

Matthew Hoffman
CpE 403 – Advanced Embedded Systems
CC1350 Lab 4

Task 1- Importing the WSN Examples

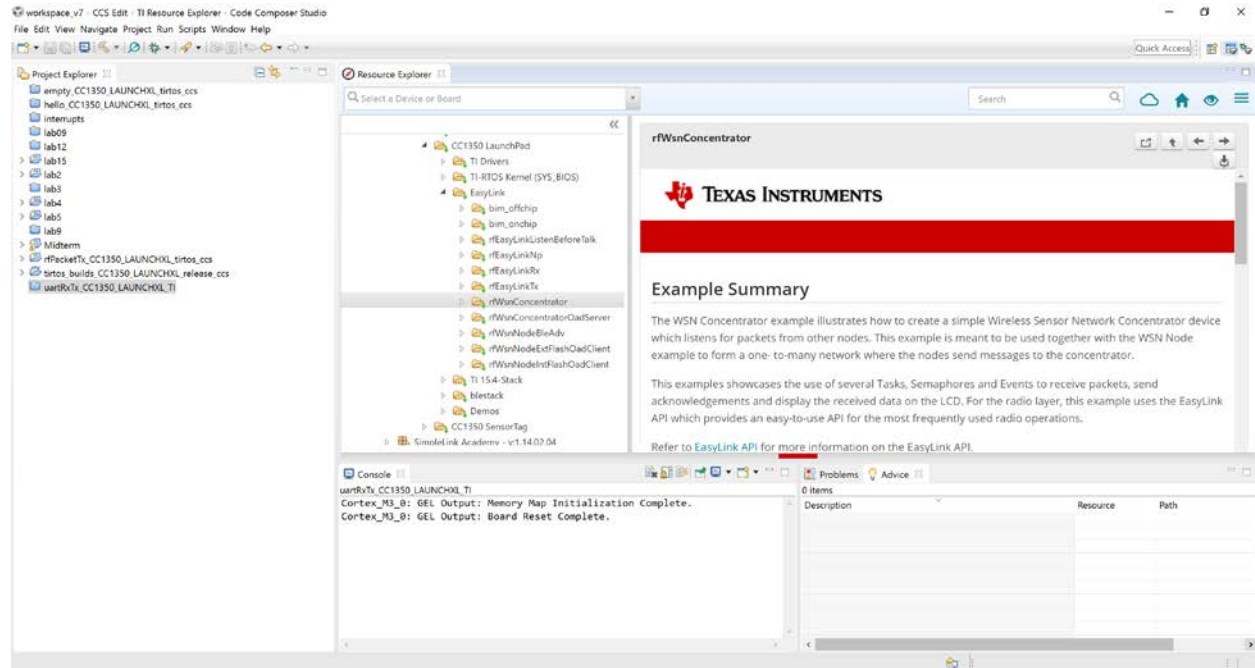


Figure 1. Task 1.a - rfWsnConcentrator

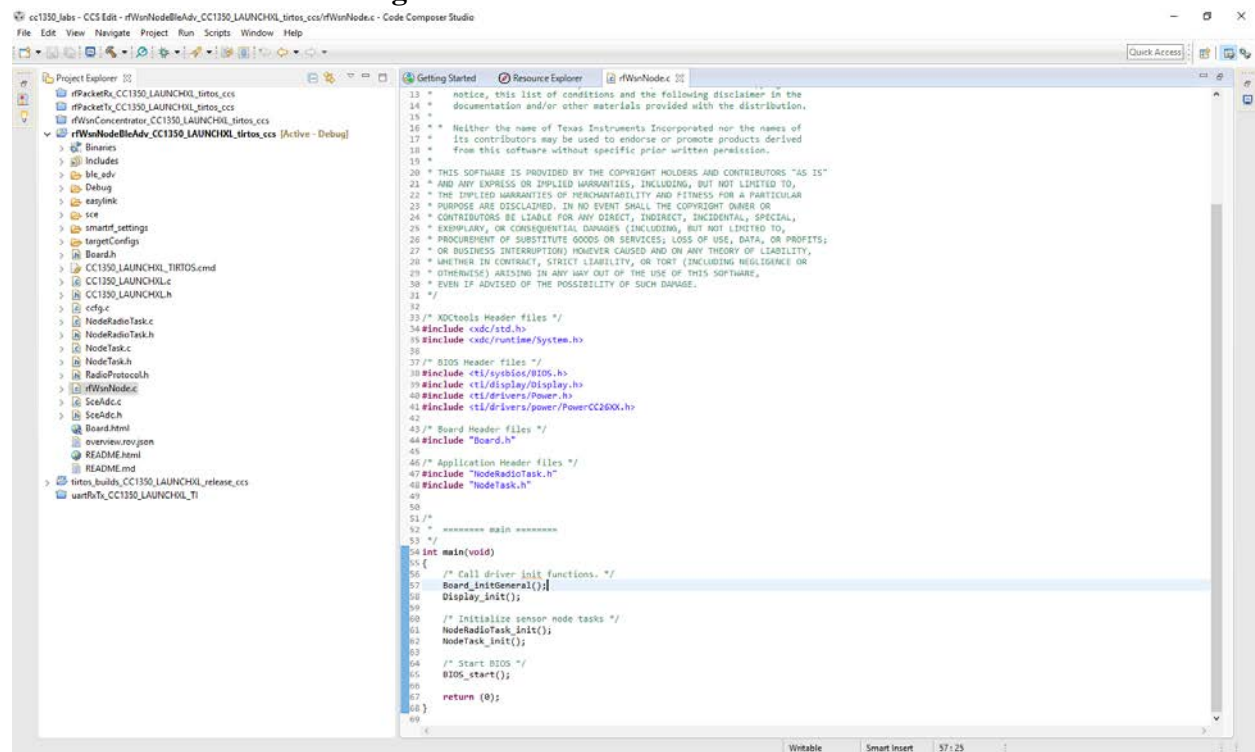


Figure 2. Task 1.b – rfWsnConcentrator

Task 2 – Putting it all to work

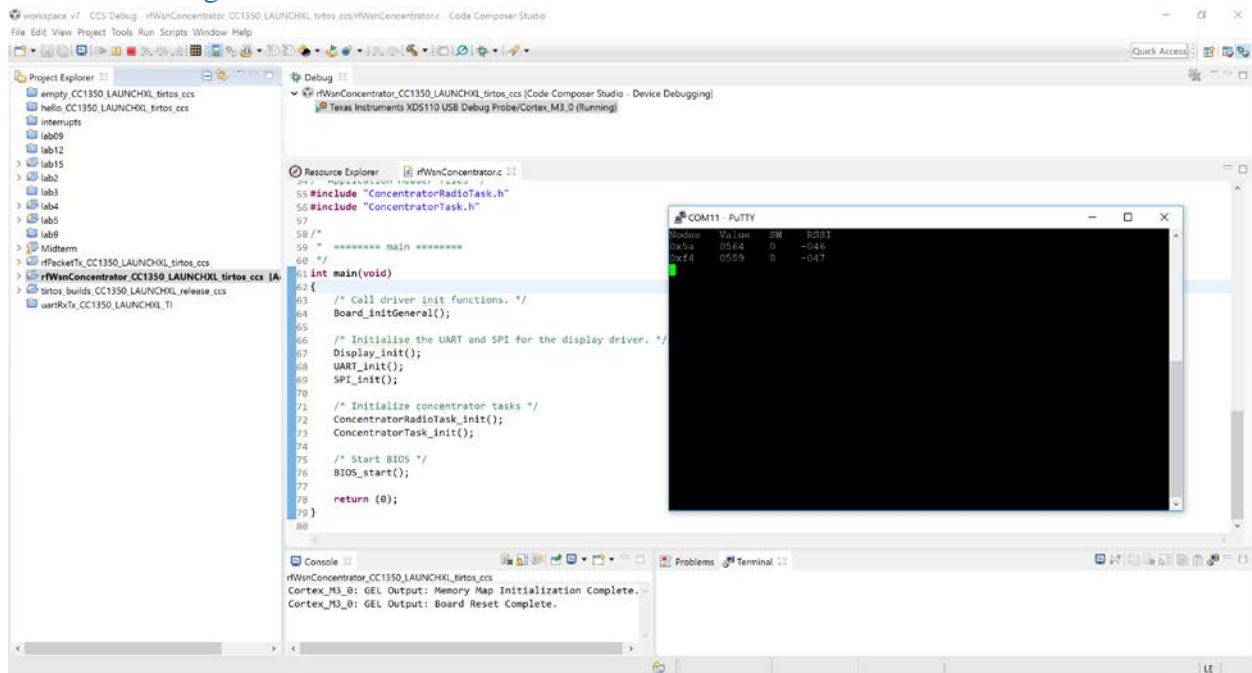


Figure 3. Task 2.a – Node connections

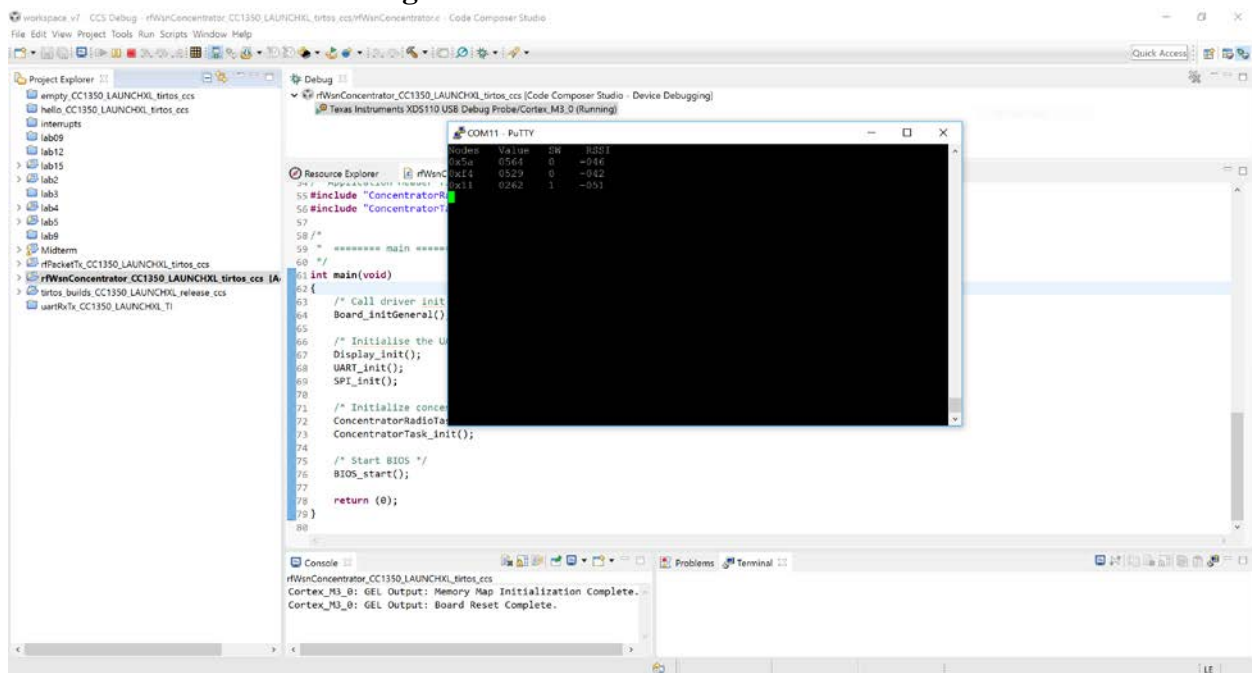


Figure 4. Task 2.b – ADC Value defaults

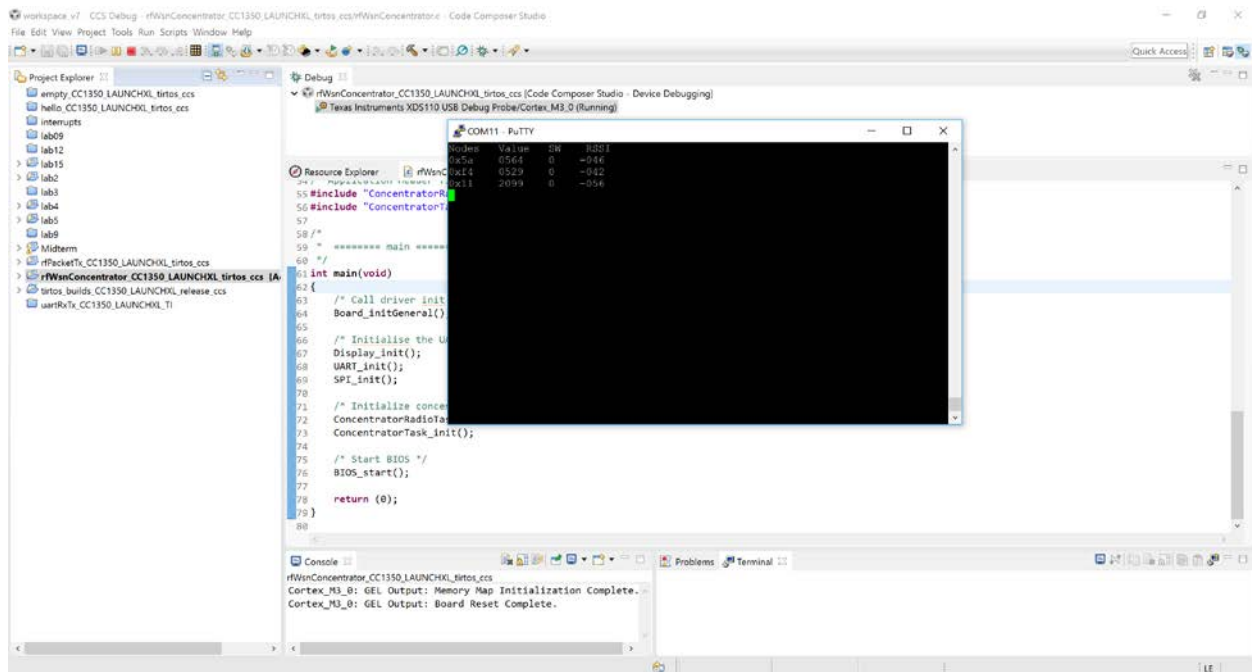


Figure 5. Task 2.c – ADC Value with changed ADV values

Task 3 – Change RF Channel

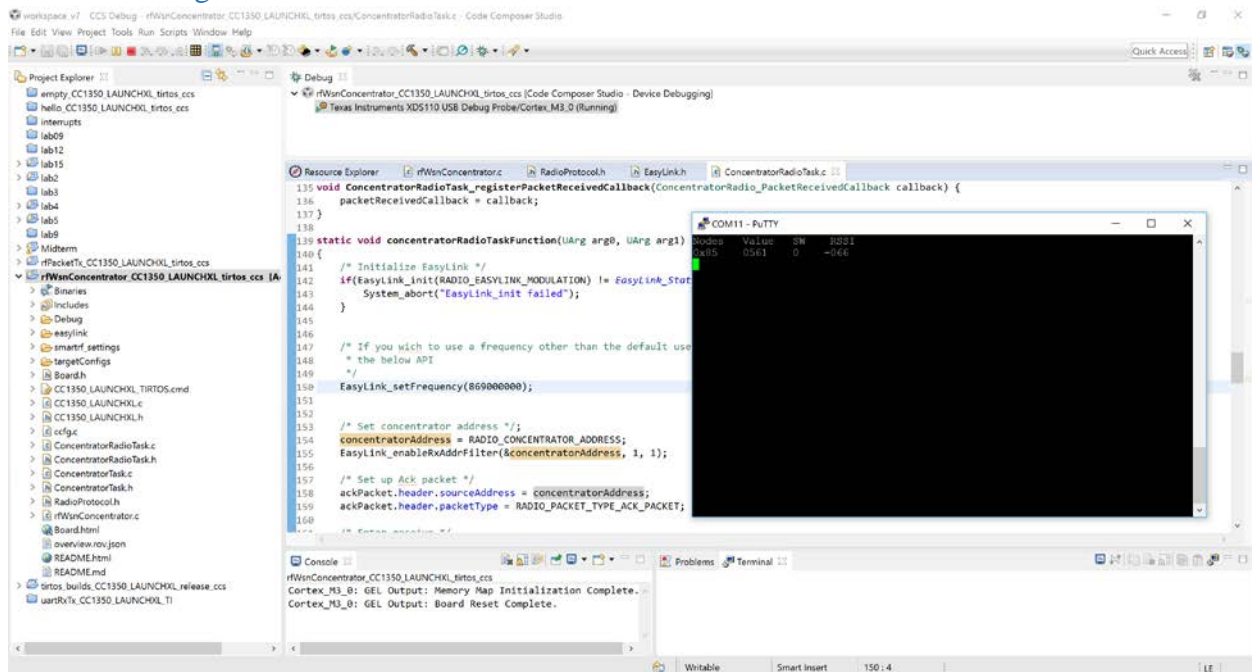


Figure 6. Task 3.a Use of original frequency 868000000

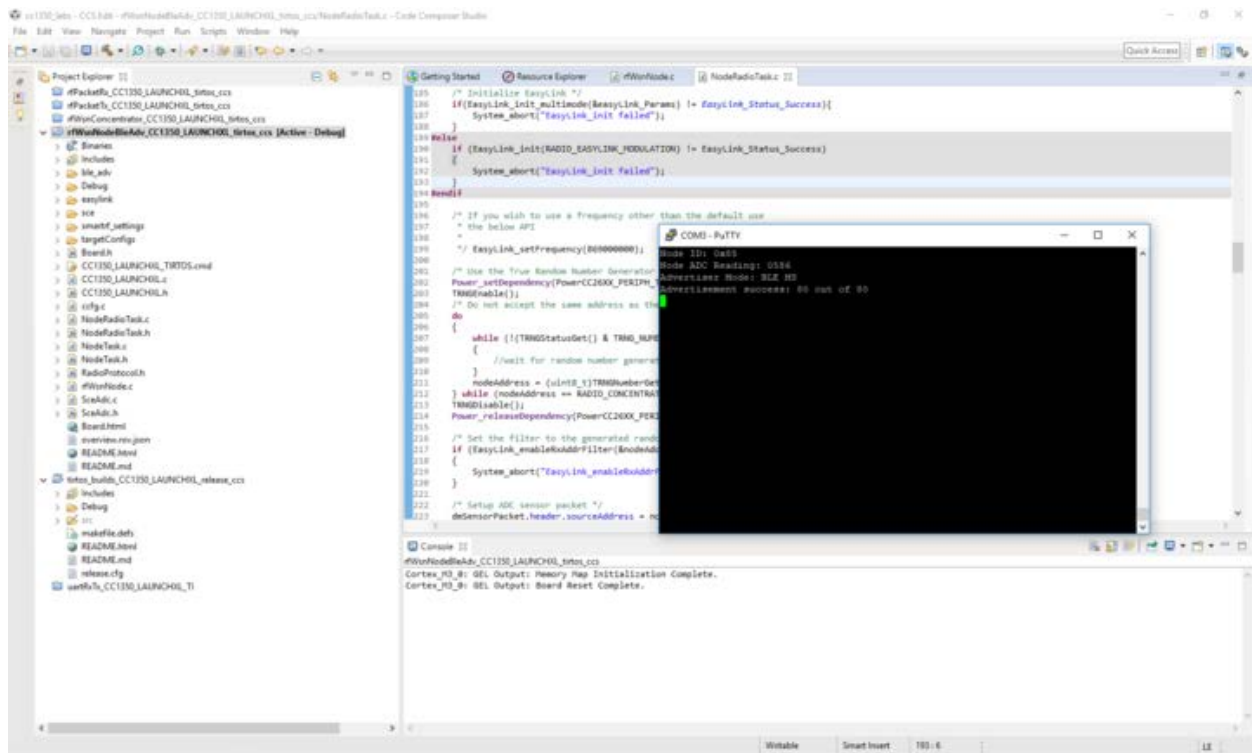


Figure 7. Task 3.b Change of original frequency to 867000000

Task 4 – Switch from 2-GFSK 50 kbps to Long Range Mode (LRM)

```

38
39 #define RADIO_CONCENTRATOR_ADDRESS    0x00
40 #define RADIO_EASYLINK_MODULATION     EasyLink_Phy_50kbps2gfsk
41
42 #define RADIO_PACKET_TYPE_ACK_PACKET    0
43 typedef enum
44 {
45     EasyLink_Phy_Custom = 0,           //!< Customer Phy specific settings exported from SmartRF Studio
46     EasyLink_Phy_50kbps2gfsk = 1,      //!< Phy settings for Sub1G 50kbps data rate, IEEE 802.15.4g GFSK.
47     EasyLink_Phy_625bpsLrm = 2,        //!< Phy settings for Sub1G 625bps data rate, Long Range Mode.
48 }

```

Figure 8. Task 4 –Change of short to long range

Task 5 – Modify Code Running on the Sensor Controller

The screenshot displays the Sensor Controller Studio 1.5.0.188 interface. The left sidebar shows the project structure with 'ADC Sample' selected, containing 'Initialization Code', 'Execution Code', and 'Termination Code'. The central pane shows the 'ADC Sample - Execution Code' with the following C code:

```

1 // Enable the ADC
2 adcEnableSync(ADC_REF_FIXED, ADC_SAMPLE_TIME_2P7_US,
3   ADC_TRIGGER_MANUAL);
4
5 // Sample the ADC
6 S16 adcValue;
7 adcGenManualTrigger();
8 adcReadFifo(adcValue);
9 output.adcValue = adcValue;
10
11 // Disable the ADC
12 adcDisable();
13
14 // Alert the driver if outside of change mask
15 U16 adcMaskedBits = adcValue & cfg.changeMask;
16 if (adcMaskedBits != state.oldAdcMaskedBits) {
17   fwGenAlertInterrupt();
18   state.samplesSinceLastReport = 0;
19 } else {
20   state.samplesSinceLastReport =
21   state.samplesSinceLastReport + 1;
22 }
23
24 //Alert driver if minimum report interval has expired
25 if(cfg.minReportInterval != 0) {
26   if(state.samplesSinceLastReport >=
27   cfg.minReportInterval) {
28     fwGenAlertInterrupt();
29     state.samplesSinceLastReport = 0;
30   }
31 }
32
33 // Save old masked ADC value
34 state.oldAdcMaskedBits = adcValue & cfg.changeMask;
35
36 // Schedule the next execution
37 fwScheduleTask(2);

```

The right sidebar contains two panels:

- Constants:** A table listing various constants and their values.

Constant	Value
BIN_COUNT	5
THRESHOLD_COUNT	6
ADC_INPUT_DCOUPL	0x0020
ADC_INPUT_VDDS	0x0080
ADC_INPUT_VSS	0x0040
ADC_REF_FIXED	0x0000
ADC_REF_VDDS_REL	0x0008
ADC_SAMPLE_TIME_10P6_US	5
ADC_SAMPLE_TIME_10P9_MS	15
ADC_SAMPLE_TIME_170_US	9
ADC_SAMPLE_TIME_1P37_MS	12
ADC_SAMPLE_TIME_21P2_US	6
- Data structures:** A table showing the initial values for data structures.

Data structure	Initial
cfg	
changeMask	0
minReportInterval	0
output	
adcValue	0
state	
oldAdcMaskedBits	0
samplesSinceLastReport	0

Below these panels are buttons for 'Add', 'Edit', and 'Remove' for both constants and data structures. At the bottom, there is a list of 'Available procedures' including functions like adcDisable, adcEnableSync, and fwScheduleTask.

Figure 9. Task 5.a – Sensor Controller Studio Execution Code

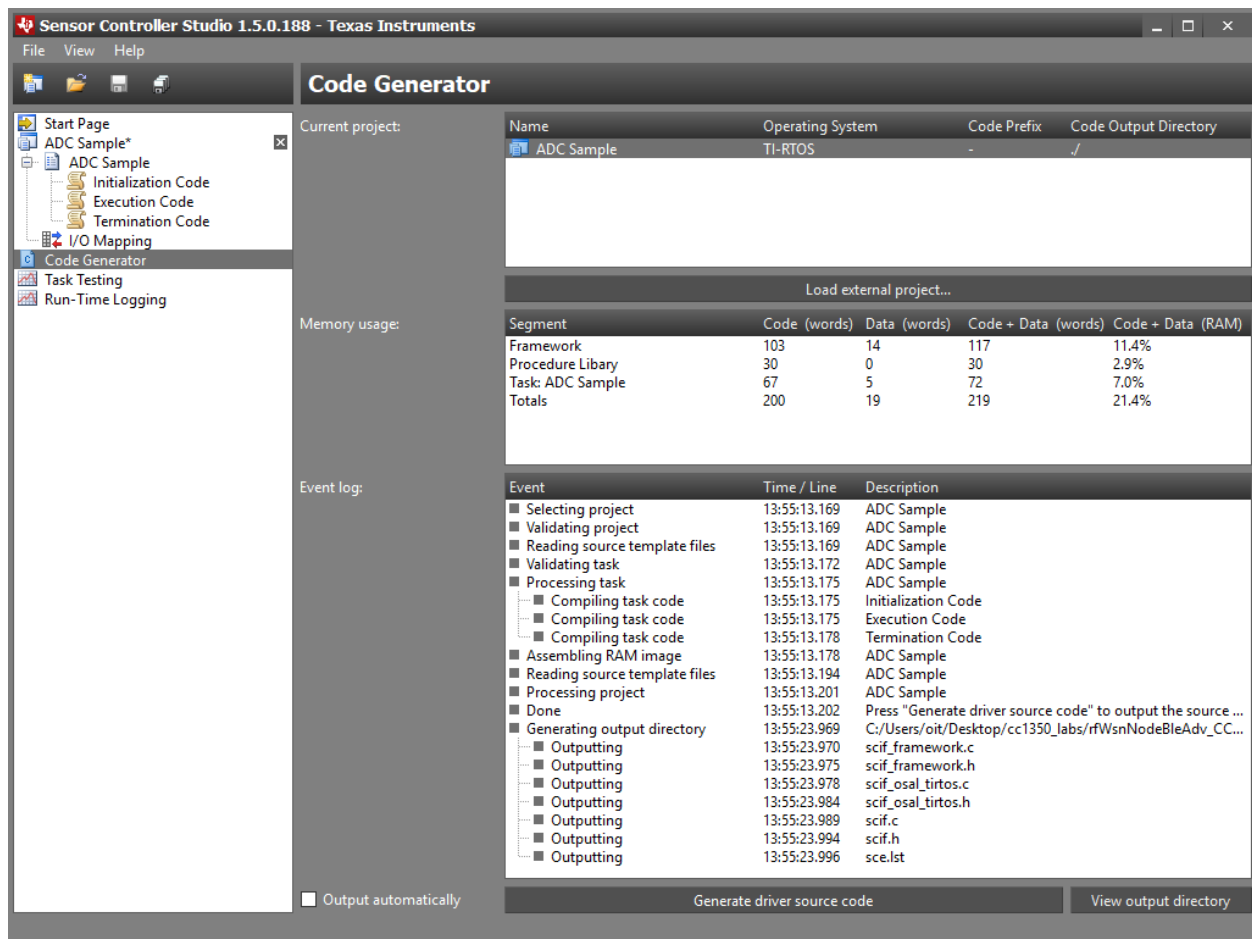


Figure 10. Task 5.b – Code Generator. Sampling Change.

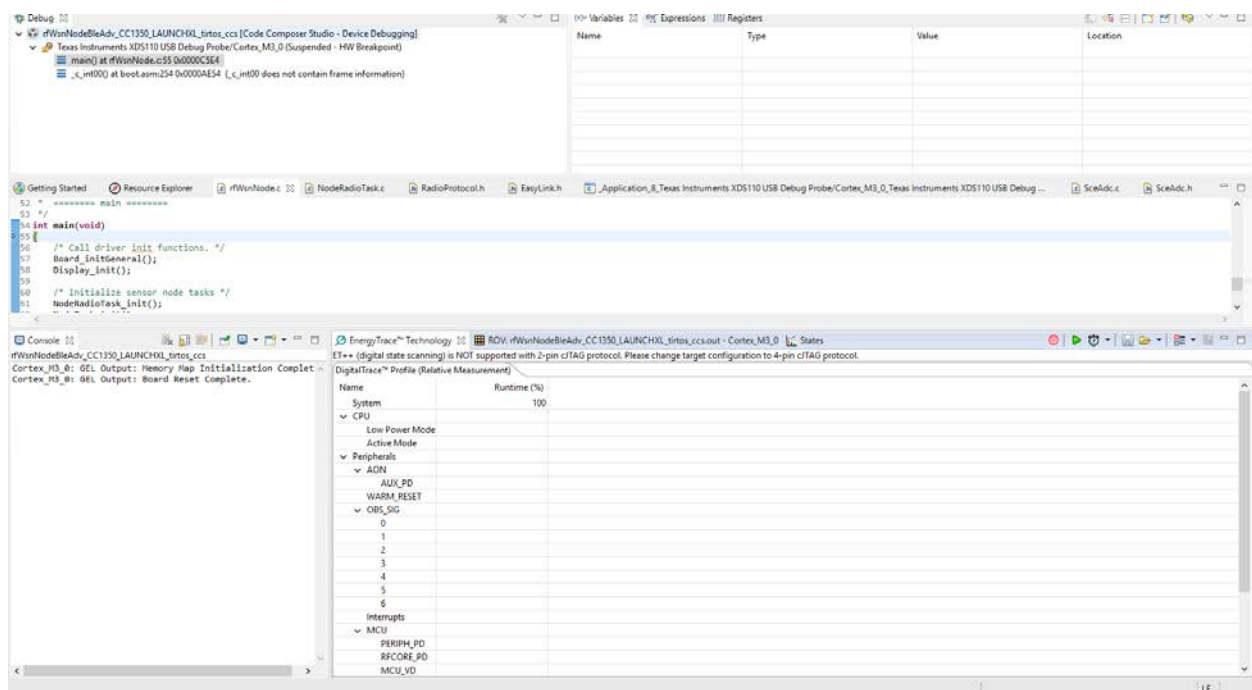


Figure 11. Task 5.c – Energy Trace