PC Command	START	COMMAND	ADRESS		PARAM_LEN	PARAM	XMODEM CRC16	
	Byte	Byte	High-Byte	Low-Byte	Byte	PARAM_LEN x Byte	High-Byte	Low-Byte
	0x2F	0x30 - 0x3F	0x00 - 0xFF	0x00 - 0xFF	1-255 // 0=256	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF

INTERFACE Answer	START	COMMAND	ADRESS		PARAM_LEN	PARAM	ACK	XMODEM CRC16	
	Byte	Byte	High-Byte	Low-Byte	Byte	PARAM_LEN x Byte	Byte	High-Byte	Low-Byte
	0x2E	0x30 - 0x3F	0x00 - 0xFF	0x00 - 0xFF	1-255 // 0=256	0x00 - 0xFF	0x00 - 0x0F	0x00 - 0xFF	0x00 - 0xFF

Field name	Min Value	Max Value	Description
START	0x2E = 46 = '.'	0x2F = 47 = '/'	Escape character: (PC) must send **0x2F** / Interface must send **0x2E** in response
COMMAND	0x30 = 48 = '0'	0x3C = 58 = '<'	All chars are printable to better control with portmonitor
ADRESS	0x0000 = 0	0xFFFF = 65535	Only **Valid if Device or EEprom Read/Write** (Big Endian) adress 0xFFFF will be ignored in non SilC2 modes (for ascending read/write)
PARAM_LEN(n)	0x01 = 1	0x00 = 256	Length-Field for the following PARAM Block. To handle the whole Byte range from 0..256 a trick is used The minimum Value is 1 so there has to be allways 1 Byte in PARAM Values from 1..255 count what they say, but 0 means 256.
PARAM	0x00 = 0	0xFF = 255	A Data-block of PARAM_LEN count of Bytes. for command w/o param set PARAM_LEN=1 and the single PARAM byte = 0
ACK	0x00 = 'OK'	0x0F	Interface Response Field with OK or Error Code. Only send by Interface. Error Codes range is from 0x01 to 0x0F (see table below)
XMODEM CRC16	0x0000 = 0	0xFFFF = 65535	As used in crc16.h of AVR-Gcc: Polynomial: x^16 + x^12 + x^5 + 1 (0x1021) Initial value: 0x0000 This is the CRC used by the Xmodem-CRC protocol. All previous bytes are calculated from START to PARAM

Rem: The last 2 byte in sequence = CRC. Hex-Values are show when they are allways equal.

Command Table	HexVal	DecVal	Ascii	Meaning
cmd_InterfaceTestAlive	**30**	**48**	**0**	May be send by PC to check: Interface and/or device still present and responding ?
PC sends:	2F 30 00 00 01 00 CF D4			param: no
Interface responds	2E 30 00 00 01 00 00 44 C2			Check device presence if connected, return ACK_OK or ACK_GENERAL_ERROR
				Rem: *BLHeliSuite sends this 1-2 times/sec to check the interface/device connection*
cmd_ProtocolGetVersion	**31**	**49**	**1**	Retrieve Interface Protocoll version
PC sends:	2F 31 00 00 01 00 65 85			param: no
Interface responds	2E 31 00 00 01 bb 00 CRC			param: bb = 1 Byte with interface protocol version number
				Rem: *The version number of this command table and handling*
cmd_InterfaceGetName	**32**	**50**	**2**	Retrieve Interface Name (Type) as text.
PC sends:	2F 32 00 00 01 00 8B 57			param: no
Interface responds	2E 32 00 00 nn abc... 00 CRC			param: nn = number of chars; abc... = chars with interface version text
				Rem: *Only the name of the interfaces (w/o the Rev. num)*
cmd_InterfaceGetVersion	**33**	**51**	**3**	Retrieve Interface version as byte value.
PC sends:	2F 33 00 00 01 00 21 06			param: no
Interface responds	2E 33 00 00 02 bb bb 00 CRC			param: bb = 2 Byte with Interface version number I.Byte= 13.2 II.Byte= .0.1
				Rem: *Rev. Number of the interface*
cmd_InterfaceExit	**34**	**52**	**4**	Exit PC Mode (SilC2: Resets the ESC's and) restarts Interface or Boxes Display Mode
PC sends:	2F 34 00 00 01 00 46 D2			param: no .
Interface responds	2E 34 00 00 01 00 00 42 63			param: no
cmd_DeviceReset	**35**	**53**	**5**	Reset connected Target (ESC)
PC sends:	2F 35 00 00 01 0n CRC			param: 00-07 select the ESC channel (MULTIPLE ESC interfaces only)
Interface responds	2E 35 00 00 01 0n 00 CRC			param: 00-07
				Rem: *SilC2: Used as a single command will restart the ESC*
cmd_DeviceGetID	**36**	**54**	**6**	REMOVED in protocol rev 6/106 -> cmd_DeviceInitFlash is used instead

Command Table	HexVal	DecVal	Ascii	Meaning
cmd_DeviceInitFlash	**37**	**54**	**6**	Enable Flash access to Target MCU and retrive MCU info
PC sends:	2F 37 00 00 01 0n CRC			param: 00-07 select the ESC channel (MULTIPLE ESC interfaces only)
Interface responds	2E 37 00 00 03 aa bb cc dd 00 CR			SilC2: param: aa=DeviceID bb=DerivativeID cc=LineState
				LineState: bit 0 = C2CK, bit 1 = C2D (0=Low/1= high) should be both high -> 11b
				Atm: param: aa=HiSign bb=LoSign cc=BootMsg last char ("*471x*") for versioning
				Sign: 2 lower bytes of Device Sign (eg. 0x9307 = Atmega8)
				SilBLB: param: aa=HiSign bb=LoSign cc=BootMsg last char ("*471x*") for versioning
				Sign: 2 bytes of DeviceName (eg. 0xF330 = C8051F330)
				All 106 dd=InterfaceMode (see cmd_InterfaceSetMode) Mode can change after autodetect
cmd_DeviceEraseAll	**38**	**56**	**8**	Erase whole memory of Target MCU
PC sends:	2F 38 00 00 01 00 CD F9			param: no
Interface responds	2E 38 00 00 01 00 00 49 80			param: no
				Rem: valid for SilC2, AtmSK not SilBLB not AtmBLB
cmd_DevicePageErase	**39**	**57**	**9**	Erase one page in memory of Target MCU
PC sends:	2F 39 00 00 01 bb CRC			param: bb = 1 Byte with the page number
Interface responds	2E 39 00 00 01 bb 00 CRC			param: bb = 1 Byte with the page number
				Rem: valid for SilC2 and SilBLB only
cmd_DeviceRead	**3A**	**58**	**:**	Read memory of Target MCU
PC sends:	2F 3A hi lo 01 nn CRC			param: hi lo = start address; nn = number of bytes to read
Interface responds	2E 3A hi lo nn bbb... 00 CRC			param: hi lo = start address; nn = number of data bytes; bbb... = data bytes
				Rem: *nn = 0 means: read 256 bytes*
cmd_DeviceWrite	**3B**	**59**	**;**	Write to memory of Target MCU
PC sends:	2F 3B hi lo nn bbb... CRC			param: hi lo = start address; nn = number of data bytes; bbb... = data bytes
Interface responds	2E 3B hi lo 01 00 00 CRC			param: hi lo = start address
				Rem: nn = 0 means: read 256 bytes
				Rem: Writes are internally verified with SilC2 only.

Command Table	HexVal	DecVal	Ascii	Meaning
cmd_DeviceC2CK_LOW	**3C**	**60**	**<**	Set Silabs C2 clock line (C2CK) to low
PC sends:	2F 3C 00 00 01 0n CRC			param: 00-07 select the ESC channel (MULTIPLE ESC interfaces only)
Interface responds	2E 3C 00 00 01 0n 00 CRC			param: 00-07
				Rem: valid for SilC2 only
cmd_DeviceReadEEprom	**3D**	**61**	**=**	Read EEprom of Target Atmel MCU
PC sends:	2F 3D hi lo 01 nn CRC			param: hi lo = start address; nn = number of bytes to read
Interface responds	2E 3D hi lo nn bbb... 00 CRC			param: hi lo = start address; nn = number of data bytes; bbb... = data bytes
				Rem: valid for Atm only. *nn = 0 means: read 256 bytes*
cmd_DeviceWriteEEprom	**3E**	**62**	**>**	Write to EEprom of Target Atmel MCU
PC sends:	2F 3E hi lo nn bbb... CRC			param: hi lo = start address; nn = number of data bytes; bbb... = data bytes
Interface responds	2E 3E hi lo 01 00 00 CRC			param: hi lo = start address
				Rem: valid for Atm only. *nn = 0 means: read 256 bytes*
cmd_InterfaceSetMode	**3F**	**63**	**?**	Set interface mode
PC sends:	2F 3F 00 00 01 0n CRC			param: 00-03 //SilC2=0, SiLBLB=1 ,AtmBLB=2, AtmSK=3
Interface responds	2E 3F 00 00 01 0n 00 CRC			param: 00-03
				Rem: valid full 4w-if interfaces only // respond ACK_OK or ACK_I_INVALID_PARAM

Errror codes

If a command sequence is send by the master and the interface fails to proceed, it will answer with an Error code.

Interface Error Response	2E cc hi lo 01 00 er CRC	Data: 00 cc = command which failed; hi+lo = address value which failed; er = Error Code

Error codes defined for ACK

ACK_OK	0x00	Operation succeeded. No Error.	
ACK_I_UNKNOWN_ERROR	0x01	Failure in the interface for unknown reason	unused
ACK_I_INVALID_CMD	0x02	Interface recognized an unknown command	
ACK_I_INVALID_CRC	0x03	Interface calculated a different CRC / data transmission form Master failed	
ACK_I_VERIFY_ERROR	0x04	Interface did a successful write operation over C2, but the read back data did not match	
ACK_D_INVALID_COMMAND	0x05	Device communication failed and the Status was 0x00 instead of 0x0D	unused
ACK_D_COMMAND_FAILED	0x06	Device communication failed and the Status was 0x02 or 0x03 instead of 0x0D	unused
ACK_D_UNKNOWN_ERROR	0x07	Device communication failed and the Status was of unknow value instead of 0x0D	unused
ACK_I_INVALID_CHANNEL	0x08	Interface recognized: unavailable ESC Port/Pin is adressed in Multi ESC Mode	
ACK_I_INVALID_PARAM	0x09	Interface recognized an invalid Parameter	
ACK_D_GENERAL_ERROR	0x0F	Device communication failed for unknown reason	

History:

V1.0 Intial release

V2.0 Added Support für Multiple BESC Handling
Interface Name starting with "m..." indicates: this is a multiple BESC Interface
The following Commands got a new parameter 0-7 which selects the BESC Channel 1..8
Once selected, the Channel will remain activ till another one is selected.
cmd_DeviceC2CK_LOW
cmd_DeviceReset
. cmd_DeviceInitFlash
To enable Interfaces with less than 8 channels ACK_I_INVALID_CHANNEL is added
Interface will respond if a Channel higher than supported is addressed.

V3.0 cmd_DeviceInitFlash returns the SiLabs device Derivative ID

V4.0 cmd_DeviceInitFlash combines cmd_DeviceReset + cmd_DeviceGetID + cmd_DeviceInitFlash
and returns DeviceID, DerivativeID and LineState for C2D and C2CK wires

V5.0 cmd_InterfaceGetVersion now returns 2 bytes.
(first byte = 2 digit main+ 1.digit sub / second byte 3. and 4. digit sub)
Length of cmd_InterfaceGetVersionStr is no longer fixed to 12 but variable length

V105 First Rev of 4way Interface (4w-if); Some Changes in Names
New Error Code ACK_I_INVALID_PARAM

V6/106 removed cmd_DeviceGetID
Internal Verify now for C2 removed / please use DeviceRead to verfiy
Fixed ACK_D_GENERAL_ERROR =0x0F onf 0xFF
Added new commands cmd_DeviceReadEEprom,cmd_DeviceWriteEEprom, cmd_InterfaceSetMode
Autodetect mode added for v106. Interface switches between BLHeli and SK bootloader Atmel/Silabs.