

S5D9 ADS1015 I2C Bus Examples (Both IIC and SCI Drivers)

By
Michael Li
(3/2/2018)

<https://www.miketechuniverse.com>

E2 Studio 5.4.0.023
SSP 1.3.0

This is a set of examples that show how to add the ADC capability to the S5D9 IOT board.

This board has two grove headers. Grove B is an IIC port while grove A is an SCI port. You creates r_iic driver for the IIC port and r_sci_i2c for the SCI port.

6 examples are included in this tutorial.

Grove B / r_iic driver (only for the fast speed mode) *

Grove B / framework with r_iic driver

Grove A / r_sci_i2c driver

Grove A / framework with r_sci_i2c driver

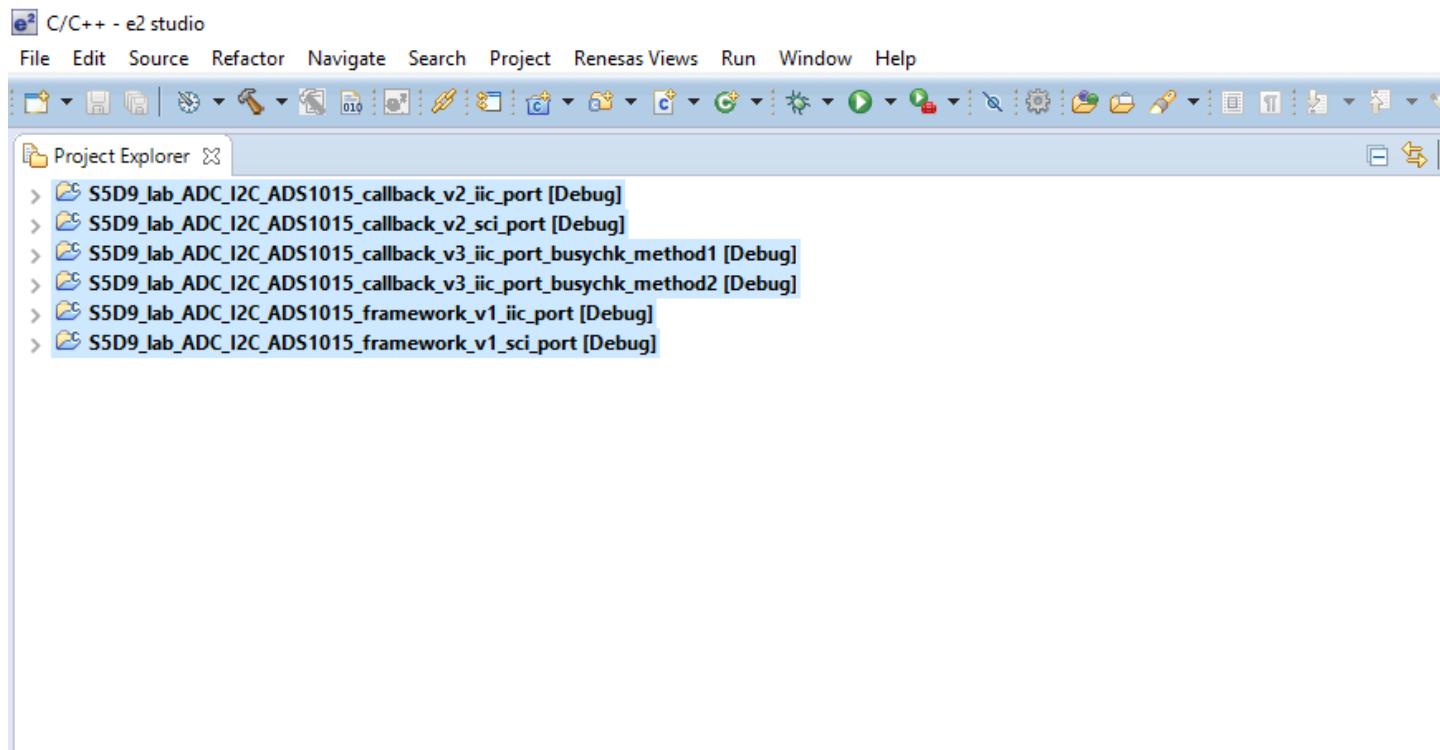
*Because r_iic has some issue for standard speed mode. Two possible workaround examples are included.

Grove B / r_iic driver / method 1 with busy bit check

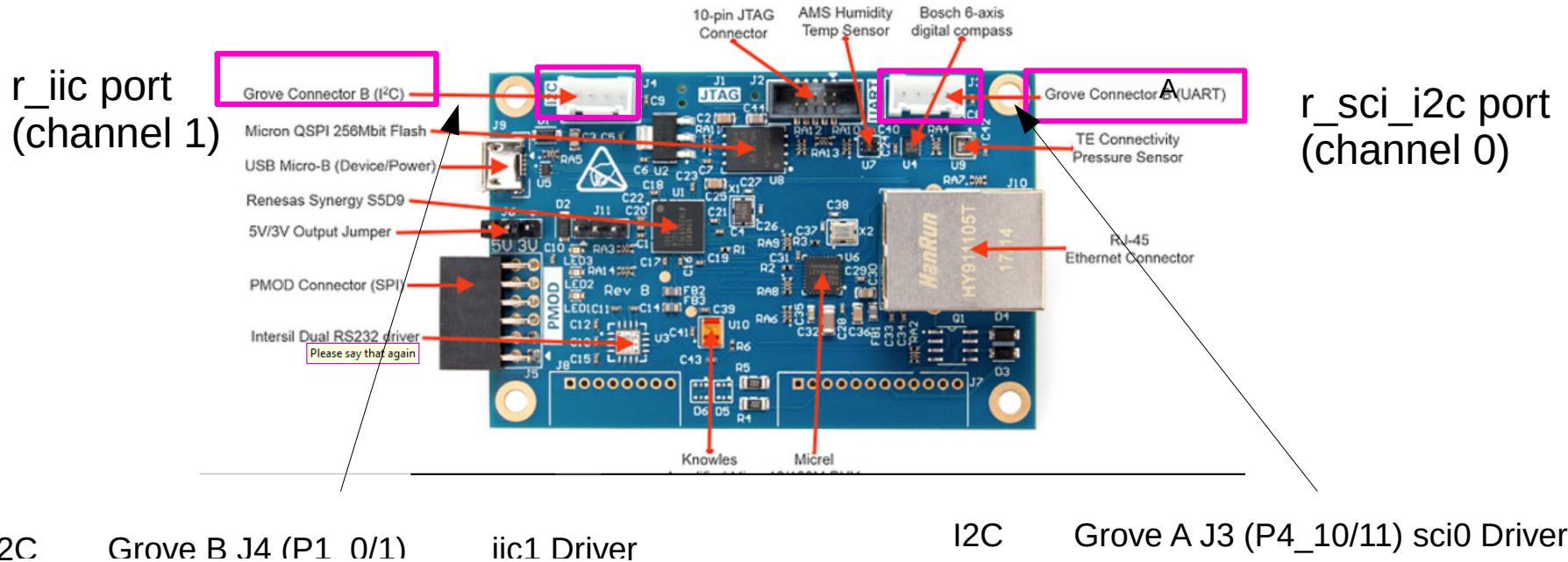
Grove B / r_iic driver / method 2 with busy bit check

Both checks for the hardware busy bit before the code starts the next i2c transmission. The above examples are done with limited testing. At this point, this ADC component seem to work well with these workaround solutions.

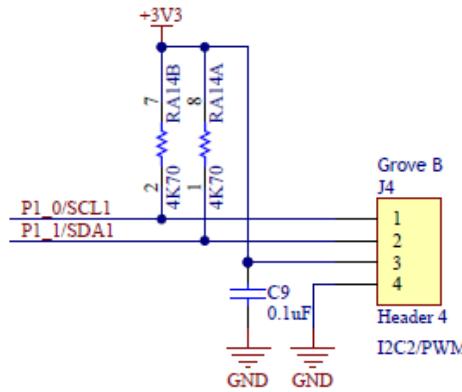
Project List



Two I2C Bus Ports' Location

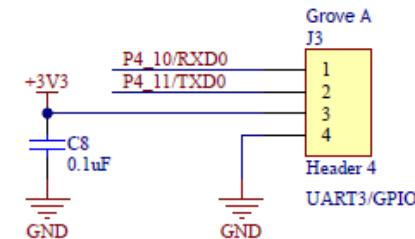


I2C Grove B J4 (P1 0/1) iic1 Driver



Need to add external 4.7K pull up resistors for P4_10/11

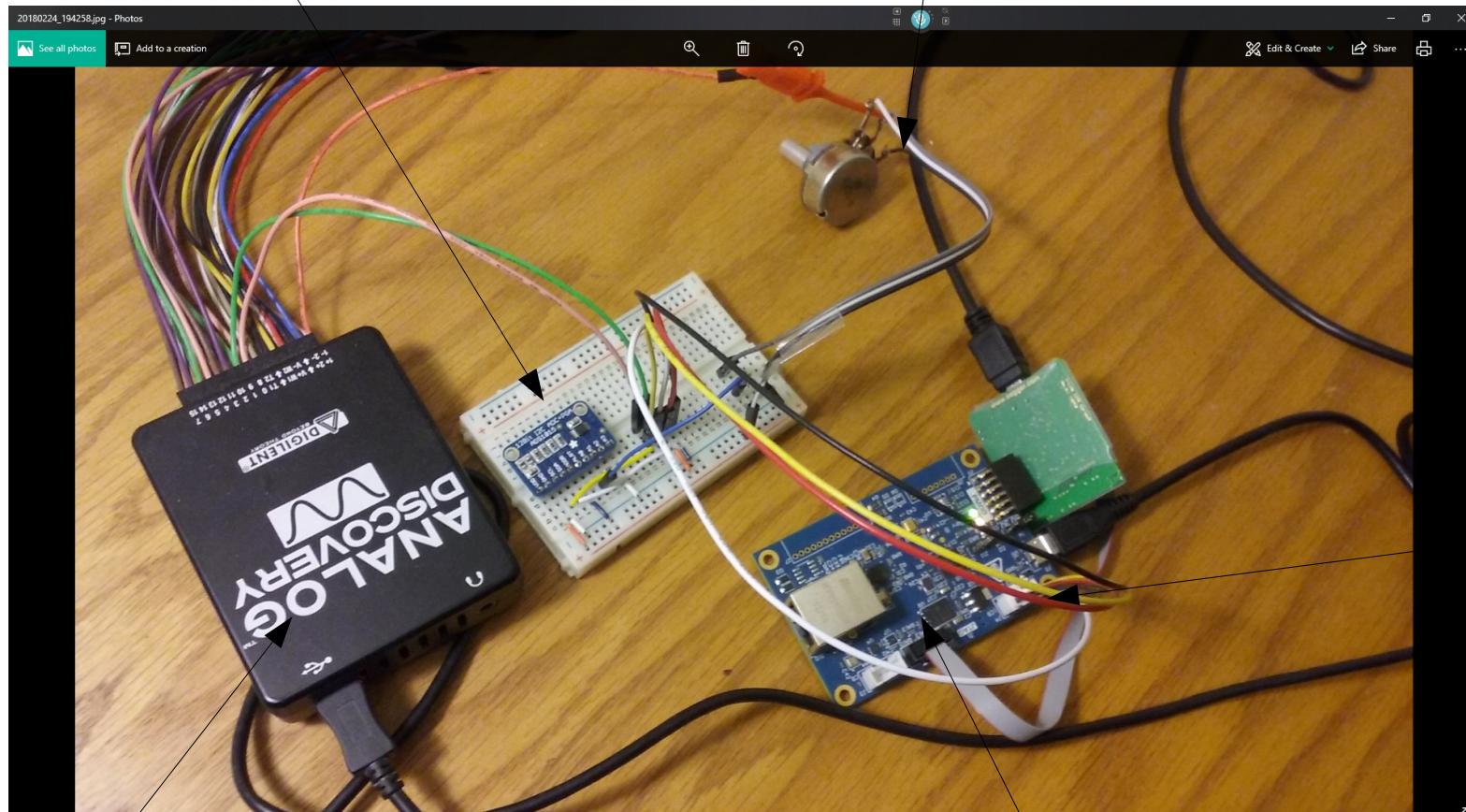
Grove A J3 (P4 10/11) sci0 Driver



Reconfigure this port as I2C instead of UART port.

Adafruit TI ADS1015 ADC

Hardware setup I2C bus
(Grove B r_iic port)
(ADC part and Potentiometer)



Scope

Michael C. Li

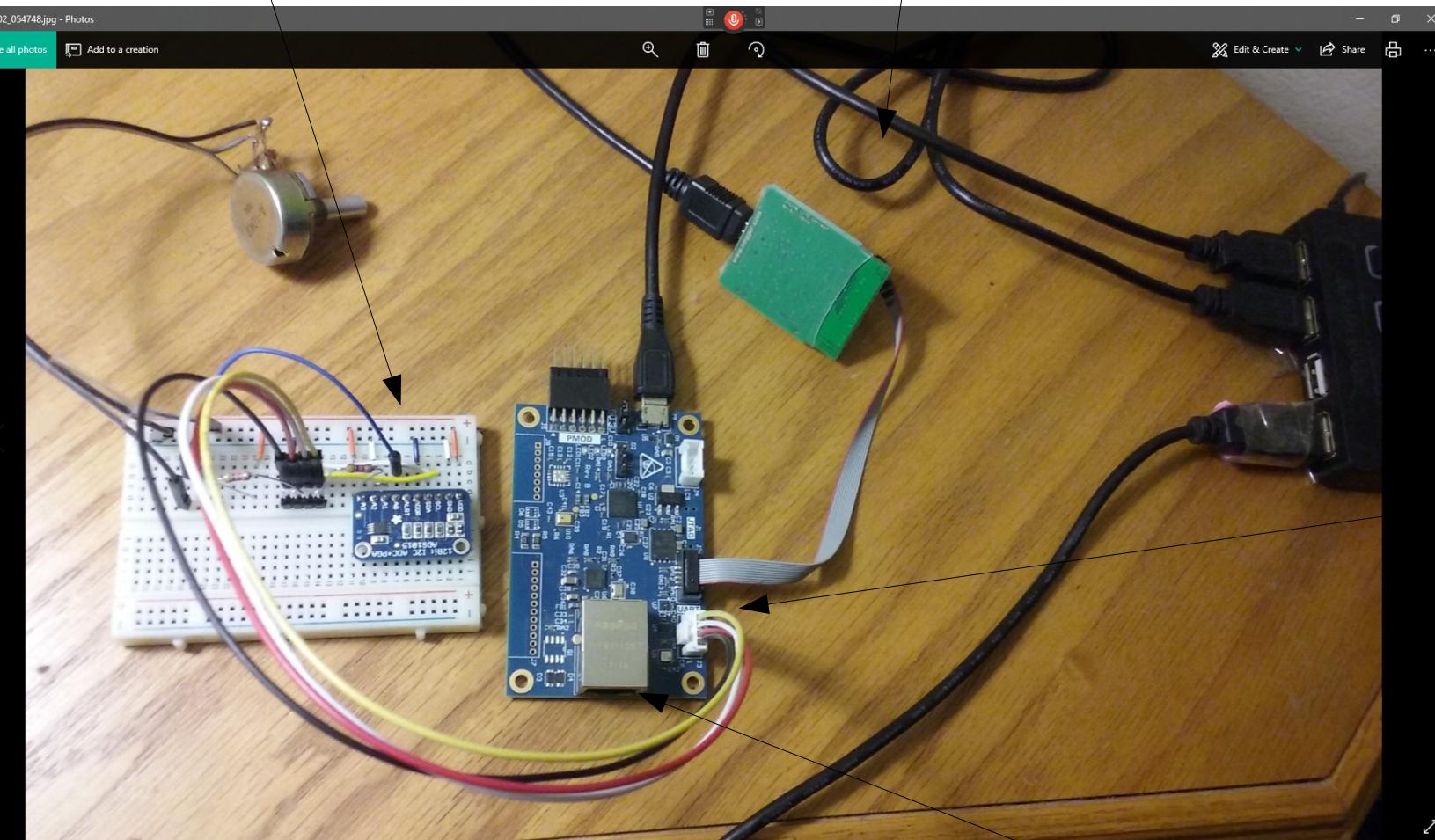
Grove B r_iic port

S5D9 IOT board

Hardware setup I2C bus
(Grove A r_sci_i2c port)
(ADC part and Potentiometer)

Adafruit TI ADS1015 ADC

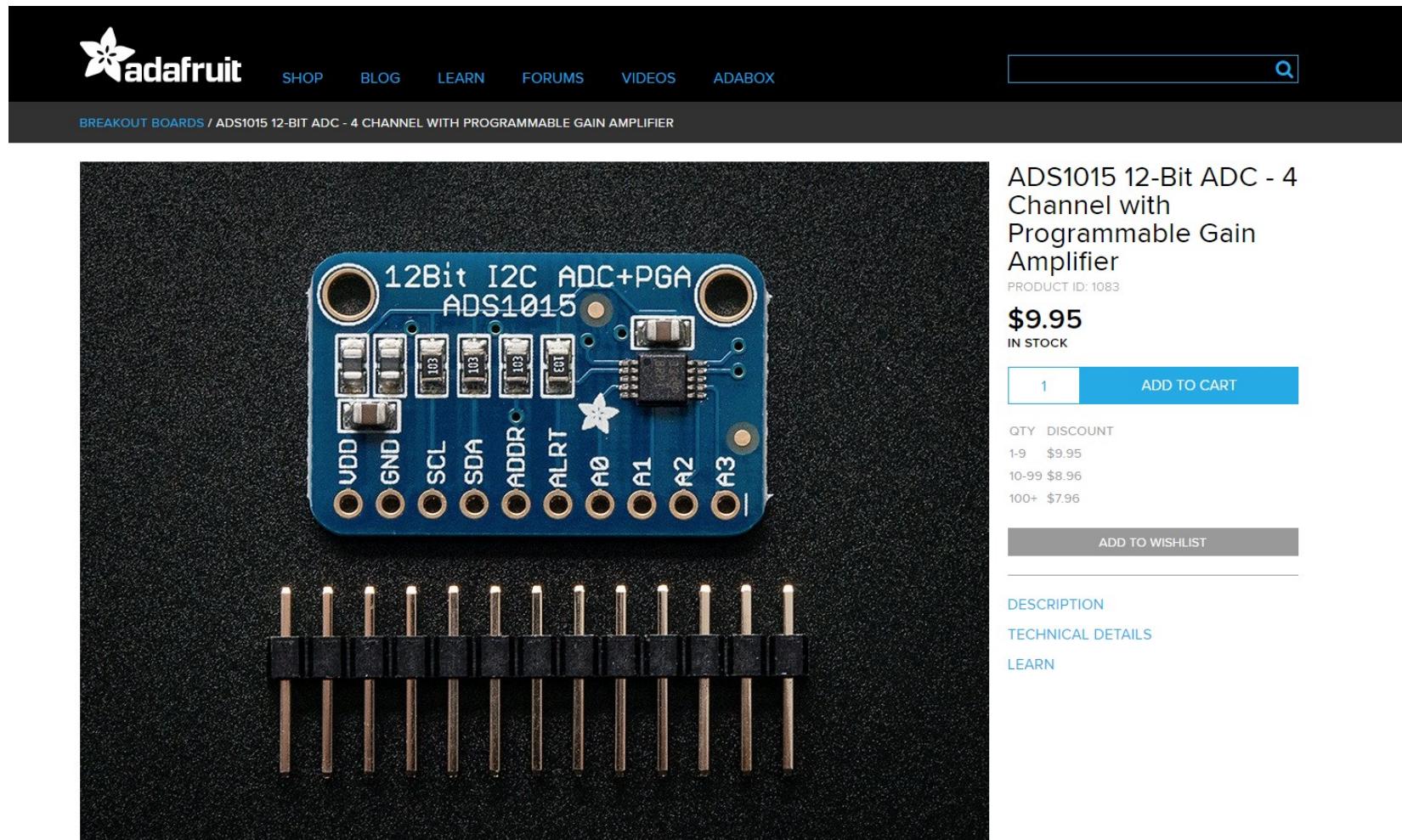
Grove B r_sci_i2c
port



S5D9 IOT board

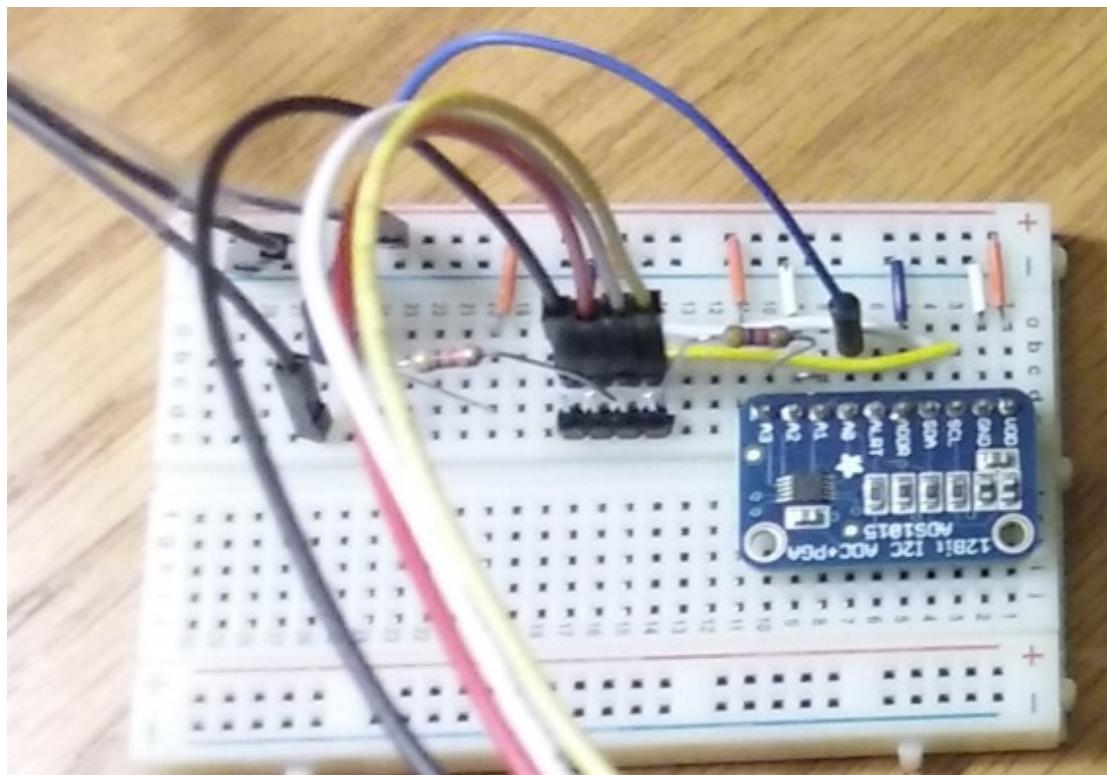
Michael C. Li

<https://www.adafruit.com/product/1083>



Michael C. Li

ADC connection



VCC = VCC
GND = GND
SCL = SCL
SDA = SDA
ADDR = GND
ALRT = NC
A0 = Potentiometer
A1/2/3 = GND (not used)

ADC Slave Address

I²C ADDRESS SELECTION

The ADS1013/4/5 have one address pin, ADDR, that sets the I²C address. This pin can be connected to ground, VDD, SDA, or SCL, allowing four addresses to be selected with one pin as shown in [Table 5](#). The state of the address pin ADDR is sampled continuously.

Table 5. ADDR Pin Connection and Corresponding Slave Address

ADDR PIN	SLAVE ADDRESS
Ground	1001000
VDD	1001001
SDA	1001010
SCL	1001011

Example: Configuration Register Read

Step 1: Slave address (48h) + w → Register address (01h)

Step 2: Slave address (48h) + r → 2 bytes of data read (8583h)

Example: Configuration Register Write and initiate one shot ADC conversion.

Step 1: Slave address (48h) + w → Register address (01h) + 2 bytes of data write (C383h)

Table 8. Conversion Register (Read-Only)

BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NAME	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

$$\text{Voltage level} = D[11:0] / 2048 \times 4.096V \text{ (using +/-4096 FSR)}$$

Example: Conversion Register Read

Step 1: Slave address (48h) + w → Register address (00h)

Step 2: Slave address (48h) + r → 2 bytes of data read (16 bits)

POINTER REGISTER

The four registers are accessed by writing to the Pointer register byte; see [Figure 16](#). [Table 6](#) and [Table 7](#) indicate the Pointer register byte map.

Table 6. Register Address

BIT 1	BIT 0	REGISTER
0	0	Conversion register
0	1	Config register
1	0	Lo_thresh register
1	1	Hi_thresh register

Default = 8583h (Config Register)

Table 9. Config Register (Read/Write)

BIT	15	14	13	12	11	10	9	8
NAME	OS	MUX2	MUX1	MUX0	PGA2	PGA1	PGA0	MODE
BIT	7	6	5	4	3	2	1	0
NAME	DR2	DR1	DR0	COMP_MODE	COMP_POL	COMP_LAT	COMP_QUE1	COMP_QUE0

Default = 8583h.

Bit [15]

OS: Operational status/single-shot conversion start

This bit determines the operational status of the device.
This bit can only be written when in power-down mode.

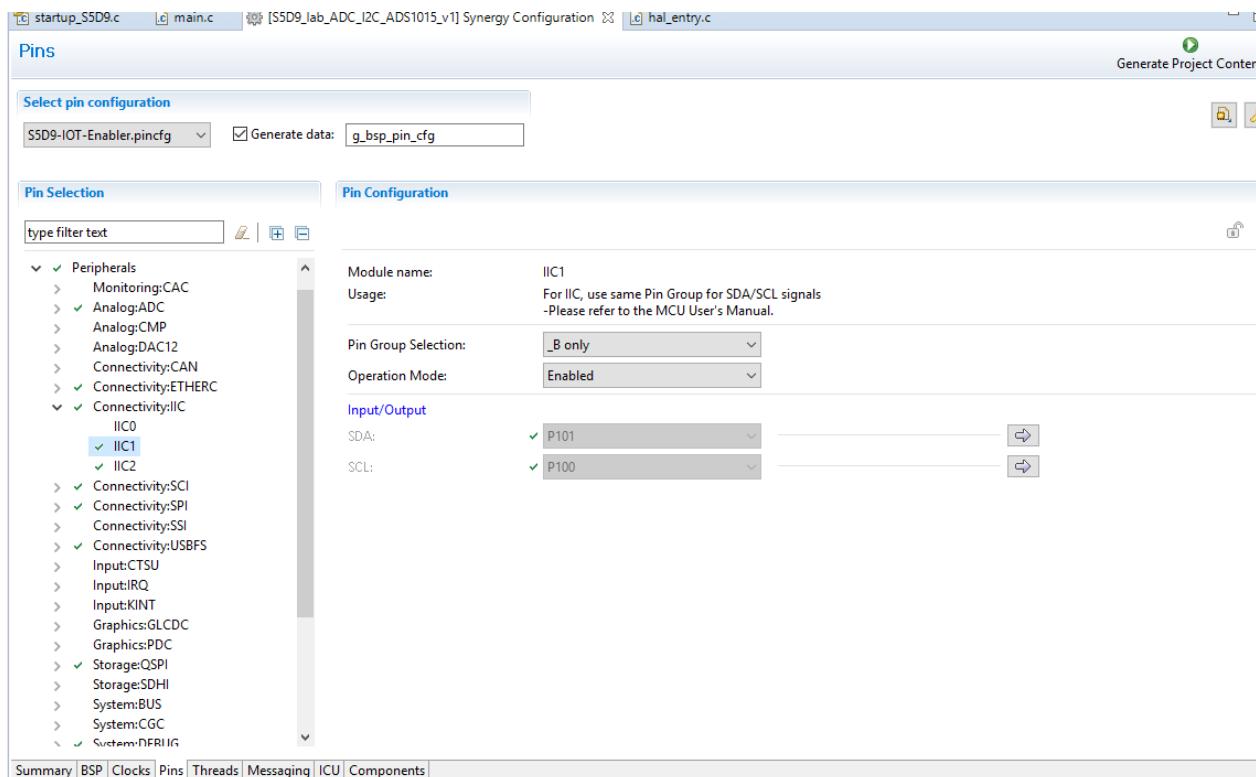
For a write status:

- 0 : No effect
- 1 : Begin a single conversion (when in power-down mode)

For a read status:

- 0 : Device is currently performing a conversion
- 1 : Device is not currently performing a conversion

IIC Channel 1



SCI Channel 0

PINS

Select pin configuration
SSD9-IOT-Enabler.pincfg Generate data: g_bsp_pin_cfg

Generate Project Content

Pin Selection Pin Configuration

type filter text

Peripherals

- > Monitoring:CAC
- > ✓ Analog:ADC
- > Analog:CMP
- > Analog:DAC12
- > Connectivity:CAN
- > ✓ Connectivity:ETHERC
- > ✓ Connectivity:I2C
- > ✓ Connectivity:SCI
 - ✓ SCI0
 - SCI1
 - SCI2
 - ✓ SCI3
 - SCI4
 - SCI5
 - SCI6
 - ✓ SCI7
 - SCI8
 - SCI9
- > ✓ Connectivity:SPI
- < Connectivity:SSI

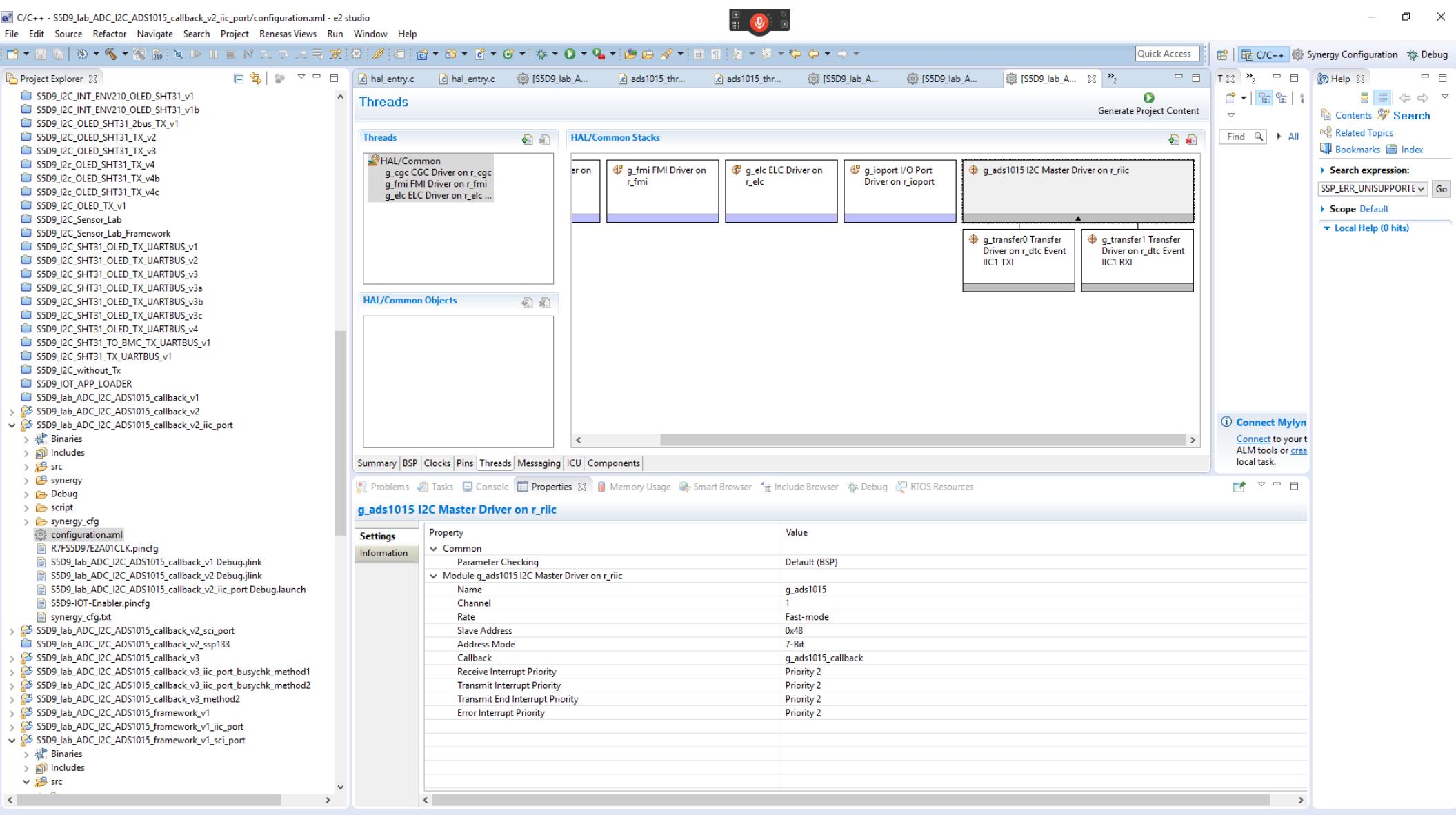
Module name: SCI0
Usage:
When using Simple I2C mode, ensure port pins output type is n-ch open drain.
When switching between I2C and other modes, first disable.

Pin Group Selection: B only
Operation Mode: Simple I2C

Input/Output

TXD_MOSI:	None	↔
RXD_MISO:	None	↔
SCK:	None	↔
CTS_RTS_SS:	None	↔
SDA:	P411	↔
SCL:	P410	↔

r_iic driver



r_sci_i2c driver

C/C++ - S5D9_lab_ADC_I2C_ADS1015_callback_v2_sci_port/configuration.xml - e2 studio

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

Project Explorer

- S5D9_I2C_ENV210_OLED_SHT31_v3c_Framework
- S5D9_I2C_EXT_OLED_v1
- S5D9_I2C_EXT_OLED_v2
- S5D9_I2C_EXT_SHT31
- S5D9_I2C_EXT_SHT31_1
- S5D9_I2C_EXT_SHT31_without_Tx
- S5D9_I2C_INT_EN210_v1
- S5D9_I2C_INT_ENV210_OLED
- S5D9_I2C_INT_ENV210_OLED_SHT31
- S5D9_I2C_INT_ENV210_OLED_SHT31_v1
- S5D9_I2C_INT_ENV210_OLED_SHT31_v1b
- S5D9_I2C_OLED_SHT31_2bus_Tx_v1
- S5D9_I2C_OLED_SHT31_TX_v2
- S5D9_I2C_OLED_SHT31_TX_v3
- S5D9_I2c_OLED_SHT31_TX_v4
- S5D9_I2c_OLED_SHT31_TX_v4b
- S5D9_I2c_OLED_SHT31_TX_v4c
- S5D9_I2C_OLED_TX_v1
- S5D9_I2C_Sensor_Lab
- S5D9_I2C_Sensor_Lab_Framework
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v1
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v2
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3a
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3b
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3c
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v4
- S5D9_I2C_SHT31_TO_BMC_TX_UARTBUS_v1
- S5D9_I2C_SHT31_TX_UARTBUS_v1
- S5D9_I2C_without_Tx
- S5D9_IOT_APP_LOADER
- S5D9_lab_ADC_I2C_ADS1015_callback_v1
- S5D9_lab_ADC_I2C_ADS1015_callback_v2
- S5D9_lab_ADC_I2C_ADS1015_callback_v2_iic_port
- S5D9_lab_ADC_I2C_ADS1015_callback_v2_sci_port
 - Binaries
 - Includes
 - src
 - synergy
 - Debug
 - script
 - synergy_cfg
 - configuration.xml
 - R7FS5D97E2A01CLK.pincfg
 - S5D9_lab_ADC_I2C_ADS1015_callback_v1 Debug.jlink
 - S5D9_lab_ADC_I2C_ADS1015_callback_v2 Debug.jlink
 - S5D9_lab_ADC_I2C_ADS1015_callback_v2_sci_port Debug.launch
 - S5D9-IOT-Enabler.pincfg
 - synergy_cfg.txt
 - S5D9_lab_ADC_I2C_ADS1015_callback_v2_ssp133

Threads

HAL/Common Stacks

- g_cgc CGC Driver on r_cgc
- g_fmi FMI Driver on r_fmi
- g_elc ELC Driver on r_elc
- g_ioport I/O Port Driver on r_ioport
- g_ads1015 I2C Master Driver on r_sci_i2c
- g_transfer2 Transfer Driver on r_dtc Event SCI0 TXI
- g_transfer3 Transfer Driver on r_dtc Event SCI0 RXI

HAL/Common Objects

Summary | BSP | Clocks | Pins | Threads | Messaging | ICU | Components |

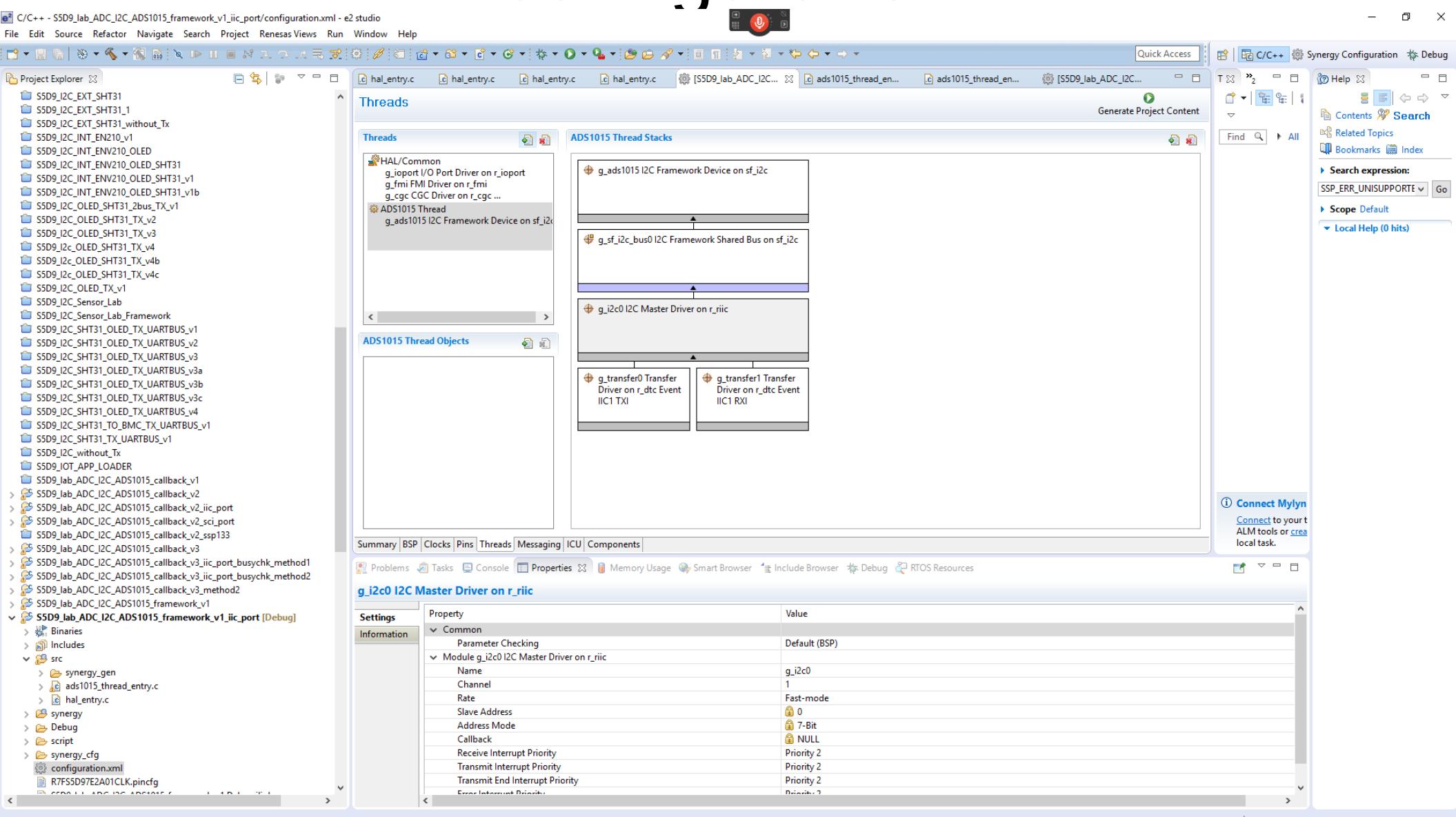
g_ads1015 I2C Master Driver on r_sci_i2c

Property	Value
Common	Default (BSP)
Module g_ads1015 I2C Master Driver on r_sci_i2c	
Name	g_ads1015
Channel	0
Rate	Fast-mode
Slave Address	0x48
Address Mode	7-Bit
SDA Output Delay (nano seconds)	300
Bit Rate Modulation Enable	Enable
Callback	g_ads1015_callback
Receive Interrupt Priority	Priority 2
Transmit Interrupt Priority	Priority 2

Problems | Tasks | Console | Properties | Memory Usage | Smart Browser | Include Browser | Debug | RTOS Resources

Common

Framework with the r_iic driver configuration



Framework with the r_sci_i2c driver configuration

The screenshot shows the e2 studio IDE interface with the following details:

- Project Explorer:** Lists numerous source files and configurations related to the S5D9_I2C project, including various I2C driver configurations (e.g., S5D9_I2C_OLED_SHT31_v1, S5D9_I2C_OLED_SHT31_v1b, etc.) and framework components.
- Threads View:** Displays the thread stack hierarchy:
 - HAL/Common (g_ioport I/O Port Driver on r_ioport, g_fmi FMI Driver on r_fmi, g_cgc CGC Driver on r_cgc ...)
 - ADS1015 Thread (g_ads1015 I2C Framework Device on sf_i2c)
 - ADS1015 Thread Stacks (g_sf_i2c_bus I2C Framework Shared Bus on sf_i2c)
 - ADS1015 Thread Objects (g_i2c0 I2C Master Driver on r_sci_i2c)
 - g_transfer2 Transfer Driver on r_dtc Event SCI0 TXI
 - g_transfer3 Transfer Driver on r_dtc Event SCI0 RXI
- Properties View:** Shows the configuration for the "g_i2c0 I2C Master Driver on r_sci_i2c" module. The "Common" section includes:

Property	Value
Parameter Checking	Default (BSP)
Name	g_i2c0
Channel	0
Rate	Fast-mode
Slave Address	0
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
- Help View:** Provides search and documentation resources.

Configuration Register read

Buf[0] = 85h and Buf[1] = 83h

The screenshot shows the e2 studio IDE interface during a debugging session for the project `SSD9_lab_ADC_I2C_ADS1015_v1`. The Variables view displays the contents of the `buf` array:

Name	Type	Value
buf	uint8_t [20]	0x1ffe2364 <g_main_stack+4036>
buf[0]	uint8_t	133 'j'
buf[1]	uint8_t	85
buf[2]	uint8_t	0 '\0'
buf[3]	uint8_t	0 '\0'
buf[4]	uint8_t	0 '\0'
buf[5]	uint8_t	16 '\020'
buf[6]	uint8_t	4 '\004'
buf[7]	uint8_t	64 '@'
buf[8]	uint8_t	1 '\001'
buf[9]	uint8_t	0 '\0'
buf[10]	uint8_t	3 '\003'
buf[11]	uint8_t	0 '\0'
buf[12]	uint8_t	0 '\0'
buf[13]	uint8_t	1 '\001'
buf[14]	uint8_t	0 '\0'
buf[15]	uint8_t	0 '\0'
buf[16]	uint8_t	254 'p'
buf[17]	uint8_t	15 '\017'
buf[18]	uint8_t	0 '\0'
buf[19]	uint8_t	0 '\0'
err	ssp_err_t	SSP_SUCCESS

The code editor shows the `main.c` file with the following code snippet:

```
26 00004ab8     while (1):
27
28
29
30     while (1) {
31         // read configuration register
32         buf[0] = ADS1015_CONFIGREG_ADDR;
33         err = g_ads1015.p_api->write(g_ads1015.p_ctrl, buf, 1, false);
34         if (SSP_SUCCESS != err) {
35             g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
36             while (1);
37         }
38         err = g_ads1015.p_api->read(g_ads1015.p_ctrl, buf, 2, false);
39         if (SSP_SUCCESS != err) {
40             g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
41             while (1);
42         }
43     }

```

The Project Explorer on the right lists several I2C driver variants:

- SSD9_I2C_SHT31_OLED_TX_UARTBUS_v3a
- SSD9_I2C_SHT31_OLED_TX_UARTBUS_v3b
- SSD9_I2C_SHT31_OLED_TX_UARTBUS_v3c
- SSD9_I2C_SHT31_OLED_TX_UARTBUS_v4
- SSD9_I2C_SHT31_TO_BMC_TX_UARTBUS_v1
- SSD9_I2C_SHT31_TX_UARTBUS_v1
- SSD9_I2C_without_Tx
- SSD9_IOT_APP_LOADER
- SSD9_lab_ADC_I2C_ADS1015_v1
 - Binaries
 - Includes
 - src
 - synergy_gen
 - hal_entry.c
 - synergy
 - Debug
 - script

Configuration Register write/read

Buf[0] = C3h and Buf[1] = 83h

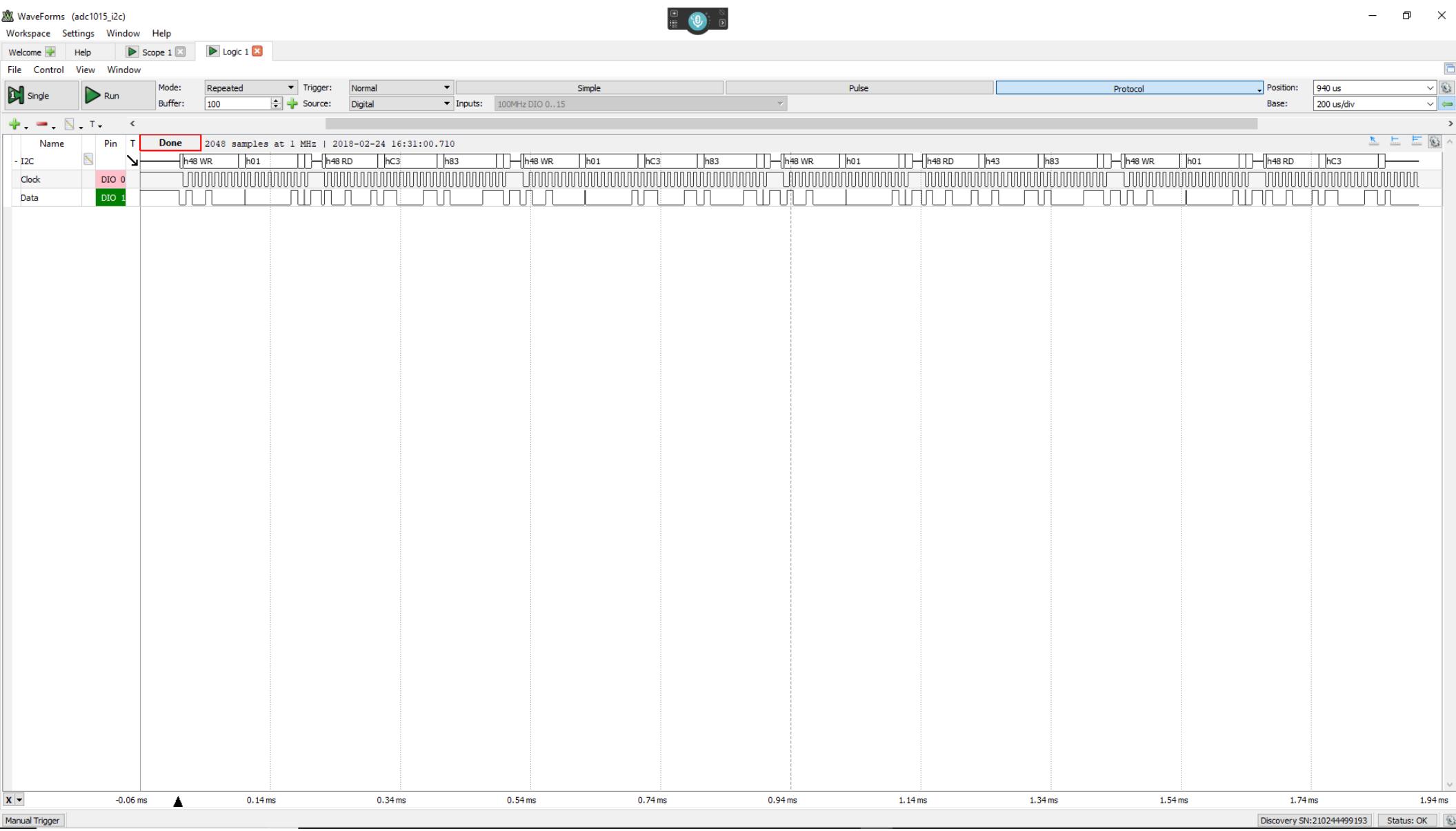
The screenshot shows the e2 studio IDE interface with the following details:

- Project:** SSD9_lab_ADC_I2C_AMS1015_v1
- File:** src/hal_entry.c
- Breakpoint:** Thread #1 (single core) (Suspended : Step)
- Registers:** Shows the state of registers for the current thread.
- Memory Dump:** A table showing the contents of the buffer (buf) array. The value at buf[0] is highlighted in yellow and has a tooltip: Name : buf[0], Details:195 'A', Default:195 'A', Decimal:195, Hex:0xc3, Binary:11000011, Octal:0303.
- Source Code:** The hal_entry.c file is open, showing the main loop and configuration register access logic.
- Outline:** Shows the project structure with various source files and configurations.
- Console:** Displays temporary breakpoint information and the current stack status.

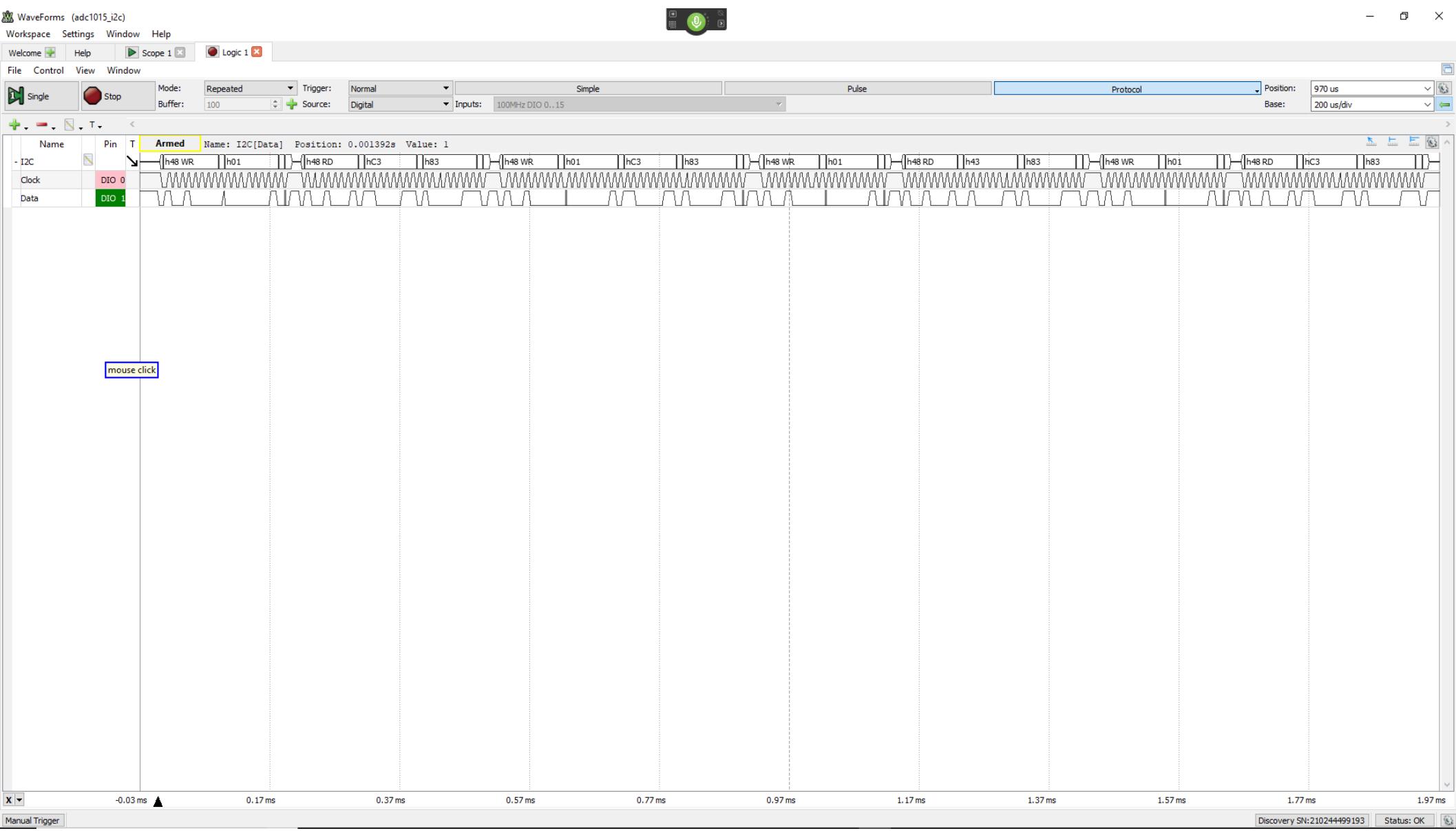
```
52 00004ble
53
54
55     while (1);
56
57 00004af4     while (1) {
58     // read configuration register
59     buf[0] = ADS1015_CONFIGREG_ADDR;
60     err = g_ads1015.p_api->write(g_ads1015.p_ctrl, buf, 1, false);
61     if (SSP_SUCCESS != err) {
62         g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
63         while (1);
64     }
65     err = g_ads1015.p_api->read(g_ads1015.p_ctrl, buf, 2, false);
66     if (SSP_SUCCESS != err) {
67         g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
68         while (1);
69     }
70 }
```

Temporary breakpoint 1, main () at/src/synergy_gen/main.c:5
5 hal_entry();

C3 (1100_0011) → 43 (0100_0011) good
WR OS bit → RD OS bit (0=busy)



Stop when OS bit = 1



$0x4450 \rightarrow 0x445$ or 1093
 $1093 / 2048 \times 4.096 = 2.186V$
 FSR +/- 4.096 and only positive side use 2048

Debug - S5D9_lab_ADC_I2C_ADS1015_v3/src/hal_entry.c - e2 studio

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

Quick Access C/C++ Synergy Configuration Debug

Help Contents Search Related Topics Bookmarks Index Search expression: P_ERR_UNSUPPORTED Scope Default Local Help (0 hits)

Variables Breakpoints Registers Modules Expressions Eventpoints IO Registers Peripherals Address

Expression	Type	Value	Address
distance		Error: Multiple errors reported.\ Failed to execute...	
g_time_start_flag		Error: Multiple errors reported.\ Failed to execute...	
g_time_end_flag		Error: Multiple errors reported.\ Failed to execute...	
g_echo_start_time		Error: Multiple errors reported.\ Failed to execute...	
g_echo_end_time		Error: Multiple errors reported.\ Failed to execute...	
(x): convdata	uint16_t	1093	0x1ffe03a8
(x): ain_level	float	2.18600011	0x1ffe03a4
Add new expression			

hal_entry.c [S5D9_lab_ADC_I2C_ADS1015_v3] Synergy Configuration startup_S5D9.c main.c

```

101 000086a8      g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
102 000086b6      while (1);
103
104 000086b8      if (buf[0] & 0x80) {
105 000086c2          g_ioport.p_api->pinWrite(YEL_LED_PIN, LED_ON); // turn on red led for error
106
107 000086d0      buf[0] = ADS1015_CONVERSION_ADDR;
108 000086d4      err = g_ads1015.p_api->write(g_ads1015.p_ctrl, buf, 1, false);
109 000086e0      if (SSP_SUCCESS != err) {
110 000086f2          g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
111 00008700          while (1);
112
113 00008702      err = g_ads1015.p_api->read(g_ads1015.p_ctrl, buf, 2, false);
114 0000871a      if (SSP_SUCCESS != err) {
115 00008720          g_ioport.p_api->pinWrite(RED_LED_PIN, LED_ON); // turn on red led for error
116 0000872e          while (1);
117
118 00008730      // convdata[11:0] = buf[0:1] >> 4
119 00008738      convdata = buf[0];
120 00008744      convdata = convdata << 8;
121 00008754      convdata = convdata + (uint16_t)buf[1];
122 00008760      convdata = convdata >> 4;
123 00008760      ain_level = (float) convdata * 4.096 / 2048; // FSR = +/-4.096. For positive voltage range, only 2048 steps.
124 000087aa      while (1);
125
126 000087ac  }
127
128  }
129

```

Outline Project Explorer

- S5D9_I2C_OLED_SHT31_TX_v2
- S5D9_I2C_OLED_SHT31_TX_v3
- S5D9_I2C_OLED_SHT31_TX_v4
- S5D9_I2C_OLED_SHT31_TX_v4b
- S5D9_I2C_OLED_SHT31_TX_v4c
- S5D9_I2C_OLED_TX_v1
- S5D9_I2C_Sensor_Lab
- S5D9_I2C_Sensor_Lab_Framework
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v1
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v2
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3a
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3b
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v3c
- S5D9_I2C_SHT31_OLED_TX_UARTBUS_v4
- S5D9_I2C_SHT31_TO_BMC_TX_UARTBUS_v1
- S5D9_I2C_SHT31_TX_UARTBUS_v1
- S5D9_I2C_without_Tx
- S5D9_IOT_APP_LOADER
- S5D9_lab_ADC_I2C_ADS1015_v1
- S5D9_lab_ADC_I2C_ADS1015_v2
- S5D9_lab_ADC_I2C_ADS1015_v3
 - Binaries
 - Includes
 - src
 - synergy_gen
 - common_data.c
 - common_data.h

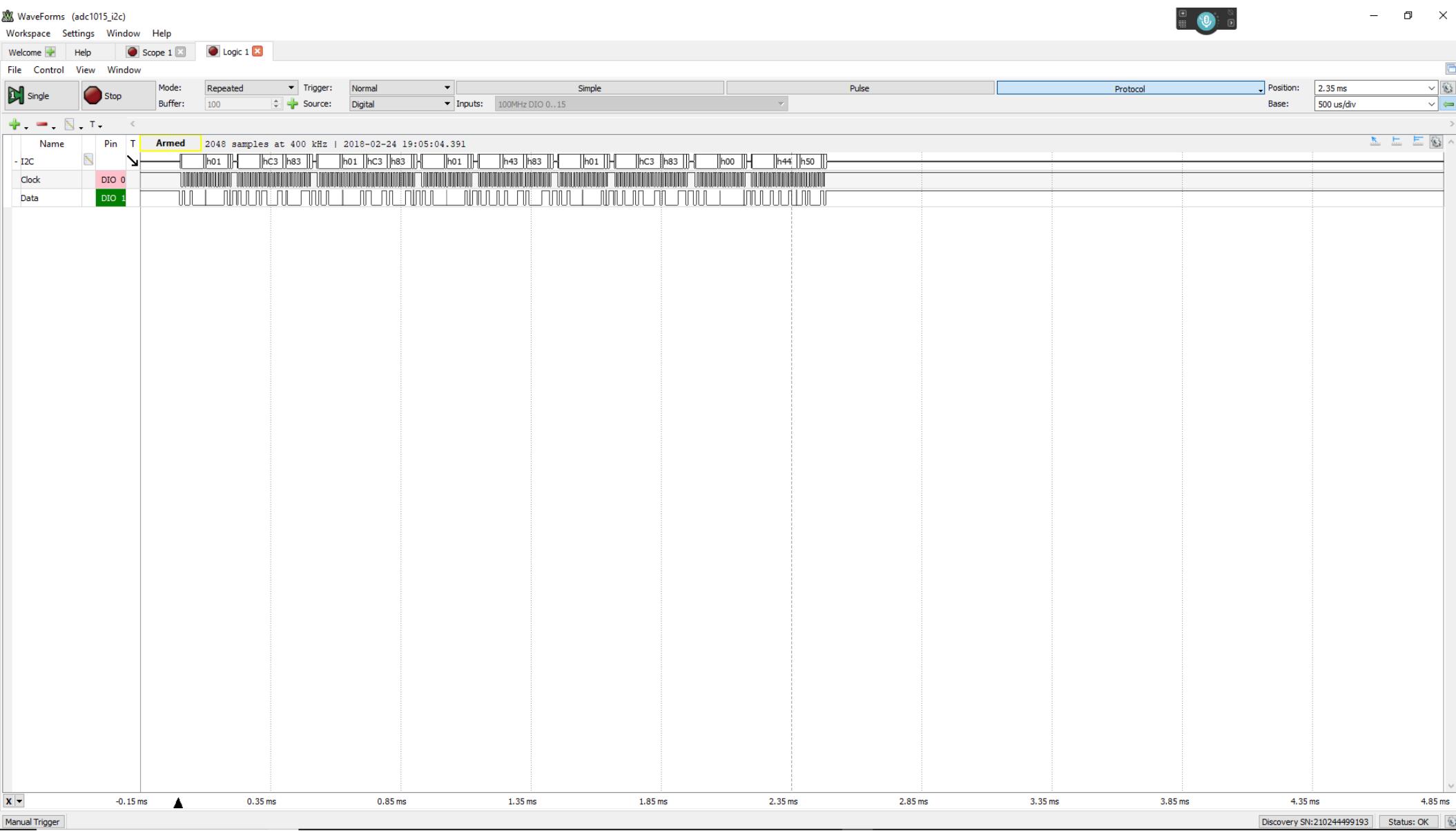
Console Tasks Renesas Coverage Memory Usage Performance Analysis Profile Real-time Chart Trace Visual Expression ARM CoreSight ITM Live Trace... Smart Browser Problems Executables Memory RTOS Resources RTOS Resources

S5D9_lab_ADC_I2C_ADS1015_v3 Debug [Renesas GDB Hardware Debugging] C:/Renesas/e2_studio540_ssp130_s5d9iot/DebugComp/arm-none-eabi-gdb (7.8.2)

hal_entry () at/src/hal_entry.c:124

124 while (1);

Suspended



0.379 KHz (fast mode)

