User Manual

1000BASE-T1 – SFP Module

Version 0.4
05. March 2018

Technica Engineering GmbH
Leopoldstraße 236
80807 München
Germany

Fax: +49-89-200-072430
Email: Info@technica-engineering.de
www.technica-engineering.de

Index

1 Feature List

The Technica Engineering 1000Base-T1 SFP Module fits into a standard Small Form-factor Pluggable slot.

It uses the SGMII and generates 1000 Mbit/s full-duplex.
Note: SERDES interface is not supported!

After power up, it self-configures to Automotive 1000Base-T1.

Registers of the integrated transceiver are accessible via I2C interface for diagnosis and re-configuration.

One Link LED shows link status.

Power requirement:	3.3 Volt DC +/- 0.03 Volt
Size:	68 x 14 x 14 mm
Weight:	0,1 kg
International Protection:	IP 2 0
Operating Temperature:	0 to +70 °Celsius

2 Warranty and Safety Information

Before operating the device, read this manual thoroughly and retain it for your reference.

Use the device only as described in this manual.
Use only in dry conditions.
Do not apply power to a damaged device.

Do not open the device. Otherwise warranty will be lost.

This device is designed for engineering purpose only.
Special care has to be taken for operation.
Do not use this device in a series production car.
As this device is likely to be used under rough conditions, warranty is limited to 1 year.
Manufacturer liability for damage caused by using the device is excluded.

3 Pinning

The 1000Base-T1 line is connected by a Molex connector.

Molex 0533250260 Header 2.0mm
Molex 510900200 Housing
Molex 50212-8000 Crimp Contact

Pin	Function	Pin	Function
1	1000Base-T1 Plus	2	1000Base-T1 Minus

SFP Socket connector:

Pin	Function	Pin	Function
1	GND	20	GND
2	Tx_Faul – Connected to GND	19	SGMII_TXD_N
3	n.c.	18	SGMII_TXD_P
4	I2C_DAT	17	GND
5	I2C_CLK	16	3.3 Volt
6	GND	15	3.3 Volt
7	n.c.	14	GND
8	Rx_Los – Connected to GND	13	SGMII_RXD_P
9	n.c.	12	SGMII_RXD_N
10	GND	11	GND

4 Startup and Configuration

4.1 Startup

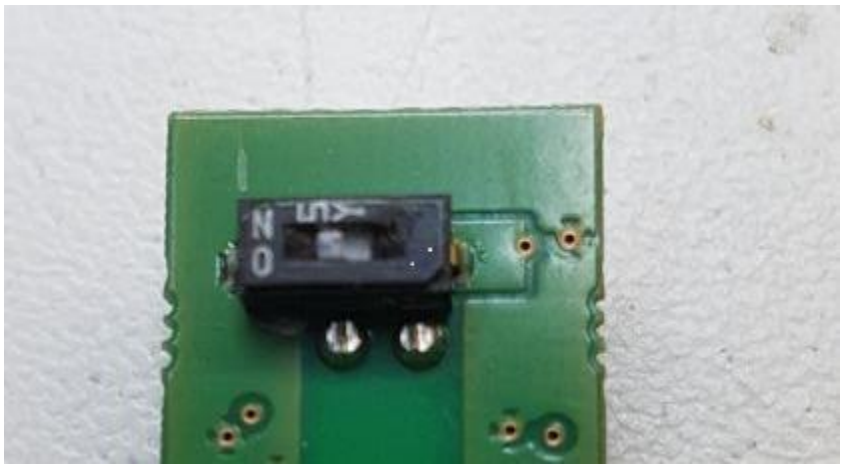
After 3.3 Volt power is applied, the SFP module starts up and self-configures the 88Q2112_Z1 transceiver by MDIO interface. This lasts 100ms. Do not apply any I2C activity on the bus during this time.

4.2 Self-Configuration

The SFP Module configures itself to Automotive 1000Base-T1 after power up.

Master / Slave Configuration is done according to the DIP switch on the bottom of the device.
The lock has to be opened if reconfiguration of the DIP switch is desired (see picture below).

ON/left = Master
OFF/right = Slave



5 I2C Interface

5.1 I2C specifications

I2C can be used 100ms after the power up of the module.

The module operates with f_{SCL} up to 53KHz without requiring clock stretching.
The module may clock stretch with f_{SCL} greater than 53KHz and up to 400 kHz.

The module processor listens as slave on the 7-bit address 0x50.
Note: b1010 000X = 0xA0
Read access beyond address 95 will return 0x00.

The 88Q2112_Z1 transceiver can be accessed at I2C slave 7-bit address 0x40. **Note:** b1000 000X = 0x80
The 88Q2112_Z1 PHY does not support I2C interfaces. However, the microcontroller acts as a bridge between the host and the PHY. Commands from the Host are processed by the microcontroller. The microcontroller accesses to the PHY trough MDIO interface and forwards the information to the Host.

5.2 Module processor I2C register map

Memory Map (Read only registers):

Data Bytes	Byte Number	Comment
0x03,	0	Identifier SFP
0x04,	1	Ext. Identifier
0x80,	2	Connector
0x00, 0x00, 0x00, 0x00,	3-6	Transceiver high
0x00, 0x00, 0x00, 0x00,	7-10	Transceiver low
0x00,	11	Encoding
0x01,	12	Bitrate Nominal in 100 MBit
0x00,	13	Reserved
0x00,	14	Link Length Fiber
0x00,	15	Link Length Fiber
0x00,	16	Link Length Fiber
0x00,	17	Link Length Fiber
0x0A,	18	Link Length Copper in meter
0x00,	19	Reserved
'T', 'e', 'c', 'h', 'n', 'i', 'c', 'a', ' ', 'E', 'n', 'g', '.', ' ', ' ', ' ',	20-35	Vendor Name
0x00,	36	Reserved
0x00, 0x00, 0x00,	37-39	Vendor ID
'1', '2', '3', '4', '5', '6', '7', '8', '9', '0', '1', '2', '3', '4', '5', '6',	40-55	PartNumber
0x00, 0x00, 0x00, 0x00,	56-59	Revision Number
0x00, 0x00, 0x00,	60-62	Reserved
0xBC,	63	Check Code for Field 0-62
0x00, 0x00,	64-65	Options
0x00,	66	Bitrate max
0x00,	67	Bitrate min
'1', '2', '3', '4', '5', '6', '7', '8', '9', '0', '1', '2', '3', '4', '5', '6',	68-83	Serial Number String
0x00, 0x00, 0x00, 0x00,	84-87	Date Code high
0x00, 0x00, 0x00, 0x00,	88-91	Date Code low
0x00, 0x00, 0x00,	92-94	Reserved
0x42	95	Check Code Extended for Field 64-94

5.3 I2C Device addressing and operation

5.3.1 I2C Current Address Read

The current read operation only requires the I2C device read word to be sent. When the acknowledge is received from the SFP module, the current address data word is serially clocked out.

Example: Read the current address of the SFP module (b1010000X)

		<-I2C_device ->																		
HOST	START	MSB						LSB	READ										NACK	STOP
		1	0	1	0	0	0	0	1	0	X	X	X	X	X	X	X	X	1	
SFP										ACK	MSB							LSB		
											<-DATA WORD->									

5.3.2 Random Read

The random address read requires two operations to perform the read.

Example: Read a random address of the SFP module (I2C device = b1010000X)

First a write operation to specify the address desired to read:

		<-I2C_device ->									<-I2C Memory address->								
HOST	START	MSB						LSB	WRITE		MSB							LSB	
		1	0	1	0	0	0	0	0	0	X	X	X	X	X	X	X	X	0
SFP										ACK									ACK

Then a read operation to read the previous address specified:

		<-I2C_device ->																		
HOST	START	MSB						LSB	READ										NACK	STOP
		1	0	1	0	0	0	0	1	0	X	X	X	X	X	X	X	X	1	
SFP										ACK	MSB							LSB		
											<-DATA WORD->									

5.3.3 Sequential read

The sequential reads are started by either a current word address read or a random address read. To specify a sequential read, the host responds with an acknowledge instead of a stop after each data word.

First a write operation to specify the address desired to read:

		<-I2C_device ->									<-I2C Memory address->								
HOST	START	MSB						LSB	WRITE		MSB							LSB	
		1	0	1	0	0	0	0	0	0	X	X	X	X	X	X	X	X	0
SFP										ACK									ACK

Then the read operations:

		<-I2C_device ->																														
HOST	START	MSB						LSB	READ										ACK											NACK	STOP
		1	0	1	0	0	0	0	1	0	X	X	X	X	X	X	X	X	1	X	X	X	X	X	X	X	X	1	
SFP										ACK	MSB							LSB		MSB							LSB				
											<-DATA WORD n->									<-DATA WORD n+1->											

5.3.4 Byte Write

The write operation requires 8-bits of data word address following the device address write word and acknowledgement.

Example: Byte write operation into the SFP module (b1010000X)

		<-I2C_device ->									<-MEMORY ADDRESS->									<-DATA WORD->										
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	1	0	0	0	0	0	0	X	X	X	X	X	X	X	X	0	X	X	X	X	X	X	X	X	0	
SPF										ACK									ACK									ACK	

5.3.5 Sequential write

The sequential write is started in the same way as a single byte write, but the host master does not send a stop condition after the first word is clocked in.

		<-I2C_device ->									<-MEMORY ADDRESS->									<-DATA WORD 1->									<-DATA WORD 2->										
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		MSB							LSB		STOP
		1	0	1	0	0	0	0	0	0	X	X	X	X	X	X	X	X	0	X	X	X	X	X	X	X	X	0	X	X	X	X	X	X	X	X	0	
SPF										ACK									ACK								ACK									ACK		

5.4 I2C access to the 88Q2112_Z1 transceiver

The 88Q2112_Z1PHY listens as slave on the 7-bit address 0x40. Every internal register of the PHY (16 bits) is accessed by defining the Device Number (1 byte) and the Register Ad-dress (2 bytes). The I2C Memory address is mapped as:

Address	Register name
0x00	Device number
0x01	Register_address_MSB
0x02	Register_address_LSB
0x03	PHY_Register_Operation
0x04	PHY_register_data_MSB
0x05	PHY_register_data_LSB

5.4.1 Register description

Device number
Defines the device number of the register to access

Register_address_MSB
Defines the Most significant byte of the register to access

Register_address_LSB
Defines the Less Significant Byte of the register to access

PHY_Register_Operation_Status

- **Bit 0 – Read Start Condition Flag (Read/Write)**
  This flag is set by the master after specifying the first 3 bytes of the register (Devices number, Register_address_MSB and Register_address_LSB) when a read operation is trigged.

- **Bit 1 – Read In Progress Flag (Read)**
  This flag is set by the slave during the reading operation

- **Bit 2 – Read Operation Done Flag (Read)**
  This flag is set by the slave when the reading operation is finished. This flag can be read after triggering the read to ensure that the data has finished reading.

- **Bit 3 – Write Start Condition Flag (Read)**
  This flag is set by the slave when a write operation is trigged.

- **Bit 4 – Write In Progress Flag (Read)**
  This flag is set by the slave during the writing operation

- **Bit 5 – Write Operation Done Flag (Read)**
  This flag is set by the slave when the writing operation is finished. This flag can be read after writing the PHY_register_data_LSB to ensure that the data has finished writing

PHY_register_data_MSB
When a reading operation, this register contains the MSB of the PHY register.
When a write operation, this register contains the MSB to write in the PHY register

PHY_register_data_LSB
When a reading operation, this register contains the LSB of the PHY register.
When a write operation, this register contains the LSB to write in the PHY register

5.5 PHY register access examples

5.5.1 Read Operation

The Read Operation is performed by accessing the PHY I2C device and loading the PHY register address into the appropriate I2C registers.
The register memory can be specified either with I2C Random Access or I2C Sequential Access

Example: Read the PHY's OUI using I2C **Random Access**. Register 1.0002 (Device number 1, register 0x0002)

		<-I2C_device ->									<-I2C Memory address->									<- Device_Number>									
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
SFP										ACK									ACK									ACK	
		0x00									0x01																		

		<-I2C_device->									<-I2C Memory address->									<Register_address_MSB>									
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0		
SFP										ACK									ACK								ACK		
											0x01									0x00									

		<-I2C_device->									<-I2C Memory address->									<Register_address_LSB>									
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0		
SFP										ACK									ACK								ACK		
											0x02									0x02									

		<-I2C_device->									<-I2C Memory address->									PHY_Register_Operation									
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0		
SFP										ACK									ACK								ACK		
											0x03									0x01									

		<-I2C_device->									<-I2C Memory address->									<-I2C_device->																					
HOST	START	MSB						LSB	WRITE		MSB							LSB		START	MSB						LSB		READ												NACK	STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0		1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	
SFP										ACK									ACK										ACK	MSB								LSB			
											0x04																			PHY_register_data_MSB											

		<-I2C_device->									<-I2C Memory address->									<-I2C_device->																			
HOST	START	MSB						LSB	WRITE		MSB							LSB		START	MSB						LSB	READ										NACK	STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0		1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	1	1	1	
SFP										ACK									ACK										ACK	MSB							LSB		
											0x05																			PHY_register_data_LSB									

Example: Read the PHY’s OUI using I2C **Sequential Access**. Register 1.0002 (Device number 1, register 0x0002)

		<-I2C_device->									<-I2C Memory address->									< Device_Number>									<Regis-ter_address_MSB>								
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		MSB							LSB	
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
SFP										ACK									ACK									ACK									ACK
											0x00									0x01									0x00								

	<Register_address_LSB>									<PHY_Register_Operation>									<-I2C_device->																	
HOST	MSB									MSB							LSB		MSB						LSB	READ										ACK
	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
SPF									ACK									ACK									ACK	MSB							LSB	
	0x02									0x01																		PHY_register_data_MSB								

HOST									NACK	STOP
	0	0	1	0	1	0	0	1	1	
SPF	MSB							LSB		
	PHY_register_data_LSB									

5.5.2 Write Operation

The Write Operation requires the PHY address word and the register memory.
The register memory can be specified either with I2C Byte Write or I2C Sequential Write

Example: Set the PHY to Slave configuration using I2C **Byte Write**. Value: 0x8001.Register 1.0834 (Device number 1, register 0x0834)

		<-I2C_device ->									<-I2C Memory address->									<- Device_Number>									
HOST	START	MSB							LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
SFP											ACK									ACK									ACK	
												0x00									0x01									

		<-I2C_device->									<-I2C Memory address->									<Register_address_MSB>									
HOST	START	MSB							LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	
SFP											ACK									ACK									ACK	
												0x01									0x80									

		<-I2C_device->									<-I2C Memory address->									<Register_address_LSB>									
HOST	START	MSB							LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	
SFP											ACK									ACK									ACK	
												0x02									0x01									

		<-I2C_device->									<-I2C Memory address->									PHY_Register_Operation										
HOST	START	MSB							LSB	WRITE		MSB							LSB		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	
SFP											ACK									ACK								ACK		
											0x03										0x00									

		<-I2C_device->									<-I2C Memory address->									<-I2C_device->									PHY_register_data_MSB									
HOST	START	MSB							LSB	WRITE		MSB							LSB		START	MSB							LSB	WRITE		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0		1	0	0	0	0	0	0	0		1	0	0	0	0	0	0	0	0	
SFP										ACK									ACK									ACK									ACK		
											0x04																				0x80									

		<-I2C_device->									<-I2C Memory address->									<-I2C_device->									PHY_register_data_LSB									
HOST	START	MSB							LSB	WRITE		MSB							LSB		START	MSB							LSB	WRITE		MSB							LSB		STOP
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0		1	0	0	0	0	0	0	0		0	0	0	0	0	0	0	1	0	
SFP										ACK									ACK									ACK									ACK		
											0x05																				0x01									

Example: Set the PHY to Master configuration using I2C **Sequential Write**. Value: 0xC001.Register 1.0834 (Device number 1, register 0x0834)

		<-I2C_device->									<-I2C Memory address->									< Device_Number>								<Regis-ter_address_MSB>								
HOST	START	MSB						LSB	WRITE		MSB							LSB		MSB							LSB		MSB							LSB	
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0
SFP										ACK									ACK								ACK									ACK
											0x00									0x01								0x08								

	<Register_address_LSB>									<PHY_Register_Operation>																											
HOST	MSB									MSB								LSB		MSB							LSB		MSB							LSB		STOP
	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
SPF									ACK									ACK								ACK									ACK		
	0x34									0x00									PHY_register_data_MSB								PHY_register_data_LSB									

6 Contact

If you have any questions regarding this product please do not hesitate to contact us:

Technica Engineering GmbH
Leopoldstraße 236
80807 München
Germany

Telefon: +49-89-200-072410
Fax: +49-89-200-072430

Info@technica-engineering.de

www.technica-engineering.de