Communication Protocol
I²C & SPI - 52 Series
AN -11

Date: 30-Aug-2019

Version: 2

I²C - Read Example(without CRC):

Byte#		0									1										2									
Send By Master		0xD8									0x2E										0xD9									
	S	1	1	0	1	1	0	0	0	A	0	0	1	0	1	1	1	0	A	P	S	1	1	0	1	1	0	0	1	A
		Address(0x6C)							W		Command											Address(0x6C)							R	

Byte#	0		1		2		3		4		5		
Receive From Slave	Low Byte	A	High Byte	A	Low Byte	A	High Byte	A	Low Byte	A	High Byte	A	P
	Temperature				Pressure				Status				

- S : Start bit
- P : Stop bit
- A : ACK
- W: i2c write mode
- R: i2c read mode
- Length = 6 bytes to read

Below equation is used to convert temperature ADC counts into deg C:
(ADC+32000)/64000*165 - 40
of which the temperature ADC counts from -32000 to +32000, which represents temperature value from -40 to +125 deg C.

```
void ReadPressure_noCRC() {
  Wire.beginTransmission(0x6C); // transmit to device #address = 0x6C
  Wire.write(0x2E);        // sends five bytes
  Wire.endTransmission();    // stop transmitting
  Wire.requestFrom(0x6C, 6); //Request registers, note that registers 2 bytes wide
//
  for (int i = 0; i < 3; i++)
  {
    buffer[i] = Wire.read(); // read low byte
    buffer[i] = buffer[i] | (Wire.read() << 8); // read high byte
  }
//
  dsp_t_sync = buffer[0]; //dsp_t register (corrected temperature measurement)
  dsp_s_sync = buffer[1]; //dsp_s register (corrected pressure measurement)
  status_r = buffer[2]; //status_sync register

  Serial.println("DSP_S: " + String(dsp_s_sync));
  Serial.println("DSP_T: " + String(dsp_t_sync));
  Serial.println("Status:" + String(status_r, BIN) + "( 0x" + String(status_r, HEX) + " )");
}
```

I²C - Read Example(with CRC):

Byte#		0									1									2										3								
Send By Master		0xDA									0x2E									0x5B										0xDB								
	S	1	1	0	1	1	0	1	0	A	0	0	1	0	1	1	1	0	A	0	1	0	1	1	0	1	1	A	P	S	1	1	0	1	1	0	1	1	A
		Address(0x6D)							W		Command									legnth-1				CRC-4								Address(0x6D)							R	

Byte#	0		1		2		3		4		5		6		
Receive From Slave	Low Byte	A	High Byte	A	Low Byte	A	High Byte	A	Low Byte	A	High Byte	A		A	P
	Temperature				Pressure				Status				CRC-8		

- S : Start bit
- P : Stop bit
- A : ACK
- W: i2c write mode
- R: i2c read mode
- Length = 6 bytes to read

TheCRC4andCRC8fieldsarecomputedinthesamebitandbyteorderasthetransmissionoverthe bus.
Their polynomial are:

- CRC4: polynomial: 0x03 initialization value: 0x0F
- CRC8: polynomial: 0xD5 initialization value: 0xFF

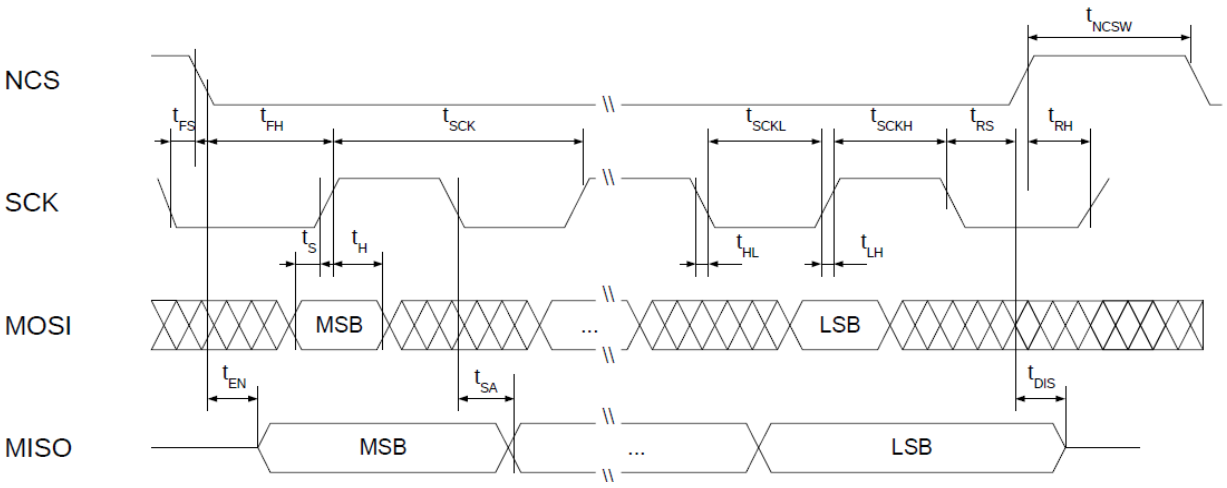
```
void ReadPressure_CRC() {
  Wire.beginTransmission(0x6D); // transmit to device #address = 0x6D
  Wire.write(0x2E);       // sends five bytes
  Wire.write(CRC4);       //  write CRC4
  Wire.endTransmission();    // stop transmitting
  Wire.requestFrom(0x6D, 7); //Request registers, note that registers 2 bytes wide
//
  for (int i = 0; i < 3; i++)
  {
    buffer[i] = Wire.read(); // read low byte
    buffer[i] = buffer[i] | (Wire.read() << 8); // read high byte
  }
//
  CRC8 = Wire.read();     //Read CRC-8
  dsp_t_sync = buffer[0]; //dsp_t register (corrected temperature measurement)
  dsp_s_sync = buffer[1]; //dsp_s register (corrected pressure measurement)
  status_r = buffer[2]; //status_sync register

  Serial.println("DSP_S: " + String(dsp_s_sync));
  Serial.println("DSP_T: " + String(dsp_t_sync));
  Serial.println("Status:" + String(status_r, BIN) + "( 0x" + String(status_r, HEX) + " )");
  Serial.println("CRC_8 " + String(CRC8));
}
```

- Pressure is from -32000 (0x8300) to 32000(0x7D00)
- STATUS:
  Bit 0 : 1: idle, 0: busy
  Bit 1 : 1: ADC_P register has been updated. Cleared when ADC_P is read
  Bit 2 : 1: ADC_T register has been updated. Cleared when ADC_S is read
  Bit 3 : 1: Pressure register has been updated. Cleared when Pressure is read
  Bit 4 : 1: Temperature register has been updated. Cleared when Temperature is read
  Bit 5 : 0
  Bit 6 : 0
  Bit 7 : 1: bridge supply failure occurred
  Bit 8 : 1: sensor bridge check failure occurred
  Bit 9 : 1: sensor acquisition chain diagnostics failure occurred
  Bit 10 : 0
  Bit 11 : 0
  Bit 12 : 0
  Bit 13 : 0
  Bit 14 : 0
  Bit 15 : 0
  Bit 16 : 0
- CRC4: polynomial: 0x03 initialization value: 0x0F
- CRC8: polynomial: 0xD5 initialization value: 0xFF

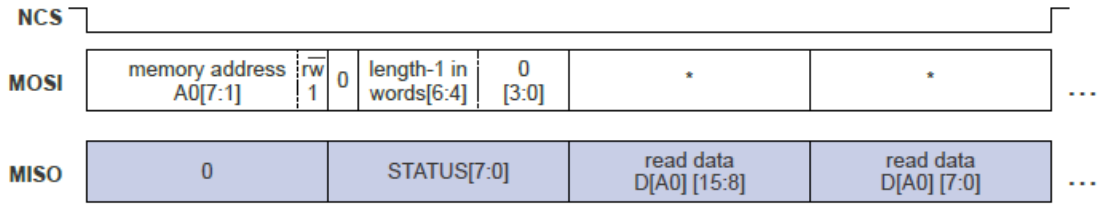
SPI - Timing Behavior

Cpha (Clock Phase of pin) = 0, 1st edge sample 2nd edge shift (adjustable)
Spol (Clock Polarity of pin) = 0, Clock off level 0 (adjustable)



SPI - Read Example(w0/Checksum):

Byte#	0	1	2	3	4	5	6	7
MOSI	0x2F	0x20	*	*	*	*	*	*
MOSI Comment	Command 0x2E+1 (byte memory address)	0x20 =0b00100000 0 : Frame without CRC 010 : length-1 =3 (words) 0000 : should be 0	Ignore					
MISO	0x00	0x1E	Low_Byte	High_Byte	Low_Byte	High_Byte	0x00	0x1E
MISO Comment		Status	Temperature		Pressure		Status	



Headquarter Switzerland:
Pewatron AG
Thurgauerstrasse 66
CH-8050 Zurich
Phone +41 44 877 35 00
info@pewatron.com

Office Germany:
Pewatron Deutschland GmbH
Edisonstraße 16
D-85716 Unterschleißheim
Phone +49 89 374 288 87-0
info.de@pewatron.com

PEWATRON
SENSORS · POWER SOLUTIONS

We are here for you. Addresses and Contacts.

Sales Germany & Austria

Postcode 00000 – 31999
Postcode 38000 – 39999
Postcode 80000 – 99999
Austria

Kurt Stritzelberger

Phone +49 89 260 52 80
Mobile +49 171 803 41 35

kurt.stritzelberger@pewatron.com

Postcode 32000 – 37999
Postcode 40000 – 79999

Gerhard Vetter

Phone +49 674 394 75 75
Mobile +49 163 762 74 30

gerhard.vetter@pewatron.com

Geometrical sensors
Sensor elements

Thorsten Ravagni

Phone +49 60 479 53 627

thorsten.ravagni@pewatron.com

Sales Switzerland & Liechtenstein

Postcode 3000 – 9999

Basil Frei

Phone +41 44 877 35 18
Mobile +41 76 279 37 26

basil.frei@pewatron.com

Postcode 1000 – 2999

Christian Mohrenstecher

Mobile +41 76 444 57 93

christian.mohrenstecher@pewatron.com

Sales International Key Accounts

Peter Felder

Phone +41 44 877 35 05
Mobile +41 79 406 49 83

peter.felder@pewatron.com

Sales Other Countries / Product Management

Pressure Sensors

Philipp Kistler
Phone +41 44 877 35 03
philipp.kistler@pewatron.com

Accelerometers / Level
Flow sensor elements

Thorsten Ravagni
Phone +49 60 479 53 627
thorsten.ravagni@pewatron.com

Drive technology
CH Postcode 5000 – 9999 / DE

Roman Homa
Mobile +41 76 444 00 86
roman.homa@pewatron.com

Gas sensors / Gas sensor modules
Load cells

Dr. Thomas Clausen
Phone +41 44 877 35 13
thomas.clausen@pewatron.com

Power supplies

Sebastiano Leggio
Phone +41 44 877 35 06
sebastiano.leggio@pewatron.com

Drive technology
CH Postcode 1000 – 4999 / AT / IT / FR

Christian Mohrenstecher
Mobile +41 76 444 57 93
christian.mohrenstecher@pewatron.com

Flow / Level / Medical products

Dr. Adriano Pittarelli
Phone +49 8245 774 95 44
adriano.pittarelli@pewatron.com

Linear position sensors
Angle sensors

Eric Letsch
Phone +41 44 877 35 14
eric.letsch@pewatron.com

Current sensors
Power solutions

Osman Coban
Phone +49 71 635 363 898
osman.coban@pewatron.com