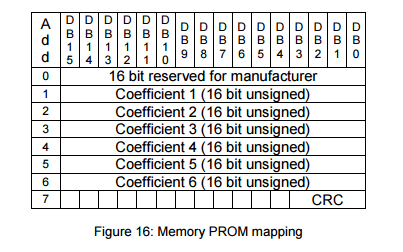
**Payload Enviro SPI Drivers**

* After sending the reset command, there’s a 3ms “reload” where we just need to keep the CS low.
* After a command for an “uncompensated” conversion, it takes 9ms before it’s ready for an ADC read. You can keep the SS low here if you want but you don’t have to. 
* The maximum clock frequency is 20 MHz for this device
* Something in the pressure\_init( ) is causing the SSM to reset.
* Pressure\_sensor\_init() is still apparently fucking things up before it even executes anything.
* Dac\_initialize to blame?
* **Nope**
* **Forgot that I can’t use the delays before I enable global interrupts since they make use of timers.**
* I don’t think I’m getting the raw values of pressure and temperature correctly.
* **It was actually just a printing error.**
* The temperature component seems to be working now after playing around with it for a bit.
* For some reason my raw pressure reading is way off from what the datasheet says is ‘typical’.
* The pressure offset is off by 10% (C2) which could be contributing to some errors.
* **Port expander needs a different slave mode than almost everything else.**
* Port\_expander\_read/\_write save and restore the spi settings.

Rahman’s SPI Settings (02/16/2016):

|  |
| --- |
| void spi\_initialize\_master(void) |
|  |
|  | { |
|  | MCUCR = MCUCR & 0b01111111; |
|  | SPCR = 0b01011111; |
|  | return; |
|  | } |

* Port\_expander\_read and port\_expander\_write now both save and restore the SPI configuration now.
* **Need to set the data direction of each port depending on the SELF\_ID**
* **Done**
* Pins 25-28 = GPIOB0-3, 1-4 = GPIOB4-7
* Pins 17-21 = GPIOA0-4, 22-24 = GPIOA5-7
* EXP\_RST should be high or low? High for normal operation.
* EXP\_RST = pin 22 on the SSM

PORT EXPANDER 000:

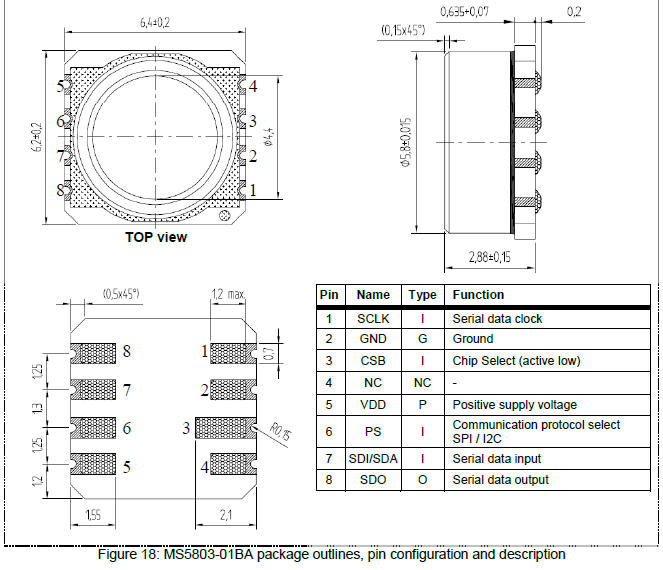
* SS\_HUM = GPA4
* SS\_TEMP = GPA3
* SS\_ACCEL = GPA2
* SS\_PRESH = GPA1
* SS\_PRESL = GPA0
* **No other pins on port expander 000 are used**

PORT EXPANDER 001:

* Heater1-5 = GPIOB0-4
* Valve1a,1b = A0,1
* Valve2a,2b = A2,3
* Valve3a,3b = A4,5
* Valve4a,4b = A6,7
* Finished writing code for SS1\_set\_high( ) and SS1\_set\_low( )
* **I’m not able to get pressure values from the payload, it’s likely that there’s something wrong with the port expander code.**

**Payload Environmental Board:**

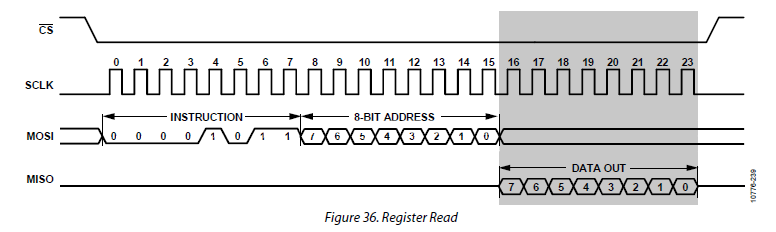
* I’m working on the payload environmental board now trying to get the sensors working.
* What’s different now is that the slave selects for all the sensors are outputs of the port expanders on this board.
* **First, let’s seem if I’m able to set the slave select for a pressure sensor low, then high.**
* **Q: Is the slave select for the port expander low?**
* Yes
* Okay, so pin 17 (SS\_INT\_PRESL) is not going low & high.
* I was writing to the wrong GPIO port on the port expander
* It’s doing something now, it’s starts off low but I don’t know if it’s also going high.
* Lengthen timeout.
* Nevermind, it’s pretty solid just 3.3V all the time.
* 0 = W, 1 = R
* Wasn’t clearing and setting the SS properly.
* Verified that SS is working properly now.
* Perhaps PEX000 was either burned or didn’t reflow properly.
* **It looks like we need to replace PEX000.**
* **Port expander working now, time to work on the pressure sensor.**
* I think the port expander is still fucking up on the pin that I want.
* **The slave select never goes low.**
* **Yeah… Let’s just switch out the broken PEX for the working one.**
* SO.. It’s not something wrong with the port expander because I swapped the “broken” one with the one that’s working and it seems to work just fine.
* What about a pin that’s not connected to a slave select?
* They work just fine….
* Something wrong with the pressure sensor? Is it on backwards?
* What about port A?
* PortA (temperature sensor) works just fine.
* This leads me to believe even more that something is wrong with the pressure sensors.

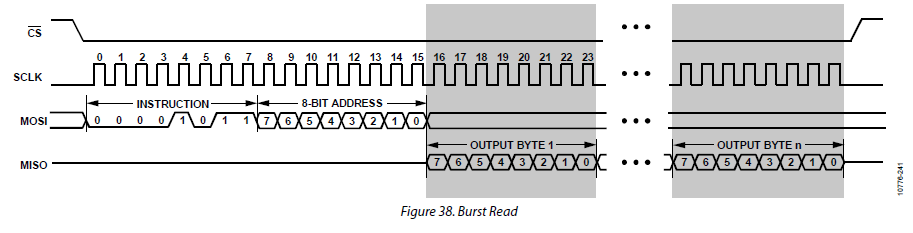


* What is the value of R26?
* 10k
* I don’t think it’s the pull-up’s fault seeing as the temperature sensor also has a pull-up on it’s slave select.
* **Reading IODIR again, I find that GPA0,1 are inputs now?**?
* **Fixed it, needed to write to the whole register for some reason.**
* The slave select is going high and low now.
* **I’m getting values that make sense now ☺**
* The values that I’m getting aren’t very correct…
* They’re both trying to tell me that temp = 28C and press = 0mbar
* I think the brackets are messed up in my code, pressure I calculated: 200mbar.
* Switched a bunch of %1u to %ld.
* I’m getting an overflow when I multiply by 2^17
* Let’s try unsigned integers
* I needed to break down the computation into multiple steps and uses prints to see where the overflow was happening.
* I can calculate pressure correctly now.
* The lower-range sensor is reading a correct pressure now of 1008mbar
* I fixed the connector which had fallen off (I need this to communicate with the payload over UART)
* I now somehow have a short between VCC & GND
* Fixed it, there was indeed a short created underneath the connector, the connector is soldered back on properly now.
* I’m getting FFF from the temperature sensor.
* Let’s try lowering the SPI speed.
* Nothing changed, still getting 0xFFF
* Try a delay after changing SS
* **The slave select pulse with the clock??**
* **Never mind, slave select is perfect.**
* **When in shutdown: 0x800F,** This is what I’m getting!!!!
* So the problem is that I need to set SS high after sending the command for continuous conversion, but I have to communicate with the port expander first.
* 4 spi\_transfer(0)s: still 0x800F
* 3 “ : still 0x800F
* Note: **setting D0-D15 to any other values may place the LM95071/LM95071-Q1 into a manufacturer's test mode, upon which the LM95071/LM95071-Q1 will stop responding as described**
* **00 = continuous conversion, FF = shutdown, other = maybe test mode?**
* **Only eight bits have been defined above since only the last eight transmitted are detected by the LM95071/LM95071-Q1, before CS is taken HIGH.**
* Unfortunately, the last byte that we send to the port expander is 0x1F, and therefore **it stays in shutdown mode.**
* Ground MOSI of the temp sensor?
* Can’t simply ground it because then MOSI won’t work for the port expander.
* **NO: ONLY MISO is connected to the temp sensor.**
* In this case, the last thing on the MISO line before CS goes high is 0xFFFE00 which at least ends in 00.
* Let’s try setting GPIOA from 0x17 to 0x1F (don’t need to read), and then simply ground miso.
* Hopefully this will put the temp sensor in continuous conversion mode.
* It worked for the first one.
* **Let’s set MISO as an output after we talk to the temp sensor and set it to zero.**
* MISO = PB0
* **FIXED IT :D**

**Accelerometer:**

* ADXL362
* Two operating modes:
* Measurement mode, wake-up mode
* Selectable measurement ranges of +-2,4,8g
* 12-bit ADC
* Can report data from 12.5 to 400 Hz
* Acceleration and temperature: 12-bit values, using 2 registers per measurement.
* To read a full sample of 3-axis acceleration data, six registers must be read
* XDATA, YDATA, ZDATA are 8-bit regs if you want to use them instead.
* CPHA, CPOL = 0 **good!**
* **SPI COMMANDS:**
* 0x0A = write reg
* 0x0B = read reg
* 0x0D = read FIFO
* </CS down> <command byte (0x0A or 0x0B)> <address byte> <data byte> <additional data bytes for multi-byte> …</CS up>
* **A register read or write command begins with the address specified in the command and auto-increments for each additional byte in the transfer.**
* **Invalid commands:** MISO remains at high impedance, and bus keeper holds MISO line at its last value.





**Addresses:**

STATUS = 0x0B

XDATAL = 0x0E

XDATAH = 0x0F

YDATAL = 0x10

YDATAH = 0x11

ZDATAL = 0x12

ZDATAH = 0x13

**Initialization:**

1. Leave filter control register alone, default: +-2g, 100Hz measurement rate.
2. Write 0x02 (POWER\_CTL) to register 0x2D: begin measurement

**ADC:**

* ADC for each MIC-Detect board pair is located on the detect board.
* The CS for the ADC comes from the Detect board.
* LED and Detect Port expanders have the same ID.
* The Port expander on the LED board controls the CS on the Detect board.
* PD1-12 are channels on the digital ADC, this is true for both MIC and Fluorescence.
* The only thing different about fluorescence is that lights are turned on for the Detect board instead of the usual LED board.
* DETECT\_CS for the Detect board PE is GB0 on the LED board = pin 25.