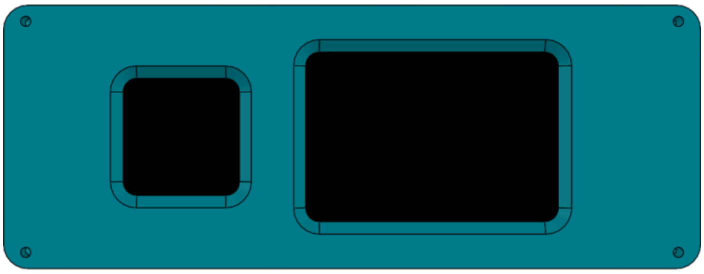
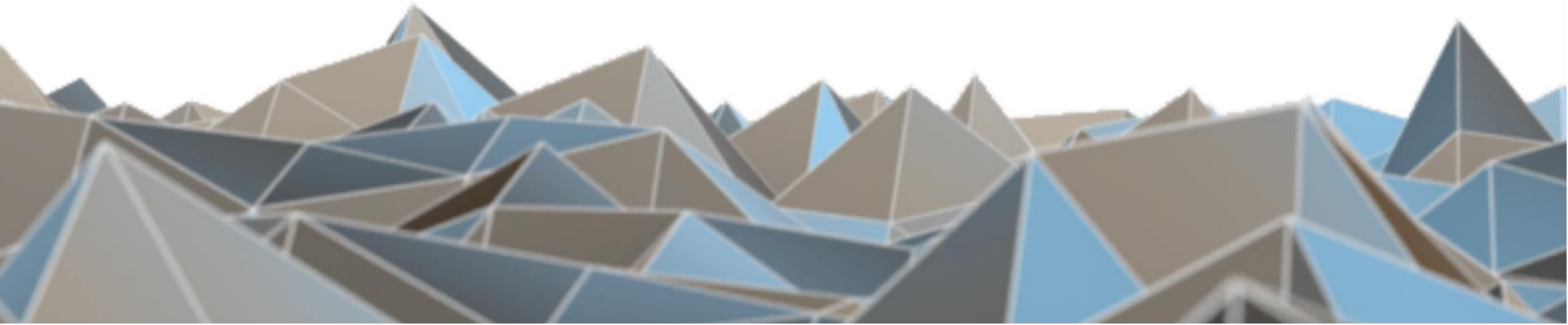
BLUETECHNIX
Embedding Ideas

Argos3D-P220



Software User Manual

Version 1



BECOM BLUETECHNIX GmbH

Gutheil-Schoder-Gasse 17
1230 Wien
AUSTRIA

office@bluetechnix.com
www.bluetechnix.com

Argos3D-P220 – Software User Manual

Document No.: 900-308 / A

Publication date: September 11, 2018

Subject to change without notice. Errors excepted.

Table of Contents

The information herein is given to describe certain components and shall not be considered as a guarantee of characteristics.

Terms of delivery and rights of technical change reserved.

We hereby disclaim any warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

BECOM BLUETECHNIX makes and you receive no warranties or conditions, express, implied, statutory or in any communication with you. BECOM BLUETECHNIX specifically disclaims any implied warranty of merchantability or fitness for a particular purpose.

BECOM BLUETECHNIX takes no liability for any damages and errors causing of the usage of this product. The user of this product is responsible by himself for the functionality of his application. He is allowed to use the product only if he has the qualification. More information is found in the General Terms and Conditions (AGB).

Information

For further information on technology, delivery terms and conditions and prices please contact BECOM BLUETECHNIX (http://www.bluetechnix.com).

Warning

Due to technical requirements components may contain dangerous substances.

1 General Information

This guide applies to the Argos3D-P220 camera from Bluetechnix. Follow this guide chapter by chapter to set up and understand your product. If a section of this document only applies to certain camera parts, this is indicated at the beginning of the respective section.

1.1 Symbols Used

This guide makes use of a few symbols and conventions:

⚠ **Warning**

Indicates a situation which, if not avoided, could result in minor or moderate injury and/or property damage or damage to the device.

⚠ **Caution**

Indicates a situation which, if not avoided, may result in minor damage to the device, in malfunction of the device or in data loss.

! **Note**

Notes provide information on special issues related to the device or provide information that will make operation of the device easier.

Procedures

A procedure always starts with a headline

1. The number indicates the step number of a certain procedure you are expected to follow. Steps are numbered sequentially.

This sign ➤ indicates an expected result of your action.

References

↳ This symbol indicates a cross reference to a different chapter of this manual or to an external document.

2 Overview

The document describes the necessary steps and settings to work with the Argos3D-P220 and describes the firmware dependent interfaces.

This document applies to firmware version 1.7.6.

For a hardware compatibility list please refer to our support site.

Software and documentation

↳ https://support.bluetechnix.com/index.html

3 Interfacing

The Argos3D-P220 provides control and data interfaces via Fast Ethernet.

The control interface is used to set and read the configuration of the Argos3D-P220 via a set of registers. Refer to Chapter 6 for a detailed register description.

The data interface provides a continuous stream of the distance and amplitude values or the XYZ data depending on the configuration.

The following types are used in the control and data streaming protocols:

- **Uint8**: 8 bit unsigned integer
- **Uint16**: 16 bit unsigned integer
- **Uint32**: 32 bit unsigned integer

!

Note

Values with ‘0x’ as prefix are hexadecimal values.

3.1 Control Interface

The Argos3D-P220 can be configured using the UPD control interface. For the control interface the Argos3D-P220 is listening to the following factory default IP settings:

- **IP-Address**: 192.168.0.10
- **Subnet mask**: 255.255.255.0
- **Network protocol**: UDP
- **UDP port**: 10003

!

Note

The Ethernet IP settings can be configured using the ***Eth0_*** registers. The changes become active on a device reset.

The Argos3D-P220 can be configured using a dedicated set of command frames. The Argos3D-P220 answers to each command frame with a dedicated response frame. The following table shows the currently supported command frames:

Command frame	Description
Register Read	Used to read one or more consecutive registers
Register Write	Used to write one or more consecutive registers
Reset	Used to reset/reboot the Argos3D-P220
Flash Update	Used to transfer files and updates
Keep Alive	Used to check device connection
Discovery	Used to look for Bluetechnix ToF cameras in the network

Table 3-1: Supported command frames

The following section describes each command frame and the expected answer in detail. To be able to communicate with the Argos3D-P220, the frame must be composed exactly as described.

3.1.1 Register read

Command frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	3	Command code for read
0x04	SubCommand	Uint8	XX	Ignored
0x05	Status	Uint8	XX	Ignored
0x06	Flags	Uint16	<flags>	[Bit 0] 1..Ignore DataCrc32
0x08	Length (high-byte first)	Uint32	<# of bytes to read>	Number of bytes to read (must be a multiple of two)
0x0C	HeaderData0 (high-byte) HeaderData1 (lowbyte)	Uint16	<Register Address>	Start register address for read command
0x0E	HeaderData2	Uint8	XX	Ignored
0x0F	HeaderData3	Uint8	XX	Ignored
0x10	CallbackIpVersion	UInt8	4	4: IPv4
0x11	CallbackIpAddr (high-byte first)	Uint32	<IP address>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x15	CallbackPort (high-byte first)	Uint16	<IP port>	The destination port for the response If set to 0, the device sends the packet back to the source port (since V1.6)
0x17	Reserved (35 bytes)	35*Uint8	XX	Ignored
0x3A	DataCrc32	Uint32	XX	Ignored
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]

Table 3-2: Register read command frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

Response frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	3	Command code for read
0x04	SubCommand	Uint8	XX	Ignore
0x05	Status	Uint8	Refer to table	Result code
0x06	Flags	Uint16	<flags>	[Bit 0] 1..Ignore DataCrc32
0x08	Length (high-byte first)	Uint32	<# of bytes read>	The number of bytes read (length of <Data> in bytes)
0x0C	HeaderData0 (high-byte) HeaderData1 (lowbyte)	Uint16	<Register Address>	Start register address of read data
0x0E	HeaderData2	Uint8	XX	Ignored

Addr	Field	Format	Value	Description
0x0F	HeaderData3	Uint8	XX	Ignored
0x10	Reserved (42 bytes)	Uint8[]	XX	reserved
0x3A	DataCrc32	Uint32	<CRC32 sum>	Checksum over <Data> 2)
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D 1)
0x40	Data	byte[]	<result data>	Result: One or more 16 bit values, each stored as big endian (high-byte first)

Table 3-3: Register read response frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.
Note 2): For the CRC32 calculation the CRC-32 is used (Polynom: 0x04C11DB7, start value: 0xFFFFFFFF). Please ask the Bluetechnix support for an implementation example of the CRC-32.

Result codes

Please refer to 3.1.8.

3.1.2 Register write

Command frame

Addr	Field	Format	Value	Description
0x00	Preamble	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	4	Command code for write
0x04	SubCommand	Uint8	XX	Ignored
0x05	Status	Uint8	XX	Ignored
0x06	Flags	Uint16	<flags>	[Bit 0] 1..Ignore DataCrc32
0x08	Length (high-byte first)	Uint32	<# of bytes to write>	The number of bytes to write (must be a multiple of two and match length of <Data> in bytes)
0x0C	HeaderData0 (high-byte) HeaderData1 (lowbyte)	Uint16	<Register Address>	Start register address for write command
0x0E	HeaderData2	Uint8	XX	Ignored
0x0F	HeaderData3	Uint8	XX	Ignored
0x10	CallbackIpVersion	UInt8	4	4: IPv4
0x11	CallbackIpAddr (high-byte first)	Uint32	<IP address>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x15	CallbackPort (high-byte first)	Uint16	<IP port>	The destination port for the response If set to 0, the device sends the packet back to the source port (since V1.6)
0x17	Reserved (35 bytes)	35*Uint8	XX	Ignored
0x3A	DataCrc32	Uint32	<CRC32 sum>	Checksum over <Data> 2)
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D 1)
0x40	Data	byte[]	<data to write>	One or more 16 bit values in a stream that should be written, each stored as big endian (high-byte first)

Table 3-4: Register write command frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

Note 2): For the CRC32 calculation the CRC-32 is used (Polynom: 0x04C11DB7, start value: 0xFFFFFFFF). Please ask the Bluetechnix support for an implementation example of the CRC-32.

Response frame

See General Response (3.1.7).

Flags

Flags	Description
Bit 0	1: Ignore DataCrc32

Table 3-5: Register write flag description

Result codes

Please refer to 3.1.8.

3.1.3 Reset

Command frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	7	Command code for reset
0x04	SubCommand	Uint8	XX	Ignored
0x05	Status	Uint8	XX	Ignored
0x06	Flags	Uint16	XX	Ignored
0x08	Length (high-byte first)	Uint32	0	No data
0x0C	HeaderData0	Uint8	XX	Ignored
0x0D	HeaderData1	Uint8	XX	Ignored
0x0E	HeaderData2	Uint8	XX	Ignored
0x0F	HeaderData3	Uint8	XX	Ignored
0x10	CallbackIpVersion	UInt8	4	4: IPv4
0x11	CallbackIpAddr (high-byte first)	Uint32	<IP address>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x15	CallbackPort (high-byte first)	Uint16	<IP port>	The destination port for the response If set to 0, the device sends the packet back to the source port (since V1.6)
0x17	Reserved (35 bytes)	35*Uint8[]	XX	Ignored
0x3A	DataCrc32	Uint32	0	No data, no checksum
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]

Table 3-6: Reset command frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

Response frame

See General Response (3.1.7).

Flags

Flags	Description
	Currently no flags defined for this command

Table 3-7: Reset flag description

Result codes

Please refer to 3.1.8.

3.1.4 Flash Update

Command frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	11, 12, 13 or 21	11: Flash Bootloader 12: Flash Application 13: Flash generic file 21: Flash Lens Calibration Data
0x04	SubCommand	Uint8	0, 1 or 2	If Command == 13 (otherwise ignored): 0: Write to SPI flash 1: Write to parallel flash
0x05	Status	Uint8	XX	Ignored
0x06	Flags	Uint16	0	Bit 0 must be cleared and DataCrc32 must be valid
0x08	Length (high-byte first)	Uint32	<# of bytes of data>	The size of the data of this packet
0x0C	HeaderData0 (high-byte) HeaderData1 HeaderData2 HeaderData3 (lowbyte)	Uint32	<Flash Address>	A generic file is flashed to this address. When Flashing a Bootloader or application it is ignored
0x10	CallbackIpVersion	UInt8	4	4: IPv4
0x11	CallbackIpAddr (high-byte first)	Uint32	<IP address>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x15	CallbackPort (high-byte first)	Uint16	<IP port>	The destination port for the response If set to 0, the device sends the packet back to the source port (since V1.6)
0x17	PacketNumber (high-byte first)	UInt32	<# current>	A consecutive numbering of the packets to send (starting at 1)

Addr	Field	Format	Value	Description
0x1B	FileSize (high-byte first)	UInt32	<file size>	Length of the binary file to flash
0x1F	FileCRC32	UInt32	<CRC32 sum>	Checksum over the complete binary file [2)]
0x23	Reserved (23 bytes)	Uint8[]	XX	Ignored
0x3A	DataCrc32	Uint32	<CRC32 sum>	Checksum over <Data> [2)] [3)]
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]
0x40	Data	byte[]	<binary loader file>	The loaderfile to flash in a bytestream

Table 3-8: Flash update command frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

Note 2): For the CRC32 calculation the CRC-32 is used (Polynom: 0x04C11DB7, start value: 0xFFFFFFFF). Please ask the Bluetechnix support for an implementation example of the CRC-32.

Note 3): The DataCrc32 is mandatory, the appropriate flag must be set to 0.

Response frame

See General Response (3.1.7).

Flags

Flags	Description
Bit 0	1: Ignore DataCrc32

Table 3-9: Flash update flag description

Result codes

Please refer to 3.1.8.

3.1.5 Keep Alive

Command frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	254	Command code for ‚Alive message‘
0x04	SubCommand	Uint8	XX	Ignored
0x05	Status	Uint8	XX	Ignored
0x06	Flags	Uint16	XX	Ignored
0x08	Length (high-byte first)	Uint32	0	No data
0x0C	HeaderData0	Uint8	XX	Ignored
0x0D	HeaderData1	Uint8	XX	Ignored
0x0E	HeaderData2	Uint8	XX	Ignored
0x0F	HeaderData3	Uint8	XX	Ignored
0x10	CallbackIpVersion	UInt8	4	4: IPv4

Addr	Field	Format	Value	Description
0x11	CallbackIpAddr (high-byte first)	Uint32	<IP address>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x15	CallbackPort (high-byte first)	Uint16	<IP port>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x17	Reserved (35 bytes)	35*Uint8[]	XX	Ignored
0x3A	DataCrc32	Uint32	0	No data, no checksum
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]

Table 3-10: Alive command frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

Response frame

See General Response (3.1.7).

Flags

Flags	Description
	Currently no flags defined for this command

Table 3-11: Alive flag description

Result codes:

Please refer to 3.1.8.

3.1.6 Discovery

Command frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	253	Command code for ‚Discovery'
0x04	SubCommand	Uint8	XX	Ignored
0x05	Status	Uint8	XX	Ignored
0x06	Flags	Uint16	<flags>	[Bit 0] 1..Ignore DataCrc32
0x08	Length (high-byte first)	Uint32	0	No data
0x0C	HeaderData0 (high-byte) HeaderData1 (lowbyte)	Uint16	Device Type	Device type to discover (0 for any device)
0x0E	HeaderData2	Uint8	XX	Ignored
0x0F	HeaderData3	Uint8	XX	Ignored
0x10	CallbackIpVersion	UInt8	4	4: IPv4

Addr	Field	Format	Value	Description
0x11	CallbackIpAddr (high-byte first)	Uint32	<IP address>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x15	CallbackPort (high-byte first)	Uint16	<IP port>	The destination address for the response If set to 0.0.0.0, the device sends the packet back to the source address (since V1.6)
0x17	Reserved (35 bytes)	35*Uint8[]	XX	Ignored
0x3A	DataCrc32	Uint32	0	No data, no checksum
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]

Table 3-12: Alive command frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

Response frame

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	253	Command code for 'Discovery'
0x04	SubCommand	Uint8	XX	Ignore
0x05	Status	Uint8	Refer to table	Result code
0x06	Flags	Uint16	<flags>	[Bit 0] 1..Ignore DataCrc32
0x08	Length (high-byte first)	Uint32	N	length of <Data> in bytes
0x0C	HeaderData0 (high-byte) HeaderData1 (lowbyte)	Uint16	XX	Ignored
0x0E	HeaderData2	Uint8	XX	Reserved
0x0F	HeaderData3	Uint8	XX	Reserved
0x10	Reserved (42 bytes)	Uint8[]	XX	Reserved
0x3A	DataCrc32	Uint32	<CRC32 sum>	Checksum over <Data> [2)]
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]
0x40	DeviceMAC	6*Uint8		Discovered device MAC
46	DeviceIpVersion	Uint8	4	4: IPv4
47	DeviceIp	UInt32		Discovered device IP
4B	SubnetMask	UInt32		Discovered device Subnet Mask
4F	GatewayIp	UInt32		Discovered device Gateway IP
53	UdpStreamIpVersion	Uint8	4	4: IPv4
54	UdpStreamIP	UInt32		Discovered device UDP stream IP
58	UdpStreamPort	Uint16		Discovered device UDP stream port
5A	UdpConfigPort	Uint16		Discovered device UDP config port
5C	Reserved	Uint16		Reserved
5E	Reserved	Uint16		Reserved
60	DeviceType	Uint16		Discovered device register DeviceType
62	DeviceSerialNumber	Uint32		Discovered device register SerialNrLow and SerialNumberHigh
66	DeviceUptime	UInt32		Discovered device register UptimeLow and UptimeHight
6A	Mode0Register	Uint16		Discovered device register Mode0

Addr	Field	Format	Value	Description
6C	StatusRegister	Uint16		Discovered device register Status
6E	FirmwareVersion	Uint16		Discovered device register FirmwareInfo

Table 3-13: Register read response frame

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.
Note 2): For the CRC32 calculation the CRC-32 is used (Polynom: 0x04C11DB7, start value: 0xFFFFFFFF). Please ask the Bluetechnix support for an implementation example of the CRC-32.

Result codes

Please refer to 3.1.8.

3.1.7 General Response

Addr	Field	Format	Value	Description
0x00	Preamble (high-byte first)	Uint16	0xa1ec	Unique identifier, start of header
0x02	ProtocolVersion	Uint8	3	This document refers to version V3.0
0x03	Command	Uint8	<command code>	Command code of the original command sent
0x04	SubCommand	Uint8	<subcommand code>	SubCommand code of the original command sent
0x05	Status	Uint8	Refer to table	Result code
0x06	Flags	Uint16	<flags>	[Bit 0] 1..Ignore DataCrc32
0x08	Length (high-byte first)	Uint32	0	Length of <Data> is zero
0x0C	HeaderData0	Uint8	<header data 0>	Same as in sent command
0x0E	HeaderData1	Uint8	<header data 1>	Same as in sent command
0x0E	HeaderData2	Uint8	<header data 2>	Same as in sent command
0x0F	HeaderData3	Uint8	<header data 3>	Same as in sent command
0x10	Reserved (42 bytes)	Uint8[]	<reserved data>	Same as in sent command
0x3A	DataCrc32	Uint32	0	No <Data> present
0x3E	HeaderCrc16	Uint16	<CRC16 sum>	Checksum over 60 bytes of Header: 0x02 – 0x3D [1)]

Table 3-14: General Response Frame description

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

3.1.8 Result codes

Status	Description
0	Ok
13	Invalid handle (internal error)
15	Illegal write: The Address is not valid or the register is not write-enabled
16	Illegal read: The Address is not valid (deprecated, replaced by 17)
17	Register end reached
248	Invalid Packet Nr
249	IP Version not supported
250	Length exceeds maximum filesize (not enough memory for file download)
251	HeaderCrc16 mismatch
252	DataCrc32 mismatch
253	Length invalid: Cannot be equal 0
254	Length invalid: Cannot be grater 0
255	Unknown command

Table 3-15: Result code list

3.2 3D Data Interface

A UDP stream delivers depth and amplitude data from the Argos3D-P220. Each UDP packet contains a header and up to 1400 bytes of data (Ethernet, IP, and UDP headers are not shown in Figure 3-1).

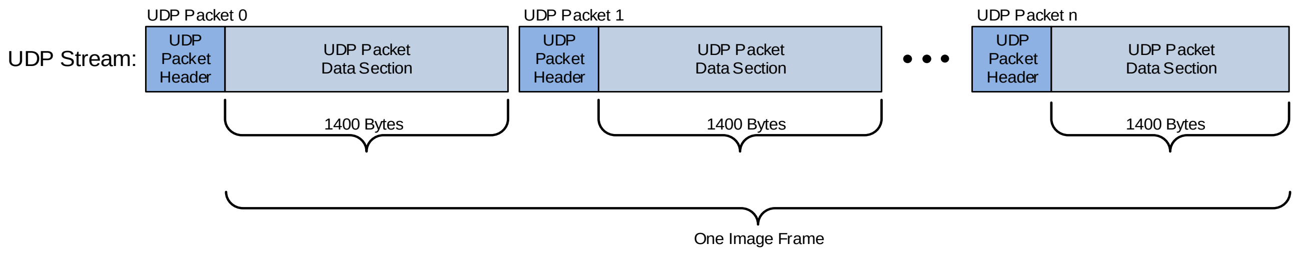


Figure 3-1: UDP streaming data format

The UDP streaming is enabled by factory default. The Argos3D-P220 streams to the following IP settings:

- **IP-Address**: Multicast address 224.0.0.1
- **UDP port**: 10002

! **Note**

The UDP stream settings can be configured using the ***Eth0_*** registers.

As multicast is used more than one host can receive the stream within the same subnet at the same time. The client has to join the appropriate multi cast group and open the port 10002 on his local network interface card (NIC) where the Argos3D-P220 is connected to. The receiver should receive the stream and interpret it as the following protocol description shows.

Note

Be aware that a multicast stream may slow down your Ethernet network as the stream may be spread to all active links of switches/hubs and routers.

3.2.1 UDP Streaming Header

The current protocol version is **1**.

Each image transmitted on the UDP stream is split into packets of max. 1432 bytes length. Each packet consists of a 32 byte packet header and up to 1400 bytes of image data section (refer to Figure 3-1).

Addr	Field	Type	Value	Description
0x00	Version	Uint16 (high byte first)	0x0001	Protocol version
0x02	FrameCounter	Uint16 (high byte first)		Continuous frame counter. On an overrun it restarts at 0.
0x04	PacketCounter	Uint16 (high byte first)		Actual packet #. The frame data must be recomposed in order of the packet #.
0x06	DataLength	Uint16 (high byte first)		Length of the image data section of the current packet.
0x08	FrameSize	Uint32 (high byte first)		Size of the image data. It may be used to calculate the expected # of packets for a frame.
0x0C	PacketCRC32	Uint16 (high byte first)		CRC32 checksum over the entire packet (pos 0 to pos n) [1)]
0x10	Flags	Uint32	Refer to Table 3-17	Optional flags
0x14	Reserved			Reserved for future use
0x20	ImageData			Image data section

Table 3-16: UDP packet header

Note 1): For the CRC32 calculation the CRC-32 is used (Polynom: 0x04C11DB7, start value: 0xFFFFFFFF). Please ask the Bluetechnix support for an implementation example of the CRC-32.

Flags

Flags	Description
Bit 0	1: Ignore DataCrc32

Table 3-17: UDP packet header flag description

3.2.2 Frame Header

The frame data itself is split into a 64 byte frame header and the frame data section. The format of the frame data depends on the selected image format and is described in chapter 4.3. Below you can find the format of the 64 byte frame header.

Addr	Field	Type	Value	Description
0x00	Reserved	Uint16	0xFFFF	
0x02	HeaderVersion	Uint16 (high byte first)	0x0003	Current header version

Addr	Field	Type	Value	Description
0x04	ImageWidth	Uint16 (high byte first)		Width of the image in pixels.
0x06	ImageHeight	Uint16 (high byte first)		Height of the image in pixels.
0x08	NofChannels	Uint8		Nof data channels. Depends on the image format
0x09	BytesPerPixel	Uint8		Bytes per pixel of the 3D image data.
0x0A	ImageFormat	Uint16 (high byte first)		The content is the same as in the register *ImageDataFormat*).
0x0C	Timestamp	Uint32 (high byte first)		Timestamp of the actual image in µs
0x10	FrameCounter	Uint16 (high byte first)		Continuous frame counter. On an overrun it restarts at 0.
0x12	Reserved			
0x1A	MainTemp	Uint8		Typically, ToF sensor temperature in °C + 50. Decrement this field by 50 to get the current temperature of the ToF sensor.
0x1B	LedTemp	Uint8		LED temperature in °C + 50. Decrement this field by 50 to get the current temperature of the illumination LEDs.
0x1C	FirmwareVersion	Uint16 (high byte first)		Content of the register *FirmwareInfo.*
0x1E	MagicV31	Uint16 (high byte first)	0x3331	These magic bytes indicate that header version is 3.1 *Valid since version 3.1*
0x20	IntegrationTime	Uint16 (high byte first)		Integration time in us. *Valid since version 3.1*
0x22	ModFreq	Uint16 (high byte first)		Modulation frequency with resolution 10 kHz (e.g., a value of 0x1234 means frequency 46.6 MHz) *Valid since version 3.1*
0x24	Temp3	Uint8		Temperature sensor #3 (Baseboard Sensor) in °C + 50. Decrement this field by 50 to get the current temperature. *Valid since version 3.1*
0x25	Reserved			
0x26	Reserved			
0x28	Reserved			
0x2A	Reserved			
0x3E	CRC16	Uint16 (high byte first)		CRC16 checksum over the header without the first two bytes and the CRC16 checksum itself (addr 0x02 to addr 0x3D) 1)
0x40	Data	Bytestream		Various channels described by the header with ToF data

Table 3-18: Image data header

Note 1): For the CRC16 calculation the CRC-CCITT is used (Polynom: 0x1021, start value: 0). Please ask the Bluetechnix support for an implementation example of the CRC-CCITT.

3.3 Manual frame triggers

The Argos3D-P220's default mode is video mode, where the camera streams continuously with configured frame rate. To use manual frame triggering, you have to disable the video mode in register ***Mode0***.

You can either trigger a frame via

- Hardware trigger: The signal is sensitive to a rising edge.
- Software trigger: See register ***Mode0***.

Both will trigger a frame capture on the ToF sensor. See chapter 4.9 for more information.

3.4 GPIOs

The camera features one general-purpose input/output and one general-purpose input on the connector. Please see the register description in chapter 4.13 for more information.

4 Camera Features

4.1 Basic Settings

The Argos3D-P220 starts according to the factory default values as described in the register description section (refer to chapter 6).

4.2 Image Processing Chain

The following flow diagram shows the Argos3D-P220's image processing chain for depth data. For amplitude data no post processing is performed.

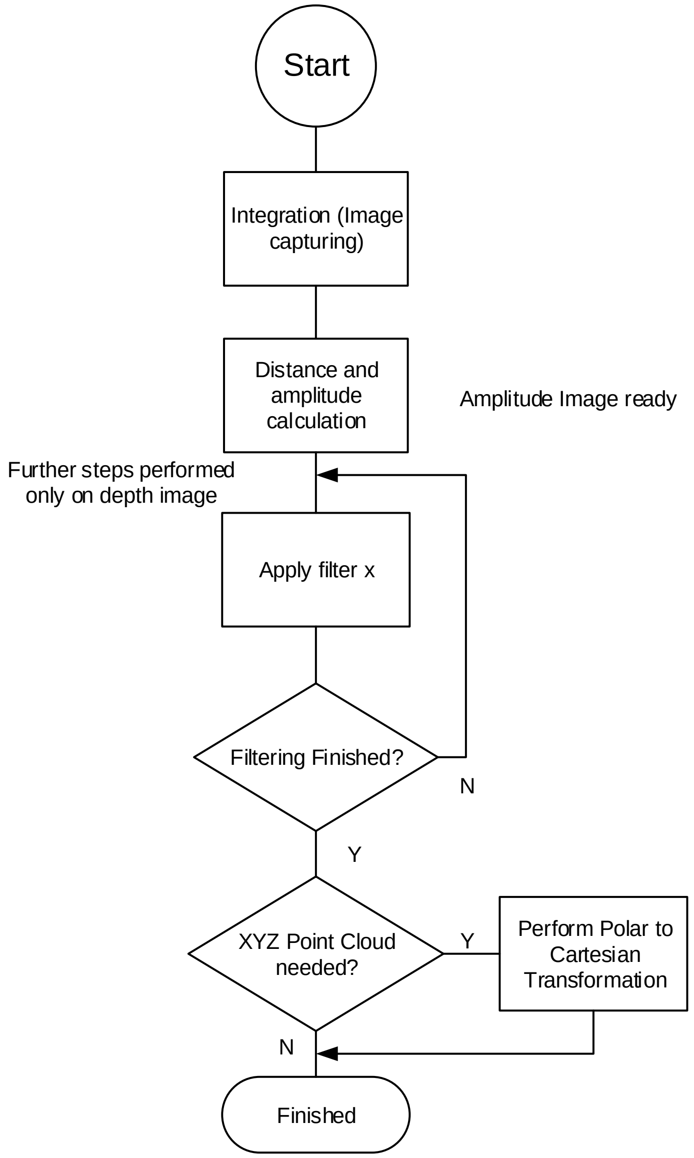

Start
Integration (Image capturing)
Distance and amplitude calculation
Amplitude Image ready
Further steps performed only on depth image
Apply filter x
Filtering Finished?
N
Y
XYZ Point Cloud needed?
Y
Perform Polar to Cartesian Transformation
N
Finished

Figure 4-1: Image processing flow

4.2.1 Image filtering

After the distance and amplitude calculation, the filters are applied to the depth data. The amplitude data will be left unfiltered. Each of the filter provides one or more configuration parameters. The iteration count for each filter can also be configured. The filters can be enabled or disabled by writing the ***ImgProcConfig*** register. Enabling more than one filter is possible but each added filter reduces the maximum achievable frame rate (as does the number of iterations).

4.2.1.1 Median Filter

A 3x3 median filter can be applied.

Register: ***FilterMedianConfig***

The number of iterations is configurable.

4.2.1.2 Bilateral filter

Registers: ***FilterBilateralConfig***

Configuration options are σ_R (weight for radius), σ_D (weight for data) and number of iterations.

4.2.1.3 Sliding Average Filter

Register: ***FilterSLAFconfig***

A sliding average filter over up to 20 frames can be applied. The number of frames is configurable. An increasing number of frames will not decrease the frame rate but may add blurring effects.

4.2.2 Pixel invalidation

The Argos3D-P220 provides an on-board check for invalid pixels.

If the amplitude of the reflected signal is below a threshold (underexposure), the distance value of the corresponding pixel will be set to 0xFFFF. If the amplitude is too high (overexposure) the distance value will be set to 0x0000. The lower and upper amplitude limit for invalidating pixels can be set by using the registers ***ConfidenceThresLow*** and ***ConfidenceThresHigh***.

For inconsistent pixels (due to unreliable data), the distance value is set to 0x0001.

4.3 Camera Coordinate System

The default coordinate system starts pixel numbering in the upper left corner of the pixel array, seen from the camera’s point of view. Also note the directions of X, Y, and Z coordinates (In XYZ image modes).

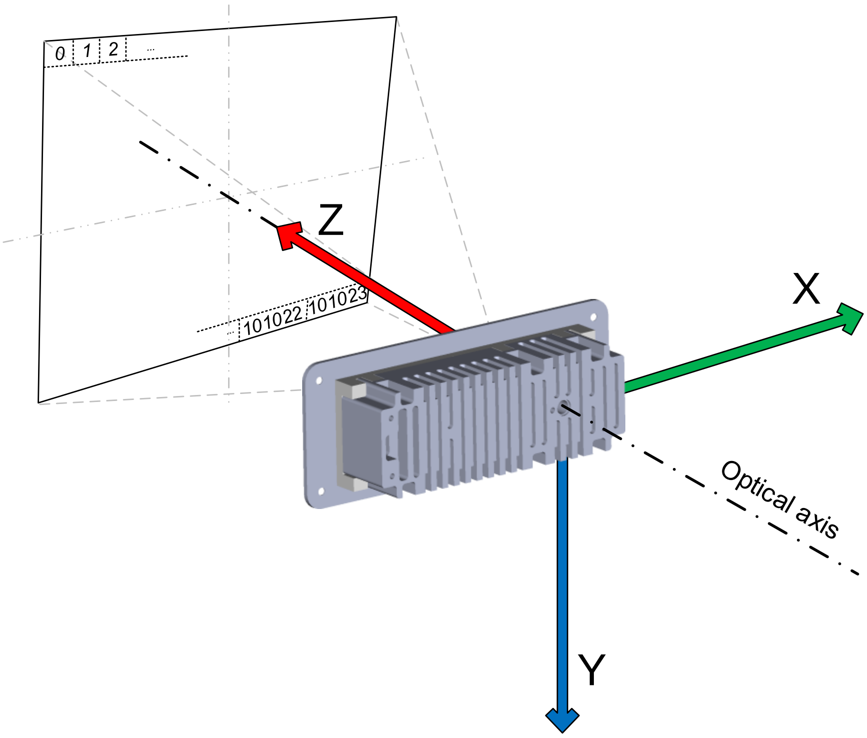


Figure 4-2: Argos3D-P220 Default Coordinate System

4.4 Camera Data Format

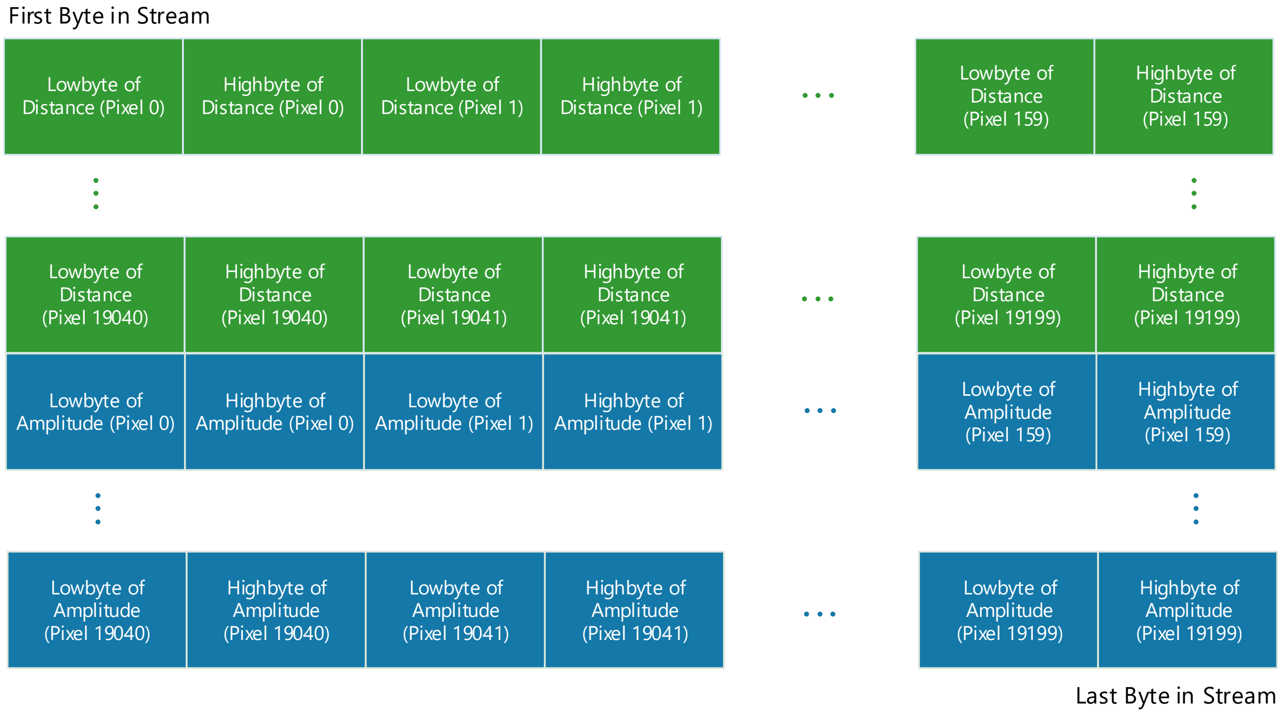
The camera provides up to four data channels. The meaning of each data channel depends on the selected data format. The factory default setting provides an array of depth data in millimeters as 16 bit unsigned (Uint16) and an array of grayscale values (Amplitudes) also as 16bit unsigned for each pixel. When changing the image data format properly, a 3D XYZ coordinate set per pixel is provided. Refer to chapter 4.3 for a description of the coordinate systems of the camera.

The image format can be selected in the register ***ImageDataFormat***. The following sections describe each of the supported formats in detail. Only the data section which contains the image data of the transferred frame will be described. For information about the packet format and meta-data please refer to chapter 3.2.

4.4.1 Distances and Amplitudes

In this mode the distances and amplitudes will be transferred in progressive mode, first the distance array, then the amplitude array. The stream starts always with pixel #0.

[ImageDataFormat = 0] The **distances** are coded in **millimeters** as **Uint16**. The **amplitudes** are also **Uint16**.



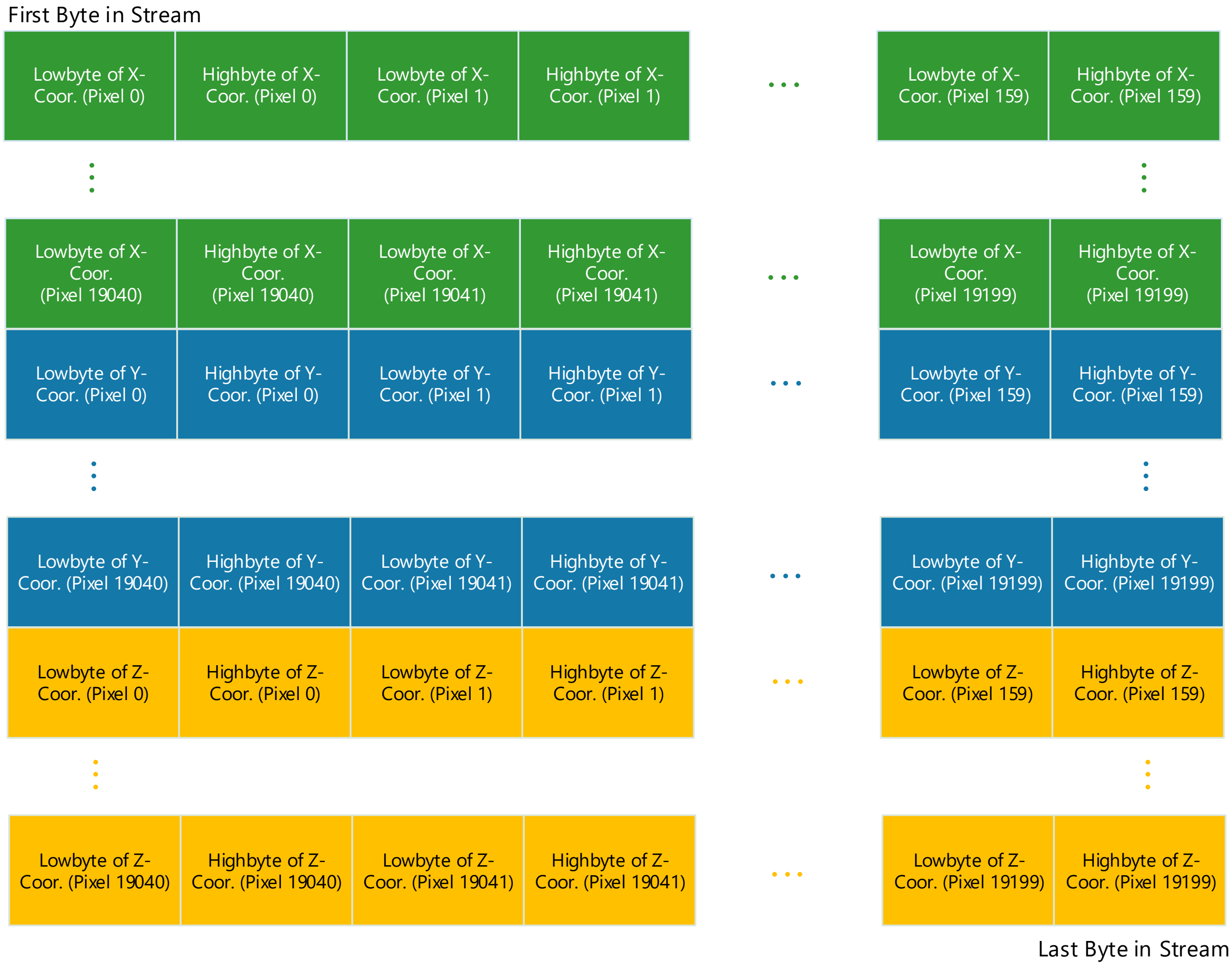
Lowbyte of Distance (Pixel 0)	Highbyte of Distance (Pixel 0)	Lowbyte of Distance (Pixel 1)	Highbyte of Distance (Pixel 1)	…	Lowbyte of Distance (Pixel 159)	Highbyte of Distance (Pixel 159)
⋮						⋮
Lowbyte of Distance (Pixel 19040)	Highbyte of Distance (Pixel 19040)	Lowbyte of Distance (Pixel 19041)	Highbyte of Distance (Pixel 19041)	…	Lowbyte of Distance (Pixel 19199)	Highbyte of Distance (Pixel 19199)
Lowbyte of Amplitude (Pixel 0)	Highbyte of Amplitude (Pixel 0)	Lowbyte of Amplitude (Pixel 1)	Highbyte of Amplitude (Pixel 1)	…	Lowbyte of Amplitude (Pixel 159)	Highbyte of Amplitude (Pixel 159)
⋮						⋮
Lowbyte of Amplitude (Pixel 19040)	Highbyte of Amplitude (Pixel 19040)	Lowbyte of Amplitude (Pixel 19041)	Highbyte of Amplitude (Pixel 19041)	…	Lowbyte of Amplitude (Pixel 19199)	Highbyte of Amplitude (Pixel 19199)

Figure 4-3: Data stream of Distance and Amplitude data

4.4.2 XYZ Point Cloud

In this mode the XYZ point cloud will be transferred in progressive mode, first the X coordinate array then the Y and Z coordinate array. The stream starts always with pixel #0.

[ImageDataFormat = 24] The **coordinates** are coded in **millimeters** as **Int16.**



First Byte in Stream

Lowbyte of X-Coor. (Pixel 0)	Highbyte of X-Coor. (Pixel 0)	Lowbyte of X-Coor. (Pixel 1)	Highbyte of X-Coor. (Pixel 1)	…	Lowbyte of X-Coor. (Pixel 159)	Highbyte of X-Coor. (Pixel 159)
⋮						⋮
Lowbyte of X-Coor. (Pixel 19040)	Highbyte of X-Coor. (Pixel 19040)	Lowbyte of X-Coor. (Pixel 19041)	Highbyte of X-Coor. (Pixel 19041)	…	Lowbyte of X-Coor. (Pixel 19199)	Highbyte of X-Coor. (Pixel 19199)
Lowbyte of Y-Coor. (Pixel 0)	Highbyte of Y-Coor. (Pixel 0)	Lowbyte of Y-Coor. (Pixel 1)	Highbyte of Y-Coor. (Pixel 1)	…	Lowbyte of Y-Coor. (Pixel 159)	Highbyte of Y-Coor. (Pixel 159)
⋮						⋮
Lowbyte of Y-Coor. (Pixel 19040)	Highbyte of Y-Coor. (Pixel 19040)	Lowbyte of Y-Coor. (Pixel 19041)	Highbyte of Y-Coor. (Pixel 19041)	…	Lowbyte of Y-Coor. (Pixel 19199)	Highbyte of Y-Coor. (Pixel 19199)
Lowbyte of Z-Coor. (Pixel 0)	Highbyte of Z-Coor. (Pixel 0)	Lowbyte of Z-Coor. (Pixel 1)	Highbyte of Z-Coor. (Pixel 1)	…	Lowbyte of Z-Coor. (Pixel 159)	Highbyte of Z-Coor. (Pixel 159)
⋮						⋮
Lowbyte of Z-Coor. (Pixel 19040)	Highbyte of Z-Coor. (Pixel 19040)	Lowbyte of Z-Coor. (Pixel 19041)	Highbyte of Z-Coor. (Pixel 19041)	…	Lowbyte of Z-Coor. (Pixel 19199)	Highbyte of Z-Coor. (Pixel 19199)

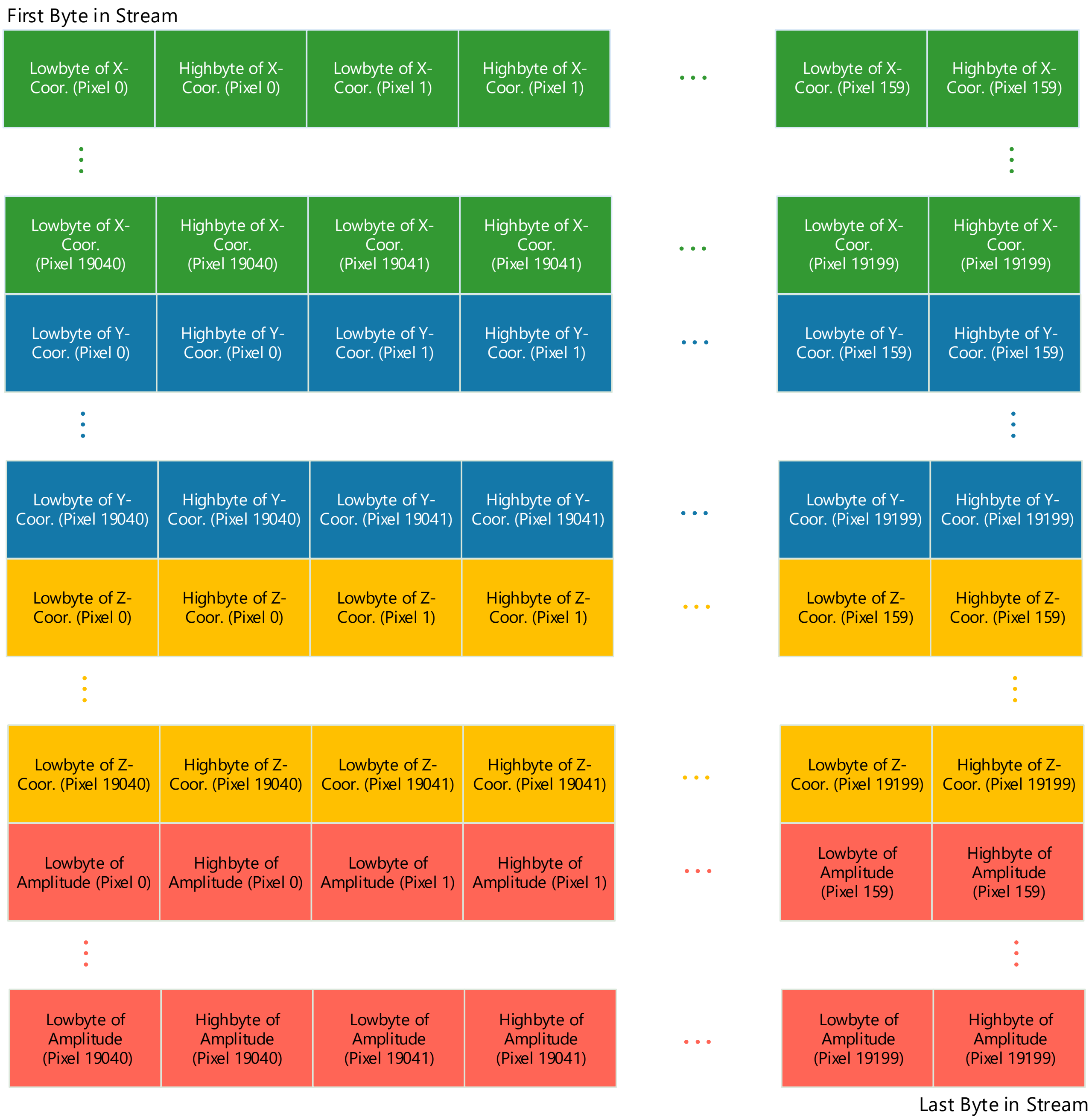
Last Byte in Stream

Figure 4-4: Data stream of XYZ Point Cloud

4.4.3 XYZ Point Cloud and Amplitude

In this mode the XYZ point cloud and the amplitude will be transferred in progressive mode. The stream starts always with pixel #0.

[ImageDataFormat = 32] The **coordinates** are coded in **millimeters** as **Int16** the **amplitudes** as **Uint16.**



First Byte in Stream

Lowbyte of X-Coor. (Pixel 0)	Highbyte of X-Coor. (Pixel 0)	Lowbyte of X-Coor. (Pixel 1)	Highbyte of X-Coor. (Pixel 1)	…	Lowbyte of X-Coor. (Pixel 159)	Highbyte of X-Coor. (Pixel 159)
⋮						⋮
Lowbyte of X-Coor. (Pixel 19040)	Highbyte of X-Coor. (Pixel 19040)	Lowbyte of X-Coor. (Pixel 19041)	Highbyte of X-Coor. (Pixel 19041)	…	Lowbyte of X-Coor. (Pixel 19199)	Highbyte of X-Coor. (Pixel 19199)
Lowbyte of Y-Coor. (Pixel 0)	Highbyte of Y-Coor. (Pixel 0)	Lowbyte of Y-Coor. (Pixel 1)	Highbyte of Y-Coor. (Pixel 1)	…	Lowbyte of Y-Coor. (Pixel 159)	Highbyte of Y-Coor. (Pixel 159)
⋮						⋮
Lowbyte of Y-Coor. (Pixel 19040)	Highbyte of Y-Coor. (Pixel 19040)	Lowbyte of Y-Coor. (Pixel 19041)	Highbyte of Y-Coor. (Pixel 19041)	…	Lowbyte of Y-Coor. (Pixel 19199)	Highbyte of Y-Coor. (Pixel 19199)
Lowbyte of Z-Coor. (Pixel 0)	Highbyte of Z-Coor. (Pixel 0)	Lowbyte of Z-Coor. (Pixel 1)	Highbyte of Z-Coor. (Pixel 1)	…	Lowbyte of Z-Coor. (Pixel 159)	Highbyte of Z-Coor. (Pixel 159)
⋮						⋮
Lowbyte of Z-Coor. (Pixel 19040)	Highbyte of Z-Coor. (Pixel 19040)	Lowbyte of Z-Coor. (Pixel 19041)	Highbyte of Z-Coor. (Pixel 19041)	…	Lowbyte of Z-Coor. (Pixel 19199)	Highbyte of Z-Coor. (Pixel 19199)
Lowbyte of Amplitude (Pixel 0)	Highbyte of Amplitude (Pixel 0)	Lowbyte of Amplitude (Pixel 1)	Highbyte of Amplitude (Pixel 1)	…	Lowbyte of Amplitude (Pixel 159)	Highbyte of Amplitude (Pixel 159)
⋮						⋮
Lowbyte of Amplitude (Pixel 19040)	Highbyte of Amplitude (Pixel 19040)	Lowbyte of Amplitude (Pixel 19041)	Highbyte of Amplitude (Pixel 19041)	…	Lowbyte of Amplitude (Pixel 19199)	Highbyte of Amplitude (Pixel 19199)

Last Byte in Stream

Figure 4-5: Data-stream of XYZ Point Cloud and Amplitude

4.4.4 Distances and XYZ Point Cloud

In this mode the distances and the XYZ point cloud will be transferred in progressive mode, first the distances array, then X, Y, and Z coordinate arrays (in this order). The stream starts always with pixel #0.

[ImageDataFormat = 72] The **distances** are coded in millimeters as Uint16. The **coordinates** are coded in **millimeters** as **Int16**.

4.4.5 X coordinate and Amplitudes

In this mode a single coordinate array, more specifically, the one belonging to the optical axis of the camera (X), is transferred, as well as the amplitudes.

[ImageDataFormat = 80] **Coordinate** values are coded in **millimeters** as **Int16**. The **amplitudes** are coded as **Uint16**.

4.4.6 Distances

In this mode a single array with distances is transferred. The stream starts always with pixel #0.

[ImageDataFormat = 96] The **distances** are coded in **millimeters** as **Uint16**.

4.5 Modulation Frequency

The modulation frequency is set to 22.5 MHz per default. Other modulation frequencies can be set using the register ***ModulationFrequency***. Be aware that this also changes the ambiguity range of the camera.

The following modulation frequencies can be selected:

Index	Frequency	Unambiguity Range
0	5 MHz	~30m
1	5.63 MHz	~26m
2	6.43 MHz	~23m
3	7.5 MHz	~20m
4	9 MHz	~16.5m
5	11.25 MHz	~13m
6	15 MHz	~10m
7	22.5 MHz	~6,6m
8	45 MHz	~3,3m

Table 4-1: Pre-defined modulation frequencies

In the register you can either write the frequency (frequency/10000) or the index as listed in Table 4-1. On a read of the register you get the currently selected modulation frequency (again, in 10-kHz-steps).

Other frequencies cannot be set.

4.6 Frame rate and Integration Time

The frame rate and the integration time can be set by using the registers ***Framerate*** and ***IntegrationTime***. The camera integration time is limited by hardware to 25 ms at maximum and 50 µs at minimum.

The maximum frame rate is ~40 fps but may be limited by the integration time. The combination of frame rate and integration time influences the input current as well as the dissipated heat and will be characterized by the *“Frame rate Integration Time Product”* (FITP) which has been defined as follows:

$$FITP = t_{INT}\,[ms] \cdot fps \left[\frac{1}{s}\right] \cdot 4$$

Caution

Be careful in setting different integration times and frame-rate combinations. Not all combinations are possible! Without appropriate cooling the device may be damaged! Refer to the Hardware User Manual for more information.

Note

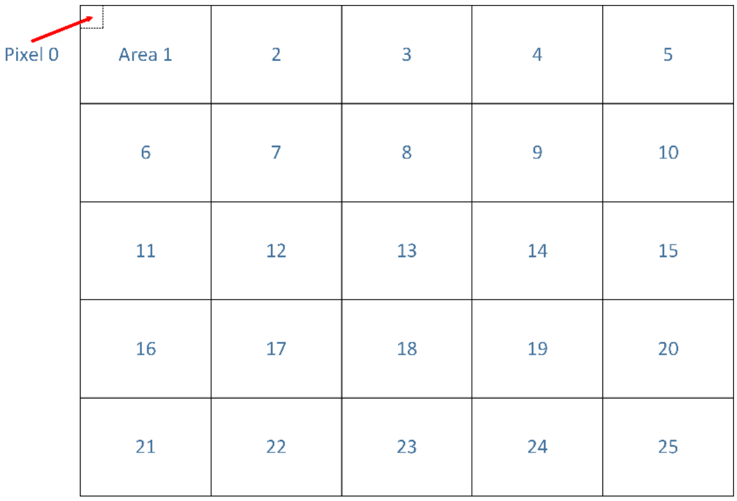
If the Auto Exposure Control is enabled the integration time will be set automatically and the register *IntegrationTime* should not be written!

4.7 Automatic Exposure Control (AEC)

The Argos3D-P220 provides an automatic exposure control feature which controls the integration time according to the currently observed amplitude data. The AEC is disabled by default and must be enabled in the register ***Mode1***.

The AEC is controlled through dedicated registers, which are listed in chapter 6.5.

The AEC algorithm also supports weighting. One may assign specific weights to each of 25 areas into which the sensor area is divided. These weights are inputs to calculate the current overall amplitude. Please see Figure 4-6: AEC weighing areas for an illustration. Each area’s weight is a Uint8 value and can range from 0% (0x0) to 100% (0xf).



Area 1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Figure 4-6: AEC weighing areas

4.8 Trigger Burst Mode

The camera can be configured to capture two images with two alternating integration times. Register ***NOF_Sequ*** (0x0120) controls the number of frames captured every burst triggered using the Manual Frame Trigger or video mode (default: 1). Register ***IntTimeSeq1*** (0x0121) controls the integration time of the second frame. It is also possible to use the Trigger Burst mode with two identical integration times.

4.9 Manual Frame Trigger

There are two types of manual trigger. To enable the manual trigger you have to disable the video mode in register ***Mode0,*** Bit[0].

4.9.1 Hardware Trigger

The camera provides an extension connector where a hardware trigger can be applied (connector pin 10). Please refer to Chapter 3.3 for interface information. Please refer to the Hardware User Manual for detailed information on the hardware trigger.

4.9.2 Software Trigger

In addition to the hardware trigger a software trigger is available. To start a frame capturing by software, set the appropriate bit (bit 4) in register ***Mode0***.

4.10 LIM Control

The internal LIM can be controlled using the ***TempDevConfig0*** register (see chapter 6.8). LIM and LIM LED segments can be deactivated using these registers.

4.11 Over Temperature Protection

The Argos3D-P220 firmware has a built-in monitoring for over-temperature condition of the LIMs. If the LIM temperature exceeds 90°C, the camera will automatically stop illumination and streaming, until temperature is below 90°C. This limit can be adjusted in register ***MaxLedTemp*** (0x0024).

4.12 Communication Keep Alive (CKA)

The communication keep alive feature should improve a stable operation in environments where a high availability of the camera services will be needed.

If the CKA feature has been enabled by writing a value >0 to the register ***CommKeepAliveTimeout*** (0x004E), the host computer must periodically write the reset value 0xCA82 to the register ***CommKeepAliveReset*** (0x004F). If the reset value will not be written within the programmed timeout the device reboots.

The timeout can be set by the register ***CommKeepAliveTimeout*** by writing the timeout value in seconds. This value will also be saved, if the register map will be saved in flash. But after a reboot the timeout check starts only after the first write of the reset value in register ***CommKeepAliveReset***.

4.13 GPIOs

The Argos3D-P220 provides 1 general-purpose input/output (connector pin 4) and 1 general-purpose input (connector pin 14). The GPIO state is mapped to register ***IOstate0*** (0x00D0) (see chapter 6.4 for details).

The general-purpose input/output can be switched from input to output using register ***IOMux1*** (0x00D3) (see chapter 6.4 for details).

Please refer to the Hardware User Manual for detailed information on the GPIOs.

4.14 Save Registers

The entire register map can be saved into the flash using the register ***CmdExec***. It will be restored from flash after a reboot or power cycle. Use this feature to save a user specific configuration.

4.15 Ethernet/IP Settings

4.15.1 MAC Address

A dedicated Ethernet MAC address from Bluetechnix MAC address pool is assigned to each Argos3D-P220 by factory default. This MAC address is saved in the OTP and cannot be changed by the user.

The user is allowed to assign a different MAC address using the registers ***Eth0Mac0*** to ***Eth0Mac2***. Be sure to make the changes persistent by saving the register map to flash using registers ***CmdEnablePasswd*** and ***CmdExec.*** Then reboot or power cycle the sensor.

If the register map in the flash is cleared, the factory default MAC address from OTP will be loaded.

4.15.2 IP/ UDP Settings

The Argos3D-P220's IP settings can be changes via the ***Eth0_**** registers. A change of the IP settings (IP address, port, subnet mask, default gateway) will take effect after a reboot. Please see the register description for details. Be sure to make the changes persistent by saving the register map to flash using registers ***CmdEnablePasswd*** and ***CmdExec.*** Then reboot or power cycle the sensor.

To change the Argos3D-P220's IP address follow these steps:

1. Convert the IP address into its hexadecimal equivalent:
   e.g.: 192.168.0.55 -> 0xC0A80037
2. Write the high word to register ***Eth0Ip1*** (0x0245) and the low word to register ***Eth0Ip0*** (0x0244).
3. Write the password 0x4877 to register ***CmdEnablePasswd*** (0x0022) to enable the ***CmdExec*** register.
4. Write 0xDD9E to register ***CmdExec*** (0x0033) to save the current register map.
5. Power cycle the Argos3D-P220.
6. Connect to the Argos3D-P220 using the new IP address.

4.16 Reset to Factory Default

The Argos3D-P220 can be reset to the factory default register settings by deleting the saved register map. This can be done by writing a dedicated value to the register ***CmdEnablePasswd*** and ***CmdExec***.

Alternatively, a factory reset is executed by setting the *nTrigger* input (connector pin 10) high for 5 seconds during boot-up. (Please consult the Hardware User Manual for details.)

4.17 Bootloader and Firmware Update

The Argos3D-P220 will be delivered with a bootloader which is capable to update the onboard firmware. The communication with the bootloader will be done using dedicated UDP command frames over the control interface connection.

Bluetechnix provides tools for updating the Argos3D-P220 firmware over Ethernet. Please refer to our support site.

Bluetechnix ToF-Suite

↳ https://support.bluetechnix.at/wiki/Argos_3D-P22x

4.17.1 Boot Sequence

After a power on or reboot the bootloader will be started. The bootloader checks if a valid firmware is installed and tries to start the firmware. If no application can be found the bootloader stays in bootloader mode waiting for incoming Ethernet connection.

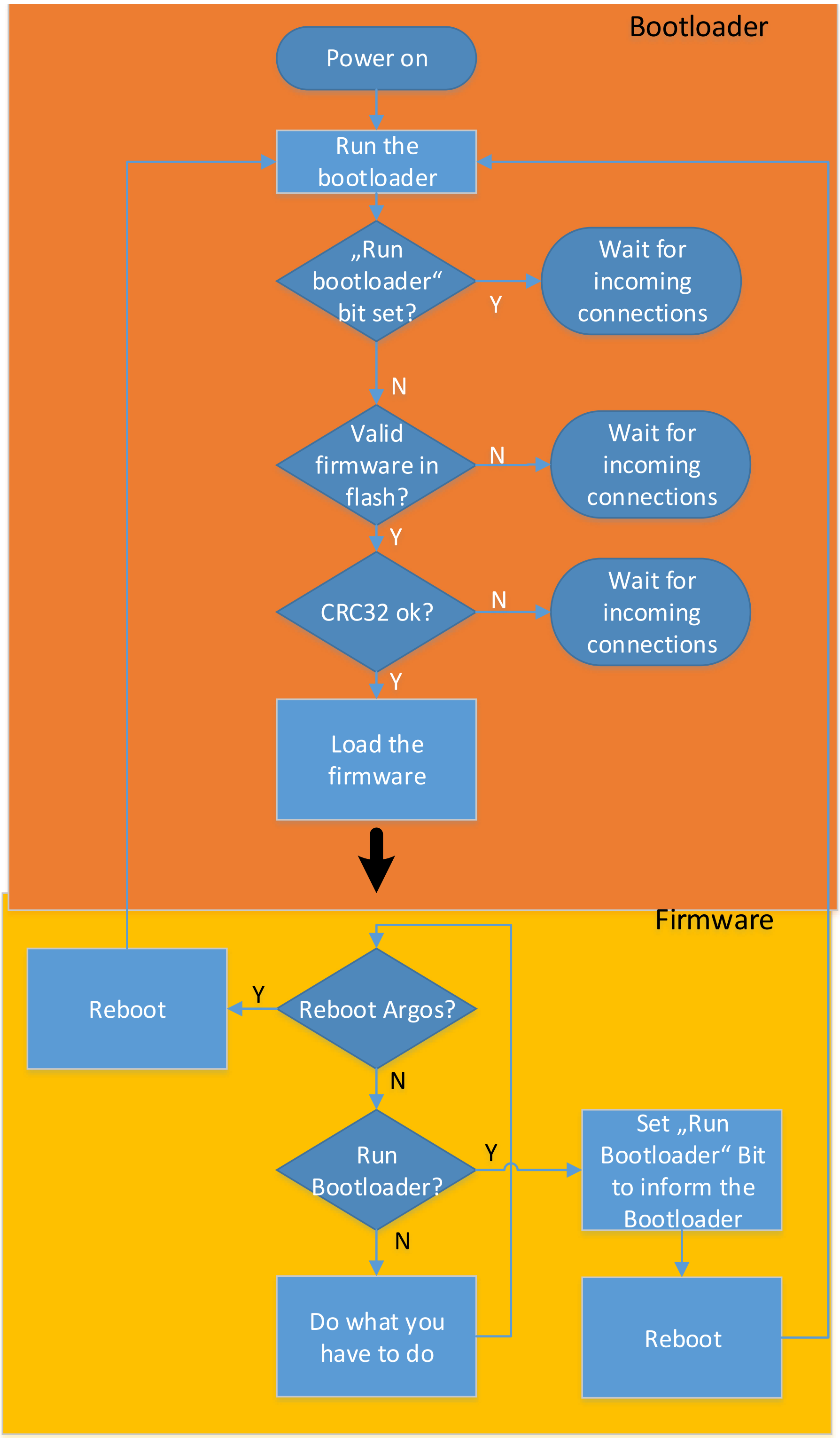


Figure 4-7: Boot sequence

4.17.2 Bootloader default settings

- **IP-Address:** 192.168.0.10
- **UDP port for the control interface:** 10003
- **MAC Address:** Factory default MAC address

4.17.3 Network Configured Bootloader

Since firmware V1.0.1 the Argos3D-P220 supports the Network Configured Bootloader Mode, which preserves the camera's network settings during the update process. The Network Configured Bootloader requires bootloader V1.0.0 and is started by setting Bit[12] of register ***Mode0*** (0x01) after writing 0x5e6b into register ***CmdEnablePasswd*** (0x0022). The bootloader will then start with the currently configured IP address, MAC address, subnet mask and gateway IP address. After updating to any firmware version supporting the Network Configured Bootloader Mode the camera will apply these network settings and will save them as a user register map.

5 Software

5.1 BltTofApi

SDK for ToF products: Bluetechnix 'Time of Flight' API

In order to create a common interface for our products we define the interfaces between a ToF device and an application. The main part of this model is the *BltTofApi* which is written in C for platform independency.

The library which provides this API for Ethernet-based devices as the Argos3D-P220 is the *BtaEthLib* (*BltTofApi* Ethernet Library).

Please visit our support Wiki to get information and to download the SDK.

Bluetechnix 'Time of Flight' API

↳ https://support.bluetechnix.at/wiki/ (Section Software)

5.2 MATLAB SDK

MATLAB SDK for ToF products: BltTofApi Matlab SDK

The MATLAB SDK is able to access the *BltTofApi* interface and will therefore be compatible with any device with an existing library implementing the *BltTofApi*.

Bluetechnix 'Time of Flight' API Matlab SDK

↳ https://support.bluetechnix.at/wiki/ (Section Software)

5.3 BltTofSuite

For the first evaluation of the camera and to evaluate different settings and configurations a .NET demo application for Microsoft Windows is provided: *BltTofSuite*. The demo application can be downloaded from our support web site.

Software and documentation

↳ https://support.bluetechnix.at/index.html

6 Register Description

!

Note

Some critical registers are password protected. To enable the functionality a specific value must be written to the ***CmdEnablePasswd*** register in advance to enable the functionality. This should prevent from accidentally executing certain functions.

6.1 General registers

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
0001	Mode0	0001	R/W	Bit[0]: 0..Manual Mode, 1.. Video Mode Bit[4]: 1..Manual Trigger (self-clearing bit) Bit[6]: 1..Clear status register Bit[8]: 1..Start Bootloader (Start Bootloader requires writing 0x5e6b into register CmdEnablePasswd (0x0022)) Bit[12]: 1..Start Bootloader with current network configuration (Start Bootloader requires writing 0x5e6b into register CmdEnablePasswd (0x0022))
0003	Status	0040	R	Bit[0]: 0..Application Mode, 1..Bootloader Mode Bit[2]: 1..Ongoing Calibration Bit[3]: 1..LED-Board temperature sensor error Bit[4]: 1..Main-Board temperature sensor error Bit[5]: 1..Calibration data missing Bit[6]: 1..Factory Regmap was loaded Bit[9]: 1..LED board over-temperature Bit[14]: 1..Base-Board temperature sensor error

0004	ImageDataFormat	0000	R/W	Bit[3:10]: 0… 2 bytes depth-data / 2 bytes amp-data 3… X/Y/Z coordinates (2 bytes in signed format for each coordinate) 4… X/Y/Z coordinates and amp-data (2 bytes in signed format for each coordinate,2 bytes unsigned for the amp value) 9… depth-data and X/Y/Z coordinates (2 byte unsigned for the depth value , 2 byte in signed format for each coordinate) 10… Optical axis coordinate (either X or Z, depending on AxisOrientation register 0x0045) and amp-data (2 bytes in signed format, 2 bytes unsigned for amp-data) 11… 4 channels Test mode: arithmetic functions with coordinates as input (2 bytes ascending index; 2 bytes constant 0xbeef; 2 bytes ascending squared index; 2 bytes constant 0x0000) 12… 2 bytes depth-data
0005	IntegrationTime	01F4	R/W	Integration Time [µs] (min: 50, max: 25000)
0006	DeviceType	795c	R	Hardware specific identification
0008	FirmwareInfo		R	Bit[0-5]: Non Functional Revision Bit[6-10]: Minor Revision Bit[11-15]: Major Revision
0009	ModulationFrequency	08ca	R/W	Modulation frequency index: 0.. 5 MHz 1.. 5.63 MHz 2.. 6.43 MHz 3.. 7.5 MHz 4.. 9 MHz 5.. 11.25 MHz 6.. 15 MHz 7.. 22.5 MHz 8.. 45 MHz
000A	Framerate	0019	R/W	Framerate [Hz]
000B	HardwareConfiguration	005A	R/W	Lens opening angle identifier
000C	SerialNumberLowWord		R	Lower 16bit of the 32bit Serial Number
000D	SerialNumberHighWord		R	Higher 16bit of the 32bit Serial Number
000E	FrameCounter		R	Frame Counter (increments on every captured frame)

000F	CalibrationCommand	0000	R/W	Bit[0:7]: Cmd code 2..Capture dist calibration image 0 3.. Capture dist calibration image 1 4.. Dist calibration calculation 13.. FPPN calibration of the current frequency 14.. Center-dist calibration calculation 16.. Clear FPPN calibration data 17.. Clear dist calibration data 18.. Clear lens calibration data 19.. Calibrate DistOffset of the current frequency Bit[9]: 1.. Output calibration result over image stream
0010	ConfidenceThresLow	012C	R/W	Amplitude threshold for valid distance data
0011	ConfidenceThresHigh	3A98	R/W	Amplitude threshold for valid distance data
0019	Mode1	0800	R/W	Bit[3]: 0..AEC Off, 1..AEC On
001A	CalculationTime		R	Calculation time for the last frame in 10[µs]. The inverse of this value shows the maximum achievable frame rate based on the CPU load.
001B	LedboardTemp		R	Temperature of LED-Board in 0,01[°C] (FFFF: Sensor not available).
001C	MainboardTemp		R	Temperature of Main-Board in 0,01[°C] (FFFF: Sensor not available).
001D	LinearizationAmplitude	0000	R/W	Amplitude for Linearization Function [float value x 10000]
001E	LinearizationPhaseShift	0000	R/W	Amplitude for Linearization Function [float value x 10000]
001F	FrameTime		R	Time between the last two frames. In 0,1[ms]
0020	RealWorldXcoordinate	0000	R/W	Distance to the calibration target [mm].
0021	CalibrationExtended	0000	R	Bit[0-7]: Status/error 0.. Idle 2.. Dist calib exposure 3.. Dist calib capturing img 4.. Dist calib saving img to flash 5.. Dist calib loading img from flash 6.. Dist calib calculation/saving result 17.. CenterDist calib loading img from flash 18.. CenterDist calib calculation/saving result 19.. FPPN calibration 20.. Erasing flash 161.. Operation done 255.. Generic error 254.. NVM error 252.. Out of memory 249.. Led board failure 248.. Invalid modulation frequency 246.. Wrong image mode (Need depth) Bit[10]: 1..Error occurred Bit[12]: 1..No FPPN Calibration data in NVM Bit[13]: 1..No Dist Calibration data in NVM Bit[14]: 1..No Lens Calibration data in NVM Bit[15]: 1..Temperature compensation error

0022	CmdEnablePasswd	0000	R/W	Set a password for critical operations: 0x4877: Register map flash operations (register CmdExec 0x0033) 0x5e6b: Test commands (register TestConfig 0x01c0)
0024	MaxLedTemp	2328	R/W	Maximum tolerable LED-Board temperature 0,01[°C]
0026	HorizontalFov	2)	R	Horizontal field of view in 0,01[°]
0027	VerticalFov	2)	R	Vertical field of view in 0,01[°]
002B	TriggerDelay	0000	R/W	Delay between trigger assertion (either software or hardware) and image capturing [ms]
002C	BootloaderStatus	4000	R	Bit[14-15]: Firmware Load Counter. This counter is reset by the firmware. It counts the boot attempts. In Bootloader mode it is used to detect a firmware load problem
002D	TempCompGradientLim		R/W	Factor ‘c’ of the illumination temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
002E	ApplicationVersion		R	See “FirmwareInfo (0x0008)” for bit description, in Bootloader mode this register contains the firmware info of the flashed application
002F	DistCalibGradient	4000	R/W	Gradient of dist value, interpreted as fixed comma shifted by 14 binary digits
0030	TempCompGradient2Lim		R/W	Factor ‘b’ of the illumination temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
0032	CPLDversion		R	Version of the firmware on CPLD. Bit description: see “FirmwareInfoReg”
0033	CmdExec	0000	R/W	Initiate an operation: Executing the following commands must be preceded by writing 0x4877 into register CmdEnablePasswd (0x0022): 0xC2AE.. Clear UserRegMap in flash 0x9E20.. Read UserRegMap from flash 0x909A.. Read FactoryRegMap 0xDD9E.. Write UserRegMap to flash
0034	CmdExecResult	0000	R	Result code of the operation initiated using CmdExec 1.. Success Other.. Error
0035	FactoryMacAddr2		R	Highest and second highest byte of the MAC address stored in OTP flash
0036	FactoryMacAddr1		R	Byte 3 and 2 of the MAC address stored in OTP flash
0037	FactoryMacAddr0		R	Byte 1 and lowbyte of the MAC address stored in OTP flash
0038	FactoryYear		R	Production year (stored in OTP flash)
0039	FactoryMonthDay		R	Bit[0-7]: Production day (stored in OTP flash) Bit[8-15]: Production month (stored in OTP flash)
003A	FactoryHourMinute		R	Bit[0-7]: Production hour (stored in OTP flash) Bit[8-15]: Production minute (stored in OTP flash)

003B	FactoryTimezone		R	Production timezone (stored in OTP flash)
003C	TempCompGradient3Lim		R/W	Factor ‘a’ of the illumination temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
003D	BuildYearMonth		R	Build date/time Bit[14-4]: Year Bit[3-0]: Month
003E	BuildDayHour		R	Build date/time Bit[9-5]: Day Bit[4-0]: Hour
003F	BuildMinuteSecond		R	Build date/time Bit[11-6]: Minute Bit[5-0]: Second
0040	UpTimeLow		R	Lower 16 bit of uptime in [s]
0041	UpTimeHigh		R	Higher 16 bit of uptime in [s]
0042	AkfPlausibilityCheckAmp Limit	0032	R/W	Limit for the akf plausibility check
0043	TimSerialLow		R	Serial Number of the TIM module, low word
0044	TimSerialHigh		R	Serial Number of the TIM module, high word
004A	TempCompGradientTim		R/W	Factor ‘c’ of the ToF sensor temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
004B	TempCompGradient2Tim		R/W	Factor ‘b’ of the ToF sensor temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
004C	TempCompGradient3Tim		R/W	Factor ‘a’ of the ToF sensor temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
004E	CommKeepAliveTimeot		R/W	Communication keepalive timeout [s] After this timeout, a watchdog reset occurs if the timeout is not reset by writing the reset value to the CommKeepAliveReset register
004F	CommKeepAliveReset		R/W	Communication keepalive write register Resets the CommKeepAlive timeout when the value 0xCA82 is written

Table 6-1: General register

Note 1): The number of median iterations may have an impact on the achievable frame rate. The frame rate may decrease on incrementing this register.

Note 2): The content depends on the mounted lens and the calibration data and represents the real viewing angles.

6.2 More General Registers

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
00C1	DistOffset0	1)	R/W	An offset for distance values when operating at modulation frequency with index 0
00C2	DistOffset1	1)	R/W	An offset for distance values when operating at modulation frequency with index 1
00C3	DistOffset2	1)	R/W	An offset for distance values when operating at modulation frequency with index 2

00C4	DistOffset3	1)	R/W	An offset for distance values when operating at modulation frequency with index 3
00C5	DistOffset4	1)	R/W	An offset for distance values when operating at modulation frequency with index 4
00C6	DistOffset5	1)	R/W	An offset for distance values when operating at modulation frequency with index 5
00C7	DistOffset6	1)	R/W	An offset for distance values when operating at modulation frequency with index 6
00C8	DistOffset7	1)	R/W	An offset for distance values when operating at modulation frequency with index 7
00C9	DistOffset8	1)	R/W	An offset for distance values when operating at modulation frequency with index 8
010A	TempCompGradient Baseboard		R/W	Factor 'c' of the ToF baseboard temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
010B	TempCompGradient 2Baseboard		R/W	Factor 'b' of the ToF baseboard temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
010C	TempCompGradient 3Baseboard		R/W	Factor 'a' of the ToF baseboard temperature compensation function: y [mm] = a/100000 * x^3 + b/10000 * x^2 + c/1000 * x + u
010D	BaseboardTemp		R	Temperature of baseboard in 0,01[°C] (FFFF: Sensor not available).
010E	PWM50Temp	0FA0	R/W	Temperature for PWM control in 0,01[°C]. Creates a PWM output with duty cycle of 50%
010F	PWM100Temp	1B58	R/W	Temperature for PWM control in 0,01[°C]. Creates a PWM output with duty cycle of 100%
0110	IllPreheatingTime	0064	R/W	Time for illumination pre heating in µs
011C	TriggerConfig	0000	R/W	Bit[0]: 1 … TriggerOut disabled Bit[1]: 0 … Transition to high: capture starts Transition to low: capture ended 1 … Transition to low: capture starts Transition to high: capture ended

Table 6-2: General registers

Note 5): This value varies from unit to unit.

6.3 Registers for Sequencing

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
0120	NofSequ	1	R/W	Number of images that are recorded without wait time in between
0121	IntTimeSeq1	500	R/W	Integration time to be used for capturing sequence 1

6.4 Registers for GPIOs

Addr (hex)	Register Name	Default Value (hex)	R/W	Description

00D0	IOstate0	0000	R/W	Bit[0]: … state of IN_0 (only R) Bit[1]: … state of IN_1 (only R) Bit[8]: … state of OUT_0 (R/W)
00D2	IOMux0	0000	R/W	Bit[0-1]: 0 … IN_0 is input Bit[2-3]: 0 … IN_1 function
00D3	IOMux1	0000	R/W	Bit[0-1]: 0 … OUT_0 is output 1 … OUT_0 is input

Table 6-3: Registers for GPIOs

6.5 Registers for Automatic Exposure Control

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
01A9	AecAvgWeight0	4444	R/W	Bit[15-12]: Weight for average, area 1 Bit[11-8]: Weight for average, area 2 Bit[7-4]: Weight for average, area 3 Bit[3-0]: Weight for average, area 4
01AA	AecAvgWeight1	44CC	R/W	Bit[15-12]: Weight for average, area 5 Bit[11-8]: Weight for average, area 6 Bit[7-4]: Weight for average, area 7 Bit[3-0]: Weight for average, area 8
01AB	AecAvgWeight2	C44C	R/W	Bit[15-12]: Weight for average, area 9 Bit[11-8]: Weight for average, area 10 Bit[7-4]: Weight for average, area 11 Bit[3-0]: Weight for average, area 12
01AC	AecAvgWeight3	FC44	R/W	Bit[15-12]: Weight for average, area 13 Bit[11-8]: Weight for average, area 14 Bit[7-4]: Weight for average, area 15 Bit[3-0]: Weight for average, area 16
01AD	AecAvgWeight4	CCC4	R/W	Bit[15-12]: Weight for average, area 17 Bit[11-8]: Weight for average, area 18 Bit[7-4]: Weight for average, area 19 Bit[3-0]: Weight for average, area 20
01AE	AecAvgWeight5	4444	R/W	Bit[15-12]: Weight for average, area 21 Bit[11-8]: Weight for average, area 22 Bit[7-4]: Weight for average, area 23 Bit[3-0]: Weight for average, area 24
01AF	AecAvgWeight6	4000	R/W	Bit[15-12]: Weight for average, area 25
01B0	AecAmpTarget	02BC	R/W	Auto exposure target amplitude value to which the controller is controlling to
01B1	AecTintStepMax	0021	R/W	Auto exposure maximum change of integration time percentage. The relative change of the integration time will be lower than this percentage
01B2	AecTintMax	2710	R/W	Auto exposure maximum integration time the controller calculates
01B3	AecKp	0028	R/W	Proportional part of the auto exposure controller in percent
01B4	AecKi	000F	R/W	Integral part of the auto exposure controller in percent
01B5	AecKd	0000	R/W	Differential part of the auto exposure controller in percent

Table 6-4: Registers for automatic exposure control

6.6 Registers for Filter Configuration

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
01E0	ImgProcConfig	7bc1	R/W	Bit[0]: 1… enable Median Filter Bit[1]: 1… enable Average Filter Bit[2]: 1… enable Gauss Filter Bit[3]: 1… enable Bilateral Filter Bit[4]: 1… enable Sliding Average Bit[6]: 1… enable wiggling compensation Bit[7]: 1… enable FPPN compensation Bit[8]: 1… enable ModFreq scaling Bit[9]: 1… enable scaling to [mm] Bit[11]: 1… enable temperature compensation Bit[12]: 1… enable scaling via register DistCalibGradient (0x002F) Bit[13]: 1… enable offsets via registers DistCalibOffsetX (0x00C1 onwards) Bit[14]: 1… enable akf plausibility check (affected pixel have a distance of 1)
01E1	FilterMedianConfig	0001	R/W	Bit[0-7]: … Nr of Median Iterations
01E2	FilterAverageConfig	0100	R/W	Bit[0-7]: 0… 3x3 Pixel 1… 5x5 Pixel Bit[8-15]: Nr of iterations
01E3	FilterGaussConfig	0100	R/W	Bit[0-7]: 0… 3x3 Pixel 1… 5x5 Pixel Bit[8-15]: Nr of iterations
01E4	FilterBilateralConfig	2082	R/W	Bit[0-5]: Sigma R (factor for weighing for radius, max: 6) Bit[6-11]: Sigma D (factor for weighting for data, max: 6) Bit[12-15]: Nr of iterations
01E5	FilterSlafConfig	0005	R/W	Bit[0-7]: …Window size

Table 6-5: Register for filter configuration

6.7 Registers for Ethernet configuration

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
0240	Eth0Config	0006	R/W	Bit[1]: 1.. Enable UDP streaming Bit[2]: 1.. Ignore CRC for UDP streaming
0241	Eth0Mac2	ACDE	R/W	Low byte and byte 1 of MAC address (default value differs in factory config)
0242	Eth0Mac1	4801	R/W	Byte 2 and byte 3 of MAC address (default value differs in factory config)
0243	Eth0Mac0	0203	R/W	Byte 4 and high byte of MAC address (default value differs in factory config)
0244	Eth0Ip0	000A	R/W	Low word of IP address

0245	Eth0Ip1	C0A8	R/W	High word of IP address
0246	Eth0Snm0	FF00	R/W	Low word of subnet mask
0247	Eth0Snm1	FFFF	R/W	High word of subnet mask
0248	Eth0Gateway0	0000	R/W	Low word of gateway
0249	Eth0Gateway1	0000	R/W	High word of gateway
024C	Eth0UdpStreamIp0	0001	R/W	Low word of IP address for UDP stream
024D	Eth0UdpStreamIp1	E000	R/W	High word of IP address for UDP stream
024E	Eth0UdpStreamPort	2712	R/W	Port for UDP streaming
0255	Eth0UdpConfigPort	2713	R/W	UDP port for UDP Control Interface
0255	Eth0UdpConfigPort	2713	R/W	UDP port for UDP Control Interface

Table 6-6: Registers for Ethernet configuration

6.8 Registers for Temperature Management

Addr (hex)	Register Name	Default Value (hex)	R/W	Description
028F	TempDevMaxIllTempOffset	000f	R/W	Bit[0-15]: … Temperature offset in °C for illumination temperature devices. If supported by the device the temperature threshold for security shutdown is set by using register MaxLedTemp (0x0024) added by this offset, in 0,01[°C]
0292	TempDevConfig0	0007	R/W	Device specific configuration of temperature device 0 LIM: Bit[0]: 1 … enable LIM Bit[1]: 1 … enable LED Segment 1 Bit[2]: 1 … enable LED Segment 2 Bit[5]: 1 … Fan manually on, 0 … Fan auto mode Bit[6-14]: reserved Bit[15]: is set by the firmware when start of configuration of the temperature device and cleared as soon as the configuration was successful.
0293	TempDevSysStatus0		R/W	System Status of temperature device 0 Bit[0]: … Device specific status has error bits set Bit[15]: … Initialization Error
02D0	TempDevTemperature0		R	Temperature of LIM, see LedboardTemp (0x001B)
02D1	TempDevTemperature1		R	Temperature of baseboard, see BaseboardTemp (0x010D)
02D2	TempDevTemperature2		R	Temperature of sensor board, see MainboardTemp (0x001C)
02E0	TempDevStatus0	0000	R	LIM temperature sensor status
02E1	TempDevStatus1	0000	R	Baseboard temperature sensor status
02E2	TempDevStatus2	0000	R	Sensor board temperature sensor status

Table 6-7: 6.8 Registers for Temperature Management

7 Support

7.1 General Support

General support for products can be found at Bluetechnix' support site

Support Link

↳ https://support.bluetechnix.com/index.html

7.2 Software Packages

Software packages and software downloads are for registered customers only

Software Package

↳ https://support.bluetechnix.com/index.html

8 Firmware History

8.1 Version Information

Please refer to our support site for additional information about firmware changes.

8.2 Anomalies

Please refer to our support site for additional information about firmware changes.

9 Document Revision History

Version	Date	Author	Description
1	2018-09-05	MKO	Initial Version

Table 9-1: Document Revision History