB_SetStabSpeed.
Speed must be positive value only.
Direction will be defined by the position value sent by STB_StabMoveAbs or STB_StabMoveRel.

4.8.6 STB_StabSpeedOn

OpCode	0x0805
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec
Return format	None
Data return unit	None
Description	Start moving in speed mode under stabilization.

Remark: This command will result continuous movement with the speed specified in the data of this message. Speed can be positive for CW Direction and negative for CCW direction.

4.8.7 STB_StabSpeedOff

OpCode	0x0806
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Stop moving in speed mode under stabilization.

Remark: This command will terminate the movement started with STB_StabSpeedOn command.

4.8.8 STB Operation Notes

This section doesn't include an operation code of a specific command.
Here we explain some operation behaviours regarding the stabilization mechanism.

4.8.8.1 While stabilization is active (after activating opcode 0x0800) the controller enters to stabilization mode. During this mode the controller can receive only several 'Get' commands (that will be described in the next paragraph) and 'StabilizationOff' command (opcode 0x0801). In any case that the user sends a command that is not described here the controller will return an invalid command error: **0xA6**.
The commands that the user can send during stabilization are:

OpCode	Command Name
0x0801	STB_StabilizationOff
0x0101 – 0x010A	Chapter 4.1 'Get data commands'
0x0E01 – 0x0E0B	Chapter 4.10 'Error control commands'

4.9 Preset commands

Presets are pre-defined points that can be saved in the controller memory. Each system can store up to 15 points.
For all preset commands, the preset number should be sent in the *axis* field of the packet.

4.9.1 PRST_GetPreset

OpCode	0x0D00
Data send format	Uint-8
Data send unit	Number
Return format	Preset format (See section 5.1.5)
Data return unit	Preset
Description	Returns the selected preset values according to Preset format.

4.9.2 PRST_SetPreset

OpCode	0x0D01
Data send format	Preset format (See section 5.1.5)
Data send unit	Preset
Return format	None
Data return unit	None
Description	Sets a new value to the selected preset.

4.9.3 PRST_GoToPreset

OpCode	0x0D02
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Drive the system axes to the selected preset.

Remark: the preset number should be sent at the axis number.

4.9.4 PRST_ClearAll

OpCode	0x0D03
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Clear all presets and sets them to 0.

4.9.5 PRST_GetPresetFlag

OpCode	0x0D13
Data send format	None
Data send unit	None
Return format	Uint-16
Data return unit	16 bit vector
Description	Returns a 16 bit vector that indicates which preset is activated.

Remark: bit 0 indicates preset 1 state, bit 1 indicates preset 2 state and so on.

4.9.6 PRST_ClearSinglePreset

OpCode	0x0D14
Data send format	Uint-8
Data send unit	Number
Return format	None
Data return unit	None
Description	Clears single preset value to 0.

Remark: the preset number should be sent at the axis number.

4.10 Error Control commands

Every 'Capture' system includes several status registers. Those registers describe the current status of each axis of the system.
The operation codes for the detailed motor registers are described in section 4.1.1. In this section we will describe the registers description.

Warnings

In any case that error that marked with keep showing please contact Capture-Systems support team in order to monitor those errors.

Most of the errors in the MER and DER registers will stop the motion. So in order to reset the errors (after your inspection) you can send the 'Reset faults' and the 'AxisOn' commands.

For each error you can read the detailed error string using the following opcodes (0x0E03-0x0E09). For each error type the controller can send several error messages. According to the returned error message the user can inspect it and operate the system according to it status.

4.10.1 ERR_MotionErrorRegister

MER is a 16-bit status register. It groups together all of the errors conditions. Most of those errors conditions trigger the Fault status.
OpCode – 0x0101

Bit	Description	Notes
0	CAN bus error	0 – No CAN bus error 1 – CAN bus error Check your CAN bus wiring and communication protocol.
1	Short-circuit protection	0 – No short circuit error 1 – Short circuit error
2	Setup table status	0 – The drive has a valid setup table. 1 – The drive has an invalid setup table
3	Control error	0 – No control error 1 – Control error Control error can occur if the motor doesn't reach the desired position or speed.
4	Communication error	0 – No serial communication error 1 – Serial communication error Please check your serial communication line with the drive.
5	Position sensor error	0 – No error 1 - Error
6	Positive limit switch active	0 – LSP is not active 1 – LSP is active
7	Negative limit switch active	0 – LSN is not active 1 – LSN is active
8	Over current error	0 - No over-current error 1 – Over-current error The system current consumption reached the protection limit.

9	I2T protection error	0 – No drive or motor I2T error 1 – Drive or motor I2T error The system current consumption over time reached the protection limit. Please check if there are no mechanical limitations that prevent the system from moving.
10	Motor over temperature error	0 – No motor over temperature error 1 – Drive motor temperature error. The motor is overheating. Make sure that your system supports your operation environment regarding the operation temperature.
11	Drive over temperature error	0 – No drive over temperature error 1 – Drive over temperature error. The system is overheating. Make sure that your system supports your operation environment regarding the operation temperature.
12	Over voltage error	0 – No over voltage error 1 – Over voltage error Check the system power supply for both logic and motor and set it to the desired level.
13	Under voltage error	0 – No under voltage error 1 – Under voltage error Check the system power supply for both logic and motor.
14	Command error	0 – No command error 1 – Command error. The bit set in two cases: a. An UPD command is received during the AXISON command execution. MER (14) = 1 & SRL (7) = 0. b. A cancelable call is received while a TML function is active following a previous cancelable call. MET (14) = 1 & SRL (7) = 1.
15	Enable status of drive	0 – Drive enabled 1 – Drive disabled Check enable input state (in supported systems only)

4.10.2 ERR_DetailedErrorRegister

DER is a 16-bit status register. It groups together more errors conditions. Most of the errors conditions trigger the Fault status.
OpCode – 0x0102

Bit	Description	Notes
0	TML stack overflow	0 – No stack overflow. 1 – Stack overflow. Check your protocol implementation.
1	TML stack underflow	0 – No stack underflow. 1 – Stack underflow. Check your protocol implementation.
2	Homing not available	0 – Not triggered. 1 – Triggered. If the homing sequence should work in your system you should contact Capture support team.
3	Function not available	0 – Not triggered. 1 – Triggered. If a customized function should work in your system you should contact Capture support team.
4	UPD ignored	0 – Not triggered. 1 – Triggered. Message ignored. Please check the communication protocol.
5	Cancelable call ignored	0 – Not triggered. 1 – Triggered.
6	Software positive limit switch	0 – Not triggered. 1 – Triggered.
7	Software negative limit switch	0 – Not triggered. 1 – Triggered.
8	Invalid S-curve profile.	0 – Not triggered. 1 – Triggered. Resend your S-curve profile with different values.
9	UPD ignored for S-curve	0 – Not triggered. 1 – Triggered. The S-curve parameters are invalid.
10	Encoder broken wire	0 – Not triggered. 1 – Triggered.
11	Motionless start fail	0 – Not triggered. 1 – Triggered. For brushless motors only. Please contact Capture.
12	TML heartbeat ignored	0 – Not triggered. 1 – Triggered. Check your communication wires.
13	Self-check error	0 – Not triggered. 1 – Triggered.
14	Reserved	
15	Reserved	

4.10.3 ERR_StatusRegisterHigh

SRH is the high part of a status register grouping together all the key status information concerning the drive/motor.
OpCode – 0x0103

Bit	Description	Notes
0	Drive initialization status	0 – Not performed. 1 – Performed.
1	Position trigger 1	0 – Not reached. 1 – Reached.
2	Position trigger 2	0 – Not reached. 1 – Reached.
3	Position trigger 3	0 – Not reached. 1 – Reached.
4	Position trigger 4	0 – Not reached. 1 – Reached.
5	Auto run mode status	0 – Disabled. 1 – Enabled.
6	Positive limit switch event/interrupt	0 – Not triggered. 1 – Triggered.
7	Negative limit switch event/interrupt	0 – Not triggered. 1 – Triggered.
8	Capture event/interrupt	0 – Not triggered. 1 – Event/interrupt triggered.
9	Target command	0 – Not reached. 1 – Target reached.
10	Motor I2T protection warning	0 – Motor I2T warning limit not reached. 1 – Motor I2T warning limit reached.
11	Drive I2T protection warning	0 – Drive I2T warning limit not reached. 1 – Drive I2T warning limit reached.
12	Reserved	
13	Gear ratio in electronic gearing mode	0 – Not reached. 1 – Reached.
14	Reference position in absolute electronic camming mode	0 – Not reached. 1 – Reached.
15	Fault status	0 – No fault 1 – Drive/motor in fault status

4.10.4 ERR_StatusRegisterLow

The status register (Low) includes the following data:
OpCode – 0x0104

Bit	Description	Notes
7	Homing / Function call warning	0 – Not triggered. 1 – Warning triggered.
8	Homing / Function call active	0 – Homing not active. 1 – Homing is active.
10	Motion is complete	0 – In motion. 1 – Motion complete.
14	Event has occurred	0 – Not triggered. 1 – Event/interrupt triggered.
15	Axis is on	0 – Axis is Off. 1 – Axis is On.

4.10.5 ERR_MotionStatusRegister

MSR is a 16-bit status register, containing information about motion system status and some specific events like: control error condition, position wrap-around, limit switches, etc.
OpCode – 0x0105

Bit	Description	Notes
0	Drive initialization status	0 – Not performed. 1 – Performed.
1	S-Curve update status	0 – S-curve updated successfully 1 – S-curve update denied.
2	Software protection status	0 – Not triggered. 1 – Triggered.
3	Control error status	0 – Not triggered. 1 – Triggered.
4	Reserved	
5	Position wrap-around	0 – Not triggered. 1 – Triggered.
6	Positive limit switch	0 – Not triggered. 1 – Triggered.
7	Negative limit switch	0 – Not triggered. 1 – Triggered.
8	Position capture	0 – Not triggered. 1 – Triggered.
9	Motion status	0 – In motion. 1 – Motion complete.
10	Contour	0 – Not in contour mode. 1 – In contour mode.
11	Events	0 – No event set, or programmed event not occurred yet. 1 – Last event reached.
12	Reserved	
13	Axis status	0 – Axis off. 1 – Axis on.
14	Event status	0 – Reset after update. 1 – Set for update.
15	Update the motion mode	0 – No update. 1 – Update.

4.10.6 ERR_CaptureMotorErrorRegister

CMER (Capture reduced Motor Error Register) is a 16-bit status resister. It groups together the most relevant data from the MER, DER, SRL and SRH into single register.
OpCode – 0x0E0B

Bit	Description	Notes
0	Fault status	0 – No fault 1 – Drive/motor in fault status
1	Axis status	0 – Axis off. 1 – Axis on.
2	Motion is complete	0 – In motion. 1 – Motion complete.
3	Enable status of drive	0 – Drive enabled 1 – Drive disabled Check enable input state (in supported systems only)
4	Under voltage error	0 – No under voltage error 1 – Under voltage error Check the system power supply for both logic and motor.
5	Over voltage error	0 – No over voltage error 1 – Over voltage error Check the system power supply for both logic and motor and set it to the desired level.
6	I2T protection error	0 – No drive or motor I2T error 1 – Drive or motor I2T error The system current consumption over time reached the protection limit. Please check if there are no mechanical limitations that prevent the system from moving.
7	Over current error	0 - No over-current error 1 – Over-current error The system current consumption reached the protection limit.
8	Negative limit switch active	0 – LSN is not active 1 – LSN is active
9	Positive limit switch active	0 – LSP is not active 1 – LSP is active
10	Control error	0 – No control error 1 – Control error The motor doesn't reach the desired position or speed.
11	Short-circuit protection	0 – No short circuit error 1 – Short circuit error
12	Encoder broken wire	0 – Not triggered. 1 – Triggered.
13	Software negative limit switch	0 – Not triggered. 1 – Triggered.
14	Software positive limit switch	0 – Not triggered. 1 – Triggered.

4.10.7 ERR_CaptureSystemRegister

OpCode	0x0E01
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Capture System Register is a 16-bit status register. It groups together data regarding the entire system status.

Bit	Description	Notes
0	Motor Error	0 – No error 1 – Error in one of the motors
1	System Error	0 – No Error. 1 – Error.
2	Base IMU error	0 – No Error. 1 – Error.
3	Load IMU error	0 – No Error. 1 – Error.
4	GPS communication error	0 – No Error. 1 – Error.
5	GPS position error	0 – No Error. 1 – Error.
6	GPS heading error	0 – No Error. 1 – Error.

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

4.10.8 ERR_ClearErrors

OpCode	0x0E02
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Clears the CSR register

4.10.9 ERR_GetMotorErrorString

OpCode	0x0E03
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the motor error description

Remark: The error messages description is in the end of this section.

4.10.10 ERR_GetSystemErrorString

OpCode	0x0E04
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the system error description

Remark: The error messages description is in the end of this section.

4.10.11 ERR_GetLoadImuErrorString

OpCode	0x0E05
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the load IMU error description

Remark: The error messages description is in the end of this section.

4.10.12 ERR_GetBaseImuErrorString

OpCode	0x0E06
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the base IMU error description

Remark: The error messages description is in the end of this section.

4.10.13 ERR_GetGpsComErrorString

OpCode	0x0E07
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the GPS communication error description

Remark: The error messages description is in the end of this section.

4.10.14 ERR_GetGpsPosString

OpCode	0x0E08
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the GPS position description

Remark: The error messages description is in the end of this section.

4.10.15 ERR_GetGpsHeadErrorString

OpCode	0x0E09
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the GPS Head error description

Remark: The error messages description is in the end of this section.

4.10.16 ERR_GetProtocolErrorString

OpCode	0x0E0A
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the protocol error description

Remark: The error messages description is in the end of this section.

4.10.17 Error messages description:

OpCode	Description	Error message
0x0E01	Get CSR register	-
0x0E02	Clear errors	-
0x0E03	Get motor error string	- No communication with Pan motor driver - No communication with Tilt motor driver - No communication with Roll motor driver - No communication with Four motor driver - No communication with Five motor driver - No communication with Six motor driver
0x0E04	Get system error string	- General system error - Fail to access memory
0x0E05	Get load IMU error string	- No communication with Load IMU - Fail to sync with Load IMU
0x0E06	Get base IMU error string	- No communication with Base IMU - Fail to sync with Base IMU
0x0E07	Get GPS communication error string	- No communication with the GPS
0x0E08	Get GPS position error string	- The position of the GPS is not ready
0x0E09	Get GPS heading error string	- The heading of the GPS is not ready
0x0E09	Get protocol error string **Remark**: It is recommended to read this error after receiving an 'A6' NACK from the controller	- Axis not exist in the system - Axis 0 doesn't exist - Command not valid while stabilizing - Command not valid while homing - Invalid command for Manual system - Invalid command for single axis pedestal - IMU commands not available - GPS commands not available - Opcode not Recognized - Target no. out of range - Target not exist

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

4.11 Video Tracking commands

All VDT commands are available for Video tracking application only.
For systems that include two video channels (cam 1 and cam 2) the camera index should be sent in the *axis number* field.
For commands that are not related to the camera channels, the axis number should be 0.

4.11.1 VDT_GetImageSize

OpCode	0x0720
Data send format	None
Data send unit	None
Return format	Uint-16
Data return unit	Pixels
Description	Returns the image size in pixels. First 2 bytes of the data related to the width and the last 2 bytes related to the height.
Camera Index	Required

4.11.2 VDT_SetTrackMode

OpCode	0x0721
Data send format	Uint-8
Data send unit	Tracking mode
Return format	None
Data return unit	None
Description	Sets the tracking mode
Camera Index	Required

Remark: The supported tracking modes are:
No Tracking - 0x00
Drone Tracking mode – 0x01
Vehicle Tracking mode - 0x02

4.11.3 VDT_GetTrackMode

OpCode	0x0722
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Tracking mode
Description	Gets the current tracking mode
Camera Index	Required

Remark: The supported tracking modes are:
No Tracking – 0x00
Drone Tracking mode – 0x01
Vehicle Tracking mode - 0x02

4.11.4 VDT_SetPtControlMode

OpCode	0x0723
Data send format	Uint-8
Data send unit	Control mode
Return format	None
Data return unit	None
Description	Sets the Pedestal movement master
Camera Index	Not Required

Remark: The pedestal can get movement commands from two instances, Joystick and C&C application. This command sets the master of the pedestal control.
Joystick mode – 0x00
Command and Control application mode - 0x01

4.11.5 VDT_GetPtControlMode

OpCode	0x0724
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Control mode
Description	Gets the current Pedestal movement master
Camera Index	Not Required

Remark: The pedestal can get movement commands from two instances, Joystick and C&C application. This command returns the current master of the pedestal control.
Joystick mode – 0x00
Command and Control application mode - 0x01

4.11.6 VDT _SetLatitude

OpCode	0x0725
Data send format	Double precision 64-bit
Data send unit	Degrees & Decimal minutes
Return format	None
Data return unit	None
Description	Sets the current system latitude
Camera Index	Not Required

Remark: The value is positive for the upper side of the latitude lines (North) and negative for the lower side (South).

4.11.7 VDT _GetLatitude

OpCode	0x0726
Data send format	None
Data send unit	None
Return format	Double precision 64-bit
Data return unit	Degrees & Decimal minutes
Description	Gets the current system latitude
Camera Index	Not Required

Remark: The value is positive for the upper side of the latitude lines (North) and negative for the lower side (South).

4.11.8 VDT _SetLongitude

OpCode	0x0727
Data send format	Double precision 64-bit
Data send unit	Degrees & Decimal minutes
Return format	None
Data return unit	None
Description	Sets the current system longitude
Camera Index	Not Required

Remark: The value is positive for the East side of the latitude lines and negative for the West side.

Both Latitude and Longitude in the format of DDMM.mmmm
DD: degrees, MM.mmmm: decimal minutes.

4.11.9 VDT _GetLongitude

OpCode	0x0728
Data send format	None
Data send unit	None
Return format	Double precision 64-bit
Data return unit	Degrees & Decimal minutes
Description	Gets the current system longitude
Camera Index	Not Required

Remark: The value is positive for the East side of the latitude lines and negative for the West side.

Both Latitude and Longitude in the format of DDMM.mmmm
DD: degrees, MM.mmmm: decimal minutes.

4.11.10 VDT _SetAltitude

OpCode	0x0729
Data send format	Float
Data send unit	Meters
Return format	None
Data return unit	None
Description	Sets the current system altitude in meters
Camera Index	Not Required

4.11.11 VDT _GetAltitude

OpCode	0x072A
Data send format	None
Data send unit	None
Return format	Float
Data return unit	Meters
Description	Gets the current system altitude in meters
Camera Index	Not Required

4.11.12 VDT _GetHeading

OpCode	0x072A
Data send format	None
Data send unit	None
Return format	Float
Data return unit	Degrees
Description	Gets the current system heading
Camera Index	Not Required

Remark: The value is accurate only after a Northing process was done. Without the northing process this command will return 0.

4.11.13 VDT_GetTrackError

OpCode	0x072C
Data send format	Uint-8
Data send unit	Target index (1-5)
Return format	Uint-32
Data return unit	Pixels
Description	Returns the tracking error from the boresight in pixels. First 2 bytes of the data related to the width and the last 2 bytes related to the height.
Camera Index	Required

4.11.14 VDT_VideoStabilization

OpCode	0x072D
Data send format	Uint-8
Data send unit	Video stabilization state
Return format	None
Data return unit	None
Description	Turn on/off the video stabilization functionality
Camera Index	Required

Remark: The video stabilization states are:
Off – 0x00
On – 0x01

4.11.15 VDT_StartTrackXy

OpCode	0x072E
Data send format	Uint-16
Data send unit	Width and Height in pixels
Return format	None
Data return unit	None
Description	Start tracking on selected pixel
Camera Index	Required

Remark: This commands start to track on an object that located in the selected pixel and draws the tracking square around it.
The first two bytes of the data refer to the object width location and the last 2 bytes refer to the object height location.

4.11.16 VDT_GoToLlaTarget

OpCode	0x072F
Data send format	Target LLA format (see section 5.1.3)
Data send unit	LLA Position
Return format	None
Data return unit	None
Description	Runs the system to the desired target
Camera Index	Not Required

4.11.17 VDT_SetHeading

OpCode	0x0730
Data send format	Float
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Runs the system Heading value
Camera Index	Not Required

Remark: The heading value is valid after system northing.

4.11.18 VDT_GetTargetsList

OpCode	0x0731
Data send format	None
Data send unit	None
Return format	Target list format (see section 5.1.6)
Data return unit	Target position in pixels
Description	Returns the targets list according to the list format.
Camera Index	Required

4.11.19 VDT_StartTrackTargetIndex

OpCode	0x0732
Data send format	Uint-8
Data send unit	Target index
Return format	None
Data return unit	None
Description	Start tracking on a target according to its index
Camera Index	Required

4.11.20 VDT_StopTrack

OpCode	0x0733
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Stops the tracking on the current target.
Camera Index	Required

Remark: This stop command should be sent to stop the current tracking (regardless on the tracking type: XY or index).

4.11.21 VDT_SetAcquisitionAssist

OpCode	0x0734
Data send format	Uint-8
Data send unit	Acquisition assist state
Return format	None
Data return unit	None
Description	Assist initialization of the track box size and location. Impacts user designated targets in all tracking modes.
Camera Index	Required

Remark: The acquisition assist states are:
Off – 0x00
On – 0x01

4.11.22 VDT_GetAcquisitionAssist

OpCode	0x0735
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Acquisition assist state.
Description	Returns the acquisition assist state
Camera Index	Required

Remark: The acquisition assist states are:
Off – 0x00
On – 0x01

4.11.23VDT_ SetIntelligentAssist

OpCode	0x0736
Data send format	Uint-8
Data send unit	Intelligent assist state.
Return format	None
Data return unit	None
Description	Tracker will look for turning tracks and if detected will attempt a reacquisition. The new track will be followed for a period of time before replacing the current track.
Camera Index	Required

Remark: Requires Acquisition Assist to be enabled
The intelligent assist states are:
Off – 0x00
On – 0x01

4.11.24VDT_ GetIntelligentAssist

OpCode	0x0737
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Intelligent assist state.
Description	Returns the intelligent assist state
Camera Index	Required

Remark: The intelligent assist states are:
Off – 0x00
On – 0x01

4.11.25VDT_TargetDetectionOn

OpCode	0x0738
Data send format	Uint-8
Data send unit	Detection mode
Return format	None
Data return unit	None
Description	Returns the intelligent assist state
Camera Index	Required

Remark: The supported detection modes are:
No Detection – 0x00
Drone Detection – 0x01
Vehicle Detection - 0x02

4.11.26 VDT_TargetDetectionOff

OpCode	0x0739
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Turns off the target detection mode
Camera Index	Required

4.11.27 VDT_SetCrossType

OpCode	0x073A
Data send format	Uint-8
Data send unit	Cross type
Return format	None
Data return unit	None
Description	Set the overlay cross type
Camera Index	Required

Remark: The supported cross types are:
No Cross – 0x00
Red Cross (+) – 0x01
Red Circle (o) - 0x02
Purple Square (■) – 0x03

4.11.28 VDT Operation Notes

This section doesn't include an operation code of a specific command.
Here we explain some operation behaviours regarding the video tracking application.

4.11.28.1 While video tracking is active (after activating opcodes 0x072E or 0x0732) the controller is in video tracking mode. During this mode the controller can receive only several 'Get' commands (that will be described in the next paragraph) and 'StopTrack' command (opcode 0x0733). In any case that the user sends a command that is not described here the controller will return an invalid command error: **0xA6**.
The commands that the user can send during video tracking are:

OpCode	Command Name
0x0733	VDT_StopTrack
0x0101 – 0x010A	Chapter 4.1 'Get data commands'
0x0E01 – 0x0E0B	Chapter 4.10 'Error control commands'

4.11.28.2 Video tracking system can be assembled with one or two cameras.
Most of the video commands requires camera index (in case that we mentioned otherwise at the opcode description).
The camera index (1 or 2) should be sent in the *axis* field of the packet. In cases that the camera index isn't required the user should send 0 in the *axis* field.

5 Appendix

5.1 Data Formats

All data that is sent and received in the Capture protocol is in the topology of "Big Endian". This means that when sending the bytes of the desired format the MSB is first to arrive in the packet.

5.1.1 Floating point 32-bit format example

The floating point number -45.87 in Hex presentation is 0xC2377AE1. Implementation with the protocol:

Data 1	Data 2	Data 3	Data 4
0XC2	0X37	0x7A	0xE1

The returned data from the controller is in the same order as mentioned above.

5.1.2 Double precision 64-bit format example

The double precision number 12.6492487231 in Hex presentation is 0x40294C6A54215E57
Implementation with the protocol:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
0x40	0x29	0x4C	0x6A	0x54	0x21	0x5E	0x57

The returned data from the controller is in the same order as mentioned above.

5.1.3 Target packet format LLA

For the commands GetTargetLLA (0x050F) and SetTargetLLA (0x0510) the format of the data in the packet is in the following way:

Longitude [8 byte double]	Latitude [8 byte double]	Altitude[4 byte float]

- Longitude – longitude of the target in decimal degrees format (DD.dddd).
  The value is positive for the East side of the longitude lines and negative for the West side.
- Latitude – latitude of the target in decimal degrees format (DD.dddd).
  The value is positive for the Northern side of the latitude lines and negative for the Southern side.
- Altitude – altitude of the target in meters.

Length byte of the packet in this case is 24 (0x18 in Hex).
* Format is the same for sending and receiving.
* Target number is specified in the Axis byte. The available targets numbers are 1-15.

5.1.4 Target packet format UTM

The Capture protocol uses standard UTM (WGS 84).
For commands GetTargetUTM (0x0511) and SetTargetUTM (0x0512) the format of the data in the packet is in the following way:

Easting [8 byte double]	Northing [8 byte double]	Zone[1 byte signed integer]	Altitude[4 byte float]

- Easting – easting of the target in meters (WGS84 standard).
  Northing – northing of the target in meters (WGS84 standard).
- Zone – The zone of the target: Positive value for Northern hemisphere and negative for Southern hemisphere.
- Altitude – altitude of the target in meters.

Length byte of the packet in this case is 25 (0x19 in Hex).
* Format is the same for sending and receiving.
* Target number is specified in the Axis byte. Targets numbers available are 1-15

5.1.5 Preset packet format

Each controller can hold position for up to 6 axes.
For commands GetPreset (0x0D00) and SetPreset (0x0D01) the format of the data in the packet is in the following way:

Name [16 bytes ASCII]	Azimuth [4 bytes float]	Elevation [4 bytes float]	Roll [4 bytes float]	X [4 bytes float]	Y [4 bytes float]	Z [4 bytes float]

- Name – The user can set a 16 bytes preset name that will be saved in the controller memory.
- Azimuth – Azimuth value of the Preset in degrees.
- Elevation – Elevation value of the Preset in degrees.
- Roll – Roll value of the Preset in degrees.
- X – X value of the Preset in degrees.
- Y – Y value of the Preset in degrees.
- Z – Z value of the Preset in degrees.

* The values are absolute as read from the encoder position (Value read by the command MOT_GetLoadPosition).

5.1.6 Targets list format

For GetTargetsList command (0x0732) the format of the data field in the packet is in the following way:

T_1Width	T_1Height	T_2Width	T_2 Height	T_3Width	T_3 Height	T_4Width	T_4 Height	T_5Width	T_5 Height

- The system can hold up to 5 targets at once.
- Each value is represented Uint-16 [2 bytes], so the length of the data field is 20 bytes.
- In case that there are less than 5 detected targets the value of the relevant target will be (0, 0).

6 Communication settings

This section will discuss the different communication options and default values as well as how to change those values if needed.
There are several ways for the user to communicate with the controller:

- Ethernet TCP
- Serial RS-232
- Serial RS-422
- Serial RS-485

6.1 Default values for Ethernet:

	IP	PORT
Controller	192.168.10.120	4949
PC	No default	No default

Table 4 – IP default settings

The controller comes with a default IP and Port as mentioned in Table 4. Since the user can connect from different PC with different IP every time, the controller listens to the Ethernet line and whenever there is a TCP connect request, the controller opens an active connection via TCP to the PC. The Ethernet communication establishing process is as follows:

1. The user sends a TCP connect request to the controller.
2. The controller sends back a COM_Connect message.
3. The user send COM_Connect message.
4. The controller returns acknowledge (0x06) to the user.

Each time the user wishes to connect to the controller those steps should be performed.
Connecting with a different PC is possible even when the program is running, the way to do that is to disconnect with the current PC and connect with the new one.

Controller IP can be changed for any new IP address. Please refer to section 4.7 for more information.

6.2 Default values for Serial ports:

All of the Serial lines have the same default values for the communication.

Baud Rate	Data bits	Parity bits	Stop bits
115200	8	None	1

Table 5 – Serial default settings

7 Packet send/receive Examples

This chapter focuses on giving the user some basic examples for each command type to be able to implement the protocol in an easier way.
There is a basic example written in C# for creating and using Ethernet TCP socket with some examples of sending and receiving of packets that can be downloaded from our website.

When sending commands to the controller there are several ways to receive feedback from the controller side signifying if the command was received successfully and executed correctly. These feedback messages are identical for all types of communication.

1. Sending wrong checksum:
   When the user sends a packet with one of Capture protocol commands and the checksum byte doesn't match the content of the packet, the controller will send a single byte of 0xF6, this means that the controller tested the checksum byte and the test failed. The controller will not execute commands for packets with wrong checksum byte.

2. Error executing the command:
   For all types of commands, when the controller encounters a problem and for some reason is not able to execute the command, it will send the user a single byte of 0xE6.

3. Sending invalid command:
   When the system is in stabilization, tracking or scan modes, there are some restrictions for what MOT commands can be sent. Meaning that if there are commands that are prohibited to be sent when the controller is in one of these modes it will not execute them. The response from the controller for invalid commands is a single byte of 0xA6.
   It is advised to check for response from the controller side, and if there is a single byte instead of a packet, the user should know that the command sent was invalid.
4. Pedestal unavailable:
   In case that the motor controllers are unavailable for some reason, the controller will send to the user a single byte of 0x16
5. Video tracker unavailable:
   For systems that includes video tracker only. In case that the video tracker is unavailable the controller will send the user a single byte of 0x76.

6. Acknowledge for valid command:
   For commands that are valid and have correct checksum, the controller will send an ACK byte of 0x06.

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel

7.1 MOT motion example

MOT commands are used for controlling the motors of the system. The following example will demonstrate a full motion profile commands that moves the Yaw axis at a certain angle.

- MOT_SetTum
- MOT_SetPositionRelative
- MOT_SetAcceleration
- MOT_SetSpeed
- MOT_SendPosition
- MOT_Update

After sending the following six commands, the specified motor will turn the Yaw axis.
MOT_Update must come last in the sequence.
Each square stands for one byte in hex representation:

User send: *MOT_ SetTum without data*

0x50	0x54	0x04	0x00	0x01	0x01	0x3F	0x45

Controller answer: *Acknowledge*

0x06

User send: *MOT_SetPositionRelative without data*

0x50	0x54	0x04	0x00	0x01	0x01	0x38	0x3E

Controller answer: *Acknowledge*

0x06

User send: *MOT_SetAcceleration with data of 100 (Degrees/sec^2)*

0x50	0x54		0x08	0x00	0x01	0x01	0x30	0x42	0xC8	0x00	0x00	0x44

Controller answer: *Acknowledge*

0x06

User send: *MOT_SetSpeed with data of 27.78(Degrees/sec)*

0x50	0x54	0x08	0x00	0x01	0x01	0x31	0x41	0xDE	0x3D	0x71	0x08

Controller answer: *Acknowledge*

0x06

User send: *MOT_SetPosition with data of 13.487(Degrees)*

0x50	0x54	0x08	0x00	0x01	0x01	0x32	0x41	0x57	0xCA	0xC1	0x5F

Controller answer: *Acknowledge*

0x06

User send: *MOT_Update without data*

0x50	0x54	0x04	0x00	0x01	0x01	0x34	0x3A

Controller answer: *Acknowledge*

0x06

7.2 Get data commands

This example will demonstrate the implementation of a single command that retrieve data from the controller.
Asking for the IMU Roll value from the system is made the following way:

User send: *IMU_GetRoll (message without data)*

0x50	0x54	0x04	0x00	0x00	0x06	0x02	0x0C

Controller answer: *return packet with data (value of 30.184 degrees, 0x41F178D5 in hex)*

0x50	0x54	0x08	0x00	0x00	0x06	0x02	0x41	0xF1	0x78	0xD5	0x8F

After the controller sends the answer, it does not wait for some kind of acknowledge from the user side.
Remark: reading IMU / GPS data from the controller is available only while the relevant sensor is installed.

T. +972-3-3038108 | Fax +972-3-6200621
sales@capture-sys.com | www.capture-sys.com
6 Ravnitzky st. Beit Shoham Petach Tikva 4900617 Israel