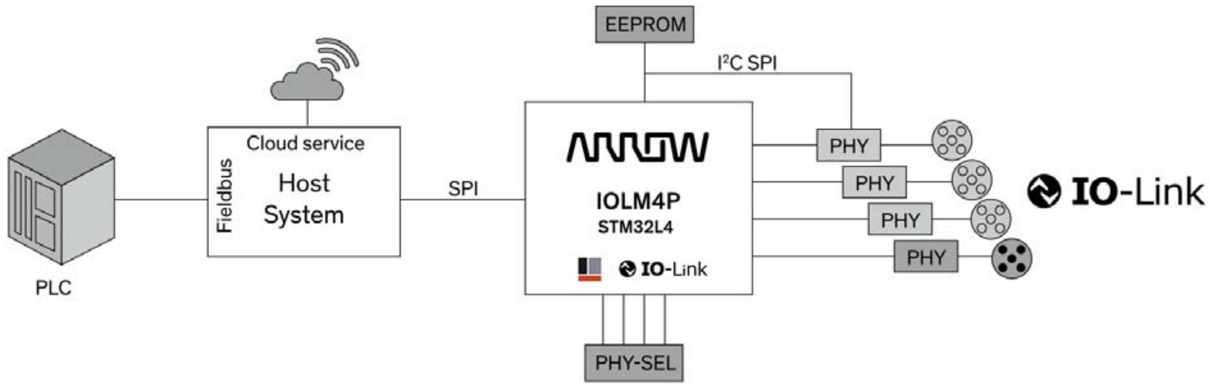
IOLM4P - Overview

The IOLM4P is based on the STM32L431 [3] and contains a full featured IO-Link Master stack that handles the IO-Link communication with up to four independent IO-Link ports.The IOLM4P is connected with a host controller that controls the IO-Link Master on the IOLM4P. The controlling function is based on a well defined simple telegram structure and uses the SPI-slave interface of the IOLM4P. A basic overview about the application of the IOLM4P is given below.

.



Feature List

- IO-Link compliant tested by approved IO-Link Master test system
- Currently supported transceiver chips (PHY's)
  - L6360 (ST)
  - LTC2784 (Analog Devices)
  - *MAX14819 (Maxim)*[1]
  - *CCE4510 (Creative Chips)*
  - *CM3010 (NXP)*
- Number of ports selectable (1 - 4)
- All IO-Link COM-speeds supported
  COM1:4.8kBit/s
  COM2:38.4kBit/s
  COM3:230.4kBit/s
- SIO-Mode supported based on capabilities of connected PHY
- Data storage support (requires external EEPROM > 8kByte)
- Crystal 14.7456MHz
- IOLM4P chip control via SPI
- Control and data exchange telegrams as defined in the IO-Link Test Specification [6, A4.4]
- PHY independent control API
- Cycle times of the 4 IO-Link Ports can be configured independently
- Shortest cycle time <= 400µs
- LQFP 64 package

1. *in preparation*

Table of Contents

	IOLM4P - Overview	1
	Feature List	1
	Table of Contents	2
1.0	References:	4
1.1	General Information	4
1.2	Evaluation Kits	4
1.3	Master Access Software-Support	4
1.4	Pinning	6
2.0	Functional Overview	10
2.1	PHY selection	10
2.2	Shared PHY signals	10
2.3	Dedicated PHY signals	11
2.4	PHY dependent connectivity	11
2.4.1	PHY-L6360	11
2.4.2	PHY-LTC2874	12
3.0	Control Interface	13
3.1	Definitions and Acronyms	13
3.2	Host-SPI Interface	13
3.2.1	Host-SPI Timing	14
4.0	Conventions for Telegram Description	15
4.1	Notational Conventions	15
4.1.1	Telegram Checksum	15
4.1.2	Element Types and Values	15
4.1.3	Data Types for Elements	16
4.1.4	Values for Elements	16
4.2	STI Handshake Convention	16
4.3	STI Communication Disconnection	16
4.4	STI ENDIANESS	16
4.5	Process Data Stream Representation	16
4.6	I/Q Configuration	17
4.7	General Status Codes	17
4.8	ISDU Codes	18
4.9	Port Configuration Codes	18
4.10	Data Storage Codes	19
5.0	Standard Telegrams	20
5.1	Get Identification E_FNCID_GETINFO - 0x00	21
5.2	Set PD Interface E_FNCID_SETPDI - 0x01	22
5.3	Get PD Interface E_FNCID_GETPDI 0x02	23
5.4	Control Status E_FNCID_CNTSTA - 0x03	23
5.5	Set Port Configuration E_FNCID_SETPORTCNFPAR0x04	25
5.6	Get Port Configuration E_FNCID_GETPORTCNFPAR0x05	27
5.7	Set Data Storage Header E_FNCID_SETDSHDR 0x06	28
5.8	Get Data Storage Header E_FNCID_GETDSHDR 0x07	28
5.9	Upload DS to Master E_FNCID_UPLDS 0x08	29

5.10 Download DS from Master E_FNCID_DWNDS 0x09 30
5.11 ISDU Transfer Initialization E_FNCID_ISDUREQ 0x0A 30
5.12 ISDU Get E_FNCID_ISDUGET 0x0B 31
5.13 Read Event E_FNCID_RDEVENT (0xC) 32
5.14 Switch OFF Power E_FNCID_SWITCHOFF 0x0D 33
5.15 Switch ON Power E_FNCID_SWITCHON 0x0E 34
5.16 Upload DS Partially E_FNCID_UPLDS_SHORT 0x018 34
5.17 Download DS Partially E_FNCID_DWNDS_SHORT 0x019 35
6.0 Extended Telegrams 37
6.1 Firmware Update Mode E_FNCID_FIRMWAREUPDATEMODE 0x10... 37
6.2 Write To Flash E_FNCID_WRITEFLASH 0x011 38
6.3 Check Flash Write E_FNCID_CHECKWRITE FLASH 0x012 38
6.4 Restart E_FNCID_RESTART 0x013 39
6.5 Set Secondary Key E_FNCID_MODIFYFWKEY 0x016 39
6.5.1 Set PD Loopback E_FNCID_SETPROCESS DATALOOPBACKMODE 0x061 40
6.5.2 Get PD Loopback E_FNCID_SETPROCESS DATALOOPBACKMODE 0x061 40
Appendix A 42
Appendix B 43
Appendix C 44
Revision History 45
List of Tables 46
List of Figures 48
Important Notice 51

1.0 References:

[1] IEC 61131-9, Programmable controllers - Part 9: Single-drop digital communication interface for small sensors and actuators (SDCI)

[2] IO-Link Interface and System Specification Version 1.1.2, July 2013.

[3] STMicroelectronics, STM32L431xx, Ultra-low-power Arm® Cortex®-M4 32-bit MCU+FPU, 100DMIPS,up to 256KB Flash, 64KB SRAM, analog, audio

[4] STMicroelectronics, L6360, IO-Link communication master transceiver IC

[5] Analog Devices, LTC2874, Quad IO-Link Master Hot Swap Controller and PHY

[6] IO-Link community: IO-Link Test Specification, Version 1.1.2, July 2014

[7]TEConcept GmbH: IO-Link Master Access

1.1 General Information

The the IOLM4P is based on the MCU STM32L431RB. For electrical characteristics please refer to [3].Currently two PHY's are supported. For details refer to [4] and [5].

1.2 Evaluation Kits

The chips are supported by the following evaluation kits from Arrow: :

Table 1: Evaluation Kits

Kit	Supported Component	Comment
ARW-IOLM4P-EVAL01	IOLM4P	Motherboard with USB and SPI connectivity
ARW-IOLM4P-EVAL02	L6360	Daughter-board supporting 4 IO-Link ports
ARW-IOLM4P-EVAL03	LTC2874	Daughter-board supporting 4 IO-Link ports

1.3 Master Access Software-Support

In order to simplify usage of the IOLM4P, a software package called „IO-LinkMasterAccess_C“ (see [7]) is provided that offers an application programming interface. The software is made available in form of C-sources and is designed for easy porting to a variety of platforms. The software is partially bundled with the chip, some high level access functions, BLOB Transfer and FW-Update need to be ordered separately.

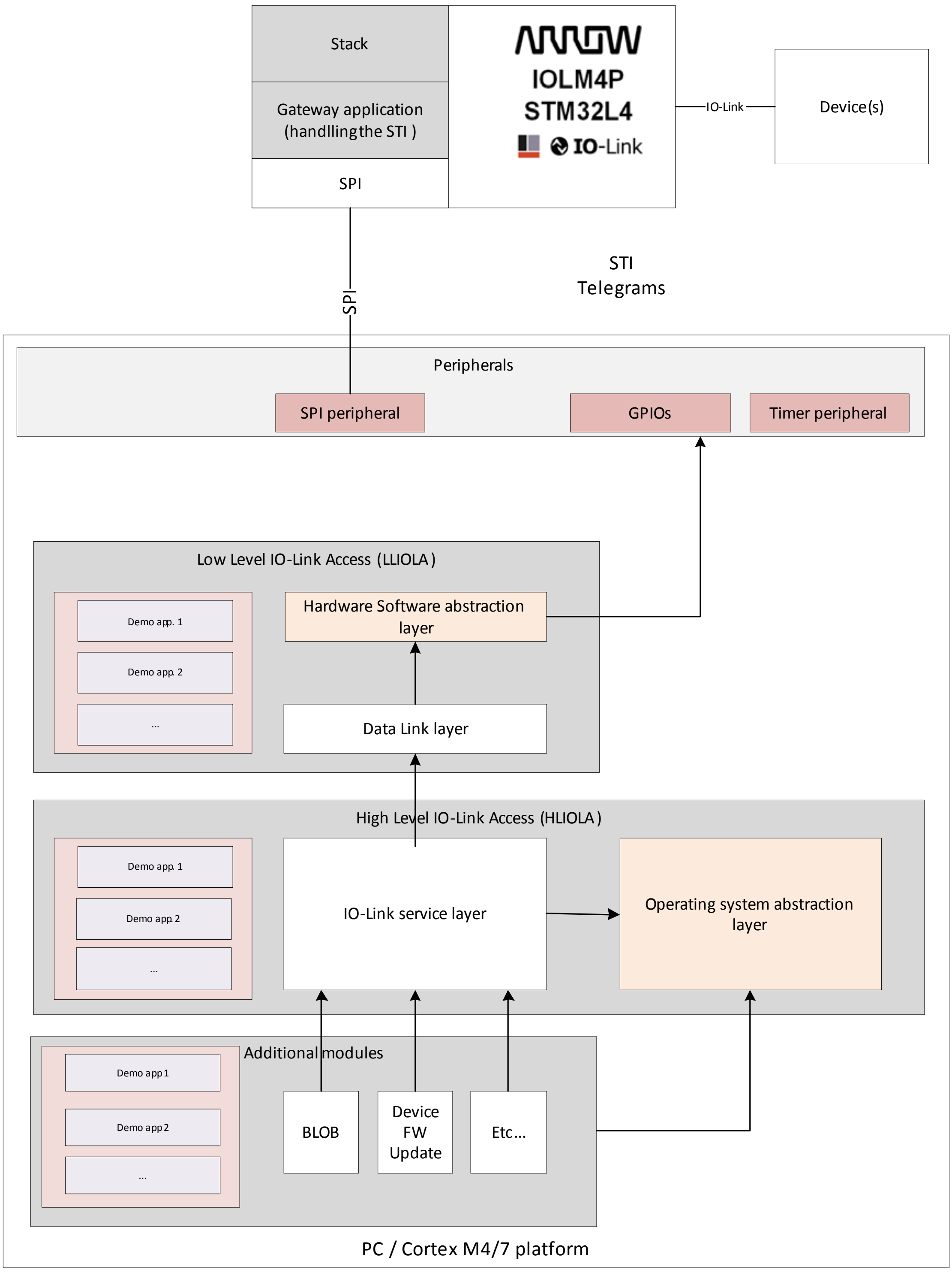


Figure 1. Connectivity Software Support for IOLM4P

1.4 Pinning

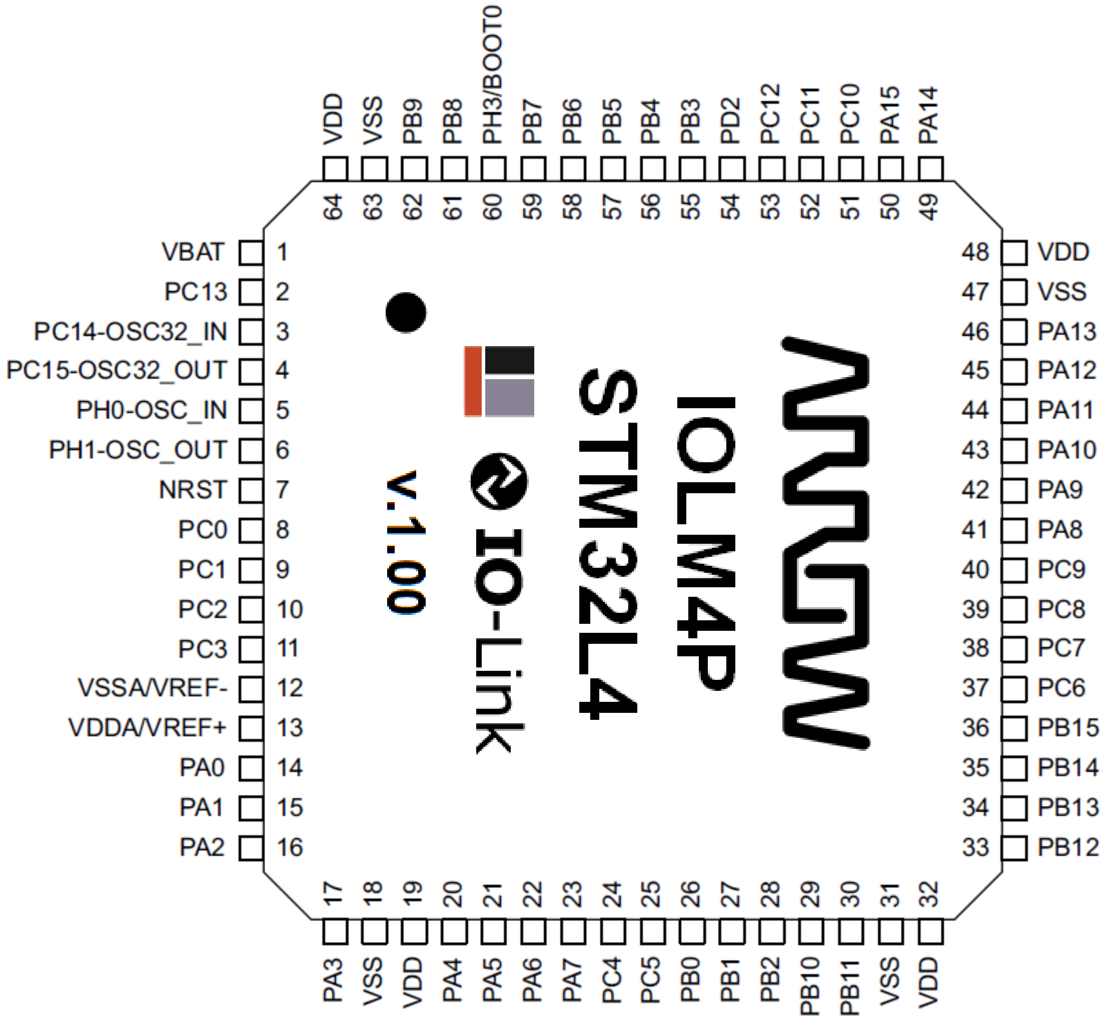


Figure 2. IOLM4P package

Table 2: Pin Description[a]

Pin No.	Type	Connect if not used	Name	Short Description
1	Supply	VDD	VBAT	Power Supply
2	IO	NC	4A	PHY A, Signal 4
3	IO	NC	4B	PHY B, Signal 4
4	IO	NC	4C	PHY C, Signal 4
5	I	X	OSC-IN	14.7456 MHz crystal input

Table 2: Pin Description[a]

Pin No.	Type	Connect if not used	Name	Short Description
6	I	X	OSC-OUT	14.7456 MHz crystal output
7	I	X	NRST	Reset Input (low active)
8	I	X	UART_RX_D	Receives UART data sent by PHY-D
9	O	X	UART_TX_D PHY_ID3	Sends UART data to PHY-D PHY identification on startup (Bit 3)
10	IO	X	1A/MISO	PHY A, Signal 1 or MISO line
11	IO	X	1B/MOSI	PHY B, Signal 1 or MOSI line
12	Supply	X	VSSA	Analog ground
13	Supply	X	VDDA	Supply voltage
14	I	VDD	WKUP	Not used
15	I	X	SPI1_SCK	SPI clock for SPI slave from host controller
16	O	X	UART_TX_B PHY_ID1	Sends UART data to PHY-B PHY identification on startup (Bit 1)
17	I	X	UART_RX_B	Receives UART data from PHY-B
18	Supply	VSS	VSS_4	Digital ground
19	Supply	VDD	VDD_4	Digital supply voltage
20	I	X	SPI1_NSS	SPI chip select from SPI host
21	I	VDD	USB_IRQN	Interrupt from FT2232
22	O	X	SPI1_MISO	SPI output to SPI host Master
23	I	VSS	SPI1_MOSI	SPI input for data from SPI host Master
24	O	X	UART_TX_C PHY_ID2	Sends UART data to PHY-C PHY identification on startup (Bit 2)
25	I	X	UART_RX_C	Receives UART data from PHY-C
26	O	X	EN_TXA	Enable TX for PHY A
27	O	X	EN_TXB	Enable TX for PHY B
28	I	VSS	Boot1	Not supported Boot Mode
29	O	X	1C/SCK	PHY C, Signal 1 or SCK to PHY board
30	IO	X	I2C2_SDA	I2C2 SCA routed to PHY board
31	I	VSS	VCAP_1	Connect to GND
32	Supply	VDD	VDD_1	Digital supply voltage
33	I	VDD	1D/NSS	PHY D, Signal 1 or NSS to PHY board
34	IO	LV	I2C2_SCL	I2C2 SCL routed to PHY board

Table 2: Pin Description[a]

Pin No.	Type	Connect if not used	Name	Short Description
35	IO	LV	TP1	Do not use
36	IO	LV	TP2	Do not use
37	I	VDD	IRQA	Interrupt line from PHY A
38	I	VDD	IRQB	Interrupt line from PHY B
39	I	VDD	IRQC	Interrupt line from PHY C
40	I	VDD	IRQD	Interrupt line from PHY D
41	IO	LV	2A	PHY A, Signal 2
42	O	LV	UART_TX_A PHY_ID0	Sends UART data to PHY A PHY identification on startup (Bit 0)
43	I	VSS	UART_RX_A	Receives UART data from PHY A
44	IO	LV	TP11	Do not use
45	I	VSS	SYNC	Sync signal as extension to SPI1 (reserved)
46	IO	LV	SWDIO	JTMS (Debug interface)
47	IO	VSS	VCAP_2	
48	Supply	VDD	VDD_2	
49	IO	LV	SWCLK	JTCK (Debug interface)
50	IO	LV	3A SPI3_NSS	PHY A, Signal 3
51	IO	LV	3B SPI3_SCK	PHY B, Signal 3 (alternative SPI3_SCK)
52	IO	LV	3C SPI3_MISO	PHY C, Signal 3 (alternative SPI3_MISO)
53	IO	LV	3D SPI3_MOSI	PHY D, Signal 3 (alternative SPI3_MOSI)
54	IO	LV	4D	PHY D, Signal 4
55	IO	LV	2B	PHY B, Signal 2
56	IO	LV	2C	PHY C; Signal 2
57	IO	LV	2D	PHY D, Signal 2
58	IO	LV	I2C1_SCL	Clock for I2C1
59	IO	LV	I2C1_SDA	Data for I2C1
60	I	GND	BOOT0	
61	O	LV	EN_TXC	TX enable for PHY C
62	O	LV	EN_TXD	TX enable for PHY D

Table 2: Pin Description[a]

Pin No.	Type	Connect if not used	Name	Short Description
63	Supply	GND	VSS_3	Digital GND
64	Supply	VDD	VDD_3	Digital Supply Voltage

a. Legend: NC = leave pin unconnected, X: mandatory, connect pin as described, LV: leave vacant, VDD: power supply, VSS: ground level

2.0 Functional Overview

The IOLM4P can handle up to 4 IO-Link ports. In addition to the IOLM4P IO-Link transceiver chips or PHY's are required to match the physical characteristics of IO-Link signal. Currently 2 different transceiver chips are supported and 3 more are in preparation[1]. As the connectivity of the PHY's varies considerably, certain pins of the IOLM4P are have different functionality depending on the connected PHY.

2.1 PHY selection

The IOLM4P checks on startup the level of the PHY_ID pins. The startup level shall be specified by an external 10k? pull-up or pull-down resistor as the pin used for PHY selection is reused as UART_TX output signal.

1. *The number of supported IO-Link transceiver chips may be extended in later versions of the IOLM4P.*

Table 3: PHY Selection on Startup

PHY	PHY_ID3	PHY_ID2	PHY_ID1	PHY_ID0
L6360	VDD_(1)	VDD_(1)	VDD_(1)	VDD_(1)
MAX14819[a]	VDD_(1)	VDD_(1)	VDD_(1)	VSS_(0)
LTC2874	VDD_(1)	VDD_(1)	VSS_(0)	VDD_(1)
CCE4510[a]	VDD_(1)	VDD_(1)	VSS_(0)	VSS_(0)
CM3010[a]	VDD_(1)	VSS_(0)	VDD_(1)	VDD_(1)
Reserved	VDD_(1)	VSS_(0)	VDD_(1)	VSS_(0)
Reserved	VDD_(1)	VSS_(0)	VSS_(0)	VDD_(1)
Reserved	VDD_(1)	VSS_(0)	VSS_(0)	VSS_(0)
Reserved	VSS_(0)	VDD_(1)	VDD_(1)	VDD_(1)
Reserved	VSS_(0)	VDD_(1)	VDD_(1)	VSS_(0)
Reserved	VSS_(0)	VDD_(1)	VSS_(0)	VDD_(1)
Reserved	VSS_(0)	VDD_(1)	VSS_(0)	VSS_(0)
Reserved	VSS_(0)	VSS_(0)	VDD_(1)	VDD_(1)
Reserved	VSS_(0)	VSS_(0)	VDD_(1)	VSS_(0)
Reserved	VSS_(0)	VSS_(0)	VSS_(0)	VDD_(1)
Reserved	VSS_(0)	VSS_(0)	VSS_(0)	VSS_(0)

a. in preparation

2.2 Shared PHY signals

PHY's that provide serial SPI or I2C bus connections can share the same serial interface signal of the IOLM4P. Additional addressing is required if interfaces are shared. In case of I2C, different I2C-addresses are used, in case of SPI, an additional chip select signal (NSS) might be used.

Table 4: Shared PHY Signals

Pin	Signal	Comment
1A	SPI2_MISO	For PHY boards that require SPI control, signals with a name starting with „1" might not be available.
1B	SPI2_MOSI	
1C	SPI2_SCK	
1D	SPI2_NSS	
3A	SPI3_NSS	For PHY boards that require SPI control interfaces, signals with a name starting with „3" might not be available.
3B	SPI3_MISO	
3C	SPI3_MOSI	
3D	SPI3_SCK	
I2C_SDA	SDA	Connected to all PHY ports. Additional addressing required
I2C_SCL	SCL	

2.3 Dedicated PHY signals

The IOLM4P can support up to four PHY's. The PHY's are referenced by „PHYA", „PHYB", „PHYC" and „PHYD". As placeholder for the PHY identifier „A", „B", „C" or „D" the symbol „*" is used in the text below. For every PHY, a number of Pins is reserved.

Table 5: Dedicated PHY Signals

Pin	Signal	Comment
UART_RX_*	UART input of IOLM4P	Connect to serial output of PHY
UART_TX_*	UART output of IOLM4P	Connect ot serial input of PHY. Used as PHY selection on startup
EN_TX*	Enables output buffer of connected PHY	
IRQ*	IRQ of PHY	Connect to IRQ-output of PHY

Table 5: Dedicated PHY Signals

Pin	Signal	Comment
1*	Signal 1	PHY dependent function
2*	Signal 2	
3*	Signal 3	
4*	Signal 4	

2.4 PHY dependent connectivity

A variety of IO-Link transceiver chips, also called PHY's can be connected to the IOLM4P.The following sections describe in detail, how PHY-signals, identified by pin names of the PHY, are connected to the IOLM4P signals. For every of the up to four connected IO-Link Ports (A...D), four dedicated (UART_RX, UART_TX, EN_TX, IRQ) and 4 generic signals (1...4) are used.

2.4.1 PHY-L6360

For reference see data sheet [4] and https://www.st.com/en/power-management/l6360.html.

The L6360 implements driving and sensing circuits for one IO-Link port including a L+ power switch, an additional digital input (I/Q) interface with configurable digital filter, embedded driver for 2 status LEDs and a linear voltage regulator. An access to the extensive chip configuration is done through the integrated I2C interface. For large capacitive loads, loads with high inrush current / high power-up energy consumption (e.g. as described in the IO-Link specification corrigendum), an additional external power switch is implemented on the PHY application board. It is suggested to activate this "boost switch" for the first 50ms time period after the L+ power-on.

Table 6: Dedicated L6360 Signals

PHY Signal	IOLM4P Signal	Comment
SCL	I2C2_SCL	Address selection via SA0,SA1, SA2
SDA	I2C2_SDA	
OUT_C/Q	UART_RX_*	

Table 6: Dedicated L6360 Signals

PHY Signal	IOLM4P Signal	Comment
IN_C/Q	UART_TX_*	
EN_CQ	EN_TX*	
IRQ	IRQ*	Open drain (low active)
Boost	Signal 1*	Ext. boost switch to increase in rush current for 50ms after power on
OUT_I/Q	Signal 2	Mirrors level of digital input port (I/Q)
EN_L+	Signal 3	L+ switch enable low active
RST	Signal 4	Low active

2.4.2 PHY-LTC2874

For reference see [5] and https://www.analog.com/en/products/ltc2874.html. The LTC2874 is a PHY that supports four ports which are identified by the suffices 1,2,3 and 4. It provides no frame-handler, supports hot-plugging and sophisticated protection features. In table 7 below, the signals that are shared for all 4 ports are described.

Table 7: Shared LTC2874 Signals

PHY Signal	IOLM4P Signal	Comment
3A	MISO3	The signals 3 {A|B|C|D} are used for SPI-control of all 4 ports
3B	MOSI3	
3C	SCK3	
3D	NSS3	
4C	IRQn	One shared interrupt signal.

In table 8 below, the port related signals are listed.

Table 8: Port specific LTC2874 Signals

PHY Signals	IOLM4P Signal	Comment
RXD {1|2|3|4}	UART_RX {A|B|C|D}	PHY data output is labeled RXD on the LTC2874.
TXD {1|2|3|4}	TX {A|B|C|D}	PHY data input is labeled TXD on the LTC2874.
TXEN {1|2|3|4}	EN_TX {A|B|C|D}	Transmit enable is active high

3.0 Control Interface

The IOLM4P contains a full featured IO-Link Master stack that handles the IO-Link communication with up to four independent IO-Link ports. The IOLM4P is connected with host controller that controls the IO-Link Master on the IOLM4P the SPI interface.

3.1 Definitions and Acronyms

Host System: System that controls the IO-Link Master via the SPI interface.

IOLM4P: IO-Link 4-port Master controller.

Host Request telegram: Telegram of length „R“ sent by the host to the Master as originally defined in the IO-Link test specification. A request telegram is called cyclic request telegram if process data are transmitted (control telegram). All other request telegrams are called acyclic request telegram.

Master Response telegram: Reply telegram of length „A“ of the Master on the host request telegram. An answer telegram is called cyclic answer telegram if process data are transmitted (status telegram). All other answer telegrams are called acyclic answer telegram.

Interleaved request telegram: Request telegram sent by the host system to the IO-Link Master with a limited number „K“ of request bytes (marked by blue boxes) and extended by a fixed number of process and control bytes.

Request block: Fixed length part of the interleaved request telegram that contain parts of the original request telegram as defined in prepended by a direction and counter information byte. Answer and request block have the same length „K“ bytes.

Answer block: Fixed length part of the interleaved answer telegram that contain parts of the original answer telegram as defined in /1/ prepended by a direction and counter information byte. Answer and request block have the same length „K“ bytes.

Interleaved answer telegram: Answer telegram of the Master as reply of an interleaved request telegram of the host with a limited number „K“ of answer bytes (marked by blue boxes) and extended by a fixed number of process and control status bytes.

Communication cycle: Sequence consisting of one request telegram and one answer telegram.

R: Length of the request telegram in bytes.

A: Length to the answer telegram in bytes.

K: Number of request/answer bytes in a request/answer block.

N : Total number of IO-Link Master ports of the Master

DS: Data Storage

PD: Process data

PDIn: Process data issues by one IO-Link Device

PDOut: Process data send to one IO-Link Device

PDInStream: Combined PDIn of all active Devices

PDOutStream: Combined PDOut for all Devices

3.2 Host-SPI Interface

The controlling function is based on a simple telegram structure and uses the SPI-slave interface of the IOLM4P.

The control interface for the host application is independent from the selected PHY.

Table 9: SPI Host Interface Pins

Pin No.	Type	Connect if not used	Name	Short Description
15	I	X	SPI1_SCK	SPI clock from host (idle at low level)
20	I	X	SPI_NSS	SPI Chip select from host (active low)

Table 9: SPI Host Interface Pins

Pin No.	Type	Connect if not used	Name	Short Description
22	O	X	SPI_MISO	SPI output to host
23	I	VSS	SPI_MOSI	SPI input from SPI host
45	I	VSS	SYNC	SYNC signal from host, PCPDA, high active

The IOLM4P acts always as slave. This means, communication between the IOLM4P and host controller is always initiated by the host controller and responded by the IOLM4P.

3.2.1 Host-SPI Timing

During a telegram exchange between the host and the IOLM4P the following steps are executed:

1. The host drives the NSS pin to low.
2. The host sends the request telegram (consisting of 3 or more bytes) by driving the CLK and the MOSI lines typically using the DMA.
3. The IOLM4P returns dummy bytes on the MISO line which are ignored by the host.
4. The host drives the NSS to high, which triggers the evaluation of the IOLM4P response.
5. The host waits for the duration t_1 to allow the IOLM4P to process the response. The time t_1 depends on the type of the requested command. It is typically 200µs except for special requests that require flashing operations. In this case it is considerably longer. (Timing does not apply to normal host applications, it will be handled by a TEConcept configuration tool.)
6. The host drives the NSS to low.
7. The host drives the clock for 2 bytes to get back the expected Function ID and the number of bytes of the device response. It places dummy data which is ignored by the IOLM4P on the MOSI line.
8. The host checks the answer telegram length byte(s) and generates clock to receive the complete answer telegram.
9. The host drives the NSS to high, which triggers the IOLM4P to wait for the next host telegram request.

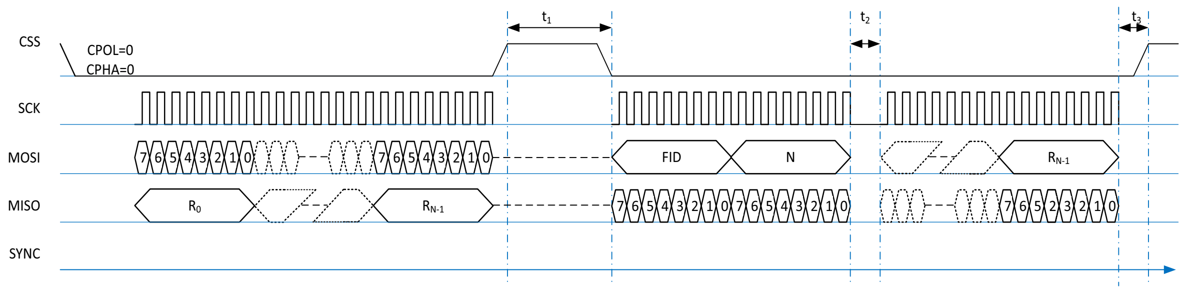


Figure 3. Host SPI Timing, SPI clock < 10MHz, 0µs < t2<100µs, t1=200µs

4.0 Conventions for Telegram Description

The basic telegram format for controlling the Master is described by using the following syntax convention.

4.1 Notational Conventions

The data transmitted over the SPI interface is defined as a sequence of bytes. The bytes are grouped in Elements that have an assigned data types.

Variable Telegram elements are specified by its name in parentheses:

```
[Element]
```

Constant Telegram elements are specified by its name or value in square brackets.

```
(0x01)
```

Repetitions of the telegram elements are indicated by enclosing the telegram elements in curly brackets. The number of repetitions is unspecified.

```
{[Element]}
```

In some cases **alternative Elements** e.g. E1 and E2 can be placed in a telegram. This is indicated by enclosing elements within acute brackets separated by vertical bars.

```
<[E1] | [E2]>
```

A telegram sequence starts with a Host request. Its first element is the function ID **FID** that is constant to specify the telegram. The second element indicates the total length of the telegram including the FID the length value itself. The telegram ends always with a checksum **CHKSUM**.

```
HRQ   (FID)
      <(N)|(EX)(NX)>
      {[Payload Bytes]}
      [CHKSUM]
```

The Master replies to a host request with the Master response **MRS**. The MRS starts with the same FID as the HRQ.

```
MRS   (FID)
      <(N)|(EX)(NE)>
      {[Payload Bytes]}
      [CHKSUM]
```

The second element in every telegram is its length.

```
<(N)|(EX)(NE)>
```

where the EX element is the size extension identifier:

```
EX: <(0)|(1)>
```

If the length of a telegram is smaller than 255 it can be indicated by one numerical Element (N). If it is longer it starts with an EX-element which has the values <0|1|> to indicate that the following element is the NE element that specifies telegram sizes between 256 and 65535 bytes.

4.1.1 Telegram Checksum

The checksum consists simply of the byte-wise x-or combination of all N bytes of the telegram.

4.1.2 Element Types and Values

Every element that is used in the HRQ or MRS telegrams is described in a subsequent telegram element description table.

Table 10: Telegram Element Description

Element	Description	Type: Value
FID	Function ID	UInt8: 0x00
N	Telegram size	UInt8: 0xNN
INFO_ID	Info ID	UInt8: 0xnn
CHKS	Checksum	UInt8: 0xnn

The last column specified the data type and the content of the elements in the HRQ or MRS telegrams. The following data types are defined for the Elements:

4.1.3 Data Types for Elements

`UIntN:` specifies an N-bit unsigned Integer

`IntN:` specifies an N-bit Integer

`Type[]:` specified an array of Type with unknown size

`Type[N]:` specified an array of Type with N elements

`s\0:` String with \0 termination

4.1.4 Values for Elements

0xnn: Indicates calculated arbitrary byte

{0xnn}: Indicates a repetition of calculates bytes

0xNN: Indicates known arbitrary 8-bit value

0x12: Fixed literal numeric 8-bit value

4.2 STI Handshake Convention

The Master shall answer for the request/status before the host sends the next telegram.

The host shall send the request/status telegram periodically. The duration of the period can vary and could be made configurable within the host application.

4.3 STI Communication Disconnection

When the PC doesn't get the answer for a telegram before it starts to send the next telegram it resends the previous. After two repetitions the telegram the host should disconnect the communication.

4.4 STI ENDIANESS

All numerical values are serialized in BIGENDIAN format. The most significant byte of an integer number are thus inserted first.

4.5 Process Data Stream Representation

The PDOutStream and the PDInStream include the outgoing and incoming process data from the C/Q line of an IO-Link port and data for all ports.

The Master is aware of the configured number of ports and the length of the process data for each ports, so it is able to compose or decompose PD-streams.

For the host side, the same information is required and needs to be stored for managing process data streams.

The process data configuration is characterized by an array of PDConfig structures as shown in Table 11 below. For every port one structure instance is used, the first PDConfig structure instance referring to the first port 0:

Table 11: PDConfig Structure

PDConfig for Port n	Description	Type: Value
Pos	Start position of process data of port	UInt8: 0xNN
Len	Length of process data of port in bytes	UInt8: 0xNN
IQ	I/Q configuration and length of port (see Table 12, see Table 13)	UInt8: 0xNN

An example of a process data stream of on an IO-Link Master with the number of ports N=4 is described by an array of 4 PDConfig structures.

```
PDConfig [4] = {
        (0)(2)(0)
// Port 0-2B PD, 0B I/Q
        (2)(2)(1)
// Port 1-2B PD, 1B I/Q
```

```
      (5)(1)(0)
// Port 2-1B PD, 0B I/Q
      (6)(3)(0)
// Port 3-3B PD, 0B I/Y
}
```

Based on the configuration given above, the corresponding PDStream would have the following structure:

```
PDStream[1]:
      [Int16: PD0]
      [Int8:  PD1]
      [Int8:  IQ1]
      [Int8:  PD2]
      [Int24: PD3]
```

If the port is in IO-Link mode, the PD are defined by the IO-Link protocol.

If the port is configured in SIO mode, the PD value is mapped to the level of the IO-Link line according to the following rule:

```
SIO line level „1“ -> PD=1
SIO line level „0“ -> PD=0
```

The same applies to the mapping of the I/Q line i digital mode.

4.6 I/Q Configuration

The I/Q line is mapped to Pin 2 of the IO-Link connector, if the IO-Link port is operating in CLASS A operation mode. The I/Q signal can be configured in terms of length in bytes (relevant in configurations where the I/Q signal does not represent digital, but analog data. It is however unidirectional. Thus, in setups, where the I/Q line is configured as input, it cannot be configured also as output. The configuration of the I/Q line is defined with one byte, the higher nibble contains the IQ configuration and the lower nibble contains the length of the digitized I/Q signal (see Table 13 and 12).

Table 12: IQ Configuration

IQ behavior	Description	Type: Value
E_IQBEHAVIOR_NOTSUPPORTED	I/Q line (Pin 2) not supported.	UInt4: 0x00
E_IQBEHAVIOR_DIGITAL	I/Q line carries digital signal	UInt4: 0x01
E_IQBEHAVIOR_ANALOG	I/Q line carries analog signal	UInt4: 0x02

Table 13: IQ Configuration and Length

IQ	Description	Bit Pos	Type: Value
IQConf	IQ configuration	7...4	UInt4: 0xNN
Len	length in bytes	3...0	UInt4: 0xNN

4.7 General Status Codes

The following table defines how a general status return code is coded:

Table 14: General Return-status Code

Status Code	Type: Value	Description
E_STAT CODE_OK	UInt8: 0x00	Request successfully completed
E_ STAT CODE_ ERR	UInt8: 0x01	Error during execution of request

1. *Int values are serialized in BIGENDIAN format*

4.8 ISDU Codes

The E_FNCID_CNTSTA response also returns the status of the last ISDU request as indicated in 15.

Table 15: ISDU Status

Execution state of last ISDU transfer	Code	Description
E_ISDUREQ STAT_IDLE	0x0	No ISDU requested
E_ISDUREQ STAT_BUSY	0x1	Service busy
E_ISDUREQ STAT_BUSY	0x2	ISDU transfer terminates with success
E_ISDUREQ STAT_ALERROR	0x3	ISDU transfer terminates with an error
E_ISDUREQ STAT_SERVICE NOTAVAILABLE	0x4	ISDU transfer service is not available

The ISDU request commands require to pass the ISDURequestType

Table 16: ISDU Request Type

ISDU Request Type	Code	Description
E_ISDUREQ_WR ITE	0x0	ISDU write request
E_ISDUREQ_RE AD	0x1	ISDU read request

The next table holds the ISDU answer codes:

Table 17: ISDU Answer Codes

ISDU transfer result	Code	Description
E_ISDUANS_ NOREQ	0x00	No ISDU requested.
E_ISDUANS_ WRITEPOS	0x01	Write request successfully completed.
E_ISDUANS_ WRITENEG	0x02	Error on write request.
E_ISDUANS_ READPOS	0x03	Read request successfully completed.
E_ISDUANS_ READNEG	0x04	Error on read request.

4.9 Port Configuration Codes

The port operating mode codes define operating modes of the related port.

Table 18: Port Operating Mode

Operating mode	Code	Description
E_OPMODE_ INACTIVE	0x0	Switched off.
E_OPMODE_DO	0x1	Digital output.
E_OPMODE_DI	0x2	Digital input.
E_OPMODE_ FIXEDMODE	0x3	Fixed mode.
E_OPMODE_ SCANMODE	0x4	Scan mode.

The port cycle configuration is used to specify how the cyclic behavior of multiple ports with respect to each other.

Table 19: Port Cyclic Operation

Port Cycle	Code	Description
E_PORTCYCLE_ FREERN	0x00	Free running
E_PORTCYCLE_ FIXVAL	0x01	Fixed value for cycle time.
E_PORTCYCLE_ MSGSYN	0x02	Message synchronization.

4.10 Data Storage Codes

Data Storage commands that transfer to the Master can make use of the following codes.

Table 20: DS Upload Commands

Operating mode	Code	Description
E_DSUPL_ SETSIZE	0x0	Set DS size.
E_DSUPL_ UPLOAD	0x1	Upload a part of the DS.
E_DSUPL_ SETVALIDITY	0x2	Set DS validity.

Data Storage commands that read information from the Master can make use of the following codes

Table 21: DS Download Commands

Port Cycle	Code	Description
E_DSDWN_GET SIZE	0x01	Get DS size.
E_DSDWN_DOW NLOAD	0x02	Download a part of the DS.
E_DSDWN_GET VALIDITY	0x03	Get DS validity.

5.0 Standard Telegrams

This subsection describes in detail all telegrams. Conventions that are applying to multiple telegrams have been specified in the prior subsections. The following table summarizes all control telegrams that are currently available:

Table 22: Overview of Standard Telegrams

FID	ILS	Function ID	Telegram Name	Description
0x00	Y	E_FNCID_GETINFO (5.1)	Get Identification	Get information about the Master
0x01	Y	E_FNCID_SETPDI (5.2)	Set PD Interface	Set the PD sizes
0x02	Y	E_FNCID_GETPDI (5.3)	Get PD Interface	Get the number of ports and the PD sizes
0x03	Y	E_FNCID_CNTSTA (5.4)	Control Status	Standard telegram for setting ports configuration and receiving port/ISDU status flags and process data streams
0x04	Y	E_FNCID_SETPORTCNF PAR (5.5)	Set Port Configuration	Set configuration parameters for one port
0x05	Y	E_FNCID_GETPORTCNF PAR	Get Port Configuration	Get configuration parameters of one port
0x06	Y	E_FNCID_SETDSHDR	Set Data Storage Header	Set Data Storage header for one port
0x07	Y	E_FNCID_GETDSHDR	Get Data Storage Header	Get Data Storage header of one port
0x08	N	E_FNCID_UPLDS	Upload Data Storage to the Master	Upload entire data storage content to Master
0x09	N	E_FNCID_DWNDS	Download Data Storage from the Master	Download entire data storage content from Master
0x0A	Y	E_FNCID_ISDUREQ	ISDU request	Issues ISDU read or write request to the Master
0x0B	Y	E_FNCID_ISDUGET	ISDU result	Receives answer or the error code of a previous ISDU request.
0x0C	Y	E_FNCID_RDEVENT	Read event	Request one element of event data from IOLM4P after an event flag was indicated
0x0D	Y	E_FNCID_SWITCHOFF	Switch OFF the Power	Switches off the 24V on the L+ line of the corresponding port.
0x0E	Y	E_FNCID_SWITCHON	Switch ON the Power	Switches on the 24V on the L+ line of the corresponding port
0x18	Y	E_FNCID_UPLDS_SHOR T	Upload a part of Data Storage to Master	Upload a specified number of bytes of the data storage content to the Master

Table 22: Overview of Standard Telegrams

FID	ILS	Function ID	Telegram Name	Description
0x19	Y	E_FNCID_DWNDS_SHORT	Download a part of the Data Storage from Master	Download a specified number of bytes of the data storage content from the Master

5.1 Get Identification E_FNCID_GETINFO - 0x00

Read out identification parameters from the Master.

```
HRQ    (E_FNCID_GETINFO)
       (N)
       [INFO_ID]
       [CHKS]
```

Table 23: Get Identification

Element	Description	Value
E_FNCID_GETINFO	Function ID	UInt8: 0x00
N	Telegram size	Uint8: 0x04
INFO_ID	Info ID	UInt8: 0xNN
CHKS	Checksum	UInt8: 0xnn

The response returns a data stream that contains relevant information about the Master.

```
MRS    (E_FNCID_GETINFO)
       [N]
       [INFO_ID]
       {[Data]}
       [CHKS]
```

Table 24: Identification Response

Element	Description	Multivalued
E_FNCID_GETINFO	Function ID	UInt8: 0x00
N	Telegram size	UInt8: 0xNN

Table 24: Identification Response

Element	Description	Multivalued
INFO_ID (see Table 25)	Info ID	UInt8: 0xNN
Data (see Table 25)	Response data	UInt8[ ]: {0xnn}
CHKS	Checksum	UInt8: 0xnn

The parameters FID and INFO are returned by the Master without change. The data content returned by the Master depends on the INFO_ID as defined in Table 25 below.

Table 25: INFO_ID and Data Content

INFO_ID	INFO_ID	Description	Data
E_INFO_VENDOR	UInt8: 0x00	Vendor Name	String: s\0
E_INFO_PRD_NAME	UInt8: 0x01	Product Name	String: s\0
E_INFO_PRDID	UInt8: 0x02	Product ID	String: s\0
E_INFO_SRNNUMB	UInt8: 0x03	Serial Number	String: s\0
E_INFO_SWREV	UInt8: 0x04	Software Rev.	String: s\0
E_INFO_HWREV	UInt8: 0x05	Hardware Rev.	String: s\0
E_INFO_DESC	UInt8: 0x06	Master Description	String: s\0
E_INFO_VENDORID	UInt8: 0x07	Unique IO-Link Vendor ID	UInt16: 0xNNNN
E_INFO_MASTERID	UInt8: 0x08	Vendor specific unique Master identification.	UInt24: 0xNNNNNN

Table 25: INFO_ID and Data Content

INFO_ID	INFO_ID	Description	Data
E_INFO_ MASTER TYPE	UInt8: 0x09	Type of Master (see Table 26)	UInt8: 0xNN
E_INFO_ PORT_ TYPES	UInt8: 0x0A	N refers to the number of supported ports. (see Table 27)	UInt8: 0xNN
E_INFO_ NOTSUP-PORTED	UInt8: 0xFF	Master replies with this INFO_ID if the requested INFO_ID is not supported	-

Table 26: E_INFO_MASTERTYPE

Master Type	Description	Type: Value
M_U	Unspecified	Int8:0x00
M_R	Reserved	Int8:0x01
M_2	V1.1 or higher	Int8:0x02
M_3	Functional Safety Master	Int8:0x03
M_4	Wireless Master	Int8:0x04
Reserved	Future Use	0x05 ... 0xFF

Table 27: E_INFO_PORT_TYPE

Port Type	Description	Type: Value
Class A	PowerOff/On supported	UInt8: 0x00
Class A+	PowerOff/On supported	UInt8: 0x01
Class B	Class B port	UInt8: 0x02
FS_A	Function safety port A without OSSDe	UInt8: 0x03
FS_A_ OSSDe	Functional safety port A with OSSDe support	UInt8: 0x04
FS_B	Functional safety port B	UInt8: 0x05

Table 27: E_INFO_PORT_TYPE

Port Type	Description	Type: Value
Wireless	Wireless Master	UInt8: 0x06

5.2 Set PD Interface E_FNCID_SETPDI - 0x01

The PDInStream sizes are configured with this telegram.PDInStreamSize and PDOutStream◘ Size are represented by 16 bit unsigned integer numbers.

```
HRQ    (E_FNCID_SETPDI)
       (N)
       [PDInStreamSize]
       [PDOutStreamSize ]
       [CHKS]
```

Table 28: Set PD Interface Request

Element	Description	Multivalued
E_FNCID_SET PDI	Function ID	UInt8: 0x01
N	Telegram size	UInt8: 0x07
PDInStream Size	Size of PDInStream in Bytes	UInt16: 0xNNNN
PDOutStream Size	Size of PDOutStream in Bytes	UInt16: 0xNNNN
CHKS	Checksum	UInt8: 0xnn

```
MRS    (E_FNCID_SETPDI)
       (N)
       [Status]
       [CHKS]
```

Table 29: Set PD Interface Response

Element	Description	Value
E_FNCID_SET PDI	Function ID	UInt8: 0x01

Table 29: Set PD Interface Response

Element	Description	Value
N	Telegram size	UInt8: 0x07
Status	Status code (see Table 14)	UInt8: 0xnn
CHKS	Checksum	UInt8: 0xnn

5.3 Get PD Interface E_FNCID_GETPDI 0x02

Acquire the number of ports and the sizes PDInstream and PDOutStream structures from the Master in Bytes.

```
HRQ   (E_FNCID_GETPDI)
      (N)
      [CHKS]
```

Table 30: Get PD Interface Request

Element	Description	Multivalued
E_FNCID_ GETPDI	Function ID	UInt8: 0x02
N	Telegram size	UInt8: 0x03
CHKS	Checksum	UInt8: 0xNN

The Master response returns in addition to the process data stream sizes the number of Master ports.

```
MRS   (E_FNCID_GETPDI)
      (N)
      [Ports]
      [PDInStreamSize]
      [PDOutStreamSize]
      [CHKS]
```

Table 31: Get PD Interface Response

Element	Description	Value
E_FNCID_ GETPDI	Function ID	UInt8: 0x02
N	Telegram size	UInt8: 0x08
Ports	Number of Ports	UInt8: 0xnn
PDInStreamSize	Number of bytes of PDInStream	UInt16: 0xnnnn
PDOutSStream Size	Number of bytes for PDOutStream	UInt16: 0xnnnn
CHKS	Checksum	UInt8: 0xnn

5.4 Control Status E_FNCID_CNTSTA - 0x03

The control status telegram is used to write configuration information to all Master ports and to send one PDOutStream to the Master. The Master returns status information for every port and the current process data as PDInStream and additionally the status of the last ISDU transfer.

```
HRQ   (E_FNCID_CNTSTA)
      [N]
      {[ControlFlags]}
      [PDOutStream]
      [CHKS]
```

The control status telegram supports configuration of all ports and allows to send one PDOutStream to the Master.

Table 32: Control Status Request

Element	Description	Multivalued
E_FNCID_ CNTSTA	Function ID	UInt8: 0x03
N	Telegram size	UInt8: 0xNN
ControlFlags (see Table 34)	Port specific configuration	UInt16:0xNN
PDOutStream	Process Data output stream (see Table 11)	UInt8[ ] : {[0xNN]}
CHKS	Checksum	UInt8: 0xNN

```
MRS    (E_FNCID_CNTSTA)
       [N]
       {[PortStatus]}
       [PDInStream]
       [CHKS]
```

The response of the Device returns the success and the current PDInStream of the Device.

Table 33: Control Status Response

Element	Description	Value
E_FNCID_CNTSTA	Function ID	UInt8: 0x03
N	Telegram size	UInt8: 0xnn
PortStatus (see Table 35)	Status bits per port	UInt16[N]: 0xnnnn
PDInStream	Process data input stream (see Table 11)	UInt8[ ]: 0xnn
CHKS	Checksum	UInt8: 0xnn

The E_FNCID_CNTSTA send control command requires for every supported port to set the prop￭er control flags as defined in Table 34.

Table 34: Port Control Flags

Control Flags	Number of Bits	Bit pos	Description
Operating Mode	1	15	Triggers a port mode change. The requested port mode is sent by the set configuration parameters telegram (see on 8.2.2.4.).
Start to Operate	1	14	Sets port to operate state, if port state is in ready to operate
DS delete	1	13	Clears data storage
Event acknowledge-ledge	1	12	Acknowledges the indicated event
PDOut-valid	1	11	Sets PDOut validity of port to valid
PDOut-invalid	1	10	Sets PDOut validity of port to invalid
DS activation start	1	9	Sets DS Activation state to DS_Enabled (changes port configuration)
DS activation stop	1	8	Sets DS Activation state to DS_Disabled (changes port configuration)
OD Start	1	7	Starts on-request data handling of the Master
OD Stop	1	6	Stops on-request data handling of the Master
DU Start	1	5	Starts diagnosis unit handling on the Master
DU Stop	1	4	Stops diagnosis unit handling on the Master
PD start	1	3	Starts the process data handling on Master
PD stop	1	2	Stops the Process data handling Master
reserved	1	1	-
reserved	1	0	-

The Master response of the E_FNCID_CNTSTA telegram returns for every Master port status bits as defined in Table 35

Table 35: Port Status Bits

Port Status Bits	Number of Bits	Bit pos	Description
Ready to operate	1	15	Indicates that the port is in Ready to operate state.
Operating	1	14	Indicates that the port is in operate state
Event indication	1	13	Indicates that a new event has been raised

Table 35: Port Status Bits

Port Status Bits	Number of Bits	Bit pos	Description
PDIn status	1	12	Indicates validity of PDIn data. (1:PDin is valid, 0: PDIn is invalid)
ODE status	1	11	Indicates that on-request-data handling is active
DU status	1	10	Indicates that diagnosis unit is active
PDE status	1	9	Indicates that process data exchange is active
DS uploaded	1	8	Indicates that DS is uploaded. The flag is cleared by executing E_FNCID_DWNDS or E_FNCID_DWNDS_SHORT requests
Fault (see Table 36)	4	7... 4	Master is in faulty state
Last ISDU status	4	3... 0	Indicates the status of last ISDU transfer (see Table 15)

The port status bits contain fault conditions that are defined in detail in Table 36 below:.

Table 36: Port Status Fault Codes

Port Status Fault Code	Code	Description
E_FAULT_NOFAULT	0x0	No Fault
E_FAULT_COMLOST	0x1	Communication to device is lost
E_FAULT_REVFAULT	0x2	Revision Fault
E_FAULT_COMPFAULT	0x3	Compatibility Fault
E_FAULT_SERNUMFAULT	0x4	Serial Number Fault
E_FAULT_DSFAULT	0x5	Data Storage Fault
E_FAULT_LICENSEFAULT	0x6	License Fault[a]
E_FAULT_TEMPERATUREFAULT	0x7	Port Temperature Fault
E_BOOTLOADER_FAULT	0x8	Master is in Bootload Mode and can't communicate with the port

a. optional, not implemented on all Masters

5.5 Set Port Configuration E_FNCID_SETPORTCNFPAR 0x04

Sends configuration for one port to the Master. The port number indicates the related IO-Link port (starting with 0).

The Master returns in his response the success status.

```
HRQ   (E_FNCID_SETPORTCNFPAR)
      (N)
      [PortNumber]
      [OperatingMode]
      [PortCycle]
      [CycleTime]
      [PDConfigLenIn]
      [PDConfigPosIn]
      [PDConfigSrcOffsetIn]
      [PDConfigLenOut]
      [PDConfigPosOut]
      [PDConfigSrcOffsetOut]
      [VendorID]
      [DeviceID]
      [SerialNumber]
      [InspectionLevel]
      [DSActivationState]
      [DSDownloadEnable]
      [DSUploadEnable]
      < - | RevisionID >
      [CHKS]
```

The set port configuration request configures one Master port in detail.

Table 37: Set Port Configuration Request

Element	Description	Value
E_FNCID_SET PORTCNFPAR	Function ID	UInt8: 0x04
N	Telegram size	UInt8: 0x28 Uint8: 0x29 (extended version)
PortNumber	Port Number	Uint8: 0x0N
OperatingMode (see Table 18)	Requested port mode	UInt8: 0x0N
PortCycle (see Table 19)	Type of port synchronization for cyclic operation	UInt8: 0x0N
CycleTime	Port cycle time with a resolution of 100 µs.	UInt16: 0xNNNN
PDConfigLenIn	Length of the incoming process data in bytes.	UInt8: 0xNN
PDConfigPosIn	Position of 1st byte of PDIn in PDInStream	UInt8: 0xNN
PDConfigLenOut	Length of the outgoing process data in bytes.	UInt8: 0xNN
PDConfigPosOut	Position of 1st byte of PDOut in PDInStream.	UInt8: 0xNN
Vendor ID	Device Vendor ID.	UInt16: 0xNNNN
Device ID	ID of connected Device	UInt16: 0xNNNNNN
SerialNumber	Device serial number.	Uint8[16]: {[0xNN]}
InspectionLevel	Inspection level.	UInt8: 0xNN
DSActivationState	Activation state of the Data Storage (0 : inactive, 1 : active).	UInt8: 0x0N
DSDownload Enable	Data Storage download enable (0 : disabled, 1 : enabled).	UInt8: 0xNN
DSUpload Enable	Data Storage upload enable (0 : disabled, 1 : enabled).	UInt8: 0xNN
RevisionID (in N==0x29)	Revision ID	0x10: V1.0 Device 0x11: V11 Device
CHKS	Checksum	UInt8: 0xNN

The Master response simply provides in a status byte the success of the E_FNCID_SETPORTCNFPAR telegram.

```
MRS    (E_FNCID_SETPORTCNFPAR)
       (N)
       [PortNumber]
       [Status]
       [CHKS]
```

The response of the Master contains simply a execution success status information.

Table 38: Set Port Configuration Response

Element	Description	Value
E_FNCID_SET PORTCNFPAR	Function ID	UInt8: 0x04
N	Telegram size	UInt8: 0x05
PortNumber	Port Number	Uint8: 0xN
Status (see Table 14)	Result state of port configuration	UInt8: 0xnn
CHKS	Checksum	UInt8: 0xnn

5.6 Get Port Configuration E_FNCID_GETPORTCNFPAR 0x05

The E_FNCID_GETPORTCNFPAR telegram is used to get the current port configuration from the Master.

```
HRQ   (E_FNCID_GETPORTCNFPAR)
      (N)
      [PortNumber]
      [CHKS]
```

Table 39: Get Set Port Configuration Request

Element	Description	Value
E_FNCID_GET PORTCNFPAR	Function ID	UInt8: 0x05
N	Telegram size	UInt8: 0x04
PortNumber	Port Number	Uint8: 0xN
CHKS	Checksum	UInt8: 0xnn

The E_FNCID_GETPORTCNFPAR response is analog to the request in E_FNCID_SETPORTCNFPAR.

```
MRS   (E_FNCID_GETPORTCNFPAR)
      (N)
      [PortNumber]
      [OperatingMode]
      [PortCycle]
      [CycleTime]
      [PDConfigLenIn]
      [PDConfigPosIn]
      [PDConfigSrcOffsetIn]
      [PDConfigLenOut]
      [PDConfigPosOut]
      [PDConfigSrcOffsetOut]
      [VendorID]
      [DeviceID]
      [SerialNumber]
      [InspectionLevel]
      [DSActivationState]
      [DSDownloadEnable]
      [DSUploadEnable]
      < - | RevisionID >
      [CHKS]
```

The set port configuration request configures one port in detail.

Table 40: Get Port Configuration Response

Element	Description	Value
E_FNCID_GET PORTCNFPAR	Function ID	UInt8: 0x05
N	Telegram size	UInt8: 27
PortNumber	Port Number	Uint8: 0x0N
OperatingMode (see Table 18)	Requested port mode	UInt8: 0x0N
PortCycle (see Table 19)	Type of port synchronization for cyclic operation	UInt8: 0x0N
CycleTime	Port cycle time with a resolution of 100 µs.	UInt16: 0xNNNN
PDConfigLenIn	Length of the incoming process data in bytes.	UInt8: 0xNN
PDConfigPosIn	Position of 1st byte of PDIn in PDInStream	UInt8: 0xNN
PDConfigLen-Out	Length of the outgoing process data in bytes.	UInt8: 0xNN
PDConfigPos-Out	Position of 1st byte of PDOut in PDInStream.	UInt8: 0xNN
VendorID	Device Vendor ID.	UInt16: 0xNNNN
DeviceID	ID of connected Device	UInt16: 0xNNNNNN
SerialNumber	Device serial number.	Uint8[16]: {[0xNN]}
InspectionLevel	Inspection level.	UInt8: 0xNN
DSActivation State	Activation state of the Data Storage (0 : inactive, 1 : active).	UInt8: 0x0N

Table 40: Get Port Configuration Response

Element	Description	Value
DSDownload Enable	Data Storage download enable (0 : disabled, 1 : enabled).	UInt8: 0xNN
DSUpload Enable	Data Storage upload enable (0 : disabled, 1 : enabled).	UInt8: 0xNN
RevisionID (in N==0x29)	Revision ID	0x10: V1.0 Device 0x11: V11 Device
CHKS	Checksum	UInt8: 0xNN

5.7 Set Data Storage Header E_FNCID_SETDSHDR 0x06

This telegram is optional. The header of the data storage array for one port is send to the Master. The Master returns a success status code.

```
HRQ   (E_FNCID_SETDSHDR)
      (N)
      [PortNumber]
      [DS_Checksum]
      [VendorID]
      [DeviceID]
      [FunctionID]
      [CHKS]
```

Table 41: Set Data Storage Header Sequest

Element	Description	Value
E_FNCID_SET DSHDR	Function ID	UInt8: 0x06
N	Telegram size	UInt8: 0x10
PortNumber	Port Number	Uint8: 0xN
DS_Checksum	Checksum of the parameters in the Data Storage (10.4.8 [2])	UInt32: 0xNNNNNNNN
VendorID	Vendor ID of connected Device	UInt16: 0xNNNN
DeviceID	Device ID of connected Device	UInt32. 0xNNNNNNNN
FunctionID	Data Storage header field	Uint16: 0xNNNN
CHKS	Checksum	UInt8: 0xnn

The Master response simply provides the success status of the E_FNCID_SETDSHDR telegram.

```
MRS   (E_FNCID_SETDSHDR)
      (N)
      [PortNumber]
      [STATUS]
      [CHKS]
```

Table 42: Set Data Storage Header Response

Element	Description	Value
E_FNCID_SET DSHDR	Function ID	UInt8: 0x06
N	Telegram size	UInt8: 0x05
PortNumber	Port Number	Uint8: 0xN
Status (see Table 14)	Result state of port configuration	UInt8: 0xnn
CHKS	Checksum	UInt8: 0xnn

5.8 Get Data Storage Header E_FNCID_GETDSHDR 0x07

This telegram is optional. The header of the Data Storage for one port is read back from the Master.

```
HRQ   (E_FNCID_GETDSHDR)
      (N)
```

```
[PortNumber]
[CHKS]
```

Table 43: Get Data Storage Header Request

Element	Description	Value
E_FNCID_GET DSHDR	Function ID	UInt8: 0x07
N	Telegram size	UInt8: 0x04
PortNumber	Port Number	Uint8: 0xN
CHKS	Checksum	UInt8: 0xnn

The Master response is analog to the request in the E_FNCID_SETDSHDR telegram.

```
MRS     (E_FNCID_GETDSHDR)
        (N)
        [PortNumber]
        [DS_Checksum]
        [VendorID]
        [DeviceID]
        [FunctionID]
        [CHKS]
```

Table 44: Get Data Storage Header Response

Element	Description	Value
E_FNCID_GET DSHDR	Function ID	UInt8: 0x07
N	Telegram size	UInt8: 0x10
PortNumber	Port Number	Uint8: 0xN
DS_Checksum	Checksum of the parameters in the Data Storage (10.4.8 [2])	UInt32: 0xNNNNNNNN
VendorID	Vendor ID of connected Device	UInt16: 0xNNNN
DeviceID	Device ID of connected Device	UInt32. 0xNNNNNNNN
FunctionID	Data Storage header field	Uint16: 0xNNNN
CHKS	Checksum	UInt8: 0xnn

5.9 Upload DS to Master E_FNCID_UPLDS 0x08

The entire Data Storage data „DSDATA“ for one port is transferred to the Master. The size of DS DATA is variable.

```
HRQ     (E_FNCID_UPLDS)
        <[N]|(EX)[N2]>
        [PortNumber]
        {[DSDATA]}
        [CHKS]
```

Table 45: Upload DS to Master Request

Element	Description	Value
E_FNCID_ UPLDS	Function ID	UInt8: 0x08
<[N] | (EX) [N2]>	Telegram size	< UInt8: 0xnn | Uint8: 0 Uint16: 0xnnnn >
PortNumber	Port Number	Uint8: 0x0N
DSDATA	Data Storage Data	UInt8[ ]: {[0xNN]}
CHKS	Checksum	UInt8: 0xnn

The Master responds with a result state.

```
MRS     (E_FNCID_UPLDS)
        (N)
        [PortNumber]
        [STATUS]
        [CHKS]
```

Table 46: Upload DS to Master Response

Element	Description	Value
E_FNCID_ UPLDS	Function ID	UInt8: 0x08
N	Telegram size	UInt8: 0xn5
PortNumber	Port Number	Uint8: 0x0N

Table 46: Upload DS to Master Response

Element	Description	Value
StatusCode (see Table 14)	Result state of port configuration	UInt8: 0xnn
CHKS	Checksum	UInt8: 0xnn

5.10 Download DS from Master E_FNCID_DWNDS 0x09

The entire Data Storage data „DSDATA“ for one port is requested from the Master. The size of DSDATA is variable.

```
HRQ    (E_FNCID_DWNDS)
       (N)
       [PortNumber]
       [CHKS]
```

Table 47: Download DS from Master Request

Element	Description	Value
E_FNCID_ DWNDS	Function ID	UInt8: 0x09
N	Telegram size	UInt8: 0x05
PortNumber	Port Number	Uint8: 0x0N
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

The Master returns the requested DS content.

```
MRS    (E_FNCID_DWNDS)
       <[N] |(EX)[N2]>
       [PortNumber]
       {[DSDATA]}
       [CHKS]
```

Table 48: Download DS from Master response

Element	Description	Value
E_FNCID_ DWNDS	Function ID	UInt8: 0x09
<[N] | (EX) [N2]>	Telegram size	< UInt8: 0xnn | Uint8: 0 Uint16: 0xnnnn >
PortNumber	Port Number	Uint8: 0x0N
DSDATA	DS content	UInt8: { 0xnn }
CHKS	Checksum	UInt8: 0xnn

5.11 ISDU Transfer Initialization E_FNCID_ISDUREQ 0x0A

An ISDU transfer can be requested with ISDU request telegram.Two types of ISDU request are existing; read- and write requests. The ISDU request type, ISDU index and sub index are passed in the request telegram. On a write request also the data bytes are passed.

```
HRQ    (E_FNCID_ISDUREQ)
       [N]
       [PortNumber]
       [ISDURequestType]
       [Index]
       [SubIndex]
       < -
       |
        {[ISDUDATA]}
       >
       [CHKS]
```

Table 49: ISDU Transfer Initialization Request

Element	Description	Value
E_FNCID_ISDU REQ	Function ID	UInt8: 0x0A

Table 49: ISDU Transfer Initialization Request

Element	Description	Value
N	Telegram size	UInt8: 0xnn
PortNumber	Port Number	UInt8: 0x0N
ISDURequest Type (see Table 16)	ISDU read or write	Uint8: 0xNN
Index	ISDU index	UInt16: 0xNNNN
SubIndex	ISDU subindex	Uint8: 0xNN
ISDUDATA (only present, if ISDURequestType = 0 (write))	ISDU parameter bytes	UInt8[ ]: 0xNN
CHKS	Checksum	UInt8: 0xnn

The Master response of the HRQ contains the success return status of the Master.

```
MRS    (E_FNCID_ISDUREQ)
       (N)
       [PortNumber]
       [Status]
       [CHKS]]
```

Table 50: ISDU Transfer Initialization Response

Element	Description	Value
E_FNCID_ISDUREQ	Function ID	UInt8: 0x0A
N	Telegram size	UInt8: 0xn5
PortNumber	Port Number	Uint8: 0x0N
Status (see Table 14)	Result state of port configuration	UInt8: 0xnn
CHKS	Checksum	UInt8: 0xnn

5.12 ISDU Get E_FNCID_ISDUGET 0x0B

After an ISDU request has been issued by using the telegram E_FNCID_ISDUREQ, the host has to check the status of the ongoing ISDU transfer by checking the ISDUstatus answer of the E_FNCID_CNTSTA telegram (see Table 15).

If the ISDU request state equals E_ISDUREQSTAT_OK or E_ISDUREQSTAT_ALERROR, a previously initiated ISDU read transfer can be completed with this telegram.

```
HRQ    (E_FNCID_ISDUGET)
       (N)
       [PortNumber]
       [CHKS]
```

Table 51: ISDU Get Request

Element	Description	Value
E_FNCID_ISDUGET	Function ID	UInt8: 0x0B
N	Telegram size	UInt8: 0xn4
PortNumber	Port Number	Uint8: 0xN
CHKS	Checksum	UInt8: 0xnn

The Master response of the HRQ depends on the ISDUAnswer element.

```
MRS    (E_FNCID_ISDUGET)
       [N]
       [PortNumber]
       [ISDUAnswer]
       < - |
         [ErrorCode]
         [Additional ErrorCode]
       |
         {[IDSUData]}
       | >
       [CHKS]]
```

Table 52: ISDU Get Response

Element	Description	Value
E_FNCID_ISDU GET	Function ID	UInt8: 0x0B
N	Telegram size	UInt8: 0xnn
PortNumber	Port Number	Uint8: 0xN
ISDUAnswer (see Table 17)	Result state of port configuration	UInt8: 0xnn
ErrorCode (inserted if ISDUANSWER= (WRITENEG or READNE)	as specified in Annex C [2])	UInt8: 0xnn
Additional ErrorCode (inserted if ISDUANSWER= (WRITENEG or READNEG)	as specified in (Annex C [2])	UInt8: 0xnn
IDSUData (inserted if ISDUANSWER= (READPOS)	ISDU value	UInit8[ ]: 0xnn
CHKS	Checksum	UInt8: 0xnn

5.13 Read Event E_FNCID_RDEVENT (0xC)

If the event flag signals that an event is pending, the E_FNCID_RDEVENT telegram shall be used to read one event from the Master. The answer telegram includes the number of remaining pending events. When all events have been read, the event acknowledge is set in the port status acquired by the E_FNCID_CNTSTA telegram.

```
HRQ    (E_FNCID_RDEVENT)
       (N)
       [PortNumber]
       [CHKS]
```

Table 53: Read Event Request

Element	Description	Value
E_FNCID_RDEVENT	Function ID	UInt8: 0x0C
N	Telegram size	UInt8: 0xn4
PortNumber	Port Number	Uint8: 0xN
CHKS	Checksum	UInt8: 0xnn

The response of the Master holds status, instance, mode, type, origin and code of the event and the number of the pending events.

```
MRS    (E_FNCID_RDEVENT)
       (N)
       [PortNumber]
       [EventStatus]
       [EventInstance]
       [EventMode]
       [EventType]
       [EventOrigin]
       [EventCode]
       EventsLeft]
       [CHKS]
```

Table 54: Read Event Response

Element	Description	Value
E_FNCID_RDEVENT	Function ID	UInt8: 0x0C
N	Telegram size	UInt8: 0x0C
PortNumber	Port Number	UInt8: 0x0n
Event Status (see Table 55)	Event Status	UInt8: 0x0n
Event Instance (see Table 56)	Event Instace	UInt8: 0x0n
Event Mode (see Table 57)	Event Mode	UInt8: 0x0n

Table 54: Read Event Response

Element	Description	Value
Event Type (see Table 58)	Event Type	UInt8: 0x0n
Event Origin (see Table 59)	Origin of Event	UInt8: 0x0n
Event Code	Event Code (see [2])	UInt16: 0xnnnn
Events Left	Number of events that needs to be read	UInt8: 0x0n
CHKS	Checksum	UInt8: 0xnn

The Event related codes are specified below.

Table 55: EventStatus Codes

Event status	Code	Description
E_EVENTSTA_OK	0x0	Triggered Event is ready to read
E_EVENTSTA_NOEVNT	0x1	No Event ready to read

Table 56: EventInstance Codes

Event instance	Code	Description
E_INSTANCE_UNKN	0x0	Instance unknown
E_INSTANCE_APPLI	0x1	Instance application

Table 57: EventMode Codes

Event mode	Code	Description
E_MODE_EVSINGSHOT	0x01	Event single shot.
E_MODPPE_EVDISA	0x02	Event disappeared.
E_MODE_EVAPP	0x03	Event appeared.

Table 58: EventType Codes

Event type	Code	Description
E_TYPE_NOT	0x01	Notification.
E_TYPE_WARN	0x02	Warning.
E_TYPE_ERR	0x03	Error.

Table 59: EventOrigin Codes

Event status	Code	Description
E_ORIGIN_REMOTE	0x00	Origin remote.
E_ORIGIN_LOCAL	0x01	Origin local.

5.14 Switch OFF Power E_FNCID_SWITCHOFF 0x0D

Switches OFF the 24V to the L+ line.

```
HRQ    (E_FNCID_SWITCHOFF)
       (N)
       [PortNumber]
       [CHKS]
```

Table 60: Switch Off Power Request

Element	Description	Value
E_FNCID_SWITCHOFF	Function ID	UInt8: 0x0D
N	Telegram size	UInt8: 0x04
PortNumber	Port Number	Uint8: 0x0N
CHKS	Checksum	UInt8: 0xnn

The Master responds with a result state.

```
MRS    (E_FNCID_SWITCHOFF)
       (N)
       [PortNumber]
```

```
      [Status]
      [CHKS]
```

Table 61: Switch Off Power response

Element	Description	Value
E_FNCID_ SWITCHOFF	Function ID	UInt8: 0x0D
N	Telegram size	UInt8: 0xn4
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

5.15 Switch ON Power E_FNCID_SWITCHON 0x0E

```
HRQ   (E_FNCID_SWITCHON)
      (N)
      [PortNumber]
      [CHKS]
```

Table 62: Switch On Power Request

Element	Description	Value
E_FNCID_ SWITCHON	Function ID	UInt8: 0x0E
N	Telegram size	UInt8: 0x04
PortNumber	Port Number	Uint8: 0x0N
CHKS	Checksum	UInt8: 0xnn

The Master responds with a result state.

```
MRS   (E_FNCID_SWITCHON)
      (N)
      [PortNumber]
      [Status]
      [CHKS]
```

Table 63: Switch On Power Response

Element	Description	Value
E_FNCID_ SWITCHON	Function ID	UInt8: 0x0E
N	Telegram size	UInt8: 0xn4
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

Table 63: Switch On Power Response

5.16 Upload DS Partially E_FNCID_UPLDS_SHORT 0x018

This telegram is optional. In some cases it is not possible for the Master to receive the whole DSData in one telegram (E_FNCID_UPLDS). The telegram E_FNCID_UPLDS_SHORT provides means to transfer DSData in smaller chunks. Depending on the content of the DSUploadCommand byte, the structure of the telegram changes as described below:

```
HRQ   (E_FNCID_UPLDS_SHORT)
      <(N)|(EX)[N2]>
      [PortNumber]
      [DSUploadCommand]
      {
       [DSSize] |
       [StartAddress]
       {[DSData]} |
       [Validity]
      }
      [CHKS]
```

Table 64: Upload DS Partially Request

Element	Description	Value
E_FNCID_UPL DS_SHORT	Function ID	UInt8: 0x18
<[N] | (EX) [N2]>	Telegram size	< UInt8: 0xnn | Uint8: 0 Uint16: 0xnnnn >
PortNumber	Port Number	Uint8: 0xNN

Table 64: Upload DS Partially Request

Element	Description	Value
DSUploadCommand „DSUC" (see Table 20)	DS upload related command	UInt8: 0xNN
DSSize (placed only, if DSUC=1)	Size of subsequent DSData chunks for E_FNCID_UPLDS_SHORT telegram	UInt16: 0xNNNN
StartAddress (placed only if DSUC=2)	Address of first transferred DSDATA byte, starting with 0	UInt16: 0xNNNN
DSDATA (placed only if DSUC=2)	Partial Data Storage Date	UInt8[ ]: {[0xNN]}
Validity (if DSUC=3)	DS Validity	UInt8: 0xNN
CHKS	Checksum	UInt8: 0xnn

The Master responds with a result state.

```
MRS    (E_FNCID_UPLDS_SHORT)
       (N)
       [PortNumber]
       [STATUS]
       [CHKS]
```

The response to the telegram contains the success status code.

Table 65: Upload DS Partially Response

Element	Description	Value
E_FNCID_UPLDS_SHORT	Function ID	UInt8: 0x18
N	Telegram size	UInt8: 0x05
PortNumber	Port Number	Uint8: 0x0N
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

5.17 Download DS Partially E_FNCID_DWNDS_SHORT 0x019

This telegram is optional. In some cases it is not possible for the Master gateway application to return the whole Data Storage in one telegram (E_FNCID_DWNDS). The telegram E_FNCID_DWNDS_SHORT provides means to return DSDATA in smaller chunks.

The Master responds with a result state.

```
HRQ    (E_FNCID_DWNDS_SHORT)
       [N]
       [PortNumber]
       [DSDownloadCommand]
       < -
       |
         [StartAddress]
         [BytesToGet]
       >
       [CHKS]
```

The response to the telegram contains the success status code.

Table 66: Download DS Partially Request.

Element	Description	Value
E_FNCID_DWNDS_SHORT	Function ID	UInt8: 0x19
N	Telegram size	UInt8: 0xnn
PortNumber	Port Number	UInt8: 0xN
DSDownloadCommand „DSDC" (see Table 21)	DS download related command	UInt8: 0xNN
StartAddress (placed only if DSDC=2 (Download))	Address of first transferred DSDATA byte, starting with 0	UInt16: 0xNNNN
Number of Bytes to get (placed only if DSUC=2 (Download))	Number of requested DSData	UInt16: 0xNNNN
CHKS	Checksum	UInt8: 0xnn

Depending on the content of the DSDownload Command byte the structure of the response telegram changes as described below:

```
MRS    (E_FNCID_DWNDS_SHORT)
       [N]
       [PortNumber]
       [StatusCode]
       <
        [DSSize]
       |
        [StartAddress]
        [BytesToGet]
        {[DSDATA]}
       |
        [Validity]
       >
       [CHKS]
```

Table 67: Download DS Partially Response

<table>
<tr><th>Element</th><th>Description</th><th>Value</th></tr>
<tr><td>E_FNCID_DWNDS_SHORT</td><td>Function ID</td><td>UInt8: 0x19</td></tr>
<tr><td><[N] | (EX) [N2]></td><td>Telegram size</td><td><
UInt8: 0xnn |
Uint8: 0
Uint16: 0xnnnn
></td></tr>
<tr><td>PortNumber</td><td>Port Number</td><td>Uint8: 0xNN</td></tr>
<tr><td>Status
(see Table 14)</td><td>Result state of port configuration</td><td>UInt8:0xnn</td></tr>
<tr><td>DSSize
(placed only if DSDC=1 (GetSize) in request</td><td>Size of DSData chunk in subsequent E_FNCID_DWNDS_SHORT (DSDC=2) telegram</td><td>UInt16: 0xNNNN</td></tr>
<tr><td>StartAddress
(placed only if DSDC=2)</td><td>Address of first transferred DSDATA byte, starting with 0</td><td>UInt16: 0xNNNN</td></tr>
<tr><td>Number of Bytes to get (placed only if DSUC=2 (Download))</td><td>Size of DSData to be transferred</td><td>UInt16: 0xNNNN</td></tr>
<tr><td>DSDATA (placed only if DSUC=2)</td><td>partial Data Storage Date</td><td>UInt8[ ]: {[0xNN]}</td></tr>
<tr><td>Validity (placed only if DSUC=3)</td><td>DS Validity</td><td>UInt8: 0xNN</td></tr>
<tr><td>CHKS</td><td>Checksum</td><td>UInt8: 0xnn</td></tr>
</table>

6.0 Extended Telegrams

The extended telegrams are not required by the IO-Link specification. They are nevertheless needed for a proper control of the Master:

Table 68: Extended Telegrams for IOLM4P communication with Host system

FID	ILS	Function ID	Telegram Name	Description
0x10		E_FNCID_FIRMWARE UPGRADEMODE	Go to Firmware Update mode	Switches Master into bootloader mode
0x11		E_FNCID_WRITEFLASH	Write to flash	Flashes a given bytestream to a given address in the local flash memory
0x12		E_FNCID_CHECKFLASHWRIT E	Check flash write	Execute XOR comparison with last flashed data values
0x13		E_FNCID_RESTART	Restart	Restarts the Master and forces to start the new FW
0x16		E_FNCID_MODIFYFWKEY	Set Secondary key	Pass secondary FW encryption key
0x61		E_FNCID_SETPROCESSDATA LOOPBACKMODE	Set Process Data Loop back mode	The process data loop-back mode (required for the physical layer test) can be switched on or off with this telegram
0x62		E_FNCID_GETPROCESSDATA LOOPBACKMODE	Set Process Data Loop back mode	The process data loop-back mode (required for the physical layer test) can be switched on or off with this telegram

6.1 Firmware Update Mode E_FNCID_FIRMWAREUPDATEMODE 0x10

The support of the firmware update mode for a productive host application is optional.

In a typical use case, the firmware update will be executed with the support of a specific configuration and parameterization tool that also supports firmware update. The firmware update files will be provided by TEConcept.

This telegram causes the Master to switch to the Firmware Update mode, by entering the bootloader. If the bootloader is active most of the telegrams are not longer supported.

```
HRQ (E_FNCID_FIRMWAREUPDATE-
    MODE)
    (N)
    [FWType]
    [CHKS]
```

Table 69: Firmware Update Mode Request

Element	Description	Value
E_FNCID_FIRMWAREUPDATEMODE	Function ID	UInt8: 0x10
N	Telegram size	UInt8: 0x04
FWType	New Firmware Type	Uint8: 0xNN
CHKS	Checksum	UInt8: 0xnn

The Master responds with a result state.

```
MRS     (E_FNCID_FIRMWAREUPDATE-
        MODE)
        (N)
        [Status]
        [CHKS]
```

Table 70: Firmware Update Mode Response

Element	Description	Value
E_FNCID_FIRMWAREUPDATEMODEN	Function ID	UInt8: 0x10
N	Telegram size	UInt8: 0xn4
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

6.2 Write To Flash E_FNCID_WRITEFLASH 0x011

A number ('L') of bytes can be written to the flash memory of the IOLM4P a location specified in the parameters. The address location is subject to interpretation of the IOLM4P and not necessarily identical to a physical location of the number ‚L' should have the divisor 4.

```
HRQ     (E_FNCID_WRITEFLASH)
        (N)
        [FWAddress]
        {[FWData]}
        [CHKS]
```

Table 71: Write To Flash Request

Element	Description	Value
E_FNCID_WRITEFLASH	Function ID	UInt8: 0x11
N	Telegram size L=N-7	UInt8: 0xNN
FWAddress	Logic start address of new FW	UInt16: 0xNNNN
FWData	Stream of L-Bytes	UIntt8[L]: {0xNN}
CHKS	Checksum	UInt8: 0xnn

The Master responds with a success status code,

```
MRS     (E_FNCID_FIRMWAREUPDATE-
        MODE)
        (N)
        [Status]
        [CHKS]
```

Table 72: Write To Flash Response

Element	Description	Value
E_FNCID_WRITEFLASH	Function ID	UInt8: 0x11
N	Telegram size	UInt8: 0xn4
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

6.3 Check Flash Write E_FNCID_CHECKWRITEFLASH 0x012

The identity of the flash content with the passed FWData is checked by this telegram and indicated in the telegram success status of the Master response.

```
HRQ     (E_FNCID_CHECKWRITE-
        FLASH)
        (N)
        [FWLength]
```

```
        [FWCS]
        [CHKS]
```

Table 73: Check Flash Write Request

Element	Description	Value
E_FNCID_CHECKWRITEFLASH	Function ID	UInt8: 0x12
N	Telegram size	UInt8: 0x08
FWLength	Total length of last firmware update	UInt32: 0xNNNNNNNN
FWCS	Checksum of last FW Update	UInt8: 0xNN
CHKS	Checksum	UInt8: 0xnn

The Master responds with a success status code.

```
MRS     (E_FNCID_CHECKWRITE-
        FLASH)
        (N)
        [Status]
        [CHKS]
```

Table 74: Check Write To Flash Response

Element	Description	Value
E_FNCID_CHECKWRITEFLASH	Function ID	UInt8: 0x12
N	Telegram size	UInt8: 0x04
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

6.4 Restart E_FNCID_RESTART 0x013

If called after successful firmware update master is restarted with new firmware.

```
HRQ     (E_FNCID_RESTART)
        (N)
        [CHKS]
```

Table 75: Restart Request

Element	Description	Value
E_FNCID_RESTART	Function ID	UInt8: 0x13
N	Telegram size	UInt8: 0x03
CHKS	Checksum	UInt8: 0xnn

The Master responds with a success status code.

```
MRS     (E_FNCID_RESTART)
        (N)
        [Status]
        [CHKS]
```

Table 76: Check Write To Flash Response

Element	Description	Value
E_FNCID_RESTART	Function ID	UInt8: 0x13
N	Telegram size	UInt8: 0x04
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

6.5 Set Secondary Key E_FNCID_MODIFYFWKEY 0x016

The secondary key might need to be set before a firmware update is executed

```
HRQ     (E_FNCID_MODIFYFWKEY)
        (N)
        [FW2KEY]
        [CHKS]
```

Table 77: Set Secondary Key Request

Element	Description	Value
E_FNCID_MODIFYFWKEY	Function ID	UInt8: 0x16
N	Telegram Size	UInt8: 0x13
FW2KEY	Secondary Key	UInt8[16]: {[0xNN]}
CHKS	Checksum	UInt8: 0xnn

The Master responds with a success status code.

```
MRS    (E_FNCID_MODIFYFWKEY)
       (N)
       [Status]
       [CHKS]
```

Table 78: MODIFY 2nd KEY response

Element	Description	Value
E_FNCID_MODIFYFWKEY	Function ID	UInt8: 0x16
N	Telegram size	UInt8: 0x04
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

6.5.1 Set PD Loopback E_FNCID_SETPROCESSDATALOOPBACKMODE 0x061

The process data loopback mode can be switched on or off with this telegram.The process data loopback mode configures the Master in line with the requirements of the EMC test (see [2], Annex G 2.2)

```
HRQ    (E_FNCID_SETPROCESSDATA-
       LOOPBACKMODE)
       (N)
       [PortNumber]
       [PDLoopBackMode]
       [CHKS]
```

Table 79: Set PD Loopback Request

Element	Description	Value
E_FNCID_SETPROCESSDATALOOPBACKMODE	Function ID	UInt8: 0x61
N	Telegram Size	UInt8: 0x5
PortNumber	Port Number	Uint8: 0xN
PDLoopBackMode	Enbables / Disabled Loopback Mode of Master	UInt8: 0x00=Off 0x01=On
CHKS	Checksum	UInt8: 0xnn

The Master responds with a success status code and the read PHY register content.

```
MRS    (E_FNCID_SETPROCESSDATA
       LOOPBACKMODE)
       (N)
       [Status]
       [CHKS]
```

Table 80: Set PD Loopback Response

Element	Description	Value
E_FNCID_SETPROCESSDATALOOPBACKMODE	Function ID	UInt8: 0x61
N	Telegram size	UInt8: 0x04
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
CHKS	Checksum	UInt8: 0xnn

6.5.2 Get PD Loopback E_FNCID_SETPROCESS

DATALOOPBACKMODE 0x061

The process data loopback mode can be queried with this telegram.The process data loopback mode configures the Master in line with the requirements of the EMC test (see [2], Annex G 2.2)

```
HRQ    (E_FNCID_GETPROCESSDATA-
       LOOPBACKMODE)
       (N)
       [PortNumber]
       [CHKS]
```

Table 81: Get PD Loopback Request

Element	Description	Value
E_FNCID_GET PROCESSDAT ALOOPBACKM ODE	Function ID	UInt8: 0x62
N	Telegram Size	UInt8: 0x4
PortNumber	Port Number	Uint8: 0xN
CHKS	Checksum	UInt8: 0xnn

The Master responds with a success status code and current status of the loopback mode

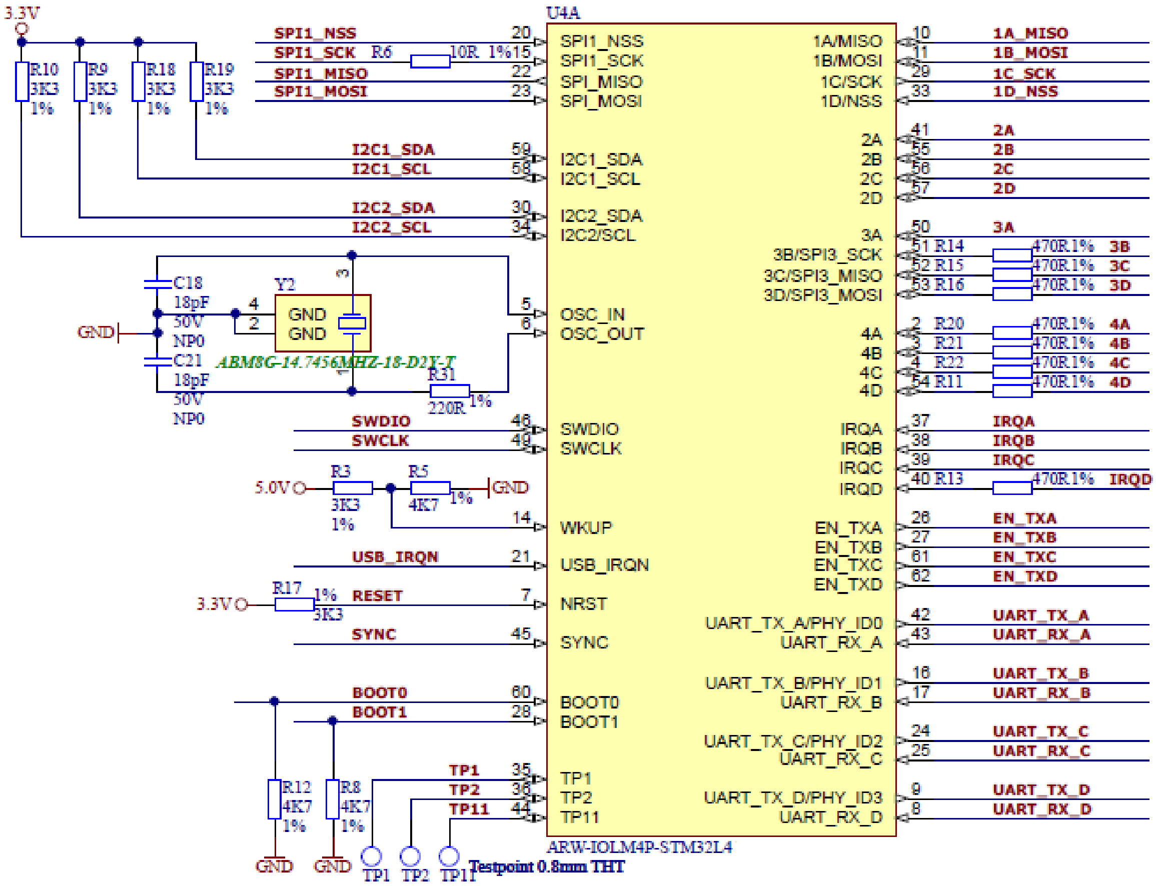
```
MRS    (E_FNCID_GETPROCESSDATA
       LOOPBACKMODE)
       (N)
       [Status]
       [PDLoopBackMode]
       [CHKS]
```

Table 82: Get Loopback mode response

Element	Description	Value
E_FNCID_SETP ROCESSDATA LOOPBACK MODE	Function ID	UInt8: 0x62
N	Telegram size	UInt8: 0x05
Status (see Table 14)	Result state of port configuration	UInt8:0xnn
PDLoopBack Mode	returns loopback mode	UInt8: 0x00=Off 0x01=On
CHKS	Checksum	UInt8: 0xnn

Table 82: Get Loopback mode response

Appendix A



Typical application for IOLM4P chip.

Appendix B

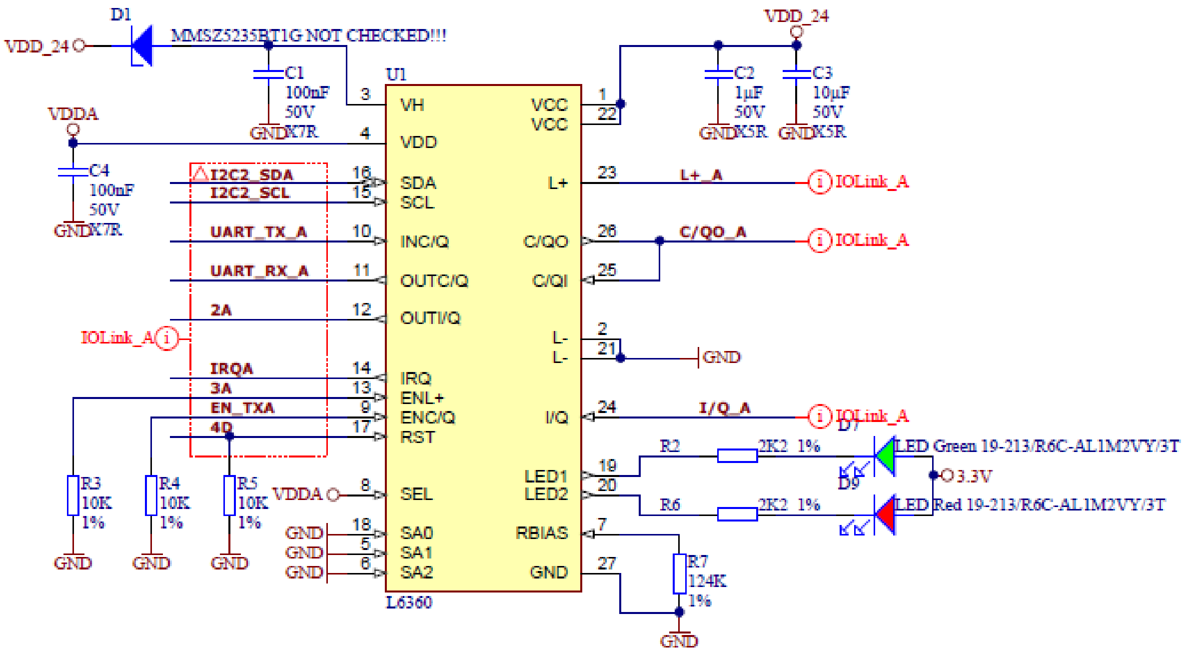


Figure 4. Typical application of L6360 [4]

Appendix C

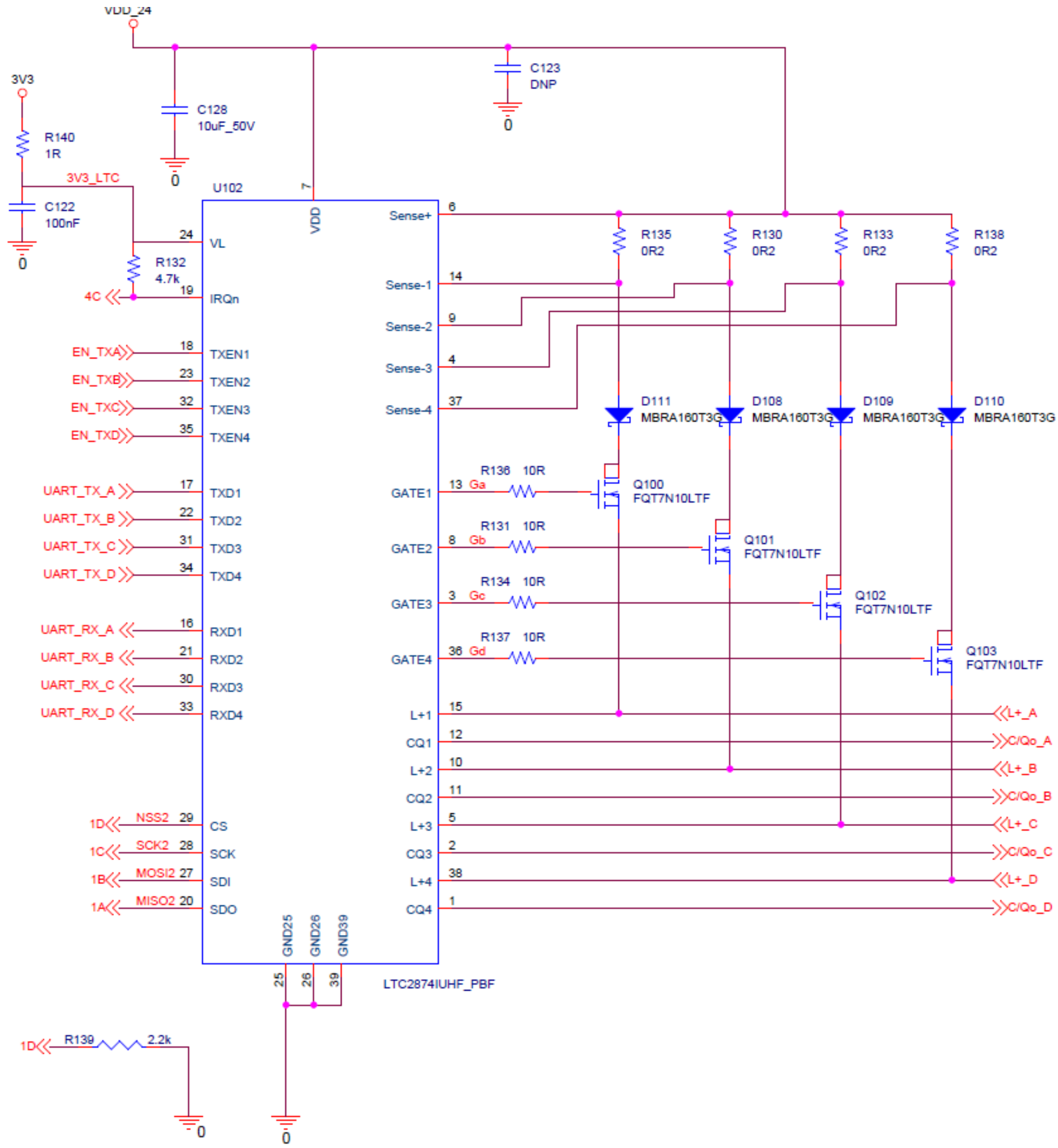


Figure 5. Typical Application of LTC2874

Revision History

- Revision 0.8 - 2018-07-01 (initial version)
- Revision 0.9 - 2018-11-01
  L6360-support, LTC2874 support
- Revision 0.9.1 - ReadPhyParam removed, I/Q table corrected, Weblinks updated

List of Tables

Table 1: Evaluation Kits 4
Table 2: Pin Description 6
Table 3: PHY Selection on Startup 10
Table 4: Shared PHY Signals 11
Table 5: Dedicated PHY Signals 11
Table 6: Dedicated L6360 Signals 11
Table 7: Shared LTC2874 Signals 12
Table 8: Port specific LTC2874 Signals 12
Table 9: SPI Host Interface Pins 13
Table 10: Telegram Element Description 15
Table 11: PDConfig Structure 16
Table 12: IQ Configuration 17
Table 13: IQ Configuration and Length 17
Table 14: General Return-status Code 17
Table 15: ISDU Status 18
Table 16: ISDU Request Type 18
Table 17: ISDU Answer Codes 18
Table 18: Port Operating Mode 18
Table 19: Port Cyclic Operation 18
Table 20: DS Upload Commands 19
Table 21: DS Download Commands 19
Table 22: Overview of Standard Telegrams 20
Table 23: Get Identification 21
Table 24: Identification Response 21
Table 25: INFO_ID and Data Content 21
Table 26: E_INFO_MASTERTYPE 22
Table 27: E_INFO_PORT_TYPE 22
Table 28: Set PD Interface Request 22
Table 29: Set PD Interface Response 22
Table 30: Get PD Interface Request 23
Table 31: Get PD Interface Response 23
Table 32: Control Status Request 23
Table 33: Control Status Response 24
Table 34: Port Control Flags 24
Table 35: Port Status Bits 24
Table 36: Port Status Fault Codes 25
Table 37: Set Port Configuration Request 26
Table 38: Set Port Configuration Response 26
Table 39: Get Set Port Configuration Request 27
Table 40: Get Port Configuration Response 27
Table 41: Set Data Storage Header Sequest 28
Table 42: Set Data Storage Header Response 28
Table 43: Get Data Storage Header Request 29

Table 44: Get Data Storage Header Response 29
Table 45: Upload DS to Master Request 29
Table 46: Upload DS to Master Response 29
Table 47: Download DS from Master Request 30
Table 48: Download DS from Master response 30
Table 49: ISDU Transfer Initialization Request 30
Table 50: ISDU Transfer Initialization Response 31
Table 51: ISDU Get Request 31
Table 52: ISDU Get Response 32
Table 53: Read Event Request 32
Table 54: Read Event Response 32
Table 55: EventStatus Codes 33
Table 56: EventInstance Codes 33
Table 57: EventMode Codes 33
Table 58: EventType Codes 33
Table 59: EventOrigin Codes 33
Table 60: Switch Off Power Request 33
Table 61: Switch Off Power response 34
Table 62: Switch On Power Request 34
Table 63: Switch On Power Response 34
Table 64: Upload DS Partially Request 34
Table 65: Upload DS Partially Response 35
Table 66: Download DS Partially Request 35
Table 67: Download DS Partially Response 36
Table 68: Extended Telegrams for IOLM4P communication with Host system 37
Table 69: Firmware Update Mode Request 37
Table 70: Firmware Update Mode Response 38
Table 71: Write To Flash Request 38
Table 72: Write To Flash Response 38
Table 73: Check Flash Write Request 39
Table 74: Check Write To Flash Response 39
Table 75: Restart Request 39
Table 76: Check Write To Flash Response 39
Table 77: Set Secondary Key Request 40
Table 78: MODIFY 2nd KEY response 40
Table 79: Set PD Loopback Request 40
Table 80: Set PD Loopback Response 40
Table 81: Get PD Loopback Request 41
Table 82: Get Loopback mode response 41

List of Figures

Figure 1. Connectivity Software Support for IOLM4P 5
Figure 2. IOLM4P package 6
Figure 3. Host SPI Timing, SPI clock < 10MHz, 0µs < t2<100µs, t1=200µs 14
Figure 4. Typical application of L6360 [4] 43
Figure 5. Typical Application of LTC2874 44

Important Notice

General Disclaimer

Information furnished by Arrow and TEConcept in this data sheet is believed to be accurate and reliable. However, no responsibility is assumed by Arrow and TEConcept for its use, nor for any infringements of patents or other rights of third parties, which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TEConcept.

Arrow and TEConcept and its subsidiaries reserve the right to make changes, corrections, enhancements, modifications, and improvements to their products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on products before placing orders.

Purchasers are solely responsible for the choice, selection, and use of Arrow products and Arrow assumes no liability for application assistance or the design of Purchasers products.

Arrow and the Arrow Logo are trademarks of Arrow, ST and the ST logo are trademarks of ST. TEConcept and the TEConcept Logo are trademarks of TEConcept.

All other product or service names are the property of their respective owners. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.