

# Securing IoT Devices using Arm TrustZone®

## Class 3: Creating your First TrustZone Application

November 28, 2018  
Jacob Beningo

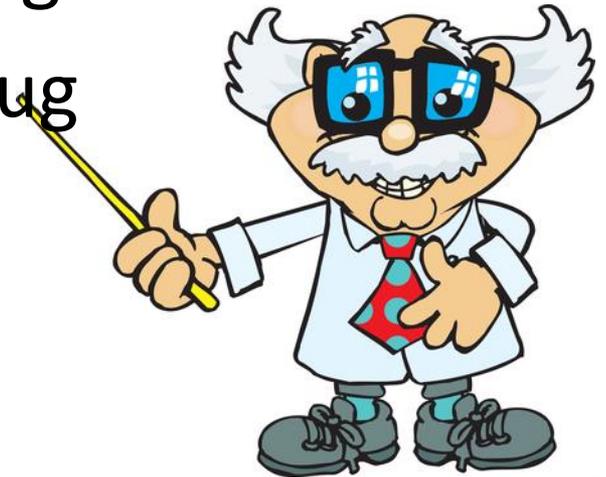
# Course Overview

## Topics:

- Understanding Embedded System Security
- Introduction to Arm TrustZone®
- **Creating your First TrustZone Application**
- Designing and Debugging a Secure Boot Solution
- Securing a RTOS Application with TrustZone

# Session Overview

- Project Setup
- The TrustZone Workspace
- The SAU
- TrustZone Example Walk-through
- Setting up simulation and debug



Presented by:

# Project Setup

The screenshot shows the Pack Installer application window. The left pane displays a tree view of devices, with 'ARM Cortex M33' selected and highlighted in red. The right pane shows a list of example packs under the 'Examples' tab, with the 'TrustZone for ARMv8-M RTOS Security Tests' example selected and highlighted in red.

Device	Summary
All Devices	5760 Devices
ABOV Semiconductor	20 Devices
Active-Semi	4 Devices
Ambiq Micro	8 Devices
Analog Devices	14 Devices
ARM	50 Devices
ARM Cortex A5	1 Device
ARM Cortex A7	1 Device
ARM Cortex A9	1 Device
ARM Cortex M0	2 Devices
ARM Cortex M0 plus	3 Devices
ARM Cortex M3	3 Devices
ARM Cortex M4	4 Devices
ARM Cortex M7	6 Devices
ARM Cortex M33	8 Devices
ARM Cortex M33 (MPS3)	3 Devices
ARM SC000	1 Device
ARM SC300	1 Device
ARMv8-M Baseline	3 Devices
ARMv8-M Mainline	10 Devices
AutoChips	1 Device
Cypress	425 Devices
GigaDevice	137 Devices
HDSC	26 Devices
Holtek	145 Devices

Example	Action	Description
CMSIS-RTOS2 Blinky (uVision Simulator)	Copy	CMSIS-RTOS2 Blinky example
CMSIS-RTOS2 RTX5 Message Queue (uVision ...)	Copy	CMSIS-RTOS2 Message Queue Example
CMSIS-RTOS2 RTX5 Migration (uVision Simul...	Copy	CMSIS-RTOS2 mixed API v1 and v2
DSP_Lib Class Marks example (uVision Simula...	Copy	DSP_Lib Class Marks example
DSP_Lib Convolution example (uVision Simul...	Copy	DSP_Lib Convolution example
DSP_Lib Dotproduct example (uVision Simula...	Copy	DSP_Lib Dotproduct example
DSP_Lib FFT Bin example (uVision Simulator)	Copy	DSP_Lib FFT Bin example
DSP_Lib FIR example (uVision Simulator)	Copy	DSP_Lib FIR example
DSP_Lib Graphic Equalizer example (uVision S...	Copy	DSP_Lib Graphic Equalizer example
DSP_Lib Linear Interpolation example (uVisio...	Copy	DSP_Lib Linear Interpolation example
DSP_Lib Matrix example (uVision Simulator)	Copy	DSP_Lib Matrix example
DSP_Lib Signal Convergence example (uVisio...	Copy	DSP_Lib Signal Convergence example
DSP_Lib Sinus/Cosinus example (uVision Sim...	Copy	DSP_Lib Sinus/Cosinus example
DSP_Lib Variance example (uVision Simulator)	Copy	DSP_Lib Variance example
NN Library CIFAR10 (uVision Simulator)	Copy	Neural Network CIFAR10 example
NN Library GRU (uVision Simulator)	Copy	Neural Network GRU example
SCVD Complex Example (uVision Simulator)	Copy	More complex Component Viewer example
SCVD Event Statistics (uVision Simulator)	Copy	Shows the usage of start/stop events with Event
SCVD Simple Example (uVision Simulator)	Copy	Simple Example introducing the Component Vi
SCVD in MyComponent (uVision Simulator)	Copy	Example showing Component Viewer and Event
SCVD printf Redirect Example (uVision Simula...	Copy	Retargeting STDOUT via Event Recorder
TrustZone for ARMv8-M No RTOS (uVision Si...	Copy	Bare-metal secure/non-secure example without
TrustZone for ARMv8-M RTOS (uVision Simul...	Copy	Secure/non-secure RTOS example with thread c
TrustZone for ARMv8-M RTOS Security Tests (...)	Copy	Secure/non-secure RTOS example with security

# Project Setup

Copy Example

Destination Folder: C:\Keil\Example

Use Pack Folder Structure     Launch µVision

OK    Cancel

SCVD printf Redirect Example (uVision Simula...	Copy	Retargeting STDOUT via Event Recorder
TrustZone for ARMv8-M No RTOS (uVision Si...	Copy	Bare-metal secure/non-secure example without
TrustZone for ARMv8-M RTOS (uVision Simul...	Copy	Secure/non-secure RTOS example with thread c
TrustZone for ARMv8-M RTOS Security Tests (...)	Copy	Secure/non-secure RTOS example with security

# The TrustZone Workspace

The screenshot displays the uVision IDE interface. The top menu bar includes File, Edit, View, Project, Flash, Debug, Peripherals, Tools, SVCS, Window, and Help. The toolbar contains various icons for file operations and simulation. The Project Explorer on the left shows a workspace with two projects: Project: CM33\_s (highlighted in green) and Project: CM33\_ns (highlighted in red). Project: CM33\_s contains folders for FVP Simulation Model, Secure Code, Interface, and Documentation, along with files main\_s.c, interface.c, and Abstract.txt. Project: CM33\_ns contains folders for FVP Simulation Model, Non-secure Code, and CMSE Library, along with files main\_ns.c and CM33\_s\_CMSE\_Lib.o. The main editor window shows the content of Abstract.txt, which describes the project setup for TrustZone on ARMv8-M applications and lists function calls for secure and non-secure states.

C:\Keil\Example\CMSIS\RTOS2\RTX\Examples\TrustZoneV8M\NoRTOS\CM33\_ns\CM33\_ns.uvprojx - uVision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

vRegTest1Task

FVP Simulation Model

Project

Workspace

Project: CM33\_s

- FVP Simulation Model
  - Secure Code
    - main\_s.c
  - Interface
    - interface.c
  - Documentation
    - Abstract.txt
  - CMSIS
  - Device

Project: CM33\_ns

- FVP Simulation Model
  - Non-secure Code
    - main\_ns.c
  - CMSE Library
    - CM33\_s\_CMSE\_Lib.o
  - CMSIS
  - Device

Abstract.txt

This ARM Cortex-M33 secure/non-secure example project that shows the setup for TrustZone for ARMv8-M applications. The application uses CMSIS and can be executed on a Fixed Virtual Platform (FVP) simulation model.

The application demonstrates function calls between secure and non-secure state.

Secure application:

- Setup code and start non-secure application.

Non-secure application:

- Calls a secure function from non-secure state.
- Calls a secure function that call back to a non-secure function.

Output:

Variables used in this application can be viewed in the Debugger Watch window.

# The TrustZone Workspace

The screenshot shows the uVision IDE interface. The title bar indicates the project path: `C:\Keil\Example\CMSIS\RTOS2\RTX\Examples\TrustZoneV8M\NoRTOS\CM33_ns\CM33_ns.uvprojx - uVision`. The menu bar includes File, Edit, View, Project, Flash, Debug, Peripherals, Tools, SVCS, Window, and Help. The toolbar contains various icons for file operations and simulation. The Project Explorer on the left shows a workspace with two projects: `Project: CM33_s` and `Project: CM33_ns`. Under `Project: CM33_ns`, the `CM33_s_CMSE_Lib.o` file is highlighted with a red box. The main editor window displays the `Abstract.txt` file with the following content:

```
This ARM Cortex-M33 secure/non-secure example project that shows the setup for TrustZone for ARMv8-M applications. The application uses CMSIS and can be executed on a Fixed Virtual Platform (FVP) simulation model.

The application demonstrates function calls between secure and non-secure state.

Secure application:
- Setup code and start non-secure application.

Non-secure application:
- Calls a secure function from non-secure state.
- Calls a secure function that call back to a non-secure function.

Output:
Variables used in this application can be viewed in the Debugger Watch window.
```

# Configuring the SAU

The screenshot displays the configuration of the Security Attribution Unit (SAU) in an IDE. The left pane shows the project structure for 'Project: CM33\_s' and 'Project: CM33\_ns'. The right pane shows the configuration options for the SAU, with several options highlighted in colored boxes:

- Enable SAU** (blue box):
  - Enable SAU:
  - When SAU is disabled:  All Memory is Secure
- Initialize Security Attribution Unit (SAU) Address Regions** (orange box):
  - Initialize SAU Region 0: 
    - Start Address: 0x0000 0000
    - End Address: 0x001F FFFF
    - Region is: Secure, Non-Secure Callable
  - Initialize SAU Region 1:
  - Initialize SAU Region 2:
  - Initialize SAU Region 3:
  - Initialize SAU Region 4:
  - Initialize SAU Region 5:
  - Initialize SAU Region 6:
  - Initialize SAU Region 7:
- Setup behaviour of Sleep and Exception Handling** (purple box):
  - Setup behaviour of Sleep and Exception Handling: 
    - Deep Sleep can be enabled by: Secure state only
    - System reset request accessible from: Secure state only
    - Priority of Non-Secure exceptions is: Lowered to 0x80-0xFF
    - BusFault, HardFault, and NMI target: Secure state
- Setup behaviour of Floating Point Unit** (light blue box):
  - Setup behaviour of Floating Point Unit: 
    - Floating Point Unit usage: Secure and Non-Secure state
    - Treat floating-point registers as Secure: Disabled
    - Clear on return (CLRONRET) accessibility: Secure and Non-Secure state
    - Clear floating-point caller saved registers on exception return: Enabled

# Configuring the SAU

The screenshot displays an IDE interface with two panes. The left pane shows a project tree for 'Project: CM33\_s' with a red box around 'partition\_ARMCM33.h (Startup)' and a red arrow pointing to the right pane. The right pane shows the configuration for 'Initialize ITNS 0 (Interrupts 0..31)' with a table of interrupt targets and their security states.

Option	Value
<input checked="" type="checkbox"/> Initialize ITNS 0 (Interrupts 0..31)	<input checked="" type="checkbox"/>
Interrupt 0	Non-Secure state
Interrupt 1	Non-Secure state
Interrupt 2	Secure state
Interrupt 3	Non-Secure state
Interrupt 4	Secure state
Interrupt 5	Non-Secure state
Interrupt 6	Secure state
Interrupt 7	Secure state
Interrupt 8	Secure state
Interrupt 9	Non-Secure state
Interrupt 10	Secure state
Interrupt 11	Secure state
Interrupt 12	Non-Secure state
Interrupt 13	Secure state
Interrupt 14	Secure state
Interrupt 15	Secure state
Interrupt 16	Secure state

# Secure Code – main.c

```
28 /* Use CMSE intrinsics */
29 #include <arm_cmse.h>
30
31 #include "RTE_Components.h"
32 #include CMSIS_device_header
33
34 /* TZ_START_NS: Start address of non-secure application */
35 #ifndef TZ_START_NS
36 #define TZ_START_NS (0x200000U)
37 #endif
38
39 /* typedef for non-secure callback functions */
40 typedef void (*funcptr_void) (void) __attribute__((cmse_nonsecure_call));
41
42 /* Secure main() */
43 int main(void) {
44     funcptr_void NonSecure_ResetHandler;
45
46     /* Add user setup code for secure part here*/
47
48     /* Set non-secure main stack (MSP_NS) */
49     __TZ_set_MSP_NS(*(uint32_t *) (TZ_START_NS));
50
51     /* Get non-secure reset handler */
52     NonSecure_ResetHandler = (funcptr_void) (*(uint32_t *) ((TZ_START_NS) + 4U));
53
54     /* Start non-secure state software application */
55     NonSecure_ResetHandler();
56
57     /* Non-secure software does not return, this code is not executed */
58     while (1) {
59         __NOP();
60     }
61 }
```

# Secure Code – interface.c

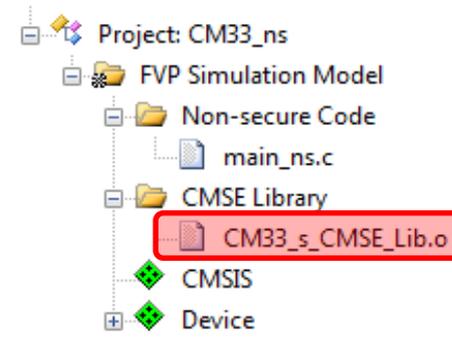
```
27 #include <arm_cmse.h>    // CMSE definitions
28 #include "interface.h"  // Header file with secure interface API
29
30 /* typedef for non-secure callback functions */
31 typedef funcptr funcptr_NS __attribute__((cmse_nonsecure_call));
32
33 /* Non-secure callable (entry) function */
34 int func1(int x) __attribute__((cmse_nonsecure_entry)) {
35     return x+3;
36 }
37
38 /* Non-secure callable (entry) function, calling a non-secure callback function */
39 int func2(funcptr callback, int x) __attribute__((cmse_nonsecure_entry)) {
40     funcptr_NS callback_NS; // non-secure callback function pointer
41     int y;
42
43     /* return function pointer with cleared LSB */
44     callback_NS = (funcptr_NS)cmse_nsfptr_create(callback);
45
46     y = callback_NS(x+1);
47
48     return (y+2);
49 }
```

# Compile Secure Code

- Right click on the secure project
- Make sure it is set as active project
- Project -> Build CM33\_s

## Build Output

```
compiling interface.c...
assembling startup_ARMCM33.s...
compiling system_ARMCM33.c...
linking...
Program Size: Code=924 RO-data=2016 RW-data=4 ZI-data=1028
".\Objects\CM33_s.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:01
```



# Non-Secure Code

```
26 #include "..\CM33_s\interface.h"           // Interface API
27
28 extern volatile int val1, val2;
29 volatile int val1, val2;
30
31 /* Non-secure function */
32 int func3 (int x);
33
34 int func3 (int x) {
35     return (x+4);
36 }
37
38 /* Non-secure main() */
39 int main(void) {
40
41     /* Call non-secure callable function func1 */
42     val1 = func1 (1);
43
44     /* Call non-secure callable function func2
45        with callback to non-secure function func3 */
46     val2 = func2 (func3, 2);
47
48     while (1);
49 }
```

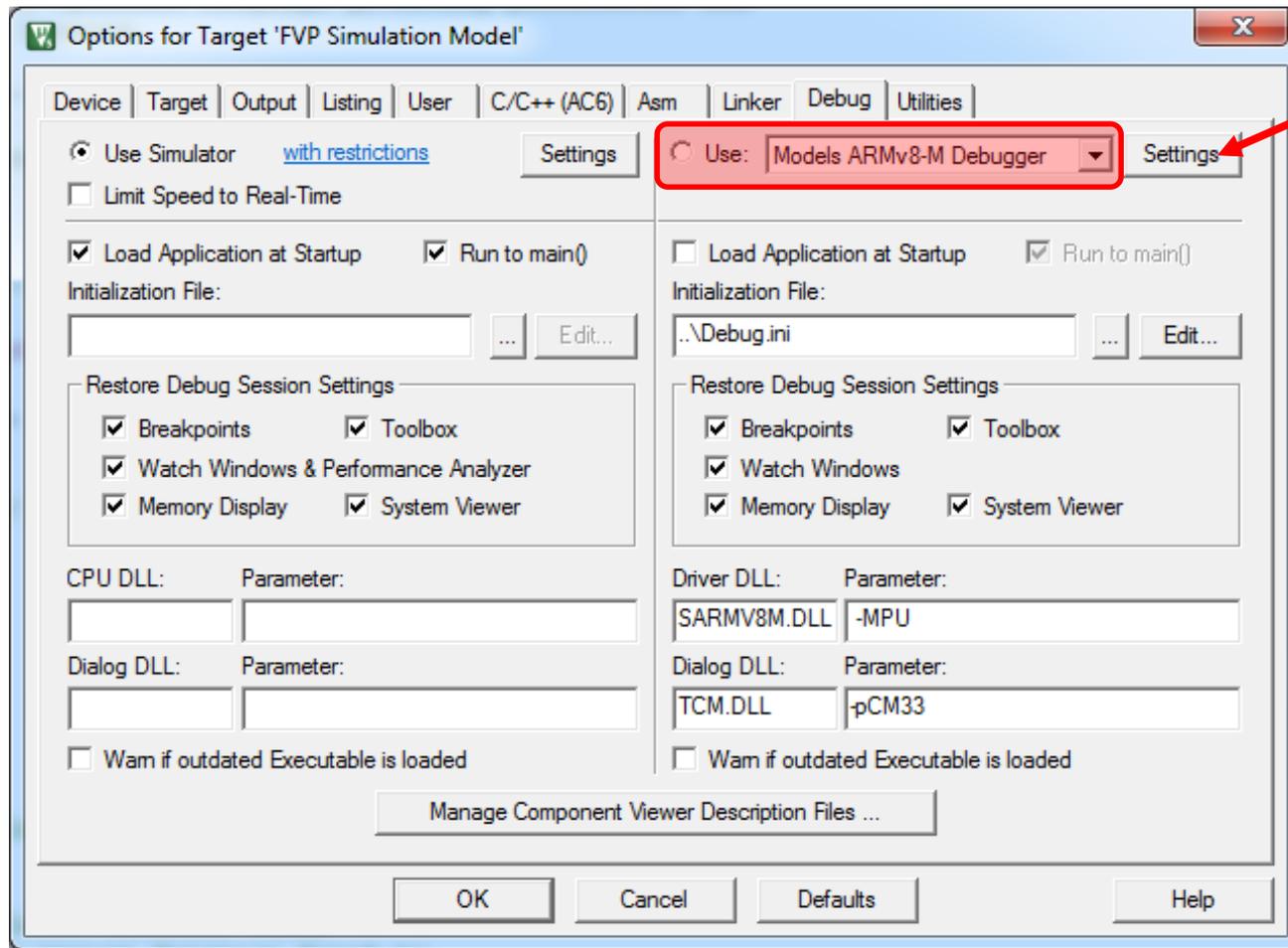
# Compile Unsecure Code

- Right click on the non-secure project
- Make sure it is the active project
- Project -> Build CM33\_ns
- (alternatively, Project -> Batch Build)

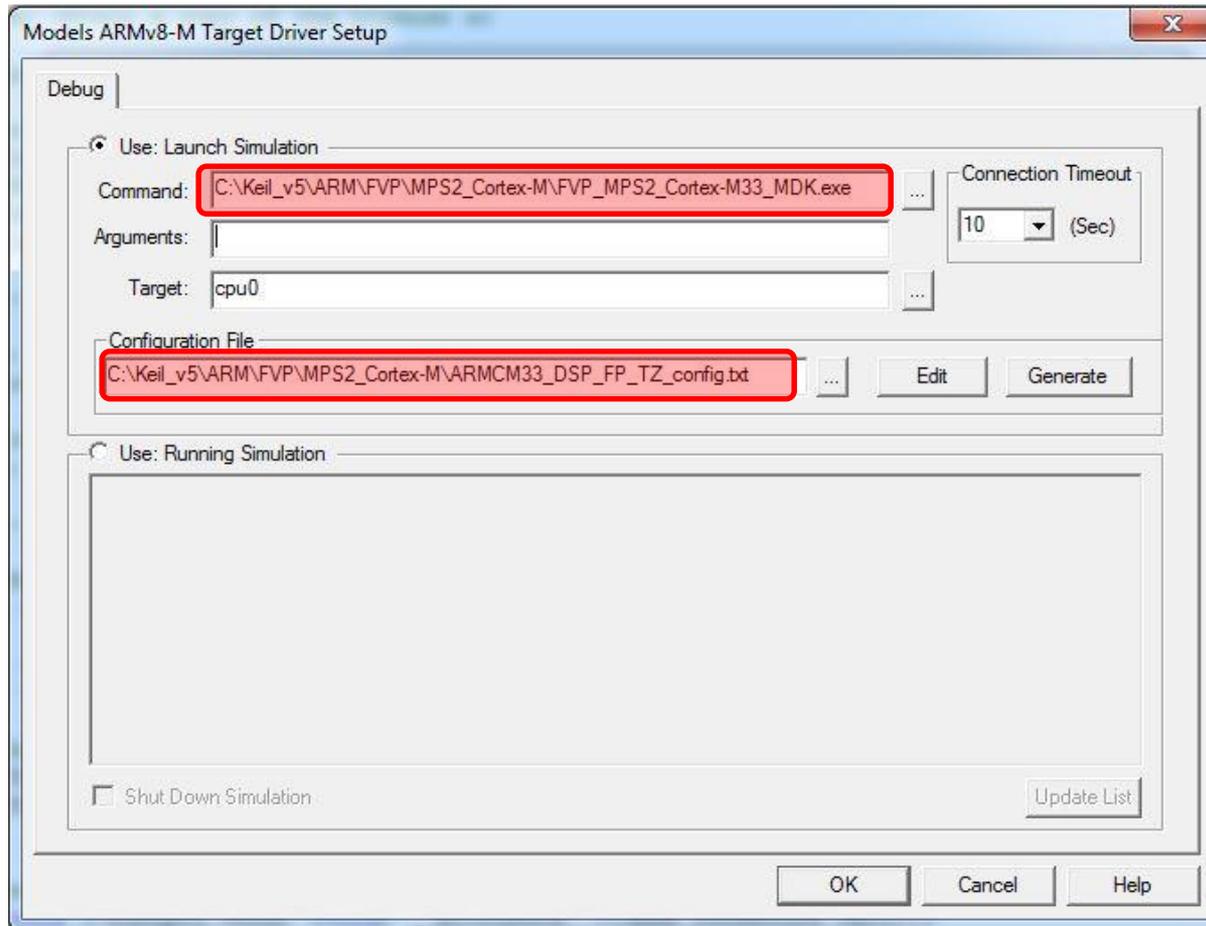
```
Build Output
compiling system_ARMCM33.c...
linking...
Program Size: Code=408 RO-data=2016 RW-data=4 ZI-data=1036
".\Objects\CM33_ns.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:01

Batch-Build summary: 2 succeeded, 0 failed, 0 skipped - Time Elapsed: 00:00:02
```

# Debugging the Application



# Debugging the Application



Register	Value
Core	
R0	0xDFDFDFCF
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0xDFDFDFCC
R14 (LR)	0xFFFFFFFF
R15 (PC)	0xCFDFDFDE
xPSR	0x01000000
Banked	
<b>Secure</b>	
MSP	0xDFDFDFCC
PSP	0x00000000
MSPLIM	0x00000000
PSPLIM	0x00000000
BASEPRI	0x00
PRIMASK	0
FAULTMASK	0
CONTROL	0x00
Non-Secure	
Internal	
Mode	Secure Thread
Privilege	Privileged
Stack	MSP
States	0
Sec	0.00000000
FPU	

```

0xCFDFDFDE 0000 MOVS r0,r0
0xCFDFDFE0 0000 MOVS r0,r0
0xCFDFDFE2 0000 MOVS r0,r0
0xCFDFDFE4 0000 MOVS r0,r0
0xCFDFDFE6 0000 MOVS r0,r0
0xCFDFDFE8 0000 MOVS r0,r0

```

Snipping Tool

Drag the cursor around the area you want to capture.

```

Abstract.txt interface.c secure.c system_saml11e16a.c secure.h main_s.c
1 /*
2  * Copyright (c) 2013-2016 ARM Limited. All rights reserved.
3  *
4  * SPDX-License-Identifier: Apache-2.0
5  *
6  * Licensed under the Apache License, Version 2.0 (the License); you may
7  * not use this file except in compliance with the License.
8  * You may obtain a copy of the License at
9  *
10 * www.apache.org/licenses/LICENSE-2.0
11 *
12 * Unless required by applicable law or agreed to in writing, software
13 * distributed under the License is distributed on an AS IS BASIS, WITHOUT
14 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 * See the License for the specific language governing permissions and
16 * limitations under the License.
17 *
18 * -----
19 *
20 * $Date:      15. October 2016
21 * $Revision:  1.1.0
22 *
23 * Project:    TrustZone for ARMv8-M
24 * Title:      Code template for secure main function
25 *
26 * -----*/
27
28 /* Use CMSE intrinsics */
29 #include <arm_cmse.h>
30
31 #include <stdio.h>
32 #include "sam.h"          /* Device header */
33
34 /* TO START UP: Check address of secure application */

```

```

Command
LOAD "..\..\nsApp\Objects\nsApp.axf" incremental
RESET

g, main

```

Name	Location/Value	Type

