

# RAM\_Run\_Function\_1 for KIT\_AURIX\_TC375\_LK

Function running from RAM

AURIX™ TC3xx Microcontroller Training  
V1.0.0



[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 (P00.5), while the flash function is toggling LED2 (P00.6).

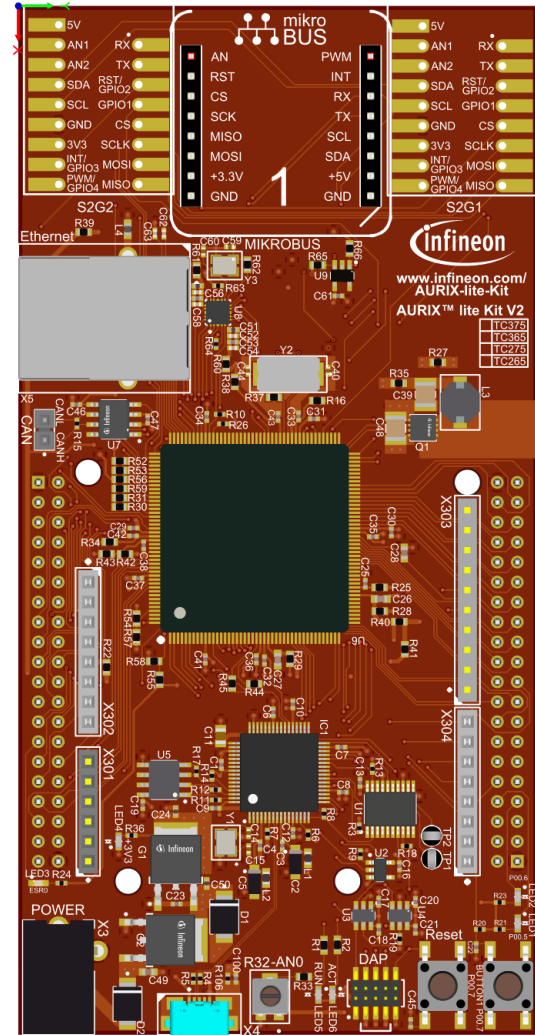
# Introduction

---

- › The TriCore™ architecture supports closely coupled program and data SRAM memory known as Program Scratch Pad RAM (PSPR) and Data Scratch Pad RAM (DSPR)
- › Program & Data Scratch-Pad RAM (PSPR/DSPR): Allows the CPU to access code/data faster compared to the other RAMs and Flashes
- › 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

# Hardware setup

This code example has been developed for the board KIT\_A2G\_TC375\_LITE.



# Implementation

---

## SRAM code section

The linker file “*Lcf\_Tasking\_Tricore\_Tc.lsf*” contains the different memory sections used by the AURIX™ microcontroller. In this example, PSPR0 (*cpu0\_psram*) is used.

**Note:** The start-up procedure can overwrite up to 1 kByte at the beginning of CPU0 PSPR. If the application needs to save program code which must be preserved through a reset, this area (the first 1 kByte) should not be used.

## 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 PSPR memory.

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

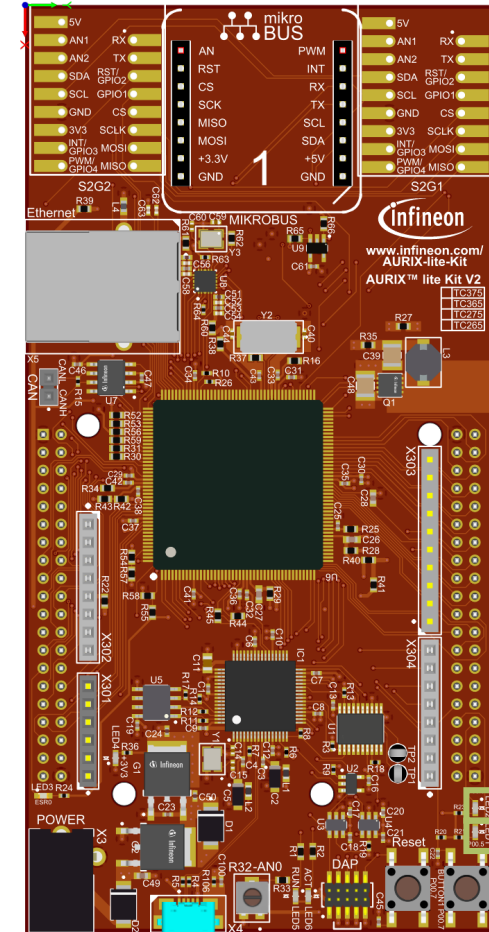
The above Port functions can be found in the iLLD header ***IfxPort.h***.

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

# Run and Test

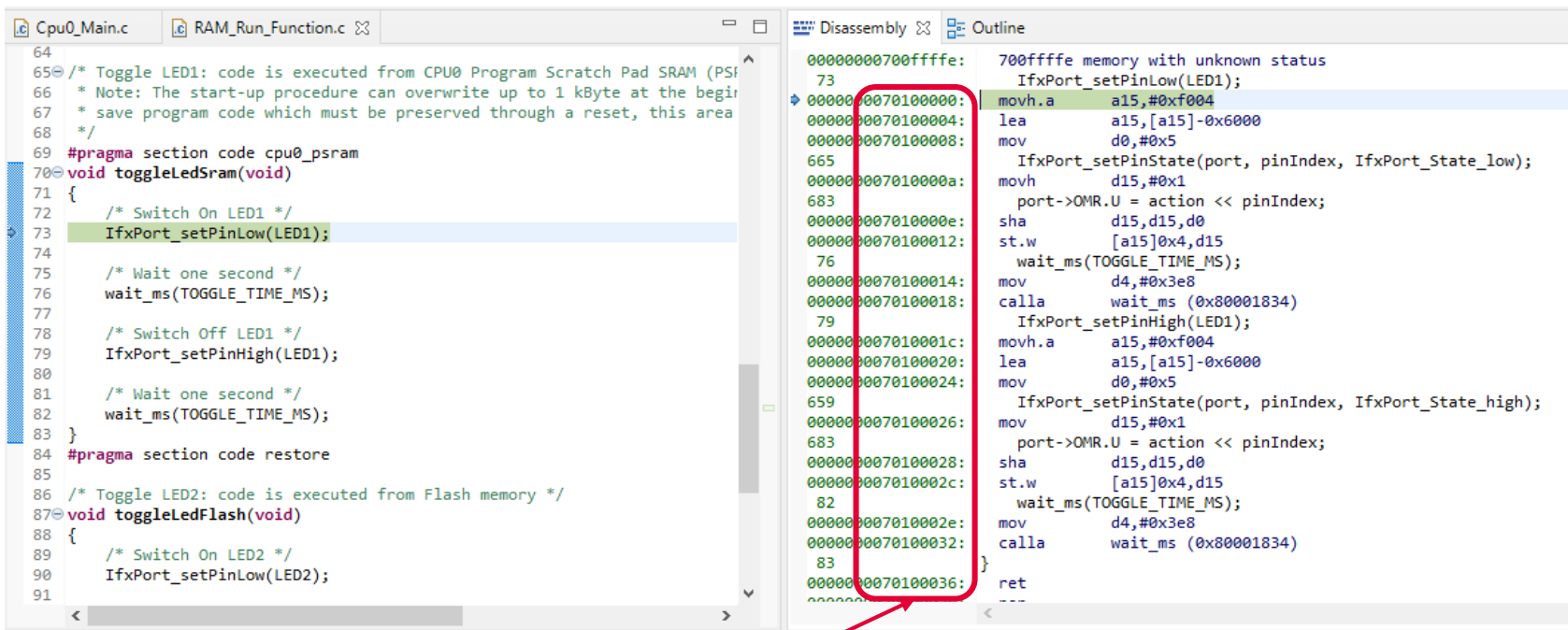
After code compilation and flashing the device, perform the following:

- › Check that LED1 (1) and LED2 (2) 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 PSPR (Addresses segment 7<sub>H</sub>).



```

Cpu0_Main.c | RAM_Run_Function.c
64
65 /* Toggle LED1: code is executed from CPU0 Program Scratch Pad SRAM (PSPR)
66 * Note: The start-up procedure can overwrite up to 1 kByte at the begin
67 * save program code which must be preserved through a reset, this area
68 */
69 #pragma section code cpu0_psr
70 void toggleLedSram(void)
71 {
72     /* Switch On LED1 */
73     IfxPort_setPinLow(LED1);
74
75     /* Wait one second */
76     wait_ms(TOGGLE_TIME_MS);
77
78     /* Switch Off LED1 */
79     IfxPort_setPinHigh(LED1);
80
81     /* Wait one second */
82     wait_ms(TOGGLE_TIME_MS);
83 }
84 #pragma section code restore
85
86 /* Toggle LED2: code is executed from Flash memory */
87 void toggleLedFlash(void)
88 {
89     /* Switch On LED2 */
90     IfxPort_setPinLow(LED2);
91
0000000700ffffe: 70ffffe memory with unknown status
73     IfxPort_setPinLow(LED1);
74     movh.a    a15,#0xf004
75     lea      a15,[a15]-0x6000
76     mov      d0,#0x5
77     IfxPort_setPinState(port, pinIndex, IfxPort_State_low);
78     movh     d15,#0x1
79     port->OMR.U = action << pinIndex;
80     sha     d15,d15,d0
81     st.w    [a15]0x4,d15
82     wait_ms(TOGGLE_TIME_MS);
83     mov     d4,#0x3e8
84     calla   wait_ms (0x80001834)
85     IfxPort_setPinHigh(LED1);
86     movh.a    a15,#0xf004
87     lea      a15,[a15]-0x6000
88     mov      d0,#0x5
89     IfxPort_setPinState(port, pinIndex, IfxPort_State_high);
90     mov     d15,#0x1
91     port->OMR.U = action << pinIndex;
92     sha     d15,d15,d0
93     st.w    [a15]0x4,d15
94     wait_ms(TOGGLE_TIME_MS);
95     mov     d4,#0x3e8
96     calla   wait_ms (0x80001834)
97 }
98     ret
99

```

- › 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>

## Trademarks

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

**Edition 2021-03**

**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\_TC375\_LK**

## 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.