Important Design Information for Hi-Resolution Digital DLHx Pressure Sensors

The sensor EEPROM does not have a write-protect, which makes it easy to corrupt the stored values during development. Sending a received status and data buffer of 3 bytes ***back*** to the sensor, over SPI, will cause writing to the active EEPROM area.

SENDING UNDOCUMENTED COMMANDS TO SENSOR WILL CORRUPT CALIBRATION AND IS NOT COVERED BY WARRANTY

Digital Interface Command Considerations – I2C

- When requesting the start of a measurement, the command length for I2C is 1 byte.
- When requesting sensor status over I2C, the host simply performs a 1-byte read transfer.
- When reading sensor data over I2C, the host simply performs a 7-byte read transfer.

Digital Interface Command Considerations - SPI

- When requesting the start of a measurement, the command length for SPI is 3 bytes, to be sent on MOSI.
- When requesting sensor status over SPI, the host MUST send the Status Read command byte (0xF0) on MOSI while reading 1 byte on MISO.
- When reading sensor data over SPI, the host MUST send the 7-byte Data Read command (0xF0 0x00 0x00 0x00 0x00 0x00 0x00) on MOSI while reading the data on MISO.

The status byte is showing as 0x44, (memory error= EEPROM checksum fail)?

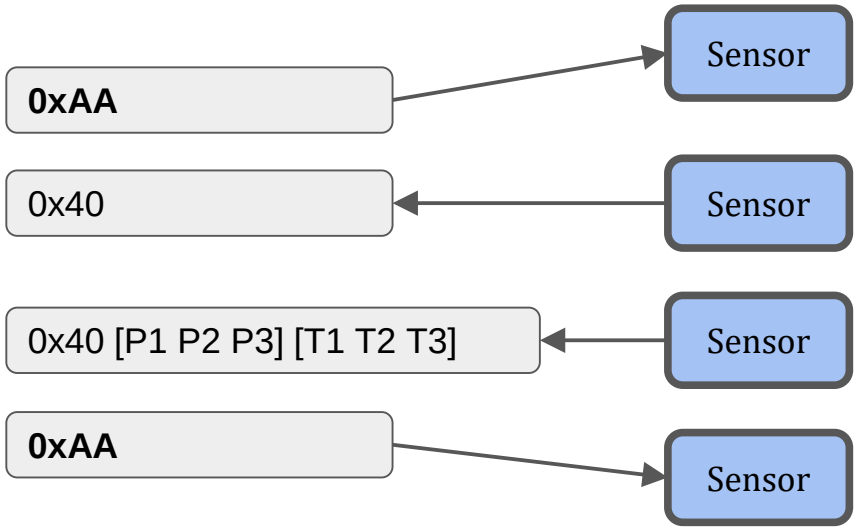
Sending 0x40 as the first byte with SPI will cause the following 2 bytes to be written to EEPROM. Generally this results in the error status, not functional failure, as 0x40 is for lot tracking and not operating parameters.

Any other value, from 0x42 to 0x57, sent as first byte, will likely affect operation.

DLH/R EEPROM corruption

- Only happens in development, *most* likely when changing from I2C to SPI communication in prototype testing.

- I2C: Correct operation:
  1. *Write*: Sensor Reading Request
  2. *Read:* Sensor Status
  3. *Read:* Sensor Output
  4. *Write*: Sensor Reading Request

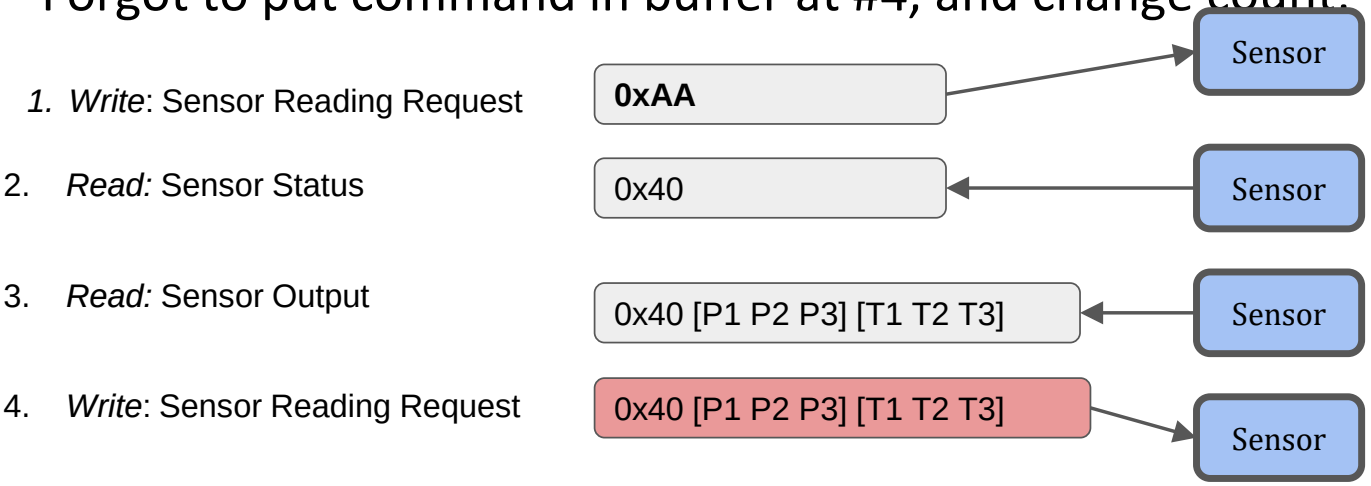


- Normal status = 0x40 (64 decimal).
  Busy = 0x60 (96 decimal).
  EEPROM error = 0x44 (68 decimal).

DLH/R EEPROM corruption

- It *is* possible to corrupt EEPROM using I2C...
- I2C: *Incorrect* operation:

  Forgot to put command in buffer at #4, and change count:



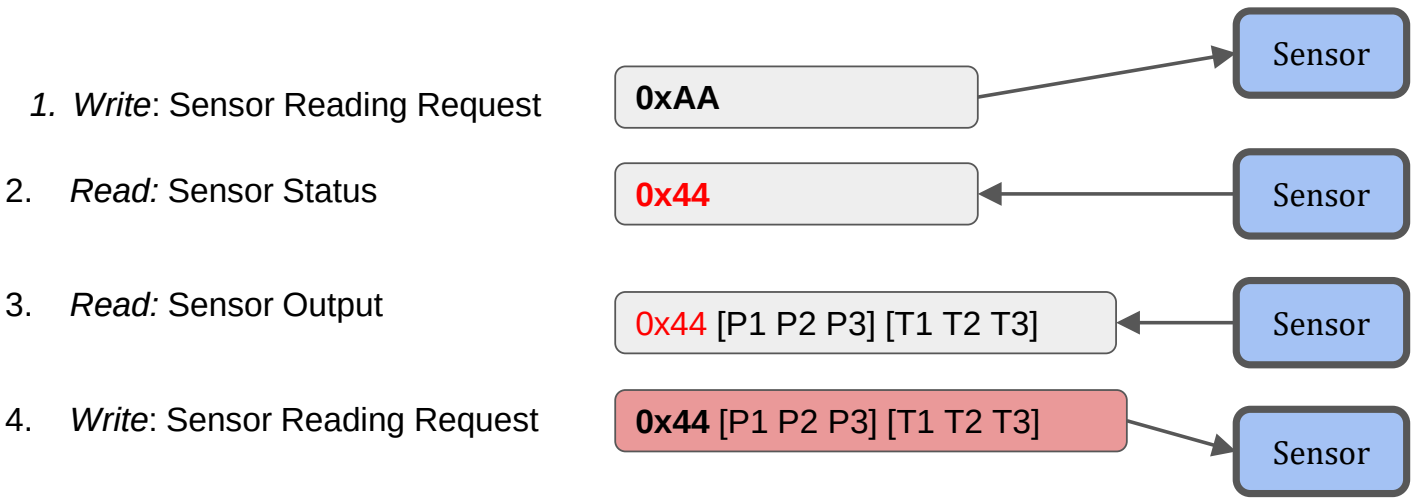
  Wrong Copy/paste of code from #3 to #4, like
  `I2C_Read(&buf,7)` to **`I2C_Write(&buf,7)`**

- #4 will write [P1 P2] to address 0, Lot ID.

DLH/R EEPROM corruption

- I2C: *Incorrect* operation, continued:

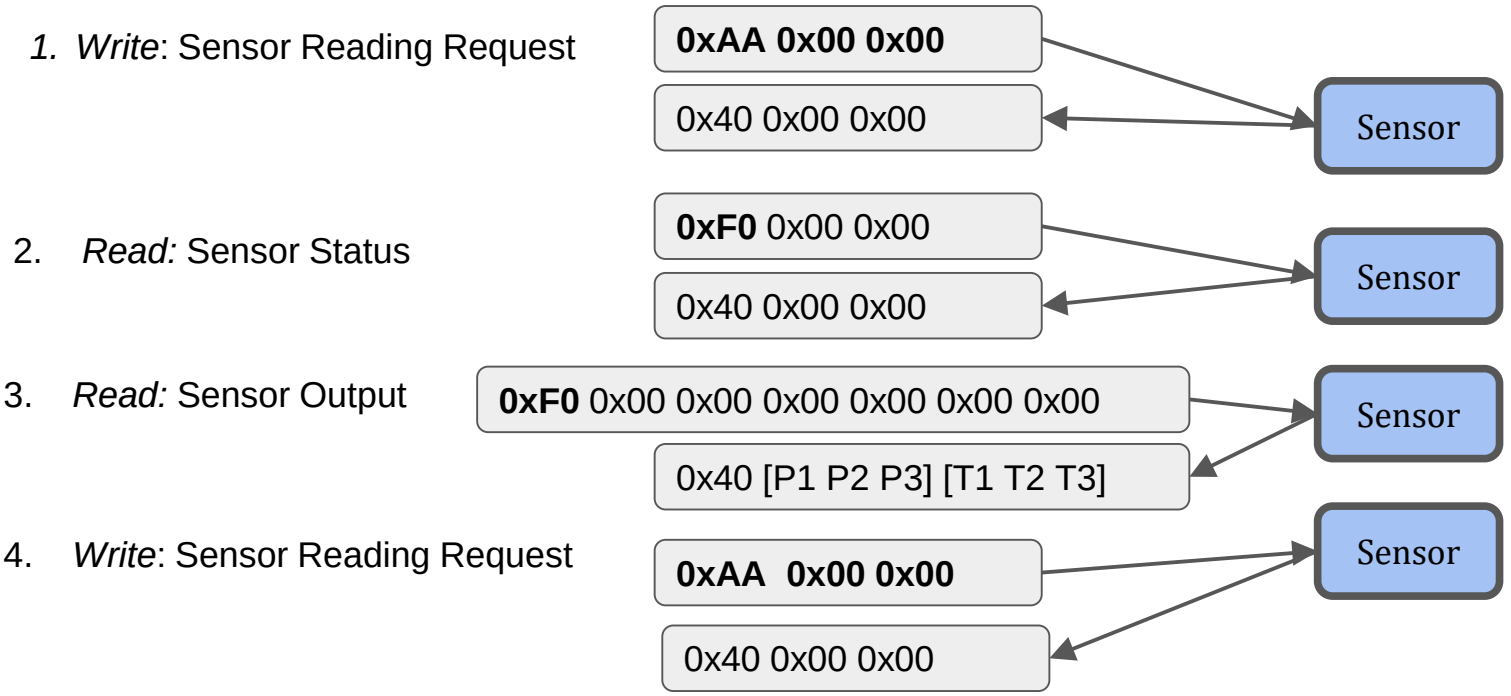
  After power-off and power on: previous Write to Address 0 causes EEPROM checksum error, status 0x44:



- #4 will write [P1 P2] to address 4, calibration Gain factor. *Now readings will be affected.*

DLH/R EEPROM corruption

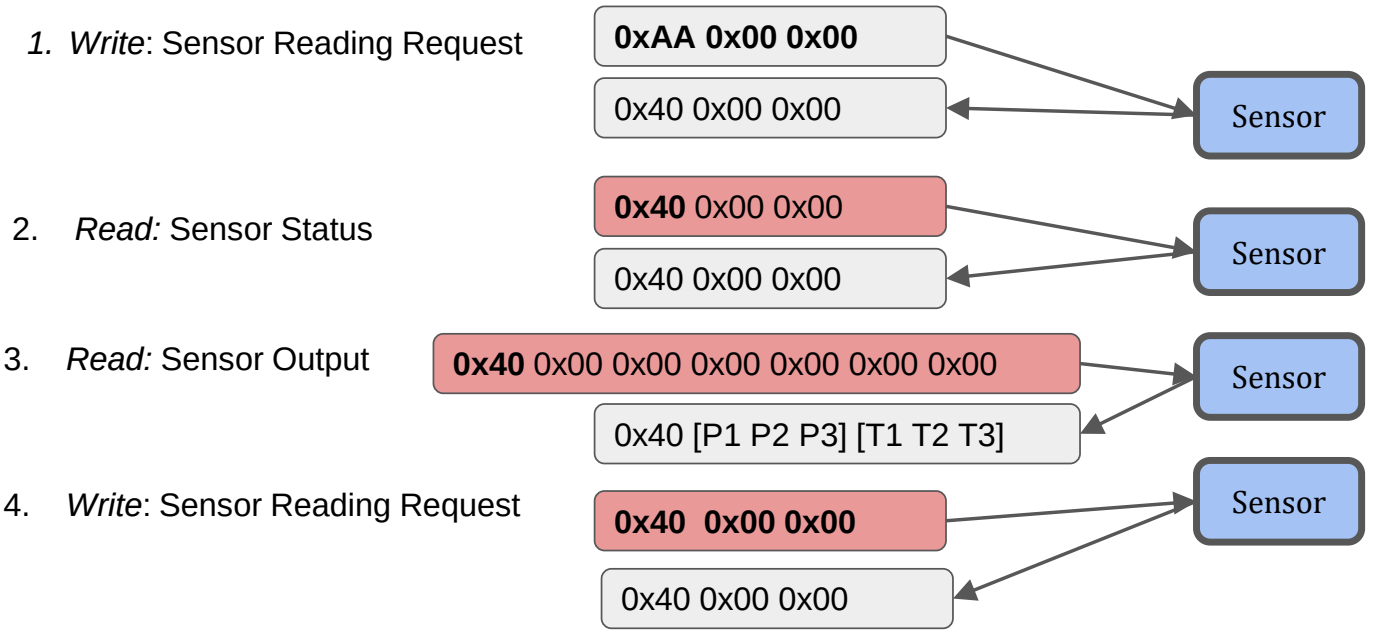
- SPI: Correct operation:



- Normal status = 0x40 (64 decimal).

DLH/R EEPROM corruption

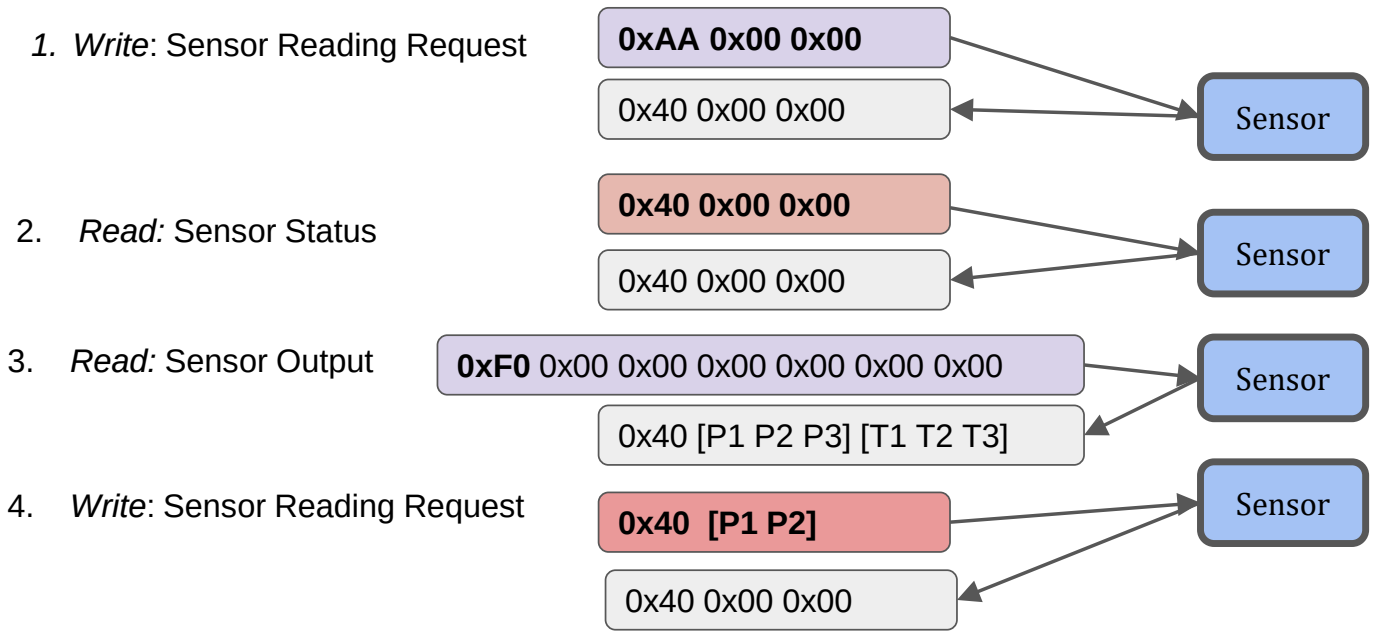
- SPI: *Incorrect* operation: Many possible errors with single buffer:



- Copy/paste #1 to #2: **`SPI_xfer(&buf, 3)`**, forgot to set '0xF0'
- - OR- Copy/paste #2 to #3: **`SPI_xfer(&buf, 3)`**, forgot to set '0xF0'
- - OR- Copy/paste #3 to #4: **`SPI_xfer(&buf, 3)`**, forgot to set '0xAA'

DLH/R EEPROM corruption

- SPI: *Incorrect* operation: dual buffer:



- Copy/paste #1 to #2: **`SPI_xfer(&buf1,&buf2, 3)`**,
  switched buffers, forgot to set '0xF0'.

- -OR- Copy/paste #2 to #4: **`SPI_xfer(&buf1,&buf2, 3)`**,
  switched buffers, forgot to set '0xAA'.