

# Avnet Visible Things UART Example

by

Michael C. Li

<https://www.miketechuniverse.com/>

4/5/2018

e2 Studio Version: 5.3.1.002

## Project Summary

Board:	gw002_rev1_3
Device:	R7FS7G27H2A01CBD
Toolchain:	GCC ARM Embedded
Toolchain Version:	4.9.3.20150529
SSP Version:	1.2.0

# PMOD Configuration for this example

Avnet Visible Things Platform

User's Manual

Gateway PMOD	I2C	Type 1 GPIO	Type 2 SPI	Type 2A Expanded SPI	Type 3 UART	Type 4 UART	Type 4A Expanded UART	Type 5 H- Bridge	Type 6 Dual H- Bridge
1	Y	Y	Y	Y	Y	Y	Y	N	N
2	Y	Y	Y	Y	Y	Y	Y	N	N
3	Y	Y	Y	Y	Y	Y	Y	N	N
4	Y	Y	Y	Y	Y	Y	Y	N	N
5	N	Y	Y	Y	N	Y	Y	Y	Y
6	Y	Y	N	N	N	N	N	N	N

Table 3 – Pmod Compatibility Chart

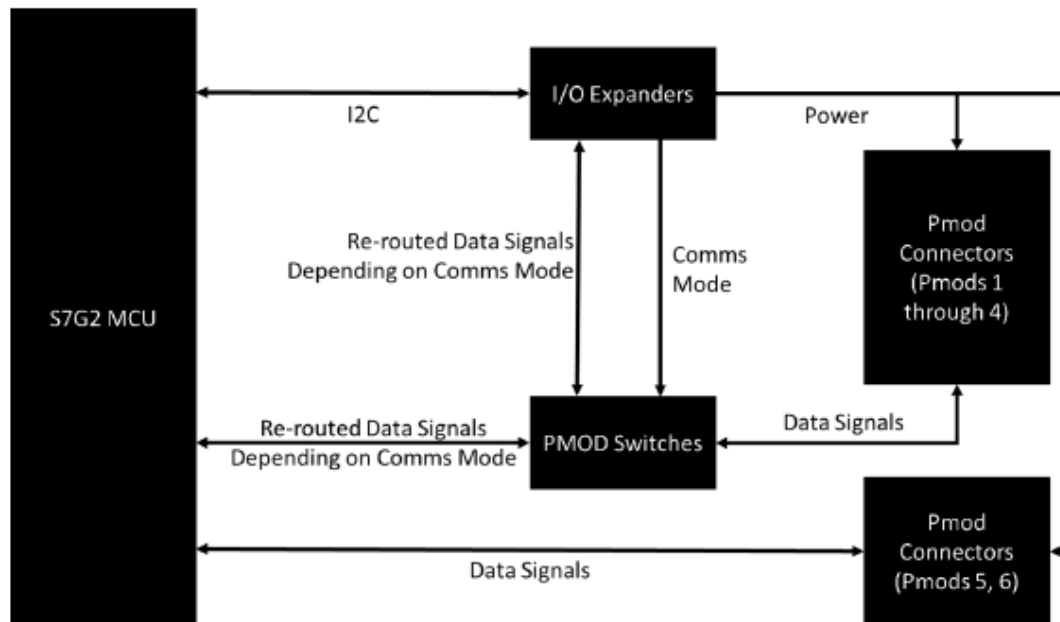


Figure 30 – Pmod, I/O Expander Interconnections Diagram

# Hardware Setup

Open or close (either way is ok for SPI mode)

Closed  
(SPI mode)



# BMC150 Module

IoT Fast Prototyping Kit (S3A7)

1. Overview



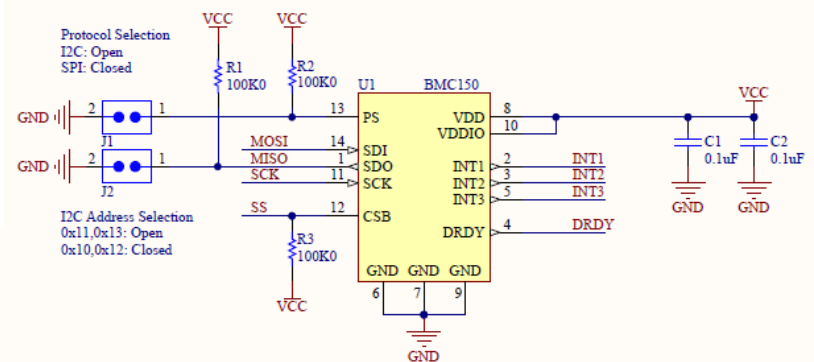
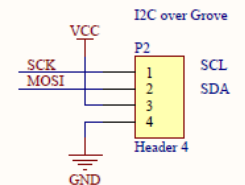
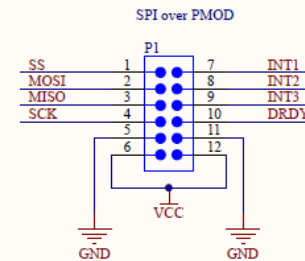
Figure 1.3 Bosch BMC150 sensor board

The active interface to be used is configured by jumper J1:

- Open = I<sup>2</sup>C (Grove)
- Closed = SPI (PMD).

In case the I<sup>2</sup>C is selected, the device addresses are dependent on the state of jumper J2:

- Open = 0x11, 0x13
- Closed = 0x10, 0x12.



# IC Chip Pin Description (not Module Pin)

Table 52: Pin description

Pin	Name	I/O Type	Sensor	Description	Connect to		
					SPI4W	SPI3W	I <sup>2</sup> C
1	SDO	Out	Mag+Acc	SPI: Data out	SDO / MISO	DNC (float)	GND for default address
2	INT1	Out	Acc	Interrupt output #1	INT 1 input or DNC if unused		
3	INT2	Out	Acc	Interrupt output #2	INT2 input or DNC if unused		
4	DRDY	Out	Mag	Data ready	DRDY input or DNC if unused		
5	INT3	Out	Mag	Interrupt output #3	INT3 input or DNC if unused		
6	GND	Supply	Mag+Acc	Ground	GND		
7	GND	Supply	Mag+Acc	Ground	GND		
8	VDD	Supply	Mag+Acc	Supply voltage	V <sub>DD</sub>		
9	GND	Supply	Mag+Acc	Ground	GND		
10	VDDIO	Supply	Mag+Acc	I/O voltage	V <sub>DDIO</sub>		
11	SCK	In	Mag+Acc	Serial clock	SCK	SCK	SCL
12	CSB	In	Mag+Acc	Chip Select	CSB	CSB	DNC (float) or V <sub>DDIO</sub>
13	PS	In	Mag+Acc	Protocol select	GND	GND	V <sub>DDIO</sub>
14	SDI	In/Out	Mag+Acc	SPI: Data in, I <sup>2</sup> C: Data	SDI / MOSI	SDA	SDA

# PMOD 1 Port

## GPIO Type 1 Pmods

If clearance is not an issue, two GPIO Type 1 standard Pmods can be used together when the connector is configured in UART/SPI Comms Mode. **Do not turn a second Type 1 Pmod upside down to get it to fit! Damage may occur as a result of inserting a Pmod upside down!**

	Pmod		S7G2	
	Pin	Pin Function	Port Pin	Pin Function
Upper Row	1	GPIO	PA05	GPIO
	2	GPIO	PA02	GPIO
	3	GPIO	PA03	GPIO
	4	GPIO	PA04	GPIO
	5	GND	GND	GND
	6	VCC (3.3V)	VCC (3.3V)	VCC (3.3V)
Lower Row	1	GPIO	P400	GPIO
	2	GPIO	I/O Expander	GPIO
	3	GPIO	I/O Expander	GPIO
	4	GPIO	I/O Expander	GPIO
	5	GND	GND	GND
	6	VCC (3.3V)	VCC (3.3V)	VCC (3.3V)

Table 7 – Pmod1 GPIO Type 1 Pin Connections (UART/SPI Mode)

# PMOD 2 Port

## UART/SPI Comms Mode

### UART Type 4 and SPI Type 2 Pmods

UART Type 4 Pmods and SPI Type 2 Pmods only have 6 pins instead of the 12 provided by the Gateway's connector. Insert these Pmods into the upper row of the connector (connector pins 1 through 6).

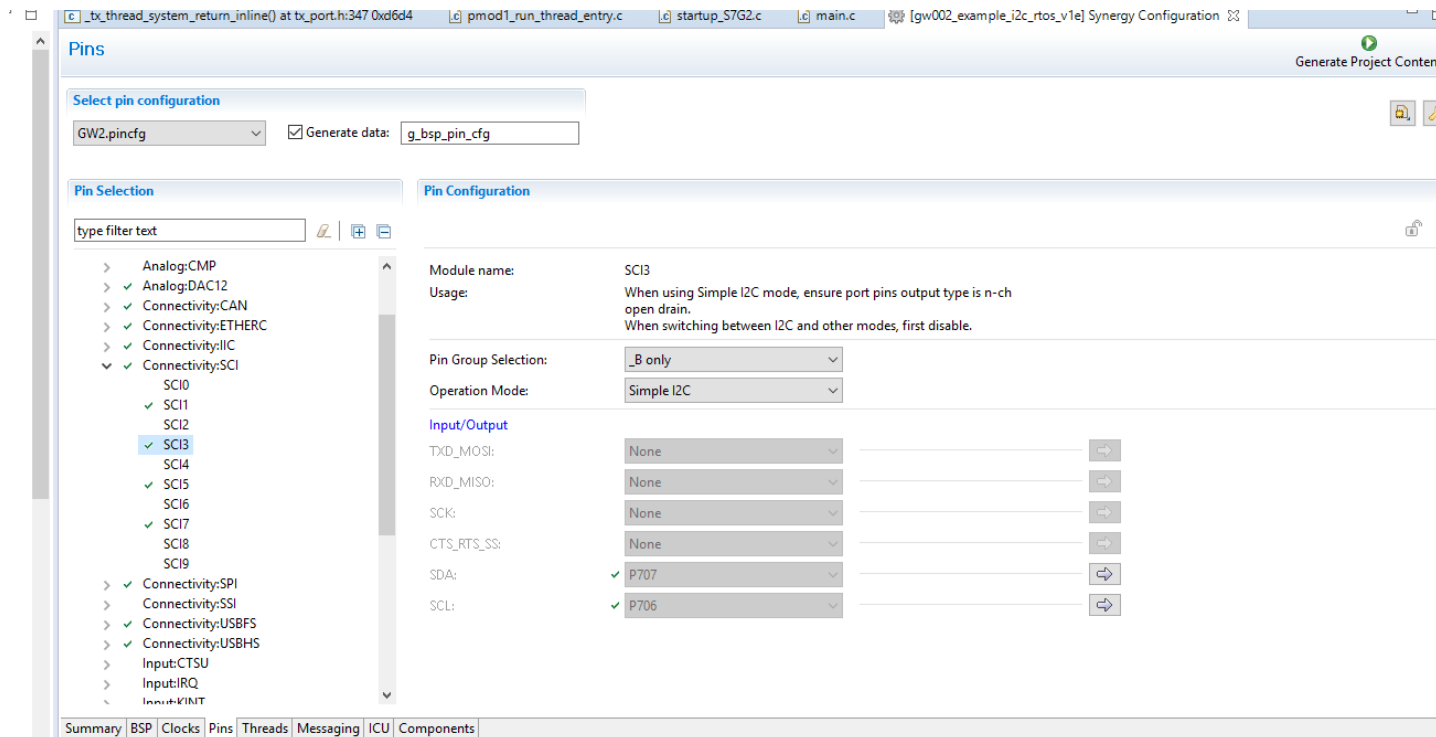
Pmod			S7G2		
Pin	Pin Function by Type		Port Pin	Pin Function	
	UART (4, 4A)	SPI (2, 2A)		UART	SPI
1	RTS	CS	PB02	CTS	CTS_RTS_SS
2	RX	MISO	PB04	TXD_MOSI	TXD_MOSI
3	TX	MOSI	PB05	RXD_MISO	RXD_MISO
4	CTS	SCK	PB03	RTS	SCK
5	GND	GND	GND	GND	GND
6	VCC (3.3V)	VCC (3.3V)	VCC (3.3V)	VCC (3.3V)	VCC (3.3V)

Table 12 – Pmod2 UART Type 4 and SPI Type 2 Pin Connections

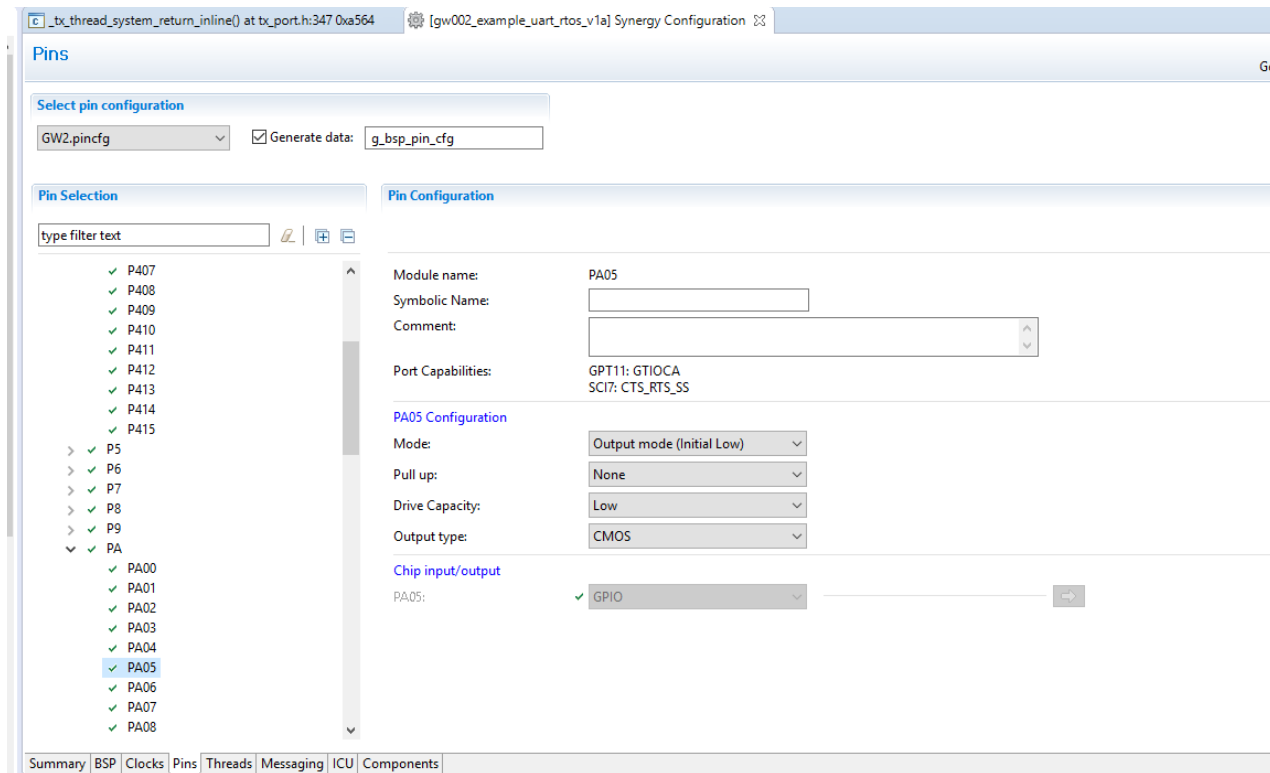




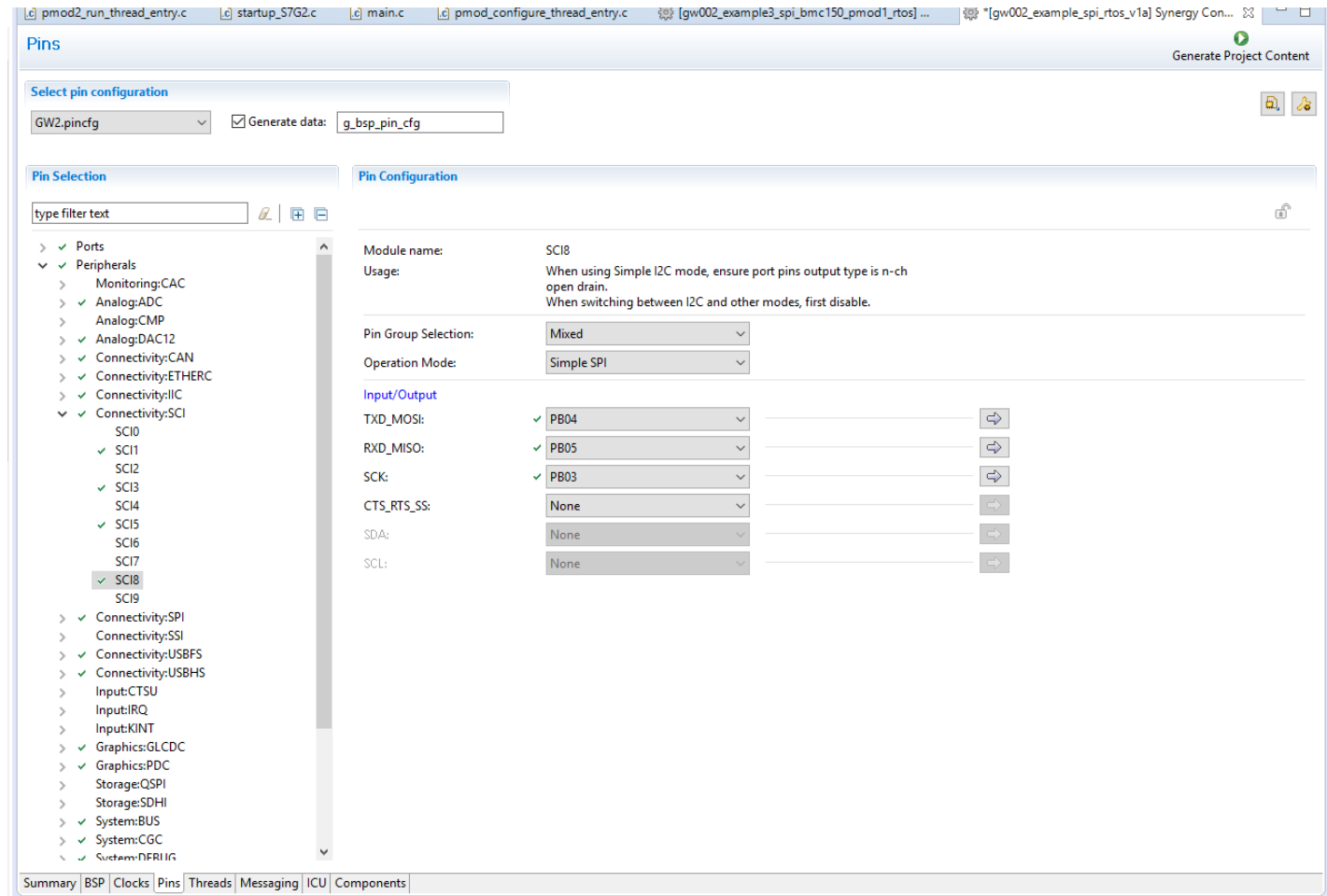
# IO Expanders U18/U19 I2C port



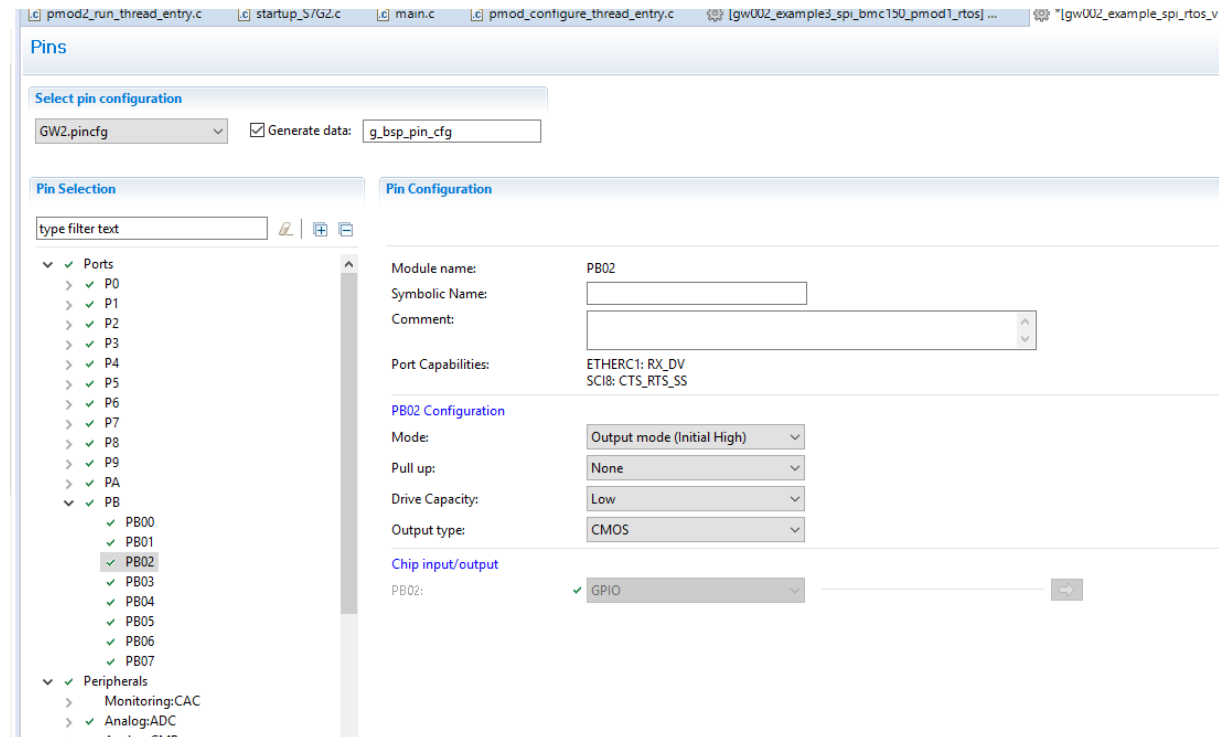
# PMOD 1 : PA02/3/4/5 Output Mode (P400 output only)



# Configure PMOD 2 (Simple SPI)

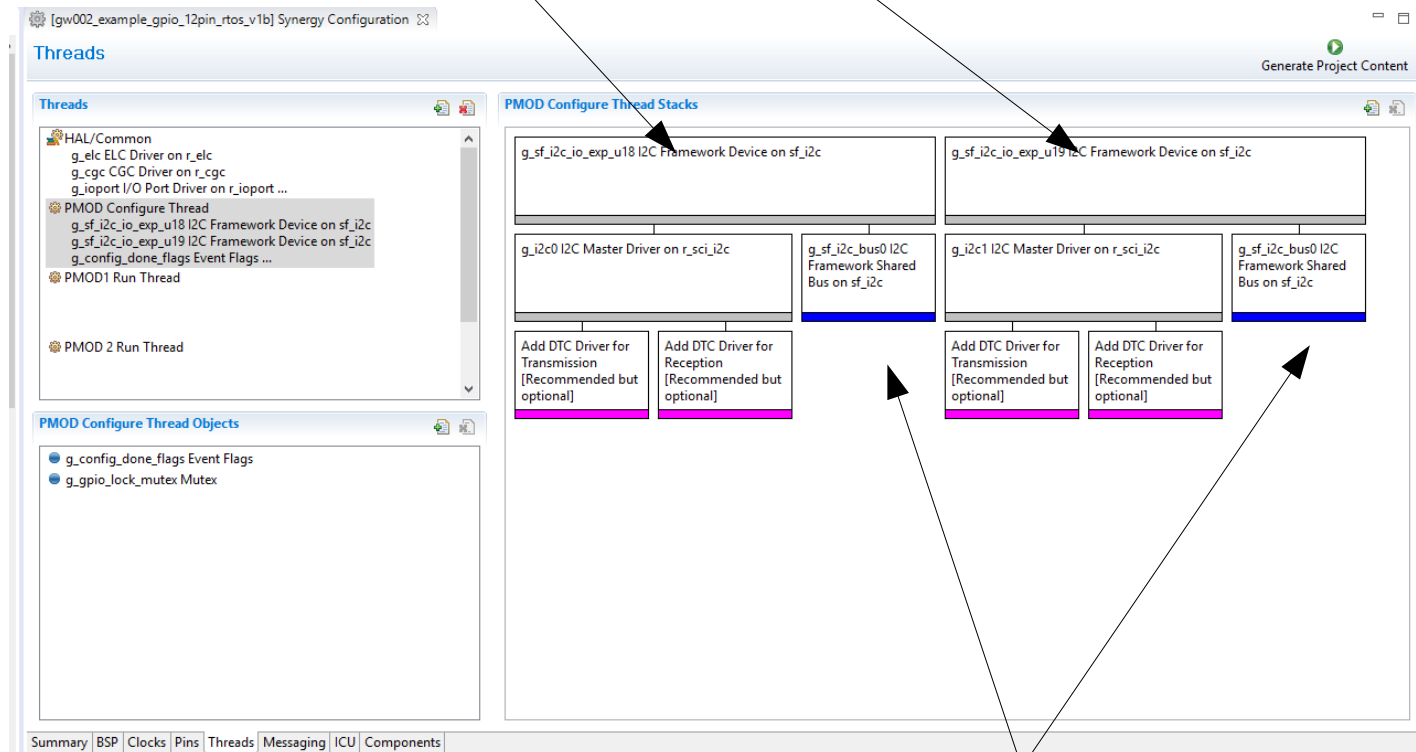


# PMOD2 : SS pin (GPIO out mode)





# PMOD Configure Thread (Create U18/U19 I2C Framework Drivers)



Both drivers share the same g\_sf\_i2c\_bus0 bus.

# Properties

PMOD Configure Thread Stacks

g\_sf\_i2c\_exp\_u18 I2C Framework Device on sf\_i2c

g\_sf\_i2c\_exp\_u19 I2C Framework Device on sf\_i2c

g\_i2c0 I2C Master Driver on r\_sci\_i2c

g\_sf\_i2c\_bus0 I2C Framework Shared Bus on sf\_i2c

g\_i2c1 I2C Master Driver on r\_sci\_i2c

g\_sf\_i2c\_bus1 I2C Framework Shared Bus on sf\_i2c

Add DTC Driver for Transmission [Recommended but optional]

Add DTC Driver for Reception [Recommended but optional]

g\_sf\_i2c\_bus0 I2C Framework Shared Bus on sf\_i2c

Property	Value
Module g_sf_i2c_bus0 I2C Framework Shared Bus c	
Name	g_sf_i2c_bus0
I2C Implementation	SCI I2C
Channel	3

g\_i2c0 I2C Master Driver on r\_sci\_i2c

Property	Value
Common	
Parameter Checking	Default (BSP)
Module g_i2c0 I2C Master Driver on r_sci_i2c	
Name	g_i2c0
Channel	3
Rate	Standard
Slave Address	0x25
Address Mode	7-Bit
SDA Output Delay (nano seconds)	300
Bit Rate Modulation Enable	Enable
Callback	NULL
Receive Interrupt Priority	Priority 2
Transmit Interrupt Priority	Priority 2
Transmit End Interrupt Priority	Priority 2

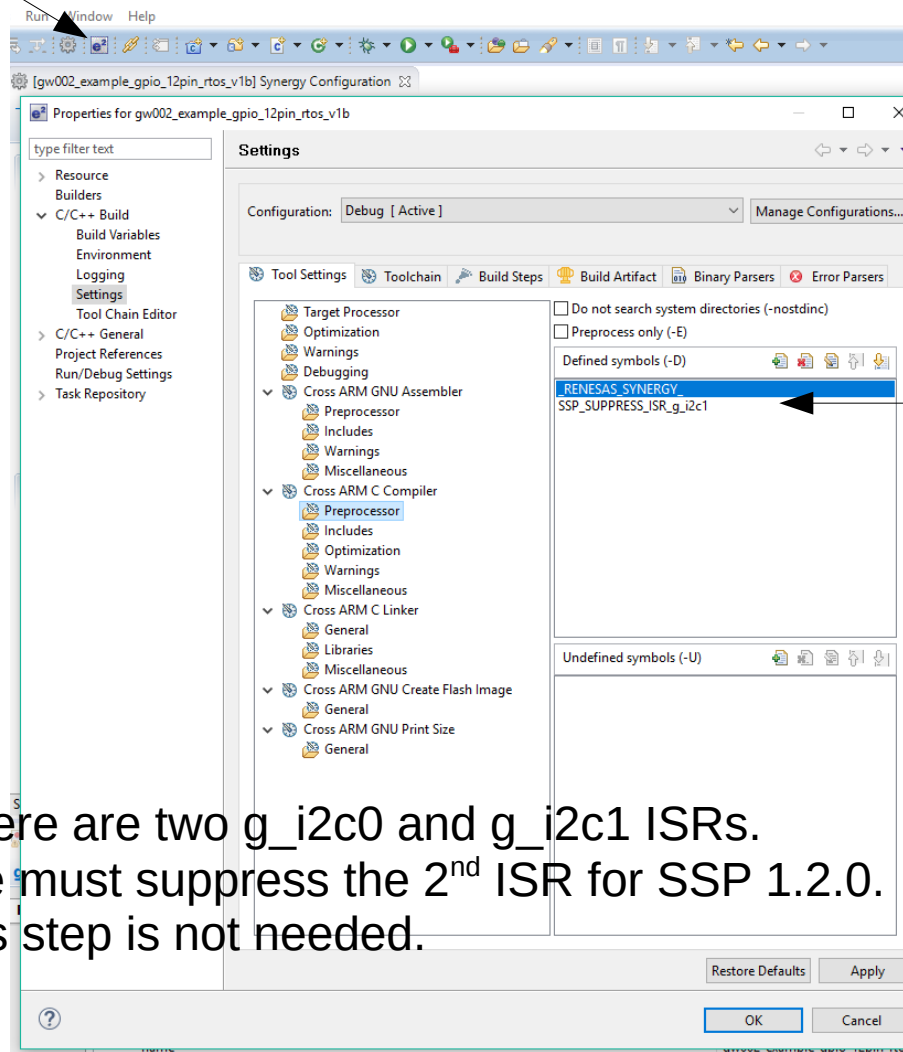
g\_i2c1 I2C Master Driver on r\_sci\_i2c

Property	Value
Common	
Parameter Checking	Default (BSP)
Module g_i2c1 I2C Master Driver on r_sci_i2c	
Name	g_i2c1
Channel	3
Rate	Standard
Slave Address	0x27
Address Mode	7-Bit
SDA Output Delay (nano seconds)	300
Bit Rate Modulation Enable	Enable
Callback	NULL
Receive Interrupt Priority	Priority 2
Transmit Interrupt Priority	Priority 2
Transmit End Interrupt Priority	Priority 2

I2C by Michael Li

# Compiler Pre-processor Requirement

Click this icon!

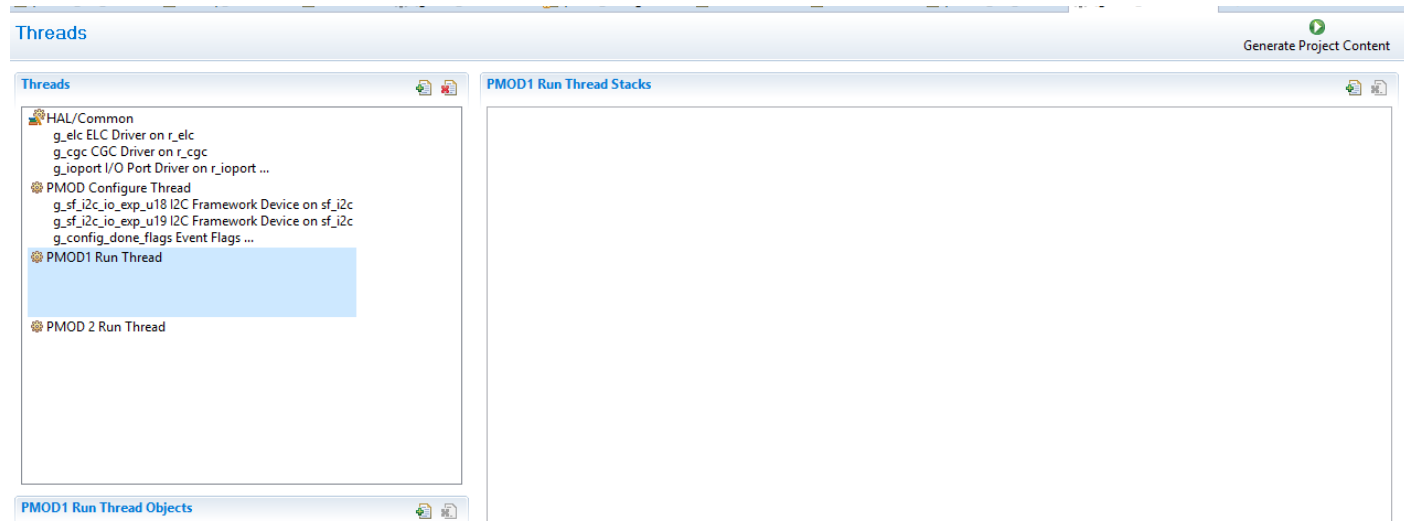


Need to Suppress 2<sup>nd</sup> i2c ISR  
(needed for SSP 1.2.0 only)

There are two g\_i2c0 and g\_i2c1 ISRs.  
We must suppress the 2<sup>nd</sup> ISR for SSP 1.2.0. For later SSP versions,  
this step is not needed.



PMOD 1 Thread has no drivers needed because the port is configured as a GPIO port.



# PMOD 2 Run Thread : SPI Framework Driver

The screenshot displays the PMOD 2 Run Thread interface, which is used for configuring and debugging the SPI Framework Driver. The interface is divided into several panels:

- Threads:** Shows the PMOD 2 Run Thread and its associated components.
- PMOD 2 Run Thread Stacks:** A diagram showing the hierarchy of the SPI Framework Driver. It includes the following components:
  - g\_sf\_spi\_device0 SPI Framework Device on sf\_spi
  - g\_spi0 SPI Driver on r\_sci\_spi
  - g\_sf\_spi\_bus0 SPI Framework Shared Bus on sf\_spi
  - g\_transfer0 Transfer Driver on r\_dtc Event SCI8 TX
  - g\_transfer1 Transfer Driver on r\_dtc Event SCI8 RX
- Property Windows:** Three windows are shown, each displaying the properties of a specific component:
  - g\_sf\_spi\_device0 SPI Framework Device on sf\_spi:**

Property	Value
Common	Default (BSP)
Parameter Checking	Default (BSP)
Module g_sf_spi_device0 SPI Framework Device on sf_spi	
Name	g_sf_spi_device0
Chip Select Port	11
Chip Select Pin	02
Chip Select Active Level	Low
  - g\_spi0 SPI Driver on r\_sci\_spi:**

Property	Value
Common	Default (BSP)
Parameter Checking	Default (BSP)
Module g_spi0 SPI Driver on r_sci_spi	
Name	g_spi0
Channel	8
Operating Mode	Master
Clock Phase	Data sampling on odd edge, data variation on even edge
Clock Polarity	Low when idle
Mode Fault Error	Disable
Bit Order	MSB First
Bitrate	10000
Bit Rate Modulation Enable	Enable
Callback	NULL
Receive Interrupt Priority	Priority 2
Transmit Interrupt Priority	Priority 2
Transmit End Interrupt Priority	Priority 2
Error Interrupt Priority	Priority 2
  - g\_sf\_spi\_bus0 SPI Framework Shared Bus on sf\_spi:**

Property	Value
Common	Default (BSP)
Parameter Checking	Default (BSP)
Module g_sf_spi_bus0 SPI Framework Shared Bus on sf_spi	
Name	g_sf_spi_bus0
SPI Implementation	SCI SPI
Channel	8

Arrows in the image indicate the following relationships:

- From **g\_sf\_spi\_device0 SPI Framework Device on sf\_spi** to **g\_sf\_spi\_device0 SPI Framework Device on sf\_spi** property window.
- From **g\_spi0 SPI Driver on r\_sci\_spi** to **g\_spi0 SPI Driver on r\_sci\_spi** property window.
- From **g\_sf\_spi\_bus0 SPI Framework Shared Bus on sf\_spi** to **g\_sf\_spi\_bus0 SPI Framework Shared Bus on sf\_spi** property window.



# Configure PMOD 1 and PMOD 2

```
*
* input: None
* Return : None
*
*/
void setup_ioexp_registers(void) {

    //////////////////////////////////////
    /// U18 setup for PMOD1 : GPIO Type 1 12 pin configuration
    /// Set IO1-5 output pins (S7G2 MCU pins)
    /// Set IO6/7/8 output pins
    /// Set IO6/7/8 =001
    //////////////////////////////////////
    pmod_bus_type_cfg[PMOD1_PORT] = GPIO_TYPE1_12PINS_COML;

    ul8port0outreg.bit.pmod1_comms = set_pmod_com_bit(pmod_bus_type_cfg[PMOD1_PORT]);

    ul8port0cfg.bit.pmod1_reset_io6 = SET_CFG_PIN_OUTPUT;
    ul8port1cfg.bit.pmod1_io7      = SET_CFG_PIN_OUTPUT;
    ul8port1cfg.bit.pmod1_io8      = SET_CFG_PIN_OUTPUT;

    ul8port0outreg.bit.pmod1_reset_io6 = 1; // make 0010 0000 initially
    ul8port1outreg.bit.pmod1_io7       = 0;
    ul8port1outreg.bit.pmod1_io8       = 0;

    //////////////////////////////////////
    /// U19 setup for PMOD1 : Power enabled
    //////////////////////////////////////
    ul9port0outreg.bit.pmod1_power = 1; // pmod1 power enabled.

    //////////////////////////////////////
    /// U18 setup for PMOD2 : UART 4 6 pin configuration
    //////////////////////////////////////
    pmod_bus_type_cfg[PMOD2_PORT] = SPI_TYPE2_6PINS_COML;

    ul8port0outreg.bit.pmod2_comms = set_pmod_com_bit(pmod_bus_type_cfg[PMOD2_PORT]);

    //////////////////////////////////////
    /// U19 setup for PMOD2 : Power enabled
    //////////////////////////////////////
    ul9port0outreg.bit.pmod2_power = 1; // pmod2 power enabled.

}
}
```

# PMOD 1 Thread Entry

```
rtos_v1c] Synergy Configuration  pmod_configure_thread_entry.c  pmod1_run_thread_entry.c  pmod2_run_thread_entry.c
#include "pmod1_run_thread.h"
#include "pmod_configure_thread_entry.h" // event flag
#include <pca9535/pca9535.h>

extern PMOD_BUS_TYPE_t      pmod_bus_type_cfg[PMOD_PORT_NUM];

/* PMOD1 Run Thread entry function */
void pmod1_run_thread_entry(void)
{
    ULONG event_flags;
    uint8_t writedata[8] = {0x33, 0x4A, 0x33, 0xF1, 0x8F, 0x41, 0xCB, 0x99}; // write data

    tx_event_flags_get(&g_config_done_flags, IOEXP_DONE_EVENT_FLAG, TX_AND, &event_flags, TX_WAIT_FOREVER); // Don't clear it. Leave it

    while (true)
    {
        for (int i=0; i<8; i++) {
            tx_mutex_get(&g_gpio_lock_mutex, TX_WAIT_FOREVER);
            write_pmode_gpio_type1_byte_port (1, writedata[i], pmod_bus_type_cfg[PMOD1_PORT]);
            tx_mutex_put(&g_gpio_lock_mutex);
            tx_thread_sleep(10);
        }
    }
}
```

# PMOD 2 Thread Entry

```
7
8
9
10
11 void pmod2_run_thread_entry(void)
12 {
13     ULONG event_flags;
14     char buf[20];
15     ssp_err_t err;
16
17     tx_event_flags_get(&g_config_done_flags, IOEXP_DONE_EVENT_FLAG, TX_AND, &event_flags, TX_WAIT_FOREVER); // Don't clear it. Leave
18
19     //read acceleration
20     err = g_sf_spi_device0.p_api->open(g_sf_spi_device0.p_ctrl, g_sf_spi_device0.p_cfg);
21     if (err)
22         g_ioport.p_api->pinWrite(LEDREDPIN, true);
23
24
25
26
27     while (1)
28     {
29         // read xyz value
30         buf[0] = (char)(0x80 | 0x02);
31         //buf[0] = (char)(0x80 | 0x00);
32         err = g_sf_spi_device0.p_api->writeRead(g_sf_spi_device0.p_ctrl, buf, &buf[7], 7, SPI_BIT_WIDTH_8_BITS, TX_WAIT_FOREVER);
33         if (err)
34             g_ioport.p_api->pinWrite(LEDREDPIN, true);
35
36         //read chip id
37         buf[0] = (char)(0x80 | 0x00);
38         err = g_sf_spi_device0.p_api->writeRead(g_sf_spi_device0.p_ctrl, buf, &buf[7], 2, SPI_BIT_WIDTH_8_BITS, TX_WAIT_FOREVER);
39         if (err)
40             g_ioport.p_api->pinWrite(LEDREDPIN, true);
41
42         //read temperature
43         buf[0] = (char)(0x80 | 0x08);
44         err = g_sf_spi_device0.p_api->writeRead(g_sf_spi_device0.p_ctrl, buf, &buf[7], 2, SPI_BIT_WIDTH_8_BITS, TX_WAIT_FOREVER);
45         if (err)
46             g_ioport.p_api->pinWrite(LEDREDPIN, true);
47
48         tx_thread_sleep(1);
49         count++;
50     }
51
52
```

Get sensor's data  
and device ID and  
temperature  
reading





# Successful Build

The screenshot displays an IDE interface with the following components:

- Project Explorer:** Lists project files including `gw002_example_gpio_12pin_rtos_v1b`, `gw002_example_gpio_6pin_rtos_v1a`, `gw002_example_gpio_i2c_spi_uart_v1h`, `gw002_example_i2c_rtos_v1e`, `gw002_example_spi_rtos_v1a [Debug]`, and `gw002_example_uart_rtos_v1d`.
- Source Editor:** Displays the code for `pmod1_run_thread_entry.c`. The code includes headers for `pmod1_run_thread.h`, `pmod_configure_thread_entry.h`, and `pca9535/pca9535.h`. It defines `PMOD_BUS_TYPE_t` and `refreshleddelay`. The `pmod1_run_thread_entry` function is shown, which includes a loop that writes data to a port and sleeps for a delay.
- Outline:** Provides a summary of the code structure, including the `pmod1_run_thread` function and its associated data and variables.
- Console:** Shows the build output for the `gw002_example_spi_rtos_v1a` target. The output indicates that the build was successful, with 0 errors and 2 warnings. The build time was 12s.853ms.

```
1  #include "pmod1_run_thread.h"
2  #include "pmod_configure_thread_entry.h" // event flag
3  #include <pca9535/pca9535.h>
4
5  extern PMOD_BUS_TYPE_t      pmod_bus_type_cfg[PMOD_PORT_NUM];
6
7  #define refreshleddelay 10    // 10x10 msec delay time.
8
9  extern uint8_t g_pmod2_port_data;
10
11 /* PMOD1 Run Thread entry function */
12 void pmod1_run_thread_entry(void)
13 {
14     ULONG event_flags;
15     //uint8_t writedata[8] = {0xAA, 0x00, 0x55, 0x00, 0x55, 0x00, 0xAA, 0x00}; // write data
16     uint8_t writedata[8] = {0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80}; // write data
17
18     tx_event_flags_get(&g_config_done_flags, IOEXP_DONE_EVENT_FLAG, TX_AND, &event_flags, TX_WAIT_FOREVER);
19
20     while (true)
21     {
22         for (int i=0; i<8; i++) {
23             tx_mutex_get(&g_gpio_lock_mutex, TX_WAIT_FOREVER);
24             write_pmod_gpio_type1_byte_port(1, ~g_pmod2_port_data, pmod_bus_type_cfg[PMOD1_PORT]);
25             tx_mutex_put(&g_gpio_lock_mutex);
26             tx_thread_sleep(refreshleddelay);
27         }
28     }
29 }
30
31
32
33
34
```

CDT Build Console [gw002\_example\_spi\_rtos\_v1a]

```
'Finished building target: gw002_example_spi_rtos_v1a.elf'
'
'Invoking: Cross ARM GNU Create Flash Image'
arm-none-eabi-objcopy -O srec "gw002_example_spi_rtos_v1a.elf" "gw002_example_spi_rtos_v1a.srec"
'Invoking: Cross ARM GNU Print Size'
arm-none-eabi-size --format=berkeley "gw002_example_spi_rtos_v1a.elf"
'Finished building: gw002_example_spi_rtos_v1a.srec'
text    data    bss    dec    hex filename
68736   988   13036   82760   14348 gw002_example_spi_rtos_v1a.elf
'Finished building: gw002_example_spi_rtos_v1a.siz'
'

14:34:00 Build Finished. 0 errors, 2 warnings. (took 12s.853ms)
```





# Correct Device ID read

Debug - gw002\_example\_spi\_rtos\_v1a/src/pmod2\_run\_thread\_entry.c - e2 studio

File Edit Source Refactor Navigate Search Project Renesas Views Run Window Help

Debug gw002\_example\_spi\_rtos\_v1a Debug [Renesas GDB Hardware Debugging]

gw002\_example\_spi\_rtos\_v1a.elf [1]

Thread #1 1 (single core) (Suspended : Step)

- pmod2\_run\_thread\_entry() at pmod2\_run\_thread\_entry.c:39 0xb4e8
- pmod2\_run\_thread\_func() at pmod2\_run\_thread.c:173 0xa5aa
- \_tx\_thread\_shell\_entry() at tx\_thread\_shell\_entry.c:164 0xc8cc
- 0xffffffff

Thread #2 1001 (single core - PMOD Configure Thread [Sleeping RC:49]) (Suspended : Signal : SIGTRAP:Trace/breakpoint trap)

- \_tx\_thread\_system\_return\_inline() at tx\_thread\_system.c:347 0xcacc
- \_tx\_thread\_system\_suspend() at tx\_thread\_system\_suspend.c:710 0xcacc
- \_tx\_thread\_sleep() at tx\_thread\_sleep.c:215 0xc952
- pmod\_configure\_thread\_entry() at pmod\_configure\_thread\_entry.c:144 0xb5d8
- pmod\_configure\_thread\_func() at pmod\_configure\_thread.c:160 0xa69e
- \_tx\_thread\_shell\_entry() at tx\_thread\_shell\_entry.c:164 0xc8cc
- 0xffffffff

Thread #3 1002 (single core - PMOD1 Run Thread [Sleeping RC:20]) (Suspended : Signal : SIGTRAP:Trace/breakpoint trap)

- \_tx\_thread\_system\_return\_inline() at tx\_thread\_system.c:347 0xcacc
- \_tx\_thread\_system\_suspend() at tx\_thread\_system\_suspend.c:710 0xcacc
- \_tx\_thread\_sleep() at tx\_thread\_sleep.c:215 0xc952
- pmod1\_run\_thread\_entry() at pmod1\_run\_thread\_entry.c:22 0xb420

(x) = Variables Breakpoints Registers Modules Expressions Eventpoints IO Registers Peripherals

Name	Type	Value
event_flags	ULONG	1
buf	char [20]	0x1ffe326c <pmod2_run_thread_stack>
buf[0]	char	-128 '\200'
buf[1]	char	-17 '\'
buf[2]	char	-17 '\'
buf[3]	char	-17 '\'
buf[4]	char	-17 '\'
buf[5]	char	-17 '\'
buf[6]	char	-17 '\'
buf[7]	char	-1 '\'
buf[8]	char	-6 '\'
buf[9]	char	5 '\005'
buf[10]	char	49 '\'
buf[11]	char	4 '\004'
buf[12]	char	-47 '\'
buf[13]	char	66 '\'
buf[14]	char	-17 '\'

Name : buf[8]  
Details: -6 '\'  
Default: -6 '\'  
Decimal: -6  
Hex: 0xfa  
Binary: 11111010  
Octal: 0372

pmod2\_run\_th... startup\_S7G2.c [gw002\_exam... [gw002\_exam... pmod\_configu... pmod1\_run\_th... pmod2\_run\_th... bmc150\_thre... startup\_S7G2.c main.c

```
33 0000b4ae if (err)
34 0000b4b4 q_ioport.p_api->pinWrite(LEDREDPIN, true);
35
36 //read chip id
37 0000b4c2 buf[0] = (char) (0x80 | 0x00);
38 0000b4c6 err = g_sf_spi_device0.p_api->writeRead(g_sf_spi_device0.p_ctrl, buf, &buf[7], 2, SPI_BIT_WIDTH_8_BITS, TX_WAIT_FOREVER);
39 0000b4e8 if (err)
40 0000b4ee q_ioport.p_api->pinWrite(LEDREDPIN, true);
41
42 //read temperature
43 0000b4fc buf[0] = (char) (0x80 | 0x08);
44 0000b500 err = g_sf_spi_device0.p_api->writeRead(g_sf_spi_device0.p_ctrl, buf, &buf[7], 2, SPI_BIT_WIDTH_8_BITS, TX_WAIT_FOREVER);
45 0000b522 if (err)
46 0000b528 q_ioport.p_api->pinWrite(LEDREDPIN, true);
```

Outline Project Explorer

- gw002\_example\_i2c\_rtos\_v1c
- gw002\_example\_i2c\_rtos\_v1d
- gw002\_example\_i2c\_rtos\_v1e
- gw002\_example\_spi\_rtos\_v1a
  - Binaries
  - Includes
  - src
    - pca9535
    - synergy\_gen
    - hal\_entry.c
    - pmod\_configure\_thread\_entry.c
    - pmod\_configure\_thread\_entry.h
    - pmod1\_run\_thread\_entry.c
    - pmod2\_run\_thread\_entry.c

Console gw002\_example\_spi\_rtos\_v1a Debug [Renesas GDB Hardware Debugging] C:/Renesas/e2\_studio531\_ssp120\_s7g2avnet/DebugComp/arm-none-eabi-gdb (7.8.2)

Program received signal 347 inc\el1\./cm4\_gcc/tx\_port.h: No such file or directory.

SIGTRAP, Trace/breakpoint trap.

\_tx\_thread\_system\_return\_inline () at inc\el1\./cm4\_gcc/tx\_port.h:347

Program received signal SIGTRAP, Trace/breakpoint trap.

\_tx\_thread\_system\_return\_inline () at inc\el1\./cm4\_gcc/tx\_port.h:347

347 in inc\el1\./cm4\_gcc/tx\_port.h

Program received signal SIGTRAP, Trace/breakpoint trap.

0x0000b4e6 in pmod2\_run\_thread\_entry () at ../src/pmod2\_run\_thread\_entry.c:38

38 err = g\_sf\_spi\_device0.p\_api->writeRead(g\_sf\_spi\_device0.p\_ctrl, buf, &buf[7], 2, SPI\_BIT\_WIDTH\_8\_BITS, TX\_WAIT\_FOREVER);

Suspended