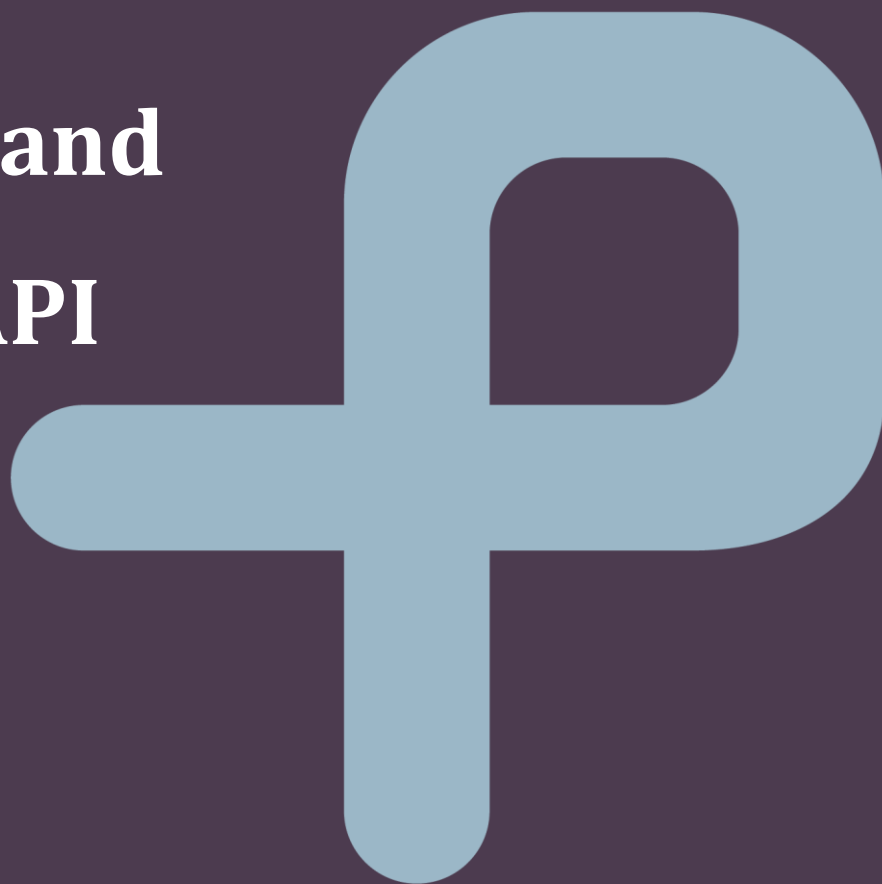


Capture Systems

Command and Control API



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: 2.20.0

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

©Copyright 2024, Capture-Systems Ltd. All Rights reserved. The Capture-Systems name and logo and all related product and service names, design marks and slogans are the trademarks, and service marks of Capture-Systems Ltd.

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.



Points on important information that can impact the system performance.

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.

Table of contents

1 INTRODUCTION	19
2 PROTOCOL PACKET STRUCTURE	20
2.1 PACKET STRUCTURE	21
2.2 REPLY PACKET STRUCTURE	23
2.2.1 Ack / Nack Reply	23
2.2.2 Data Reply	24
3 COMMANDS DESCRIPTION	25
4 COMMANDS	26
4.1 GET DATA COMMANDS	26
4.1.1 MOT_MerRegister	26
4.1.2 MOT_DerRegister	26
4.1.3 MOT_SrhRegister	26
4.1.4 MOT_SrlRegister	26
4.1.5 MOT_MsrRegister	27
4.1.6 MOT_GetMotorCurrent	27
4.1.7 MOT_GetMotorVoltage	27
4.1.8 MOT_GetMotorPosition	27
4.1.9 MOT_GetLoadPosition	28
4.1.10 MOT_GetMotorSpeed	28
4.1.11 MOT_GetShortPath	28
4.1.12 MOT_GetMaxCurrent	29
4.1.13 MOT_GetPositionErr	29
4.1.14 MOT_GetIncPosition	29
4.2 SET DATA COMMANDS	30
4.2.1 MOT_SetAcceleration	30
4.2.2 MOT_SetSpeed	30
4.2.3 MOT_SendPosition	30
4.2.4 MOT_SetActualPosition	31
4.2.5 MOT_Update	31
4.2.6 MOT_Homing	31
4.2.7 MOT_SetPositionRelative	32
4.2.8 MOT_SetPositionAbsolute	32
4.2.9 MOT_SetSpeedMode	32
4.2.10 MOT_SetPositionMode	33
4.2.11 MOT_AxisOn	33
4.2.12 MOT_AxisOff	33
4.2.13 MOT_AxisReset	34
4.2.14 MOT_SetTum	34
4.2.15 MOT_ResetFaults	34
4.2.16 MOT_SetShortPath	35

4.2.17 MOT_SaveMotorSetting	35
4.2.18 MOT_SetMaxCurrent	35
4.2.19 MOT_SetMotionComplete	36
4.3 SCANNING COMMANDS	37
4.3.1 SCN_SetYawMin	38
4.3.2 SCN_SetYawMax	38
4.3.3 SCN_SetPitchMin	38
4.3.4 SCN_SetNumSteps	39
4.3.5 SCN_SetStepHeight	39
4.3.6 SCN_SetScanSpeed	39
4.3.7 SCN_SetShortPath	40
4.3.8 SCN_IsScanOn	40
4.3.9 SCN_StopScan	40
4.3.10 SCN_StartScanZigZag	41
4.3.11 SCN_StartScanSnake	41
4.3.12 SCN_StartScanSquare	41
4.3.13 SCN Operation Notes	41
4.4 GPS COMMANDS	42
4.4.1 GPS_GetLongitude	42
4.4.2 GPS_GetLatitude	42
4.4.3 GPS_GetAltitude	42
4.4.4 GPS_GetSatellites	43
4.4.5 GPS_PositionReady	43
4.4.6 GPS_HeadingReady	43
4.4.7 GPS_isConnectedGPS	43
4.4.8 GPS_GetTargetLLA	44
4.4.9 GPS_SetTargetLLA	44
4.4.10 GPS_GoToTarget	44
4.4.11 GPS_ClearAllTargets	45
4.4.12 GPS_StartTargetLooping	45
4.4.13 GPS_StopTargetLooping	45
4.4.14 GPS_StartTargetTracking	45
4.4.15 GPS_StopTargetTracking	46
4.4.16 GPS_GetStartupStatus	46
4.4.17 GPS_SetTrackFreq	46
4.4.18 GPS_GetTrackFreq	46
4.4.19 GPS_SetInsertedLLA	47
4.4.20 GPS_GetInsertedLLA	47
4.4.21 GPS_SetInsertedHeading	47
4.4.22 GPS_GetInsertedHeading	47
4.4.23 GPS_SaveInsertedPos	48
4.4.24 GPS_SetOvdPosGPS	48
4.4.25 GPS_GetOvdPosGPS	48
4.4.26 GPS_SetGpsIP	49
4.4.27 GPS_GetGpsIP	49

4.4.28 GPS_EnGpsRead	49
4.4.29 GPS_GetEnGpsRead	49
4.5 IMU COMMANDS	50
4.5.1 IMU_IsReadyImu	50
4.5.2 IMU_GetRoll	50
4.5.3 IMU_GetPitch	50
4.5.4 IMU_GetYaw	51
4.5.5 IMU_GetGyroX	51
4.5.6 IMU_GetGyroY	51
4.5.7 IMU_GetGyroZ	51
4.5.8 IMU_ResetSensor	52
4.5.9 IMU_GetTemperatue	52
4.5.10 IMU_GetPressure	52
4.5.11 IMU_Leveling	52
4.5.12 IMU_setLevelingValue	53
4.5.13 IMU_getLevelingValue	53
4.5.14 IMU_setIMUBaseOffset	53
4.5.15 IMU_getIMUBaseOffset	53
4.5.16 IMU_setIMUBodyOffset	54
4.5.17 IMU_getIMUBodyOffset	54
4.5.18 IMU_setIMULoadOffset	54
4.5.19 IMU_getIMULoadOffset	54
4.5.20 IMU_setGyroFilterGain	55
4.5.21 IMU_getGyroFilterGain	55
4.5.22 IMU_getPanPositionByGyro	55
4.5.23 IMU_getTiltPositionByGyro	56
4.5.24 IMU_getRollPositionByGyro	56
4.5.25 IMU_getPanSpeedByGyro	56
4.5.26 IMU_getTiltSpeedByGyro	57
4.5.27 IMU_getRollSpeedByGyro	57
4.6 COMMUNICATION COMMANDS	58
4.6.1 COM_Reboot	58
4.6.2 COM_Connect	58
4.6.3 COM_Disconnect	58
4.6.4 COM_IsConnected	59
4.6.5 COM_StartKeepAlive	59
4.6.6 COM_IsKeepAliveOn	59
4.6.7 COM_setKeepAliveTimeout	60
4.6.8 COM_getKeepAliveTimeout	60
4.6.9 COM_setKeepAliveCount	60
4.6.10 COM_getKeepAliveCount	60
4.6.11 COM_SetComType	61
4.6.12 COM_SysState	61
4.6.13 COM_GetPn	62
4.6.14 COM_GetSn	62

4.6.15 COM_GetFw	62
4.6.16 COM_GetHw	62
4.7 IP CHANGE COMMANDS	63
4.7.1 IP_SetControllerIP	63
4.7.2 IP_GetControllerIP	63
4.7.3 IP_SetControllerPort	64
4.7.4 IP_GetControllerPort	64
4.7.5 IP_SetControllerSubnetMask	64
4.7.6 IP_GetControllerSubnetMask	65
4.7.7 IP_SaveIP	65
4.8 STABILIZATION COMMANDS	66
4.8.1 STB_StabilizationOn	66
4.8.2 STB_StabilizationOff	66
4.8.3 STB_GetStabError	66
4.8.4 STB_SetStabMinAccel	67
4.8.5 STB_GetStabMinAccel	67
4.8.6 STB_SetSpdKp	67
4.8.7 STB_GetSpdKp	67
4.8.8 STB_SetSpdKi	68
4.8.9 STB_GetSpdKi	68
4.8.10 STB_SetSpdKd	68
4.8.11 STB_GetSpdKd	68
4.8.12 STB_SetPosKp	69
4.8.13 STB_GetPosKp	69
4.8.14 STB_SetPosKi	69
4.8.15 STB_GetPosKi	69
4.8.16 STB_SetPosKd	70
4.8.17 STB_GetPosKd	70
4.8.18 STB_SetDistanceToTarget	70
4.8.19 STB_GetDistanceToTarget	70
4.8.20 STB_SetCentralAxes	71
4.8.21 STB_GetCentralAxes	71
4.8.22 STB_SetAxisRadiusCompensation	71
4.8.23 STB_GetAxisRadiusCompensation	72
4.8.24 STB_SetRebalanceOn	72
4.8.25 STB_GetRebalanceOn	72
4.8.26 STB_GetRebalanceStatus	73
4.8.27 STB_SetFeedforwardGain	73
4.8.28 STB_GetFeedforwardGain	73
4.8.29 STB_SaveStabilizationCfg	74
4.8.30 STB_SetAxisStabEnable	74
4.8.31 STB_GetAxisStabEnable	74
4.9 PRESET COMMANDS	75
4.9.1 PRST_GetPreset	75
4.9.2 PRST_SetPreset	75

4.9.3	PRST_GoToPreset	75
4.9.4	PRST_ClearAll	75
4.9.5	PRST_StartPresetLooping	76
4.9.6	PRST_StopPresetLooping	76
4.9.7	PRST_GetPresetFlag	76
4.9.8	PRST_ClearSinglePreset	76
4.9.9	PRST_SetLoopingDelay	77
4.9.10	PRST_GetLoopingDelay	77
4.10	ERROR CONTROL COMMANDS	78
4.10.1	ERR_MotionErrorRegister	78
4.10.2	ERR_DetailedErrorRegister	80
4.10.3	ERR_StatusRegisterHigh	81
4.10.4	ERR_StatusRegisterLow	82
4.10.5	ERR_MotionStatusRegister	83
4.10.6	ERR_CaptureMotorErrorRegister	84
4.10.7	ERR_CaptureSystemRegister	85
4.10.8	ERR_ClearErrors	86
4.10.9	ERR_GetDriverErrorString	86
4.10.10	ERR_GetSystemErrorString	86
4.10.11	ERR_GetLoadImuErrorString	86
4.10.12	ERR_GetBaseImuErrorString	87
4.10.13	ERR_GetGpsComErrorString	87
4.10.14	ERR_GetGpsPosString	87
4.10.15	ERR_GetGpsHeadErrorString	87
4.10.16	ERR_GetProtocolErrorString	88
4.10.17	ERR_GetAbsEncComErrorString	88
4.10.18	ERR_GetAbsEncCRCErrorString	88
4.10.19	ERR_GetBodyImuErrorString	88
4.10.20	ERR_GetHomingErrorString	89
4.10.21	ERR_OperationRegister	89
4.10.22	Error messages description:	90
4.11	VIDEO TRACKING COMMANDS	91
4.11.1	VDT_GetImageSize	91
4.11.2	VDT_SetTrackMode	91
4.11.3	VDT_GetTrackMode	91
4.11.4	VDT_SetPtControlMode	92
4.11.5	VDT_GetPtControlMode	92
4.11.6	VDT_SetLatitude	93
4.11.7	VDT_GetLatitude	93
4.11.8	VDT_SetLongitude	93
4.11.9	VDT_GetLongitude	94
4.11.10	VDT_SetAltitude	94
4.11.11	VDT_GetAltitude	94
4.11.12	VDT_GetHeading	95
4.11.13	VDT_GetTrackError	95

4.11.14VDT_VideoStabilization	95
4.11.15VDT_StartTrackXy	96
4.11.16VDT_GoToLlaTarget	96
4.11.17VDT_SetHeading	96
4.11.18VDT_GetTargetsList	97
4.11.19VDT_StartTrackTargetIndex	97
4.11.20VDT_StopTrack	97
4.11.21VDT_SetAcquisitionAssist	98
4.11.22VDT_GetAcquisitionAssist	98
4.11.23VDT_SetIntelligentAssist	98
4.11.24VDT_GetIntelligentAssist	99
4.11.25VDT_TargetDetectionOn	99
4.11.26VDT_TargetDetectionOff	99
4.11.27VDT_SetCrossType	100
4.11.28VDT Operation Notes	100
4.12 CONFIGURATION COMMANDS	101
4.12.1 CFG_setStartupReg	101
4.12.2 CFG_getStartupReg	101
4.12.3 CFG_SetGeneralSpd	101
4.12.4 CFG_GetGeneralSpd	102
4.12.5 CFG_setPosRange	102
4.12.6 CFG_getPosRange	102
4.13 PELCO-D SETTINGS COMMANDS	103
4.13.1 PELCOD_EnPelcoD	103
4.13.2 PELCOD_IsActivePelcoD	103
4.13.3 PELCOD_SetAddress	103
4.13.4 PELCOD_GetAddress	104
4.13.1 PELCOD_SetAxisNum	104
4.13.2 PELCOD_GetAxisNum	104
4.13.3 PELCOD_SetMinSpd	104
4.13.4 PELCOD_GetMinSpd	105
4.13.5 PELCOD_SetMaxSpd	105
4.13.6 PELCOD_GetMaxSpd	105
4.13.7 PELCOD_SetTurboSpd	105
4.13.8 PELCOD_GetTurboSpd	106
4.13.9 PELCOD_SetAccel	106
4.13.10PELCOD_GetAccel	106
4.14 RELAYS COMMANDS	107
4.14.1 RLY_ActivateModule	107
4.14.2 RLY_IsActivateModule	107
4.14.3 RLY_SetRelay	107
4.14.4 RLY_GetRelay	108
4.14.5 RLY_SetRelayPeriod	108
4.14.6 RLY_SetRelayJogging	108
4.14.7 RLY_GetModuleErr	109

4.14.8 RLY_RstComModule	109
4.14.9 RLY_SetModuleIP	109
4.14.10 RLY_GetModuleIP	110
4.14.11 RLY_SetNumRelays	110
4.14.12 RLY_GetNumRelays	110
4.15 DUAL GIMBALS COMMANDS	111
4.15.1 DG_SetSyncMode	111
4.15.2 DG_SetInnerMode	111
4.15.3 DG_IsSyncMode	112
4.15.4 DG_IsInnerMode	112
4.15.5 DG_GetPosDiff	112
4.15.6 DG_SetMaxDiff	112
4.15.7 DG_GetMaxDiff	113
4.15.8 DG_IsAxesSynced	113
4.15.9 DG_GoToCamera	113
4.15.10 DG_SetSyncSpd	113
4.15.11 DG_SetMainGimbal	114
4.15.12 DG_GetMainGimbal	114
4.15.13 DG_CTC	114
4.15.14 DG_GetCTCOffset	115
4.15.15 DG_ResetCTCOffset	115
4.15.16 DG_GetSyncSpd	115
4.15.17 DG_SetActiveWeapon	116
4.15.18 DG_GetActiveWeapon	116
4.15.19 DG_SetActiveCamera	116
4.15.20 DG_GetActiveCamera	116
4.15.21 DG_IsBoresightEn	117
4.15.22 DG_GetBoresightOffset	117
4.15.23 DG_SetBallisticOffset	117
4.15.24 DG_GetBallisticOffset	118
4.15.25 DG_GetSafetyStatus	118
4.15.26 DG_GetLoadStatus	118
4.15.27 DG_GetCapsnapIOStatus	119
4.15.28 DG_GetNumBullets	119
4.15.29 DG_ResetNumBullets	120
4.15.30 DG_IsCapSnapReady	120
4.16 SOFTWARE LIMIT SWITCH	121
4.16.1 SWLS_SetNegSWLS	121
4.16.2 SWLS_SetPosSWLS	121
4.16.3 SWLS_ActivateSWLS	122
4.16.4 SWLS_GetNegSWLS	122
4.16.5 SWLS_GetPosSWLS	122
4.16.6 SWLS_IsActiveSWLS	122
4.16.7 SWLS_SetHandler	123
4.16.8 SWLS_GetHandler	123

4.16.9 SWLS_SetOffsetValue	124
4.16.10 SWLS_GetOffsetValue	124
4.16.11 SWLS_SetSlowdownValue	124
4.16.12 SWLS_GetSlowdownValue	124
4.17 LRF COMMANDS	125
4.17.1 LRF_SetManualDistance	125
4.17.2 LRF_GetManualDistance	125
4.17.3 LRF_SetMinMeasurement	125
4.17.4 LRF_GetMinMeasurement	126
4.17.5 LRF_SetMaxMeasurement	126
4.17.6 LRF_GetMaxMeasurement	126
4.17.7 LRF_SetFrequency	126
4.17.8 LRF_GetFrequency	127
4.17.9 LRF_StartMeasurement	127
4.17.10 LRF_StopMeasurement	127
4.17.11 LRF_GetLRFDistance	127
4.17.12 LRF_SetDistanceSource	128
4.17.13 LRF_GetDistanceSource	128
4.17.14 LRF_GetTargetDistance	128
4.17.15 LRF_IsConnected	129
4.18 WEAPON CONTROL UNIT COMMANDS.....	130
4.18.1 WCU_GetFirmware	130
4.18.2 WCU_ComStatus	130
4.18.3 WCU_OpenSafety	130
4.18.4 WCU_GetSafetyStatus	131
4.18.5 WCU_StartLoading	131
4.18.6 WCU_GetLoadStatus	131
4.18.7 WCU_SetTrigger	132
4.18.8 WCU_GetTriggerStatus	132
4.18.9 WCU_GetIOStatus	132
4.18.10 WCU_GetNumBullets	133
4.18.11 WCU_ResetNumBullets	133
5 APPENDIX	134
5.1 DATA FORMATS	134
5.1.1 Floating point 32-bit format example	134
5.1.2 Double precision 64-bit format example	134
5.1.3 Target packet format LLA	134
5.1.4 Preset packet format	135
5.1.5 Targets list format	135
6 COMMUNICATION SETTINGS	136
6.1 DEFAULT VALUES FOR ETHERNET:	136
6.2 DEFAULT VALUES FOR SERIAL PORTS:	136
7 PACKET SEND/RECEIVE EXAMPLES	137

7.1 MOT MOTION EXAMPLE	138
7.2 GET DATA COMMANDS	139
7.3 GO TO TARGET COMMANDS	140
7.3.1 <i>Non Tracker system</i>	140
7.3.2 <i>Tracker system</i>	141

Revision History

Revision	Date	Changes
2.20.0	05/02/25	1 LRF commands were added: 1.1 LRF_SetMinMeasurement – 0x0302 1.2 LRF_GetMinMeasurement – 0x0303 1.3 LRF_SetMaxMeasurement – 0x0304 1.4 LRF_GetMaxMeasurement – 0x0305 1.5 LRF_SetFrequency – 0x0306 1.6 LRF_GetFrequency – 0x0307 1.7 LRF_StartMeasurement – 0x0308 1.8 LRF_StopMeasurement – 0x0309 1.9 LRF_GetLRFDistance – 0x030A 1.10 LRF_SetDistanceSource – 0x030B 1.11 LRF_GetDistanceSource – 0x030C 1.12 LRF_GetTargetDistance – 0x030D 1.13 LRF_IsConnected – 0x030E 2 Weapon control unit commands were added: 2.1 WCU_GetFirmware – 0x0B04 2.2 WCU_ComStatus – 0x0B05 2.3 WCU_OpenSafety – 0x0B06 2.4 WCU_GetSafetyStatus – 0x0B07 2.5 WCU_StartLoading – 0x0B08 2.6 WCU_GetLoadStatus – 0x0B09 2.7 WCU_SetTrigger – 0x0B0A 2.8 WCU_GetTriggerStatus – 0x0B0B 2.9 WCU_GetIOStatus – 0x0B0C 2.10 WCU_GetNumBullets – 0x0B0D 2.11 WCU_ResetNumBullets – 0x0B0E
2.19.0	05/11/24	1 GPS commands were added: 1.1 GPS_SetGpsIP – 0x0535 1.2 GPS_GetGpsIP – 0x0536 1.3 GPS_EnGpsRead – 0x0537 1.4 GPS_GetEnGpsRead – 0x0538 2 Dual Gimbals command was added: 2.1 DG_IsAxesSynced – 0x0FA8 3 Stabilization commands were added: 3.1 STB_SetAxisStabEnable – 0x0857

		3.2 STB_GetAxisStabEnable – 0x0858 Remarks: - Negative absolute position commands are enabled again.
2.18.0	09/09/24	1 IMU command was added: 1.1 IMU_getGyroFilterGain - 0x0640 1.2 IMU_setGyroFilterGain - 0x0641 1.3 IMU_getPanPositionByGyro - 0x064A 1.4 IMU_getTiltPositionByGyro - 0x064B 1.5 IMU_getRollPositionByGyro - 0x064C 1.6 IMU_getPanSpeedByGyro - 0x064D 1.7 IMU_getTiltSpeedByGyro - 0x064E 1.8 IMU_getRollSpeedByGyro - 0x064F 2 Stabilization commands were added: 2.1 STB_SetFeedforwardGain – 0x0852 2.2 STB_GetFeedforwardGain – 0x0853
2.17.1	14/07/24	1 Dual Gimbals command was added: 1.1 DG_IsBoresightEn – 0x0FB9 1.2 DG_GetBoresightOffset – 0x0FBC 2 Dual Gimbals commands- Ballistic offset were added: 2.1 DG_SetBallisticOffset – 0x0FBD 2.2 DG_GetBallisticOffset – 0x0FBE 3 Dual Gimbals commands- reading from CapSnap unit were added: 3.1 DG_GetSafetyStatus – 0x0FBF 3.2 DG_GetLoadStatus – 0x0FC0 3.3 DG_GetCapsnapIOStatus – 0x0FC1 3.4 DG_GetNumBullets – 0x0FC2 3.5 DG_ResetNumBullets – 0x0FC3 3.6 DG_IsCapSnapReady – 0x0FC4 4 Stabilization commands were added: 4.1 STB_SetRebalanceOn – 0x084F 4.2 STB_GetRebalanceOn – 0x0850 4.3 STB_GetRebalanceStatus – 0x0851 5 Stabilization commands were removed: 5.1 STB_SetStabInterval – 0x082F 5.2 STB_GetStabInterval – 0x0830 5.3 STB_SetStabCtrlType – 0x0835 5.4 STB_GetStabCtrlType – 0x0836 5.5 STB_SetStabAccelMulSpd – 0x0828 5.6 STB_GetStabAccelMulSpd – 0x0829

		5.7 STB_SetStabAccelMulPos – 0x0837 5.8 STB_GetStabAccelMulPos – 0x0838 5.9 STB_setGyroFilter – 0x082D 5.10 STB_getGyroFilter – 0x082E 5.11 STB_SetEulerFilter – 0x083B 5.12 STB_GetStabEulerLPF – 0x083C 5.13 STB_SetStabCspd – 0x0831 5.14 STB_GetStabCspd – 0x0832
2.17.0	27/06/24	1 Dual Gimbals commands were added: 1.1 DG_SetMaxDiff – 0x0FA6 1.2 DG_GetMaxDiff – 0x0FA7 1.3 DG_SetSyncSpd – 0x0FAA 1.4 DG_GetSyncSpd – 0x0FB5 1.5 DG_SetActiveWeapon – 0x0FB6 1.6 DG_GetActiveWeapon – 0x0FB7 1.7 DG_SetActiveCamera – 0x0FBA 1.8 DG_GetActiveCamera – 0x0FBB 2 LRF commands were added: 2.1 LRF_SetRange – 0x0300 2.2 LRF_GetRange – 0x0301 3 MOT_GetIncPosition – 0x017C 4 Rename ‘sighting in’ commands to ‘CTC’.
2.16.0	20/12/23	1 Software limit switch commands shown in separate section. New commands were added: 1.1 Setting SWLS handler 1.2 Setting offset value. 1.3 Setting slowing down value. 2 New Error register was added operations error – 0x0E12 3 Mot_GetPositionErr command was added – 0x010E
2.15.1	12/11/23	1. Dual Gimbals ‘sighting in’ commands were added: 1.1 DG_SetSightingIn – 0x0FB0 1.2 DG_GetSightingInOffset – 0x0FB1 1.3 DG_ResetSightingInOffset – 0x0FB2
2.14.0	01/08/23	1. Settings commands of Pelco-D protocol were added. 2. Relays commands were added. 3. Dual Gimbals commands were added. 4. Remarks: - Negative commands will be ignored with absolute mode.

		- Name of 'PresetSpd' commands were changed to 'GeneralSpd'. - Name of 'HomingImu' commands were changed to 'Leveling' - SWLS: negative limit must be less than positive limit.
2.13.6	03/05/23	1. Stabilization configuration commands were added. 2. Setting offset for IMU reading commands were added: 2.1. IMU_setIMUBaseOffset – 0x0C7A 2.2. IMU_getIMUBaseOffset – 0x0C7B 2.3. IMU_setIMUBodyOffset – 0x0C78 2.4. IMU_getIMUBodyOffset – 0x0C79 2.5. IMU_setIMULoadOffset – 0x0C6D 2.6. IMU_getIMULoadOffset – 0x0C6E
2.12.0	15/03/22	1. Keep alive mechanism was added with its commands: 1.1. COM_StartKeepAlive – 0x0705 1.2. COM_IsKeepAliveOn – 0x0706 1.3. COM_setKeepAliveTimeout – 0x0708 1.4. COM_getKeepAliveTimeout – 0x0709 1.5. COM_setKeepAliveCount – 0x071C 1.6. COM_getKeepAliveCount – 0x071D 2. More error indications were added to CaptureSystemRegister and the following error descriptions: 2.1. ERR_GetAbsEncComErrorString – 0x0E0D 2.2. ERR_GetAbsEncCRCErrorString – 0x0E0E 2.3. ERR_GetBodyImuErrorString – 0x0E0F 2.4. ERR_GetHomingErrorString – 0x0E10 3. The following configuration commands were added: 3.1. CFG_setStartupReg – 0x0C51 3.2. CFG_getStartupReg – 0x0C52 3.3. CFG_setPosRange – 0x0C70 3.4. CFG_setPosRange- 0x0C71 4. The following commands were added: 4.1. PRST_StartPresetLooping – 0x0D11 4.2. PRST_StopPresetLooping – 0x0D12 4.3. PRST_SetLoopingDelay – 0x0D1C 4.4. PRST_GetLoopingDelay – 0x0D1D 4.5. PRST_SetPresetSpd – 0x0D1E 4.6. PRST_GetPresetSpd – 0x0D1F

		4.7. GPS_StartTargetLooping – 0x0516 4.8. GPS_StopTargetLooping – 0x0517 4.9. IMU_HomingIMU – 0x016C 4.10. IMU_setHomeValueIMU – 0x0C6B 4.11. IMU_getHomeValueIMU – 0x0C6C 4.12. COM_IsConnected– 0x0704 4.13. COM_SysState – 0x0713
2.10.1	14/11/21	1. The following commands were added in Firmware 2.10.1: 1.1 MOT_GetNegSWLS – 0x010C 1.2 MOT_GetPosSWLS – 0x010D 1.3 MOT_IsActiveSWLS – 0x0110 1.4 MOT_SetNegSWLS – 0x0136 1.5 MOT_SetPosSWLS – 0x0137 1.6 MOT_ActivateSWLS – 0x0146 1.7 MOT_SetActualPosition – 0x0133 1.8 MOT_SaveMotorSetting – 0x0165
2.10.0	20/10/21	1. Enable 'Targets' functions for non-tracker systems by insert the system's position and heading. 2. The following commands were added in Firmware 2.10.0: 2.1 GPS_SetTrackFreq– 0x0529 2.2 GPS_GetTrackFreq – 0x052A 2.3 GPS_SetInsertedLLA– 0x052C 2.4 GPS_GetInsertedLLA– 0x052D 2.5 GPS_SetInsertedHeading– 0x052E 2.6 GPS_GetInsertedHeading– 0x052F 2.7 GPS_SaveInsertedPos– 0x0530 2.8 GPS_SetOvdPosGPS– 0x0531 2.9 GPS_GetOvdPosGPS– 0x0532
2.9.0	14/9/21	1. The following commands removed in Firmware version 2.9.0: 1.1. GPS_GetHeading – 0x0502 1.2. GPS_GetTargetUTM– 0x0511 1.3. GPS_SetTargetUTM– 0x0512 1.4. STB_StabMoveRel – 0x0802 1.5. STB_StabMoveAbs – 0x0803 1.6. STB_SetStabSpeed – 0x0804 1.7. STB_StabSpeedOn – 0x0805 1.8. STB_StabSpeedOff – 0x0806

		2. Add the flowing commands: 2.1. IMU_GetGyroX – 0x0605 2.2. IMU_GetGyroY – 0x0606 2.3. IMU_GetGyroZ – 0x0607 2.4. IMU_GetTemperature – 0x0637 2.5. IMU_GetPressure – 0x0638 3. When using 'IMU' and 'GPS' commands (0x0500 – 0x06FF) the axis value should be zero. 4. Removed UTM targets type.
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

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 four types of control levels: Manual, Stabilized, Tracker and Video.

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 desired communication type should be mentioned in your order documents.

Manual system will allow the user to control all axes in terms of motion profile and retrieve data regarding the status of the motors. Systems with connected sensors 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

Table 1 - Capture 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
--------------------	--------------------	--------	-------------	------------	----------------	---------------	-----------	-----------	-----	-----------	--------------

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 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_GetRoll*).

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.

GPS_commands:

Retrieve information about the system GPS position for tracker systems only. Send/receive targets and track targets.

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.

4.1.11 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.

Remark: This command sets a short path for movement in absolute mode. When the value is 1 the movement will be the minimal difference between the current location and the target location, when the value is 0 the axis will reach the exact desired location.

For example, if the current location is 0, the target location is -10°. The axis will move 10° in the negative direction, with or without short path. When the target location is 350°, with short path the axis will again move 10° in the negative direction, without the short path, it will move 350 degrees in the positive direction. For absolute systems, only the short path option is available.

4.1.12 MOT_GetMaxCurrent

OpCode	0x012E
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	milli-Amps
Description	Get the maximum allowed current.

4.1.13 MOT_GetPositionErr

OpCode	0x010E
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	degree
Description	Get the value of position error

Remark: The position error is the difference between the current position and the desired target position. For absolute mode only.

4.1.1 MOT_GetIncPosition

OpCode	0x017C
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	degree
Description	Get the position of the incremental encoder

Remark: Get the incremental position in addition to the absolute position, when the load position ([MOT_GetLoadPosition](#)) is obtained from the absolute encoder. For supported systems only.

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.

Remark: The acceleration value should be greater than 0.

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_SetActualPosition

OpCode	0x0133
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the actual position to the motor.

Remark: This command will change the motor position to the input data. Be aware to the software limit switch, setting position outside the limits will cause a motor error.

4.2.5 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...

4.2.6 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.

Remark: After this command the motors will go to a 'mechanical' constant position and will set the reading motor position to 0.

4.2.7 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.

4.2.8 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.

4.2.9 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.

4.2.10 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.

4.2.11 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.

4.2.12 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.

4.2.13 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.

4.2.14 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.

4.2.15 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

4.2.16 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.

4.2.17 MOT_SaveMotorSetting

OpCode	0x0165
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Save the motor setting to the memory.

Remark: Do not turn off your system until the save is complete.

4.2.18 MOT_SetMaxCurrent

OpCode	0x0166
Data send format	Floating point 32-bit
Data send unit	milli-Amps
Return format	None
Data return unit	None
Description	Set the maximum allowed current.

4.2.19 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:

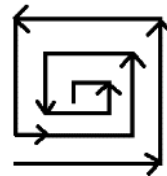
- Set the system to the initial position that declared by YawMin and PitchMin variables.
- Run the Yaw axis to the YawMax position.
- Raise the Pitch axis with the StepHeight value.
- Run the Yaw axis back to the YawMin value.
- Raise the Pitch axis with the StepHeight value.
- Repeat those steps until 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 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.



While
and

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 milliseconds 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. In any case that the user sends a motion command during this mode, the controller will stop the scanning and will execute the received command.

4.4 GPS Commands

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

4.4.1 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.
For Tracker systems only.

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

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). For Tracker systems only.

4.4.3 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. For Tracker systems only.

4.4.4 GPS_GetSatellites

OpCode	0x0509
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Byte
Description	Get the number of connected satellites

Remark: For Tracker systems only.

4.4.5 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. For Tracker systems only.

4.4.6 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. For Tracker systems only.

4.4.7 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 the GPS unit is connected and communicating or 0 if not. For Tracker systems only.

These 'Targets' commands relevant also for manual systems by insert the system's position and heading.

4.4.8 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.

4.4.9 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.

4.4.10 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 on the selected target.

Remark: the target number is set in the Axis byte of the packet. For non-tracker systems, to activate this function the 'Override GPS' flag should be set ([4.4.22](#)) and the system position ([4.4.17](#)) and heading ([4.4.19](#)) should be inserted.

4.4.11 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.

4.4.12 GPS_StartTargetLooping

OpCode	0x0516
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start looping over the all saved targets.

Remark: You can add a delay between the targets by setting the 'Looping delay' ([4.9.9](#))

4.4.13 GPS_StopTargetLooping

OpCode	0x0517
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Stop looping over the all saved targets.

4.4.14 GPS_StartTargetTracking

OpCode	0x0519
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Lock on the selected target.

Remark: the target number is set in the Axis byte of the packet. For Tracker systems only.

4.4.15 GPS_StopTargetTracking

OpCode	0x0518
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Release the selected target.

Remark: the target number is set in the Axis byte of the packet. For Tracker systems only.

4.4.16 GPS_GetStartupStatus

OpCode	0x0522
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Byte
Description	Get the GPS estimated startup completion percentage.

Remark: For Tracker systems only.

4.4.17 GPS_SetTrackFreq

OpCode	0x0529
Data send format	Unsigned Integer (8-bit)
Data send unit	Hz
Return format	None
Data return unit	none
Description	Set the target tracking frequency

Remark: For Tracker systems only.

4.4.18 GPS_GetTrackFreq

OpCode	0x052A
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Hz
Description	Get the target tracking frequency

Remark: For Tracker systems only.

4.4.19 GPS_SetInsertedLLA

OpCode	0x052C
Data send format	None
Data send unit	TargetLLA format (See section 5.1.3)
Return format	None
Data return unit	None
Description	Set the system position (LLA format)

4.4.20 GPS_GetInsertedLLA

OpCode	0x052D
Data send format	None
Data send unit	None
Return format	TargetLLA format (See section 5.1.3)
Data return unit	None
Description	Get the system position which was inserted by the user (LLA format)

4.4.21 GPS_SetInsertedHeading

OpCode	0x052E
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the system's Heading

4.4.22 GPS_GetInsertedHeading

OpCode	0x052F
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the system Heading which was inserted by the user

4.4.23 GPS_SaveInsertedPos

OpCode	0x0530
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Save the system position, heading and 'override GPS' flag

4.4.24 GPS_SetOvdPosGPS

OpCode	0x0531
Data send format	Single byte (value of 0 or 1)
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Set the 'Override GPS position' flag

Remark: When the value is set to 1 the GPS position will override by system position which inserted by the user, when set to 0 the inserted position values will not affect. This Flag is 0 by default.

4.4.25 GPS_GetOvdPosGPS

OpCode	0x0532
Data send format	None
Data send unit	None
Return format	Single byte (value of 0 or 1)
Data return unit	Boolean
Description	Get the 'Override GPS position' flag

Remark: When the value is set to 1 the GPS position will override by system position which inserted by the user, when set to 0 the inserted position values will not affect. This Flag is 0 by default

4.4.26 GPS_SetGpsIP

OpCode	0x0535
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 GPS unit.

Remark: The data in this message is the new IP address for the GPS in 4 bytes format. Note the controller IP address. The new IP address will be activated after reboot.

4.4.27 GPS_GetGpsIP

OpCode	0x0536
Data send format	None
Data send unit	None
Return format	4 byte hex value
Data return unit	IP Address
Description	Get the current IP address of the GPS.

Remark: The IP address will return as a set of 4 bytes.

4.4.28 GPS_EnGpsRead

OpCode	0x0537
Data send format	Unsigned integer (8-bit)
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Enable/disable GPS reading via TCP

Remark: Send 0 to disable GPS reading (to close the socket between the controller and GPS), send 1 to enable GPS reading (to open the socket between the controller and GPS)

4.4.29 GPS_GetEnGpsRead

OpCode	0x0538
Data send format	None
Data send unit	None
Return format	Unsigned integer (8-bit)
Data return unit	Boolean
Description	Get the flag that indicate if the GPS reading is enabled/disabled

Remark: Return value is 1 if the reading is enabled, and 0 if not.

4.5 IMU Commands

These commands retrieve data from the connected IMU unit.

In all of IMU get data commands, the axis byte in the packet should be 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.

4.5.5 IMU_GetGyroX

OpCode	0x0605
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/sec
Description	Get the current gyroscope X axis reading.

4.5.6 IMU_GetGyroY

OpCode	0x0606
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/sec
Description	Get the current gyroscope Y axis reading.

4.5.7 IMU_GetGyroZ

OpCode	0x0607
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/sec
Description	Get the current gyroscope Z axis reading.

4.5.8 IMU_ResetSensor

OpCode	0x060E
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Resets the IMU sensor.

4.5.9 IMU_GetTemperatue

OpCode	0x0637
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Celsius
Description	Get the current temperature reading.

4.5.10 IMU_GetPressure

OpCode	0x0638
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Kilopascal (kPa)
Description	Get the current pressure measurement.

4.5.11 IMU_Leveling

OpCode	0x016C
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Activate Leveling according to IMU angles.

Remark: This operation is valid only for system with IMU on the load. At 'axis' byte send the number of axis to execute this operation. The axis will arrive to 0 (or to an angle which setting-[IMU_setLevelingValue](#)) of IMU angle.

4.5.12 IMU_setLevelingValue

OpCode	0x0C6B
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the value for leveling according to IMU angles.

Remark: At 'axis' byte send the number of axis that the send value is related to.

4.5.13 IMU_getLevelingValue

OpCode	0x0C6C
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the value for leveling according to IMU angles.

Remark: The number of axis that the value is related to will be send at 'axis' byte.

4.5.14 IMU_setIMUBaseOffset

OpCode	0x0C7A
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set an offset for IMU reading- Installation at the base

Remark: This command will change the original reading from the IMU (the offset will be subtracted).
For a sensor in the system base. Axis definition is required (1- yaw, 2- pitch, 3 – roll).

4.5.15 IMU_getIMUBaseOffset

OpCode	0x0C7B
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the offset for IMU reading- Installation at the base

Remark: For a sensor in the system base. Axis definition is required (1- yaw, 2- pitch, 3 – roll).

4.5.16 IMU_setIMUBodyOffset

OpCode	0x0C78
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set an offset for IMU reading- Installation at the body

Remark: This command will change the original reading from the IMU (the offset will be subtracted).
For a sensor in the system body. Axis definition is required (1- yaw, 2- pitch, 3 – roll).

4.5.17 IMU_getIMUBodyOffset

OpCode	0x0C79
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the offset for IMU reading- Installation at the body

Remark: For a sensor in the system body. Axis definition is required (1- yaw, 2- pitch, 3 – roll).

4.5.18 IMU_setIMULoadOffset

OpCode	0x0C6D
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set an offset for IMU reading- Installation at the load

Remark: This command will change the original reading from the IMU (the offset will be subtracted).
For a sensor in the system load. Axis definition is required (1- yaw, 2- pitch, 3 – roll).

4.5.19 IMU_getIMULoadOffset

OpCode	0x0C6E
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the offset for IMU reading- Installation at the load

Remark: For a sensor in the system load. Axis definition is required (1- yaw, 2- pitch, 3 – roll).

4.5.20 IMU_setGyroFilterGain

OpCode	0x0640
Data send format	Floating point 32-bit
Data send unit	None
Return format	None
Data return unit	None
Description	Set the gain for gyro filter

Remark: Axis definition is required (1 - base, 2 - load, 3 - middle) based on the system's configuration.

When the gain value is closer to 0 the gyro will be smoother, when the gain is closer to 1 the gyro after the filter will be closer to the original data.

4.5.21 IMU_getGyroFilterGain

OpCode	0x0641
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	None
Description	Get the gain for gyro filter

Remark: Axis definition is required (1 - base, 2 - load, 3 - middle) based on the system's configuration.

When the gain value is closer to 0 the gyro will be smoother, when the gain is closer to 1 the gyro after the filter will be closer to the original data.

4.5.22 IMU_getPanPositionByGyro

OpCode	0x064A
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the pan angle calculated by gyro

Remark: Axis definition is required (0 – client, 1 - base, 2 - load, 3 - middle) based on the system's configuration.

4.5.23 IMU_getTiltPositionByGyro

OpCode	0x064B
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the tilt angle calculated by gyro

Remark: Axis definition is required (0 – client, 1 - base, 2 - load, 3 - middle) based on the system's configuration.

4.5.24 IMU_getRollPositionByGyro

OpCode	0x064C
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the roll angle calculated by gyro

Remark: Axis definition is required (0 – client, 1 - base, 2 - load, 3 - middle) based on the system's configuration.

4.5.25 IMU_getPanSpeedByGyro

OpCode	0x064D
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/sec
Description	Get the pan speed calculated by gyro

Remark: Axis definition is required (0 – client, 1 - base, 2 - load, 3 - middle) based on the system's configuration.

4.5.26 IMU_getTiltSpeedByGyro

OpCode	0x064E
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/sec
Description	Get the tilt speed calculated by gyro

Remark: Axis definition is required (0 – client, 1 - base, 2 - load, 3 - middle) based on the system's configuration.

4.5.27 IMU_getRollSpeedByGyro

OpCode	0x064F
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/sec
Description	Get the roll speed calculated by gyro

Remark: Axis definition is required (0 – client, 1 - base, 2 - load, 3 - middle) based on the system's configuration.

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_IsConnected

OpCode	0x0704
Data send format	None
Data send unit	None
Return format	UInt-8
Data return unit	Boolean
Description	Get the indication if the controller is ready and the initialization is completed

Remark: This flag indicates only if the initialization was completed and doesn't indicate if the motors or external sensor are ready.

4.6.5 COM_StartKeepAlive

OpCode	0x0705
Data send format	UInt-8
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Start/ Stop the Keep alive mechanism

Remark: send 1 to start the Keep alive mechanism, or 0 to stop it. You should activate this mechanism after each system's restarting.

Keep alive mechanism detects communication failure. When the timer reaches its timeout ([COM_getKeepAliveTimeout](#)) without receiving a new command, the counter raises. When the counter reaches its limit ([COM_getKeepAliveCount](#)) the controller will immediately stop any motion and turn off the axes.

4.6.6 COM_IsKeepAliveOn

OpCode	0x0706
Data send format	None
Data send unit	None
Return format	UInt-8
Data return unit	Boolean
Description	Ask the controller if the Keep alive mechanism is activate

Remark: Return value is 1 if the keep alive is activate, and 0 if not.

4.6.7 COM_setKeepAliveTimeout

OpCode	0x0708
Data send format	Uint-16
Data send unit	millisecond
Return format	None
Data return unit	None
Description	Set the keep alive timeout in millisecond

4.6.8 COM_getKeepAliveTimeout

OpCode	0x0709
Data send format	None
Data send unit	None
Return format	Uint-16
Data return unit	millisecond
Description	Get the keep alive timeout in millisecond

4.6.9 COM_setKeepAliveCount

OpCode	0x071C
Data send format	Uint-8
Data send unit	Number
Return format	None
Data return unit	None
Description	Set the maximum number of keep alive cycles allowed before failure

4.6.10 COM_getKeepAliveCount

OpCode	0x071D
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Number
Description	Get the maximum number of keep alive cycles allowed before failure

4.6.11 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.

4.6.12 COM_SysState

OpCode	0x0713
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	System state vector 8 bit
Description	Get the system state register to be informed which operation is active

Bit	Description	Notes
0	Stabilization	0 – Not active 1 – Active
1	Tracking	0 – Not active 1 – Active
2	Scanning	0 – Not active 1 – Active
3	Presets Looping	0 – Not active 1 – Active
4	Targets Looping	0 – Not active 1 – Active

4.6.13 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.

4.6.14 COM_GetSn

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

4.6.15 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.

4.6.16 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.

Note the IP addresses of external devices connected to the system via Ethernet communication.

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	0x071B
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

Stabilization commands are enabled in supported systems only.
The system has been calibrated and the saved configuration provides the best performance.
Changing this configuration may damage system performance, it is the user's responsibility.

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.

Remark: In order to perform movement while the stabilization mechanism is on you should the standard motion commands that described in chapter 4.2.

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_GetStabError

OpCode	0x0822
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree
Description	Get the stabilization position error.

Remark: Axis definition is required.

4.8.4 STB_SetStabMinAccel

OpCode	0x0839
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec ²
Return format	None
Data return unit	None
Description	Set a minimum value for calculated stabilization acceleration.

Remark: Axis definition is required.

4.8.5 STB_GetStabMinAccel

OpCode	0x083A
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/Sec ²
Description	Get a minimum value for calculated stabilization acceleration.

Remark: Axis definition is required.

4.8.6 STB_SetSpdKp

OpCode	0x0D04
Data send format	Floating point 32-bit
Data send unit	Factor
Return format	None
Data return unit	None
Description	Set the gain factor Kp of the PID speed controller.

Remark: Axis definition is required.

4.8.7 STB_GetSpdKp

OpCode	0x0D08
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Factor
Description	Get the gain factor Kp of the PID speed controller.

Remark: Axis definition is required.

4.8.8 STB_SetSpdKi

OpCode	0x0D05
Data send format	Floating point 32-bit
Data send unit	Factor
Return format	None
Data return unit	None
Description	Set the integral gain Ki of the PID speed controller.

Remark: Axis definition is required.

4.8.9 STB_GetSpdKi

OpCode	0x0D09
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Factor
Description	Get the integral gain Ki of the PID speed controller.

Remark: Axis definition is required.

4.8.10 STB_SetSpdKd

OpCode	0x0D06
Data send format	Floating point 32-bit
Data send unit	Factor
Return format	None
Data return unit	None
Description	Set the derivative gain Kd of the PID speed controller.

Remark: Axis definition is required.

4.8.11 STB_GetSpdKd

OpCode	0x0D0A
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Factor
Description	Get the derivative gain Kd of the PID speed controller.

Remark: Axis definition is required.

4.8.12 STB_SetPosKp

OpCode	0x0D16
Data send format	Floating point 32-bit
Data send unit	Factor
Return format	None
Data return unit	None
Description	Set the gain factor Kp of the PID position controller.

Remark: Axis definition is required.

4.8.13 STB_GetPosKp

OpCode	0x0D17
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Factor
Description	Get the gain factor Kp of the PID position controller.

Remark: Axis definition is required.

4.8.14 STB_SetPosKi

OpCode	0x0D18
Data send format	Floating point 32-bit
Data send unit	Factor
Return format	None
Data return unit	None
Description	Set the integral gain Ki of the PID position controller.

Remark: Axis definition is required.

4.8.15 STB_GetPosKi

OpCode	0x0D19
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Factor
Description	Get the integral gain Ki of the PID position controller.

Remark: Axis definition is required.

4.8.16 STB_SetPosKd

OpCode	0x0D1A
Data send format	Floating point 32-bit
Data send unit	Factor
Return format	None
Data return unit	None
Description	Set the derivative gain Kd of the PID position controller.

Remark: Axis definition is required.

4.8.17 STB_GetPosKd

OpCode	0x0D1B
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Factor
Description	Set the derivative gain Kd of the PID position controller.

Remark: Axis definition is required.

4.8.18 STB_SetDistanceToTarget

OpCode	0x082B
Data send format	Floating point 32-bit
Data send unit	Distance (Meter)
Return format	None
Data return unit	None
Description	Set the distance to target for calculating roll compensation.

4.8.19 STB_GetDistanceToTarget

OpCode	0x082A
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Distance (Meter)
Description	Get the distance to target.

4.8.20 STB_SetCentralAxes

OpCode	0x0825
Data send format	Floating point 32-bit
Data send unit	Distance (Meter)
Return format	None
Data return unit	None
Description	Set central axes distance

Remark: The distance measure between sensor axes to the center of the motion.
Axis definition is required (0- x axis, 1- y axis, 2 – z axis).

4.8.21 STB_GetCentralAxes

OpCode	0x0826
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Distance (Meter)
Description	Get central axes distance

Remark: Get the distance measure between sensor axes to the center of the motion.
Axis definition is required (0- x axis, 1- y axis, 2 – z axis).

4.8.22 STB_SetAxisRadiusCompensation

OpCode	0x0854
Data send format	Unsigned Integer (8 bit).
Data send unit	Boolean 1 or 0
Return format	None
Data return unit	None
Description	Set enable/disable Tilt/Roll Compensation.

Remark: set 1 to enable and 0 to disable a compensation for Tilt/Roll error with Pan and Tilt axes.
Axis definition is required (2 - Tilt Correction, 3 – Roll Correction).
Required to define first distance to target ([SetDistanceToTarget](#)) and central axes ([SetCentralAxes](#)).

4.8.23 STB_GetAxisRadiusCompensation

OpCode	0x0855
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	Boolean 1 or 0
Description	Get the status of Tilt/Roll Compensation.

Remark: 1 if the Roll Compensation is active, 0 if the Roll Compensation is inactive.

Axis definition is required (2 - Tilt Correction, 3 – Roll Correction).

4.8.24 STB_SetRebalanceOn

OpCode	0x084F
Data send format	Unsigned Integer (8 bit).
Data send unit	Boolean 1 or 0
Return format	None
Data return unit	None
Description	Set enable/disable Rebalance function.

Remark: Set 1 to enable and 0 to disable a Rebalance function.

The Rebalance function identifies when the system is unbalanced and acts to calm it down and restore balance.

The Rebalance function does not resolve the root causes of the imbalance. It only calms the system temporarily. If the Kp, Ki, or Kd values are too high and result in instability, it is advisable to adjust them.

Axis definition is required.

4.8.25 STB_GetRebalanceOn

OpCode	0x0850
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	Boolean 1 or 0
Description	Get the status if Rebalance function is enabled/disable.

Remark: 1 if the Rebalance function is enabled, 0 if the Rebalance function is disable.

4.8.26 STB_GetRebalanceStatus

OpCode	0x0851
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	Boolean 1 or 0
Description	Get the status of Rebalance (while the function is enabled).

Remark: While the Rebalance function is enabled, the status will indicate whether the axis is in normal mode or deceleration mode. 1 – deceleration, 0- normal.

4.8.27 STB_SetFeedforwardGain

OpCode	0x0852
Data send format	Floating point 32-bit
Data send unit	Percent
Return format	None
Data return unit	None
Description	Set percentages for Feedforward Control.

Remark: Feedforward Control - Sending a speed value to the motor in addition to the PID controller output. A value of 100% will result in a quick response, while a value of 0% will result in a slower response.

Axis definition is required.

4.8.28 STB_GetFeedforwardGain

OpCode	0x0853
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Percent
Description	Get percentages of Feedforward Control.

4.8.29 STB_SaveStabilizationCfg

OpCode	0x0D07
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Saving the current stabilization settings to the controller memory.

4.8.30 STB_SetAxisStabEnable

OpCode	0x0857
Data send format	Unsigned Integer (8 bit).
Data send unit	Boolean 1 or 0
Return format	None
Data return unit	None
Description	enable/disable stabilization for each axis individually

Remark: Set 1 to enable and 0 to disable stabilization for each axis. Axis definition is required.

4.8.31 STB_GetAxisStabEnable

OpCode	0x0858
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	Boolean 1 or 0
Description	get whether stabilization for axis is enabled

Remark: Axis definition is required.

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_StartPresetLooping

OpCode	0x0D11
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start looping over the all saved presets

Remark: You can add a delay between the presets by setting the 'Looping delay' ([4.9.9](#))

4.9.6 PRST_StopPresetLooping

OpCode	0x0D12
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Stop looping over the all saved presets

4.9.7 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.

4.9.8 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.

4.9.9 PRST_SetLoopingDelay

OpCode	0x0D1C
Data send format	float
Data send unit	millisecond
Return format	None
Data return unit	None
Description	Set the delay time between points in looping mode

Remark: The value should be in multiples of 80 millisecond. This setting is also for [GPS StartTargetLooping](#).

4.9.10 PRST_GetLoopingDelay

OpCode	0x0D1D
Data send format	None
Data send unit	None
Return format	float
Data return unit	millisecond
Description	Get the delay time between points in looping mode

Remark: This setting is also for [GPS StartTargetLooping](#).

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 register. 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.
15	CAN Bus error	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	Driver communication Error	0 – No error 1 – Error. No communication with one of the driver.
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.
7	Protocol error	0 – No Error. 1 – Error.
8	Body IMU error	0 – No Error. 1 – Error.
9	Absolute encoder CRC error	0 – No Error. 1 – Error.
10	Absolute encoder communication error	0 – No Error. 1 – Error.
11	Homing is not completed	0 – No Error- Homing is not active now. 1 – Error- Homing is active now.

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_GetDriverErrorString

OpCode	0x0E03
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the driver communication 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 ERR_GetAbsEncComErrorString

OpCode	0x0E0D
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the absolute encoder communication error description

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

4.10.18 ERR_GetAbsEncCRCErrorString

OpCode	0x0E0E
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the absolute encoder CRC error description

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

4.10.19 ERR_GetBodyImuErrorString

OpCode	0x0E0F
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the body IMU error description

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

4.10.20 ERR_GetHomingErrorString

OpCode	0x0E10
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the Homing not completed error description

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

4.10.21 ERR_OperationRegister

OpCode	0x0E12
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit).
Data return unit	16 bit vector
Description	Operation Register is a 16-bit status register. It aggregates data regarding the status of several motion operations.

Remark: An error will occur after several unsuccessful attempts to execute the specific operation and the axis will stop trying to execute it. This register is for indication purposes only and does not affect the system's operation. Axis destination is required.

Bit	Description	Notes
0	Position Error	0 – No error 1 – Error, a deviation between the current angle and the desired target angle of the axis (in absolute mode only). The deviation value can be read at MOT_GetPositionErr .
1	Homing	0 – completed. 1 – not completed.
2	Levelling	0 – completed. 1 – not completed.
3	Synchronizing (dual gimbals only)	0 – completed. 1 – not completed.

4.10.22 Error messages description:

OpCode	Description	Error message
0x0E01	Get CSR register	-
0x0E02	Clear errors	-
0x0E03	Get driver communication error string	- No communication with axes (bitmap): 0xXX
0x0E04	Get system error string	- General system error - Fail to access memory
0x0E05	Get load IMU error string	- No communication with IMU - Fail to sync with IMU
0x0E06	Get base IMU error string	- No communication with IMU - Fail to sync with 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
0x0E0A	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 - Invalid Axis 0 in motor command - 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 - Invalid input
0x0E0D	Get absolute encoder communication error string	- No communication with absolute encoder
0x0E0E	Get absolute encoder CRC error string	- Absolute encoder CRC Fail
0x0E0F	Get body IMU error string	- No communication with IMU - Fail to sync with IMU
0x0E10	Get Homing not completed error string	- Pan Homing is not completed - Tilt Homing is not completed - Pan Homing is not completed - Roll Homing is not completed - Four axis Homing is not completed - Five axis Homing is not completed - Six axis Homing is not completed - Pan & Tilt Homing are not completed - Pan & Roll Homing are not completed - Roll & Tilt Homing are not completed - Pan & Tilt & Roll Homing are not completed

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

4.11.23 VDT_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

4.11.24 VDT_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

4.11.25 VDT_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.

4.12 Configuration commands

4.12.1 CFG_setStartupReg

OpCode	0x0C51
Data send format	Unsigned Integer (8 bit).
Data send unit	8 bit vector
Return format	None
Data return unit	None
Description	Set the register according to the description for activate operations at start up

Remark: This command require specifying the axis destination.

Bit	Description	Operation	Value	Notes
0	Motor Homing mode	MOT Homing	0 – Not activate 1 – Activate	Homing function must be stored on motor memory
1	Stabilization mode	STB StabilizationOn	0 – Not activate 1 – Activate	The system must be stabilized or tracker
2	IMU Homing mode	IMU HomingIMU	0 – Not activate 1 – Activate	The system must have IMU on the load

4.12.2 CFG_getStartupReg

OpCode	0x0C52
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	8 bit vector
Description	Get the activated operations at start up register.

Remark: This command require specifying the axis destination.

4.12.3 CFG_SetGeneralSpd

OpCode	0x0D1E
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec
Return format	None
Data return unit	None
Description	Set the speed of each axis for general operations.

*Remark: Axis definitions is required. The value can't be negative.
(The speed of presets commands, Targets commands etc.).*

4.12.4 CFG_GetGeneralSpd

OpCode	0x0D1F
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/Sec
Description	Get the speed of each axis for general operations.

Remark: Axis definitions is required (The speed of presets commands, Targets commands etc.)

4.12.5 CFG_setPosRange

OpCode	0x0C70
Data send format	Unsigned Integer (8 bit).
Data send unit	Range type
Return format	None
Data return unit	None
Description	Set the angle range for load position (MOT_GetLoadPosition)

Remark: Axis definitions is required.

The angle range types are:

- 0x00 : $-\infty$ to ∞
- 0x01 : -360° to 360°
- 0x02 : -180° to 180°
- 0x03 : 0° to 360°

4.12.6 CFG_getPosRange

OpCode	0x0C71
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	Range type
Description	Get the angle range for load position (MOT_GetLoadPosition)

Remark: This command require specifying the axis destination.

4.13 Pelco-D settings commands

Pelco-D protocol is enabled in supported systems only via RS485 communication. When Pelco-D protocol is supported, the axes will execute the desired movement commands by the protocol messages. Camera commands will be ignored.

4.13.1 PELCOD_EnPelcoD

OpCode	0x0C80
Data send format	Unsigned Integer (8-bit)
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Enable / disable Pelco-D protocol support.

Remark: When the value is 1 Pelco-D will be supported, when the value is 0 Pelco-D will not be supported.

4.13.2 PELCOD_IsActivePelcoD

OpCode	0x0C81
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Boolean
Description	Get the state of Pelco-D protocol support.

Remark: When the value is 1 Pelco-D is supported, when the value is 0 Pelco-D is not supported.

4.13.3 PELCOD_SetAddress

OpCode	0x0C82
Data send format	Unsigned Integer (8-bit)
Data send unit	Integer Number
Return format	None
Data return unit	None
Description	Set the device address.

4.13.4 PELCOD_GetAddress

OpCode	0x0C83
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Integer Number
Description	Get the device address.

4.13.1 PELCOD_SetAxisNum

OpCode	0x0C84
Data send format	Unsigned Integer (8-bit)
Data send unit	Integer Number
Return format	None
Data return unit	None
Description	Match the Pelco-D protocol axis to the system axis.

Remark: Axis definitions is required- 0x01 for pan, 0x02 for tilt. The value should be an axis number supported by the system.

4.13.2 PELCOD_GetAxisNum

OpCode	0x0C85
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Integer Number
Description	Get the system axis that corresponds to the Pelco-D protocol axis.

Remark: Axis definitions is required- 0x01 for pan, 0x02 for tilt.

4.13.3 PELCOD_SetMinSpd

OpCode	0x0C86
Data send format	Floating point 32-bit
Data send unit	Degree/Second
Return format	None
Data return unit	None
Description	Set the minimum value of the speed range.

Remark: Axis definitions is required.

4.13.4 PELCOD_GetMinSpd

OpCode	0x0C87
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree/Second
Description	Get the minimum value of the speed range.

Remark: Axis definitions is required.

4.13.5 PELCOD_SetMaxSpd

OpCode	0x0C88
Data send format	Floating point 32-bit
Data send unit	Degree/Second
Return format	None
Data return unit	None
Description	Set the maximum value of the speed range.

Remark: Axis definitions is required.

4.13.6 PELCOD_GetMaxSpd

OpCode	0x0C89
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree/Second
Description	Get the maximum value of the speed range.

Remark: Axis definitions is required.

4.13.7 PELCOD_SetTurboSpd

OpCode	0x0C8A
Data send format	Floating point 32-bit
Data send unit	Degree/Second
Return format	None
Data return unit	None
Description	Set the value for the turbo speed.

Remark: For Pan Axis only.

4.13.8 PELCOD_GetTurboSpd

OpCode	0x0C8B
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree/Second
Description	Get the value of the turbo speed.

Remark: For Pan Axis only.

4.13.9 PELCOD_SetAccel

OpCode	0x0C8C
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec^2
Return format	None
Data return unit	None
Description	Set the value of the acceleration.

Remark: Axis definitions is required.

4.13.10 PELCOD_GetAccel

OpCode	0x0C8D
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/Sec^2
Description	Get the value of the acceleration.

Remark: Axis definitions is required.

4.14 Relays commands

Relays commands are enabled in supported systems only.

4.14.1 RLY_ActivateModule

OpCode	0x0D24
Data send format	Uint-8
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Activate the relays module

Remark: When the value is 1 the module will be activated, when the value is 0 the module will be deactivated. The module must be activated in order to set the relays.

4.14.2 RLY_IsActivateModule

OpCode	0x0D25
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Boolean
Description	Get the state of the relays module

Remark: When the value is 1 the module is activated, when the value is 0 the module is not activated.

4.14.3 RLY_SetRelay

OpCode	0x0D21
Data send format	Uint-8
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Turn on the selected relay

Remark: The relay number is set in the Axis byte of the packet. When the value is 1 the relay will be turn on, when the value is 0 the relay will be turn off.

4.14.4 RLY_GetRelay

OpCode	0x0D22
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	8 bit vector
Description	Get the relays state

Remark: Returns an 8 bit vector that indicates which relay is activated. Bit 0 indicates relay 1 state, bit 1 indicates relay 2 state and so on.

4.14.5 RLY_SetRelayPeriod

OpCode	0x0D23
Data send format	Unsigned Integer (16 bit).
Data send unit	Second
Return format	None
Data return unit	None
Description	Turn on the selected relay for the specified period

Remark: The relay number is set in the Axis byte of the packet. The Data should contain the period (seconds) for activating the relay. The relay will be turn off immediately after the time period will end.

4.14.6 RLY_SetRelayJogging

OpCode	0x0D2C
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Turn on the selected relay for 0.5 second only

Remark: The relay number is set in the Axis byte of the packet. The relay will be turn off immediately after 0.5 second.

4.14.7 RLY_GetModuleErr

OpCode	0x0D27
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Boolean
Description	Get an indication of communication error with the relay module

Remark: When the value is 0 there is no error communication, when the value is 1 there is no communication with the relays module.

4.14.8 RLY_RstComModule

OpCode	0x0D28
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reset the communication with the relays module

4.14.9 RLY_SetModuleIP

OpCode	0x0D29
Data send format	4-byte hex value
Data send unit	IP Address
Return format	None
Data return unit	None
Description	Set the IP address of the relays module.

Remark: The data in this message is the new IP address for the relays module 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.

4.14.10RLY_GetModuleIP

OpCode	0x0D2A
Data send format	None
Data send unit	None
Return format	4-byte hex value
Data return unit	IP Address
Description	Get the current IP address relays module.

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.

4.14.11RLY_SetNumRelays

OpCode	0x0D2E
Data send format	Unsigned Integer (8 bit).
Data send unit	Number of relays
Return format	None
Data return unit	None
Description	Set the number of the relays on the module

Remark: The maximum supported number of relays is 8.

4.14.12RLY_GetNumRelays

OpCode	0x0D2D
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit).
Data return unit	Number of relays
Description	Get the number of the relays on the module

4.15 Dual Gimbals commands

Dual gimbals commands are enabled in supported systems only. The previous commands has the same format when the available axes are: Pan (0x01), Tilt (0x02), Pan 2 (0x04) and Tilt 2 (0x05).

4.15.1 DG_SetSyncMode

OpCode	0x0FA0
Data send format	UInt-8
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Activate / deactivate the synchronization mode

Remark: When the value is 1 the synchronization operation will start and the sync mode will be activated. From now the both gimbals will be as one. Only Pan and Tilt axes will be available, any command for Pan 2 and Tilt 2 axes will be ignored.

When the value is 0 the sync mode will be disabled, the both gimbals will be operated separately.

When sync mode is active the two pairs of axes (Pan and Pan 2, Tilt and Tilt 2) will be parallel. For systems with 'Boresight' mechanism enabled, the all axes will overcome the parallelism and will point on the same target in a defined range ([LRF SetRange](#)).

When 'Sync mode' is active and one of the axes is 'Axis off', the mode will switch to 'Outer mode'.

4.15.2 DG_SetInnerMode

OpCode	0x0FA1
Data send format	UInt-8
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Activate / deactivate the inner mode

Remark: This mode is NOT relevant to sync mode. When the value is 1 the inner mode will be activated.

Gimbal 2 (inner) will keep its azimuth and will not be affected by Azimuth changes of Gimbal 1 (outer).

When the value is 0 the outer mode will be activated. The Azimuth of gimbal 2 (inner) will be affected by Azimuth changes of gimbal 1 (outer).

4.15.3 DG_IsSyncMode

OpCode	0x0FA2
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Boolean
Description	Get the state of sync mode

Remark: When the value is 1 the sync mode is active (all axes are synchronized), when the value is 0 the sync mode is disabled.

4.15.4 DG_IsInnerMode

OpCode	0x0FA3
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Boolean
Description	Get the state of inner mode

Remark: When the value is 1 the inner mode is active, when the value is 0 the outer mode is active.

4.15.5 DG_GetPosDiff

OpCode	0x0FA5
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree
Description	Get the angle difference between the axes

Remark: Axis definition is required. Axis 1 for angle difference between Pan axes of Gimbal 1 and Gimbal 2, axis 2 for angle difference between Tilt axes.

4.15.6 DG_SetMaxDiff

OpCode	0x0FA6
Data send format	Floating point 32-bit
Data send unit	Degree
Return format	None
Data return unit	None
Description	Set the maximum allowed difference between axes in 'Sync' mode

Remark: Axis definition is required, axis 1 for Pan axes and axis 2 for Tilt axes. The minimum (and the default) value is 0.0026 degree.

4.15.7 DG_GetMaxDiff

OpCode	0x0FA7
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree
Description	Get the maximum allowed difference between axes in 'Sync' mode

Remark: Axis definition is required, axis 1 for Pan axes, axis 2 for Tilt axes.

4.15.8 DG_IsAxesSynced

OpCode	0x0FA8
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Boolean
Description	Get the indication if the axes are synchronized or not

Remark: When the value is 1 the axes are synchronized- the Pan axes and the Tilt axes, have a position difference smaller than the defined maximum value. When the value is 0 the axes are NOT synchronized- at least one pair of axes has a position difference greater than the allowed maximum value.

4.15.9 DG_GoToCamera

OpCode	0x0FA9
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start 'Go to Camera' operation

Remark: This operation sends Gimbal 1 to the current position of Gimbal 2.

4.15.10 DG_SetSyncSpd

OpCode	0x0FAA
Data send format	Floating point 32-bit
Data send unit	Degrees/Sec
Return format	None
Data return unit	None
Description	Set the speed of the synchronization operation

Remark: Set the speed of the synchronization operation. Axis definition is required (Gimbal 1 axes only).

4.15.11DG_SetMainGimbal

OpCode	0x0FAB
Data send format	Unsigned Integer (8-bit)
Data send unit	Gimbal Number
Return format	None
Data return unit	None
Description	Select the main Gimbal

Remark: select the main Gimbal to activate the various motion operation (such as Scanning, Targets etc.). When the value is 1 Gimbal 1 (Yaw and Pitch axes) will be the main, when the value is 2 Gimbal 2 (Yaw 2 and Pitch 2) will be the main. When Sync mode is active, Gimbal 1 is the main.

4.15.12DG_GetMainGimbal

OpCode	0x0FAC
Data send format	Unsigned Integer (8-bit)
Data send unit	Gimbal Number
Return format	None
Data return unit	None
Description	Get the number of the main Gimbal

Remark: When the value is 1 Gimbal 1 (Yaw and Pitch axes) is the main, when the value is 2 Gimbal 2 (Yaw 2 and Pitch 2) is the main.

4.15.13DG_CTC

OpCode	0x0FB0
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Center to Center, Calculate the offset of pan 2 & tilt 2 when the center of gimbal 1 points to the same position of the center of gimbal 2

Remark: This command changes the origin positions of the Gimbal 2 axes to compensate for the parallelism between the axes. First, all axes need to point at the same target and then set the 'CTC' command. The current offsets correspond to the current range to the target (if 'boresight' mechanism is enabled, the range to target should be defined [LRF SetRange](#)).

4.15.14DG_GetCTCOffset

OpCode	0x0FB1
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the offset position of the Gimbal 2 axes when all axes point to the same target.

Remark: 'CTC offset' is the offset of the load position reading of each axis, calculated by 'center to center' procedure. Axis definition is required (Gimbal 2 axes only).

4.15.15DG_ResetCTCOffset

OpCode	0x0FB2
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reset the 'CTC' offset position of the Gimbal 2 axes

Remark: reset the stored 'Center to center' offset of gimbal 2's axes.

4.15.16DG_GetSyncSpd

OpCode	0x0FB5
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees/Sec
Description	Get the speed of the synchronization operation

Remark: Get the speed of the synchronization operation. Axis definition is required (Gimbal 1 axes only).

4.15.17DG_SetActiveWeapon

OpCode	0x0FB6
Data send format	Unsigned Integer (8-bit)
Data send unit	Weapon Number
Return format	None
Data return unit	None
Description	Set the active Weapon to select its dimensions in 'Boresight' mechanism.

Remark: 1 for weapon 1, 2 for weapon 2, up to 2 weapons. Set the active weapon to select its dimensions in 'Boresight' mechanism.

4.15.18DG_GetActiveWeapon

OpCode	0x0FB7
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Weapon Number
Description	Get the active weapon

4.15.19 DG_SetActiveCamera

OpCode	0x0FBA
Data send format	Unsigned Integer (8-bit)
Data send unit	Camera Number
Return format	None
Data return unit	None
Description	Set the active camera to select its dimensions in 'Boresight' mechanism.

Remark: 1 for camera 1, 2 for camera 2 etc. up to 4 cameras. Set the active camera to select its dimensions in 'Boresight' mechanism.

4.15.20 DG_GetActiveCamera

OpCode	0x0FBB
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Camera Number
Description	Get the active camera

4.15.21 DG_IsBoresightEn

OpCode	0x0FB9
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8-bit)
Data return unit	Boolean
Description	Get the indication if 'Boresight' mechanism is enabled or not

Remark: if the value is 1 'Boresight' mechanism is enabled, if the value is 0 'Boresight' mechanism is disabled. 'Boresight' mechanism calculate the position of gimbal 1 depending of distance to target and mechanical position of gimbal 2 so pointing axis of gimbal 1 intersects with pointing axis of gimbal 2.

4.15.22DG_GetBoresightOffset

OpCode	0x0FBC
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree
Description	Get the offset of each axis which calculated by the boresight mechanism

Remark: 'Boresight offset' is the addition to the 'Go To Camera' position command of each axis, calculated by the boresight mechanism (used in 'Sync mode'). Axis definition is required.

4.15.23DG_SetBallisticOffset

OpCode	0x0FBD
Data send format	Floating point 32-bit
Data send unit	Degree
Return format	None
Data return unit	None
Description	Set a ballistic offset in 'Sync mode'

Remark: 'Ballistic offset' is a client offset. Another addition to the 'Go To Camera' position command of Pan and Tilt axes, due to ballistic calculations (used in 'Sync mode' only). Axis definition is required- only Pan and Tilt.

4.15.24DG_GetBallisticOffset

OpCode	0x0FBE
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degree
Description	Get the stored ballistic offset

Remark: Axis definition is required- only Pan and Tilt.

4.15.25DG_GetSafetyStatus

OpCode	0x0FBF
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	Boolean
Description	Get the safety status reading from 'CapSnap' unit

Remark: When value is 1- the safety is open, when value is 0- the safety is closed.

4.15.26DG_GetLoadStatus

OpCode	0x0FC0
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	Load status
Description	Get the load status reading from 'CapSnap' unit

Remark: The Load status values are:

Value	Description
0	static
1	backward movement
2	forward movement
3	controlled movement
4	static at the back
5	forward movement (boot after turning on the system)

4.15.27DG_GetCapsnapIOStatus

OpCode	0x0FC1
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	Load status
Description	Get the status of each CapSnap I/O

Remark: The IO status values are:

Bit	Description
0	Weapon Type (0 – Mag, 1 – M2)
1	Loading button
2	Safety button
3	NC
4	NC
5	Front loading handle sensor
6	Rear loading handle sensor Mag
7	Rear loading handle sensor M2

4.15.28DG_GetNumBullets

OpCode	0x0FC2
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit)
Data return unit	Number of bullets
Description	Get the number of bullets reading from 'CapSnap' unit

4.15.29DG_ResetNumBullets

OpCode	0x0FC3
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reset the number of bullets reading from 'CapSnap' unit to default value 0.

4.15.30DG_IsCapSnapReady

OpCode	0x0FC4
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	Boolean
Description	Get the status of the communication with CapSnap unit

Remark: When value is 1- the communication with CapSnap unit is active, when value is 0- there is no communication with CapSnap unit.

4.16 Software limit switch

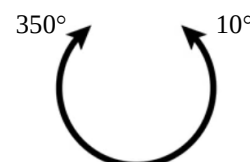
The software limit switch defines the possible range for the axis. The negative limit will be the minimum position the axis can reach, and the positive limit will be the maximum position. The value of the negative limit must be smaller than the value of the positive limit. All software limit switch commands require specifying the axis destination.

The following examples describe identical positions on the circle, but the limit's values will vary according to the required range:

- **A range between 10° to 350°:**

The minimum position is 10 ° - The negative limit value will be 10°.

The maximum position is 350°- The positive limit value will be 350°.



- **A range between -10° to 10°:**

The minimum position is -10 ° - The negative limit value will be -10°.

The maximum position is 10°- The positive limit value will be 10°.



4.16.1 SWLS_SetNegSWLS

OpCode	0x0136
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the negative software limit switch

Remark: The value should be less than the positive software limit switch.

4.16.2 SWLS_SetPosSWLS

OpCode	0x0137
Data send format	Floating point 32-bit
Data send unit	Degrees
Return format	None
Data return unit	None
Description	Set the positive software limit switch

Remark: The value should be greater than the negative software limit switch.

4.16.3 SWLS_ActivateSWLS

OpCode	0x0146
Data send format	Uint-8
Data send unit	Boolean
Return format	None
Data return unit	None
Description	Enable the software limit switch

Remark: When the value is 1 the SWLS will be activated, when the value is 0 the SWLS will be deactivated. If the SWLS is enabled, a motor error will occur when the motors reach the limit. To activate the SWLS, the current motor position should be in a valid range.

4.16.4 SWLS_GetNegSWLS

OpCode	0x010C
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the negative software limit switch

4.16.5 SWLS_GetPosSWLS

OpCode	0x010D
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Degrees
Description	Get the positive software limit switch

4.16.6 SWLS_IsActiveSWLS

OpCode	0x0110
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Boolean
Description	Get the state of software limit switch

Remark: This command will send the state of software limit switch. When the value is 1 the SWLS is activated, when the value is 0 the SWLS is NOT activated.

4.16.7 SWLS_SetHandler

OpCode	0x0176
Data send format	Uint-8
Data send unit	SWLS Handler
Return format	None
Data return unit	None
Description	Set the handler of software limit switch

Remark: The value should be one of the following

SWLS handler	Value	Description
Controller	0	Control over motion commands sent to the axis and preventing limit crossing even at high speeds. For position mode - the final angle will vary depending on the limits. For speed mode - the speed will be reduced when approaching the limits (SWLS SetSlowdownValue).
Driver	1	For safety reasons. Control over the motor current. When the driver detects a limit crossing, it immediately stops the motion (with a rapid deceleration) and blocks any command towards the limit direction.
Both (controller and driver)	2	<i>Combines both methods.</i> <i>Remark: With this option the values of the driver's limits will be with an offset (SWLS SetOffsetValue).</i>

4.16.8 SWLS_GetHandler

OpCode	0x0177
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	SWLS Handler
Description	Get the handler of software limit switch

4.16.9 SWLS_SetOffsetValue

OpCode	0x0178
Data send format	Floating point 32-bit
Data send unit	degree
Return format	None
Data return unit	None
Description	Set an offset for the limits defined in the driver.

Remark: This offset is relevant when both controller and driver are the SWLS handlers. The driver's limits will be the defined limits (posSWLS and negSWLS) with the offset. The value can be positive only. The offset will be available to change only if the software limit switch is not active.

4.16.10 SWLS_GetOffsetValue

OpCode	0x0179
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	degree
Description	Get the offset for the limits defined in the driver.

4.16.11 SWLS_SetSlowdownValue

OpCode	0x017A
Data send format	Floating point 32-bit
Data send unit	degree
Return format	None
Data return unit	None
Description	Set the distance from the limits where the axis will start to slow down

Remark: When the axis crosses the defined distance, the allowed maximum speed will be determined by the current distance from the limit until it reaches 0 and stops at the limit. This mechanism allows stopping at the limit in speed mode. The value can be positive only and if it is 0, the mechanism will not be active. This mechanism NOT available when the SWLS handler is the driver.

4.16.12 SWLS_GetSlowdownValue

OpCode	0x017B
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	degree
Description	Get the distance from the limits where the axis will start to slow down

4.17 LRF Commands

4.17.1 LRF_SetManualDistance

OpCode	0x0300
Data send format	Floating point 32-bit
Data send unit	Meter
Return format	None
Data return unit	None
Description	Set the distance to the target manually

Remark: Set the range to the target. With a dual gimbals system, in sync mode while the 'Boresight' mechanism is active, each range update will recalculate the motors position.

4.17.2 LRF_GetManualDistance

OpCode	0x0301
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Meter
Description	Get the manually distance to the target

4.17.3 LRF_SetMinMeasurement

OpCode	0x0302
Data send format	Uint-16
Data send unit	Meter
Return format	None
Data return unit	None
Description	Set the minimum of measurement range

4.17.4 LRF_GetMinMeasurement

OpCode	0x0303
Data send format	None
Data send unit	None
Return format	Uint-16
Data return unit	Meter
Description	Get the minimum of measurement range

4.17.5 LRF_SetMaxMeasurement

OpCode	0x0304
Data send format	Uint-16
Data send unit	Meter
Return format	None
Data return unit	None
Description	Set the maximum of measurement range

4.17.6 LRF_GetMaxMeasurement

OpCode	0x0305
Data send format	None
Data send unit	None
Return format	Uint-16
Data return unit	Meter
Description	Get the maximum of measurement range

4.17.7 LRF_SetFrequency

OpCode	0x0306
Data send format	Uint-8
Data send unit	Hz
Return format	None
Data return unit	None
Description	Set the measurement frequency

4.17.8 LRF_GetFrequency

OpCode	0x0307
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	Hz
Description	Get the measurement frequency

4.17.9 LRF_StartMeasurement

OpCode	0x0308
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Start measurement

4.17.10 LRF_StopMeasurement

OpCode	0x0309
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Stop measurement

4.17.11 LRF_GetLRFDistance

OpCode	0x030A
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	Meter
Description	Get distance measured from LRF

Remark: In each measurement, the LRF can detect up to 5 different targets. The axis number represents the measurement number. Axis 1 will always show the current distance or the last one measured.

4.17.12 LRF_SetDistanceSource

OpCode	0x030B
Data send format	Uint-8
Data send unit	None
Return format	None
Data return unit	None
Description	Set which distance source to use: the manual distance or the distance measured by the LRF. (0 = Manual 1 = LRF)

4.17.13 LRF_GetDistanceSource

OpCode	0x030C
Data send format	None
Data send unit	None
Return format	Uint-8
Data return unit	None
Description	Get which distance source is currently in use. (0 = Manual 1 = LRF)

4.17.14 LRF_GetTargetDistance

OpCode	0x030D
Data send format	None
Data send unit	None
Return format	Floating point 32-bit
Data return unit	None
Description	Get the final distance to the target. The final distance to the target is determined by the Distance Source, based on LRF measurements or Manual Distance.

4.17.15 LRF_IsConnected

OpCode	0x030E
Data send format	None
Data send unit	None
Return format	UInt-8
Data return unit	None
Description	Get whether the LRF is connected. 0 = Disconnected 1 = Connected

Remark: In each measurement, the LRF can detect up to 5 different targets. The axis number represents the measurement

4.18 Weapon control unit Commands

The following commands execute operations and retrieve statuses from the Weapon Control Unit. No axis definition is required. For supported systems only.

4.18.1 WCU_GetFirmware

OpCode	0x0B04
Data send format	None
Data send unit	None
Return format	ASCII string
Data return unit	None
Description	Get the firmware revision of the weapon control unit

4.18.2 WCU_ComStatus

OpCode	0x0B05
Data send format	None
Data send unit	None
Return format	Unsigned integer (8-bit)
Data return unit	Boolean
Description	Get the status of communication between the controller and the weapon control unit

Remark: When the value is 1 the communication between the controller and the weapon control unit is active, when the value is 0 the communication is inactive.

4.18.3 WCU_OpenSafety

OpCode	0x0B06
Data send format	Unsigned Integer (8 bit)
Data send unit	None
Return format	None
Data return unit	None
Description	Open/close the weapon safety

Remark: send 1 to open the safety and send 0 to close the safety. For M4 only, send 2 to open the safety in auto mode.

4.18.4 WCU_GetSafetyStatus

OpCode	0x0B07
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	None
Description	Get the status of the weapon safety

4.18.5 WCU_StartLoading

OpCode	0x0B08
Data send format	Unsigned Integer (8 bit)
Data send unit	None
Return format	None
Data return unit	None
Description	Start/ stop the weapon loading

Remark: send 1 to start a full loading movement (backward and then forward), send 2 to start a backward loading (a backward movement only), send 3 to start a forward loading (a forward movement only), send 0 to stop the loading handle immediately.

4.18.6 WCU_GetLoadStatus

OpCode	0x0B09
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	None
Description	Get the status of the loading

Remark: when the value is 0 the loading handle stops in the front, when the value is 1 the loading handle is moving backward, when the value is 2 the loading handle is moving forward, when the value is 3 the loading handle position is back and when the value is 4 the loading handle position is unknown.

4.18.7 WCU_SetTrigger

OpCode	0x0B0A
Data send format	Unsigned Integer (8 bit)
Data send unit	None
Return format	None
Data return unit	None
Description	Start/ stop the firing

Remark: Send 1 for firing, send 0 to stop firing. To maintain continuous firing, a firing value (1) should be sent at intervals of less than 100ms.

4.18.8 WCU_GetTriggerStatus

OpCode	0x0B0B
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	None
Description	Get the status of the loading

Remark: when the value is 1 the firing is active, when the value is 0 the firing is inactive.

4.18.9 WCU_GetIOStatus

OpCode	0x0B0C
Data send format	None
Data send unit	None
Return format	Unsigned Integer (8 bit)
Data return unit	None
Description	Get the status of each I/O of the weapon control unit.

Remark: The IO status values are:

Bit	Description
0	Weapon Type (0 – Mag, 1 – M2)
1	Loading button
2	Safety button
3	NC
4	NC
5	Front loading handle sensor
6	Rear loading handle sensor Mag
7	Rear loading handle sensor M2

4.18.10 WCU_GetNumBullets

OpCode	0x0B0D
Data send format	None
Data send unit	None
Return format	Unsigned Integer (16 bit)
Data return unit	Number of bullets
Description	Get the number of bullets that have been fired.

4.18.11 WCU_ResetNumBullets

OpCode	0x0B0E
Data send format	None
Data send unit	None
Return format	None
Data return unit	None
Description	Reset the number of bullets in the cartridge to default value 0.

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 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).

5.1.5 Targets list format

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

T ₁ Width	T ₁ Height	T ₂ Width	T ₂ Height	T ₃ Width	T ₃ Height	T ₄ Width	T ₄ Height	T ₅ Width	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.

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_SetPositionMode
- MOT_SetTum (as needed)
- MOT_SetPositionRelative
- MOT_SetAcceleration
- MOT_SetSpeed
- MOT_SendPosition
- MOT_Update

After sending the following seven 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_SetPositionMode without data*

0x50	0x54	0x04	0x00	0x01	0x01	0x3B	0x41
------	------	------	------	------	------	------	------

Controller answer: *Acknowledge*

0x06

User send: *MOT_SetTum without data*

0x50	0x54	0x04	0x00	0x01	0x01	0x3B	0x
------	------	------	------	------	------	------	----

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.

7.3 Go to target commands

This example will demonstrate the implementation of commands that send the system load to look on a target. This example show the different sequence between tracker system and non-tracker system.

7.3.1 Non Tracker system

To activate this command set 'Override GPS position flag', then set the system position and heading.

User send: *GPS_SetOvdPosGPS with data 0x01*

0x50	0x54	0x05	0x00	0x00	0x05	0x031	0x01	0x3C
------	------	------	------	------	------	-------	------	------

Controller answer: *Acknowledge*

0x06

User send: *GPS_SetInsertedLLA with system LLA (Target packet format)-*

Longitude: 34.8915 = 0x4041721CAC083127

Latitude: 32.1040 = 0x40400D4FDF3B645A

Altitude: 65.43 = 0x4282DC29

0x50	0x54	0x18	0x00	0x00	0x05	0x2C	0x40	0x41	0x72	0x1C	0xAC	0x08	0x31
0x27	0x40	0x40	0x0D	0x4F	0xDF	0x3B	0x64	0x5A	0x42	0x82	0xDC	0x29	0xE1

Controller answer: *Acknowledge*

0x06

User send: *GPS_SetInsertedHeading with system heading*

0x50	0x54	0x08	0x00	0x00	0x05	0x2E	0x42	0x20	0x00	0x00	0x9D
------	------	------	------	------	------	------	------	------	------	------	------

Controller answer: *Acknowledge*

0x06

User send: *GPS_SetTargetLLA with Taregt LLA (Target packet format), target number 3 in axis byte-
Longitude: 34.8921 = 0x404172305532617C
Latitude: 32.1050 = 0x40400D70A3D70A3D
Altitude: 64.97 = 0x4281F0A4*

0x50	0x54	0x18	0x00	0x03	0x05	0x2C	0x40	0x41	0x72	0x30	0x55	0x32	0x61
0x7C	0x40	0x40	0x0D	0x70	0xA3	0xD7	0x0A	0x3D	0x42	0x81	0xF0	0xA4	0xCC

Controller answer: *Acknowledge*

0x06

User send: *GPS_GoToTarget number 3 (message without data)*

0x50	0x54	0x04	0x00	0x03	0x05	0x13	0xBC
------	------	------	------	------	------	------	------

Controller answer: *Acknowledge*

0x06

Remark: This function available in non-tracker system only if 'Override GPS' Flag is 0x01

7.3.2 Tracker system

The system position is received from the GPS unit, for using GPS values reset the 'Override GPS' flag.

User send: *GPS_SetOvdPosGPS with data 0x00*

0x50	0x54	0x05	0x00	0x00	0x05	0x031	0x00	0x3B
------	------	------	------	------	------	-------	------	------

Controller answer: *Acknowledge*

0x06

User send: *GPS_SetTargetLLA with Taregt LLA (Target packet format), target number 3 in axis byte-
Longitude: 34.8921 = 0x404172305532617C
Latitude: 32.1050 = 0x40400D70A3D70A3D
Altitude: 64.97 = 0x4281F0A4*

0x50	0x54	0x18	0x00	0x03	0x05	0x2C	0x40	0x41	0x72	0x30	0x55	0x32	0x61
0x7C	0x40	0x40	0x0D	0x70	0xA3	0xD7	0x0A	0x3D	0x42	0x81	0xF0	0xA4	0xCC

Controller answer: *Acknowledge*

0x06

User send: *GPS_GoToTarget number 3 (message without data)*

0x50	0x54	0x04	0x00	0x03	0x05	0x13	0xBC
------	------	------	------	------	------	------	------

Controller answer: *Acknowledge*

0x06
