

# RAM\_Run\_Function\_1 for KIT\_AURIX\_TC297\_TFT

Function running from RAM

AURIX™ TC2xx Microcontroller Training  
V1.0.2



[Please read the Important Notice and Warnings at the end of this document](#)

# Scope of work

---

## **A function is stored and executed from SRAM.**

This example implements twice the same function which toggles an LED with a wait loop. One function is implemented to be executed from SRAM and the other one from Flash memory.

The SRAM function is toggling LED1 (P13.0), while the flash function is toggling LED2 (P13.1).

# Introduction

---

- › The Local Memory Unit (LMU) SRAM can be used for **code execution**, data storage or overlay memory
- › The LMU can be accessed via cached (segment  $9_H$ ) or via non-cached (segment  $B_H$ ) memory addresses
- › If a code is programmed to be executed from SRAM memory, it is copied from Flash to SRAM by the Start-up Software (SSW) code



# Implementation

## SRAM code section creation

The linker file “*Lcf\_Tasking\_Tricore\_Tc.lsf*” is updated by adding a memory section (called ***code\_lmuram\_nc***) for code execution from LMURAM memory.

The memory section should be assigned to the **non-cached** memory addresses (segment B<sub>H</sub>) to avoid any data inconsistency.

```
group code_lmuram_nc (ordered, attributes=rwx, copy, run_addr=mem:lmuram/not_cached)
{
    ...select "(.text.not_cached_lmuram*)";
    ...select "(.text.lmuram_not_cached*)";
}
```

## Locating function code in a specific memory section

The ***pragma*** compiler keyword with the attribute ***section code*** “<***section identifier***>” is used to specify the memory section from which the implemented function code will be fetched and executed.

The ***section code restore*** attribute is used after the function implementation to ensure that next implemented functions will be located in the default code memory section (Flash memory).

# Implementation

---

## LED Toggling

Two functions are implemented, ***toggleLedSram()*** and ***toggleLedFlash()***, to toggle two LEDs from different memory regions.

Using the previously mentioned ***pragma*** compiler keyword, the ***toggleLedSram()*** can be executed from LMURAM memory not-cached addresses segment.

Both functions are implemented as following:

- Switch On the LED by calling ***IfxPort\_setPinLow()***
- Wait for a one second delay
- Switch Off the LED by calling ***IfxPort\_setPinHigh()***
- Wait for a one second delay

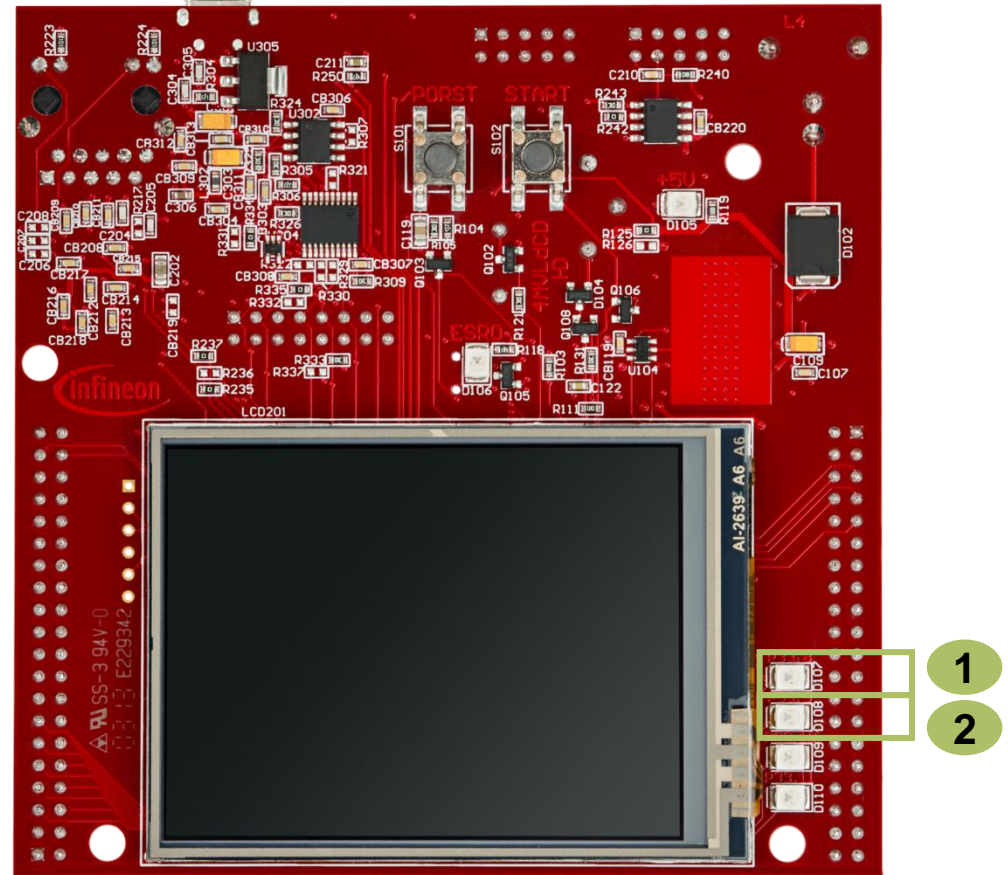
The above Port functions can be found in the iLLD header ***IfxPort.h***.

**Note:** The LEDs on the used board are low-level active.

# Run and Test

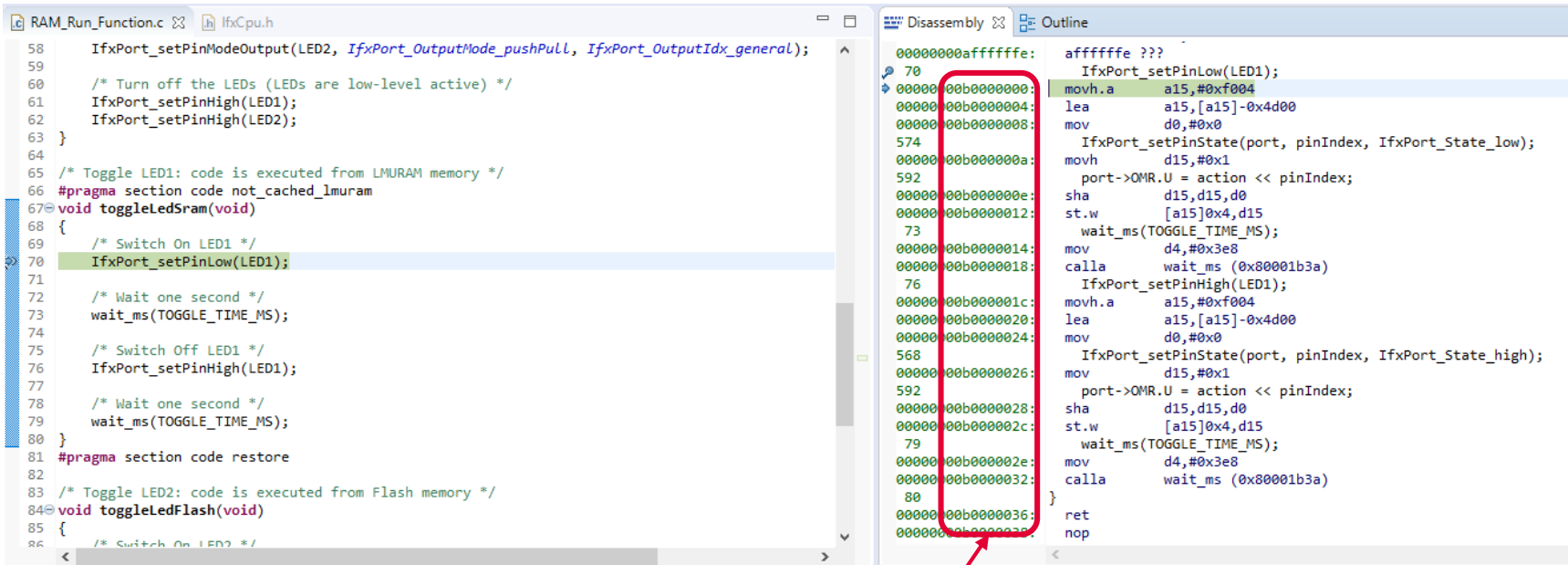
After code compilation and flashing the device:

- › Check that LED1 (D107) and LED2 (D108) are toggling



# Run and Test

Additionally, the execution from RAM can be checked by adding a breakpoint inside the ***toggleLedSram()*** function and verify in the disassembly window of the debugger that the CPU is executing it from LMURAM (Addresses segment B<sub>H</sub>).



```

RAM_Run_Function.c | IfxCpu.h
58   IfxPort_setPinModeOutput(LED2, IfxPort_OutputMode_pushPull, IfxPort_OutputIdx_general);
59
60   /* Turn off the LEDs (LEDs are low-level active) */
61   IfxPort_setPinHigh(LED1);
62   IfxPort_setPinHigh(LED2);
63 }
64
65 /* Toggle LED1: code is executed from LMURAM memory */
66 #pragma section code not_cached_lmuram
67 void toggleLedSram(void)
68 {
69     /* Switch On LED1 */
70     IfxPort_setPinLow(LED1);
71
72     /* Wait one second */
73     wait_ms(TOGGLE_TIME_MS);
74
75     /* Switch Off LED1 */
76     IfxPort_setPinHigh(LED1);
77
78     /* Wait one second */
79     wait_ms(TOGGLE_TIME_MS);
80 }
81 #pragma section code restore
82
83 /* Toggle LED2: code is executed from Flash memory */
84 void toggleLedFlash(void)
85 {
86     /* Switch On LED2 */

```

```

Disassembly | Outline
00000000affffffe:  afffffff ???
70   IfxPort_setPinLow(LED1);
71   movh.a   a15,#0xf004
0000000b00000000:  lea      a15,[a15]-0x4d00
0000000b00000004:  mov      d0,#0x0
0000000b00000008:  IfxPort_setPinState(port, pinIndex, IfxPort_State_low);
574  movh     d15,#0x1
0000000b0000000a:  port->OMR.U = action << pinIndex;
592  sha     d15,d15,d0
0000000b0000000e:  st.w    [a15]0x4,d15
0000000b00000012:  wait_ms(TOGGLE_TIME_MS);
73   mov     d4,#0x3e8
0000000b00000014:  calla   wait_ms (0x80001b3a)
0000000b00000018:  IfxPort_setPinHigh(LED1);
76   movh.a   a15,#0xf004
0000000b0000001c:  lea     a15,[a15]-0x4d00
0000000b00000020:  mov     d0,#0x0
0000000b00000024:  IfxPort_setPinState(port, pinIndex, IfxPort_State_high);
568  mov     d15,#0x1
0000000b00000026:  port->OMR.U = action << pinIndex;
592  sha     d15,d15,d0
0000000b00000028:  st.w    [a15]0x4,d15
0000000b0000002c:  wait_ms(TOGGLE_TIME_MS);
79   mov     d4,#0x3e8
0000000b0000002e:  calla   wait_ms (0x80001b3a)
80   }
0000000b00000036:  ret
0000000b00000038:  nop

```

- › Addresses from where the ***toggleLedSram()*** function is executed



# References



- › AURIX™ Development Studio is available online:
- › <https://www.infineon.com/aurixdevelopmentstudio>
- › Use the „*Import...*“ function to get access to more code examples.



- › More code examples can be found on the GIT repository:
- › [https://github.com/Infineon/AURIX\\_code\\_examples](https://github.com/Infineon/AURIX_code_examples)



- › For additional trainings, visit our webpage:
- › <https://www.infineon.com/aurix-expert-training>



- › For questions and support, use the AURIX™ Forum:
- › <https://www.infineonforums.com/forums/13-Aurix-Forum>

# Revision history

---

<b>Revision</b>	<b>Description of change</b>
V1.0.2	Added screenshot of .lsl file in Implementation slides
V1.0.1	Changed picture in Run and Test slide
V1.0.0	Initial version

## Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2021-06**

**Published by**

**Infineon Technologies AG  
81726 Munich, Germany**

**© 2021 Infineon Technologies AG.  
All Rights Reserved.**

**Do you have a question about this document?**

**Email: [erratum@infineon.com](mailto:erratum@infineon.com)**

**Document reference**

**RAM\_Run\_Function\_1\_KIT\_TC297\_TFT**

## IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”).

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office ([www.infineon.com](http://www.infineon.com)).

## WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.