Memory Analyzer

User guide

About this document

Scope and purpose

The Memory Analyzer is a graphical visualization tool that analyzes the memory consumption of the following Infineon microcontroller families:

- TLE987x
- TLE986x
- TLE985x
- TLE984x

A typical use case in embedded computing is debugging and optimizing the microcontroller program code. Mainly, detailed information about the memory utilization and linking of programs and data is very important. In most cases, the standard C-compiler and debugger do not support an easy-to-use graphical visualization of the size and location of embedded software artifacts. Especially when it comes to size and speed optimization or bug fixing of embedded programs, a graphical analyzing tool speeds up the programmer's work and becomes essential.

The Memory Analyzer supports searching for functions, objects, files, and symbols, which can be sorted by name, address location, size, and region. It analyzes the size of software modules and shows a graphical visualization of memory sections, it exports to *.pdf and *.csv files, and allows the build process integration via script language interface.

The Memory Analyzer analyzes the following linker files, ARM linker output format (AXF), an IAR linker file with extension ILINK and XLINK (OUT), and executable and linkable format (ELF).

Intended audience

This document is intended for microcontroller-embedded software developers.

Table of contents

1 Introduction

1.1 Tool overview

This tool consists of a graphical user interface (GUI) divided into four main windows tabs:

- Available Devices
- File
- Analysis
- Graphics

These tabs will be described in detail in the further sections of this document.

Moreover, the tools' main features are:

- Allow Infineon microcontroller selection
- Show memory regions of available devices
- Show header and section elements of linker file
- Check if given linker file fits into memory of selected Infineon microcontroller
- Show all functions and objects, and their memory information
- Sort linker file output by type (files, functions, objects and symbols) and category (name, type, start- and end- address, region)
- Search for a type, highlight and calculate its total size
- Export the file statistics to CSV format
- Show memory region distribution in a pie diagram
- Export information to CSV and PDF
- Drag and drop of linker files onto the tool

1.2 Supported linker file types

The tool supports the following linker file types:

- "*.axf" extension: ARM object file format
- "*.out" extension: IAR linker file extension ILINK and XLINK
- "*.elf" extension: executable and linkable format

1.3 How to get the tool

The Memory Analyzer is a tool available in the Infineon Developer Center (IDC). Open this link to access the Infineon Developer Center.

A detailed description on how to install and use the IDC is available here.

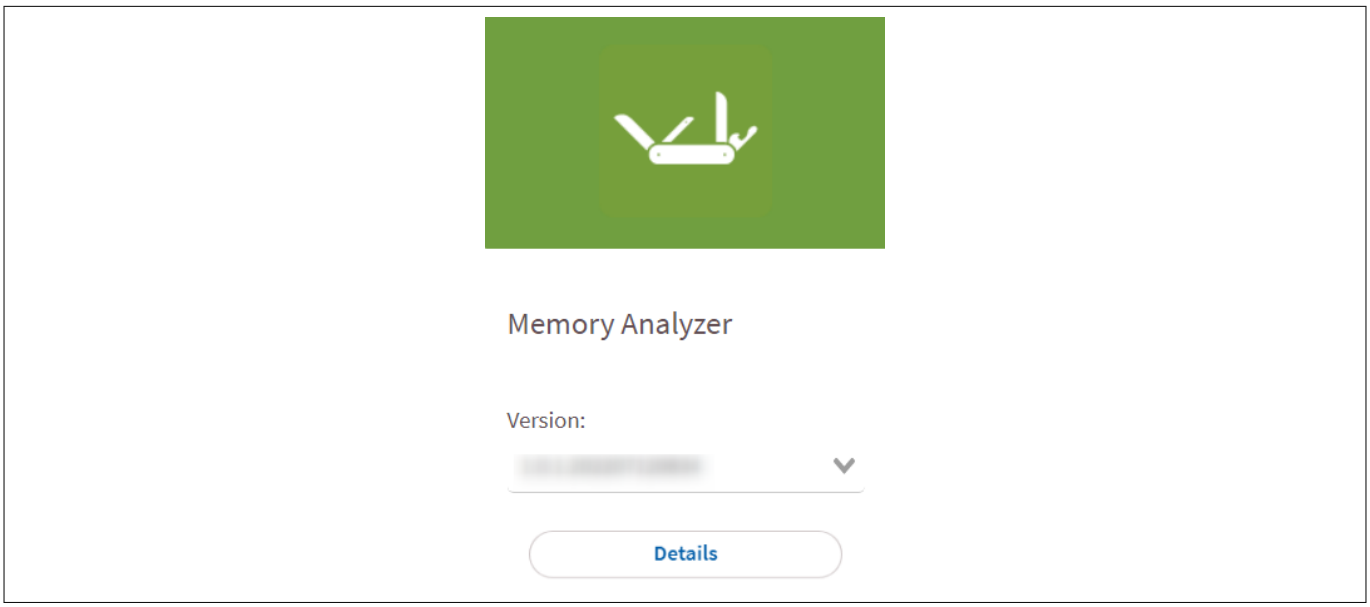


Figure 1 Memory Analyzer

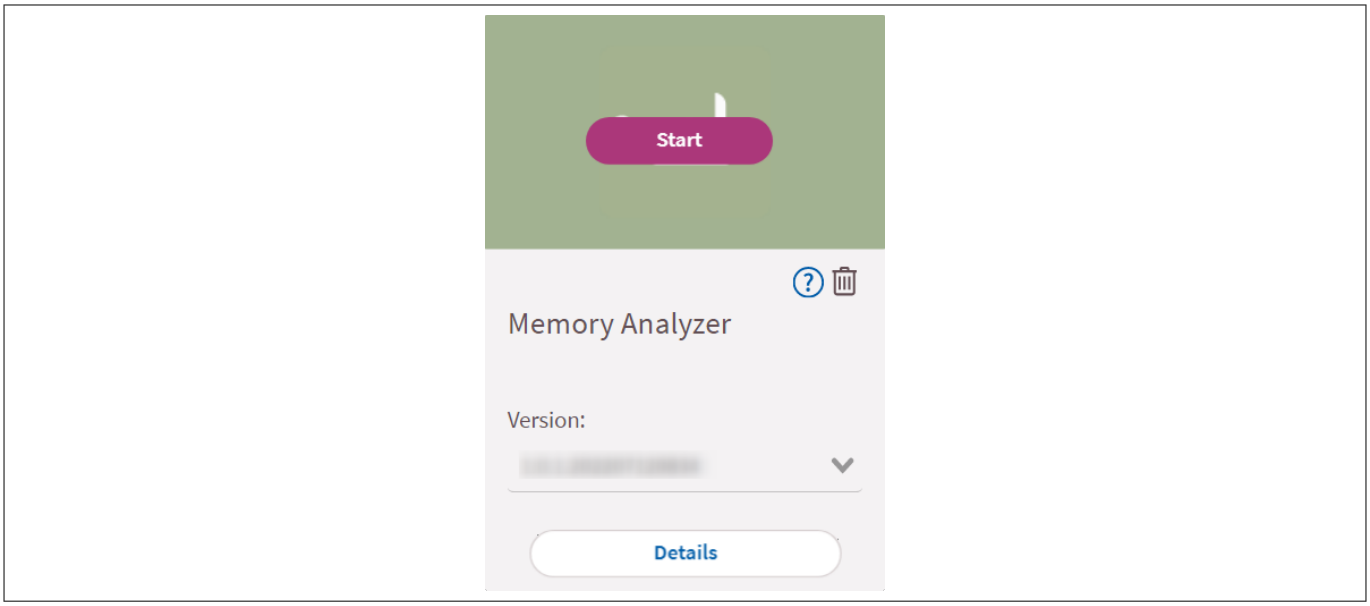


Figure 2 Start the Memory Analyzer

1.4 User guide

To open the user guide and find GUI explanations, general information, and recommendations, click on the question mark icon in the upper right corner of the tool.

Figure 3 Opening the user guide

2 GUI explanation

2.1 Available devices tab

The available devices tab shows the information about the Infineon microcontrollers, which can be selected from the drop-down list. The user can select a microcontroller according to the product family and device type.

The figure below shows the memory regions which are the physical on-chip available memories. This can be for example Data Flash RAM, Program-RAM, or any other memory region described in the datasheet of the microcontroller.

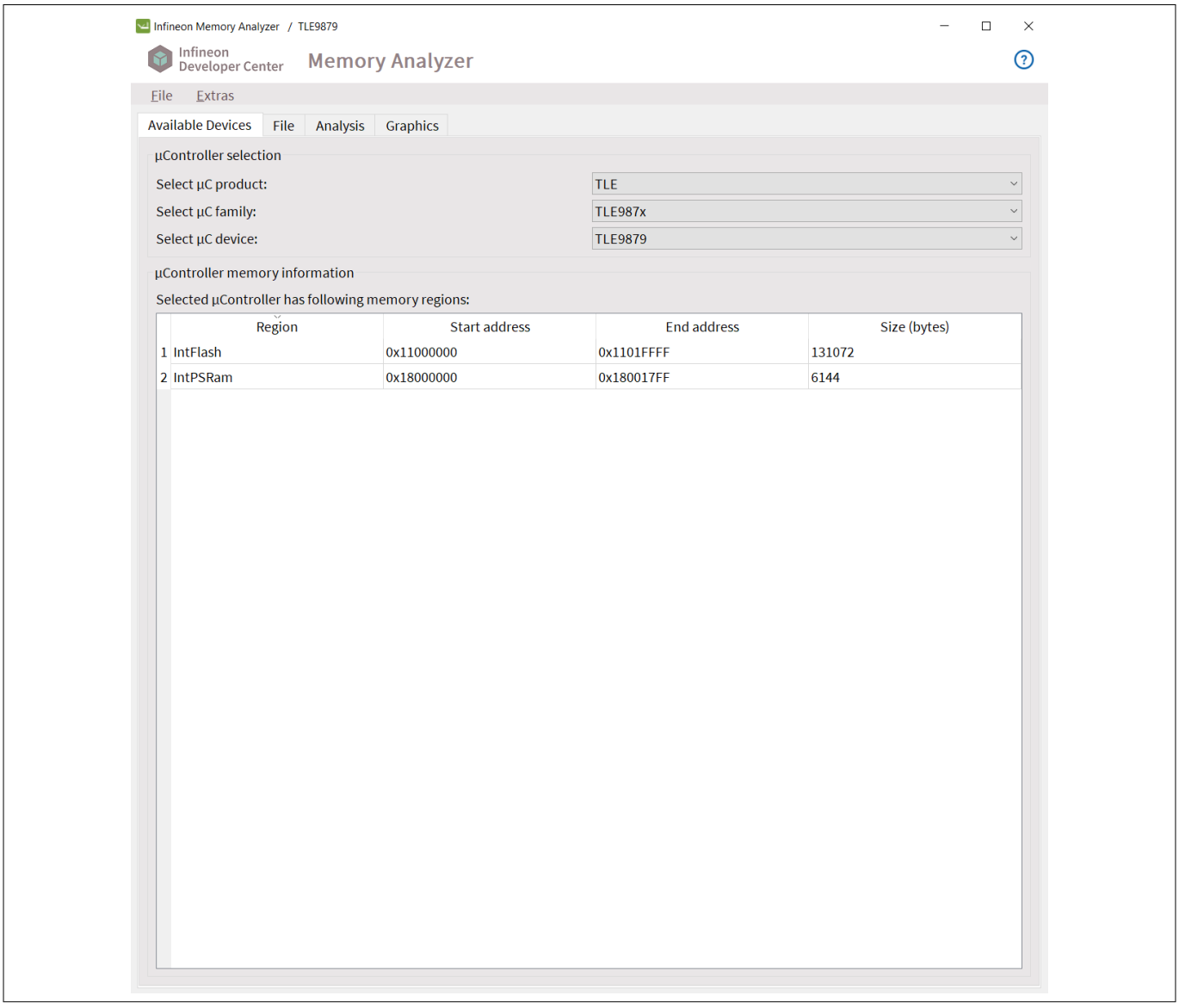


Figure 4 Available devices tab

2.2 File tab

The content of the **File** tab can be seen after loading a linker output file. It contains basic information about the file such as complete file path, headers and sections.

To analyze a linker file with supported extensions "*.axf", "*.out" and "*.elf" click on **Browse** to open the file.

After loading a linker file, the Memory Analyzer tool shows the corresponding headers and sections, the table can be sorted by clicking on the category: name, type, start address, end address, offset, size or flags.

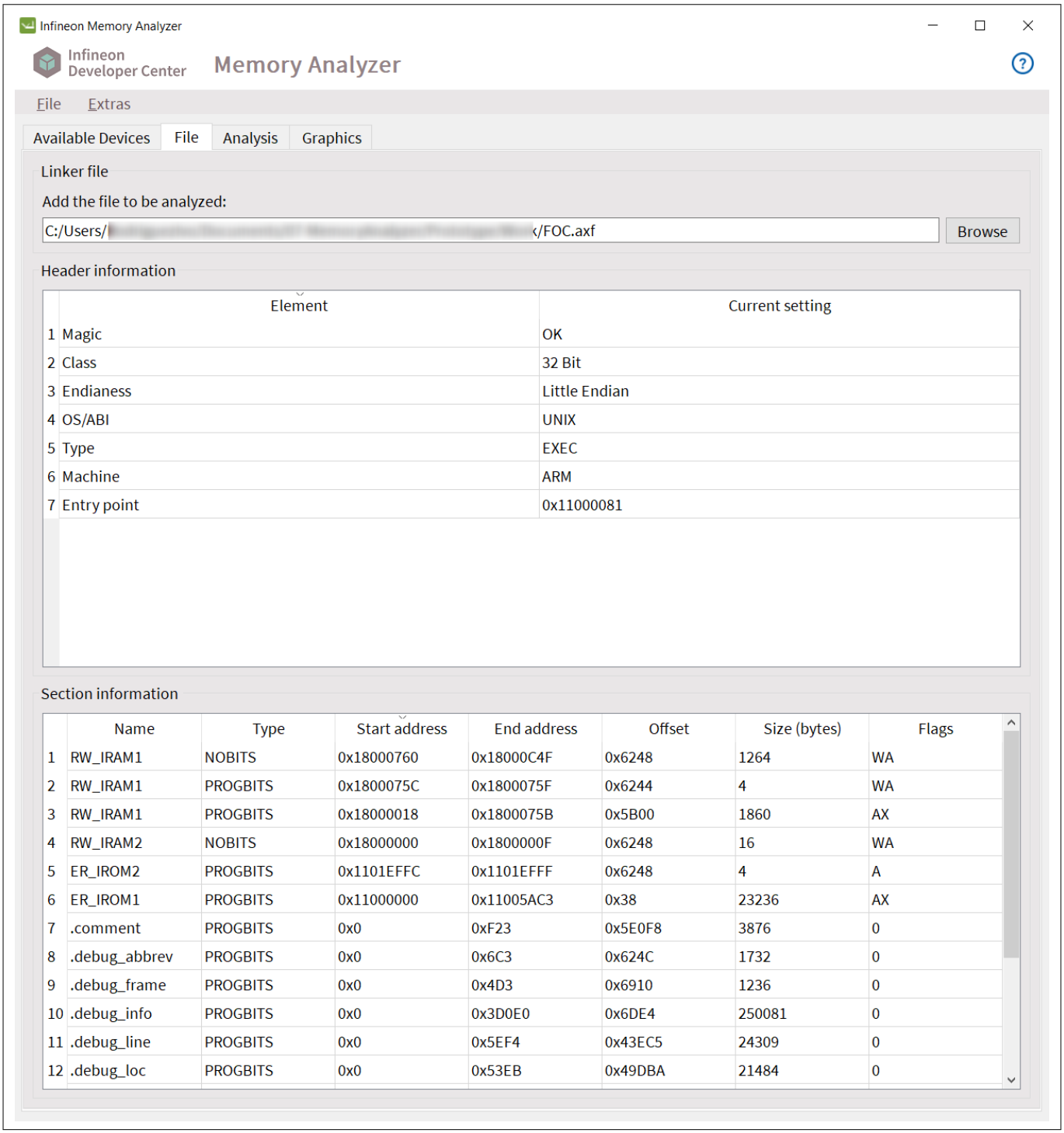


Figure 5 File tab

Header: this group box shows the header information of the linker file:

- Magic:
  identification number of an ELF-File
- Class:
  states if it is in 32- or 64-bit format
- Endianess:
  states if it is in little- or big-endian format
- OS/ABI:
  contains the name of the hosting OS. Keil compilers mostly write “UNIX” into this field, even if no OS is running on the microcontroller
- Type:
  contains the type of the binary, normally “EXEC” for a program but could also be a static or dynamic library
- Machine:
  specifies the instruction set of the controller, for example, ARM
- Entry point:
  specifies the start address of the code

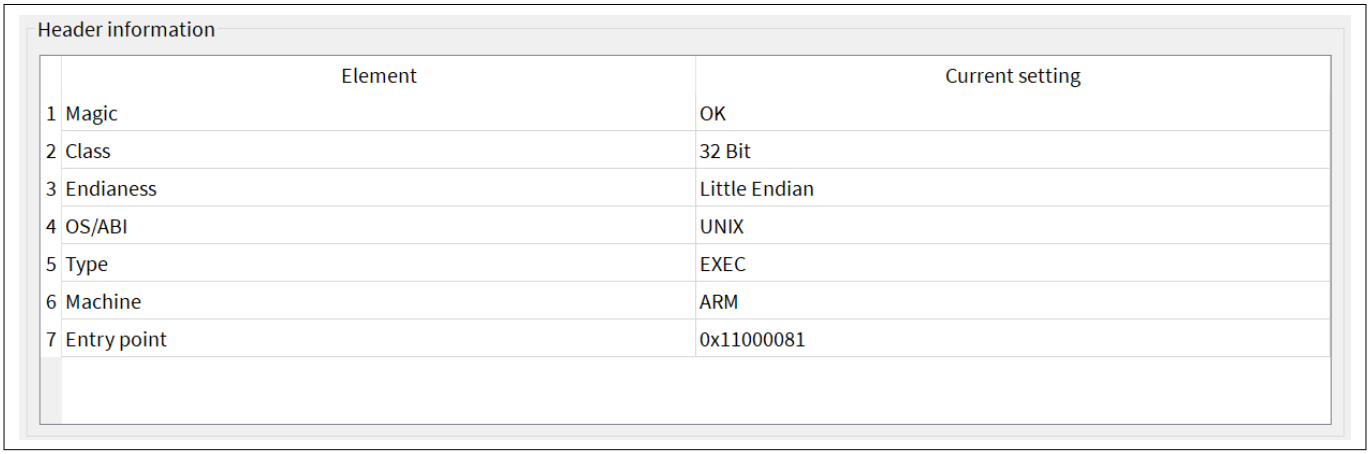


Figure 6 File tab – file header information

Section: this group shows the sections contained in the linker file:

- Name:
  contains the name of the section
- Type:
  tells the type of the section, for example:
  - PROGBITS bits of the program
  - SYMTAB symbol table; an array of ELF symbol structures
  - STRTAB string table; holds null-terminated strings
  - RELA relocation table
  - HASH hash table used by RTLD to speed symbol lookup
  - DYNAMIC dynamic tags used by RTLD, same as PT DYNAMIC
  - NOBITS zero-initialized data
- Address:
  designates the address of this section in memory
- Offset:
  is the offset within the linker file

- Size:
  is the size of the section
- Flags:
  contains several attributes of the section. Key to Flags: W (write), A (alloc), X (execute), M (merge), S (strings), l (large), I (info), L (link order), G (group), T (TLS), E (exclude), x (unknown) O (extra OS processing required), o (OS specific), p (processor specific)

Note: *Regarding sections, a whole row is shown in red font if the section is out of range, compared to the region of the selected microcontroller.*

Section information

	Name	Type	Start address	End address	Offset	Size (bytes)	Flags
1	ER_IROM1	PROGBITS	0x11000000	0x11005AC3	0x38	23236	AX
2	RW_IRAM1	PROGBITS	0x18000018	0x1800075B	0x5B00	1860	AX
3	RW_IRAM1	PROGBITS	0x1800075C	0x1800075F	0x6244	4	WA
4	RW_IRAM1	NOBITS	0x18000760	0x18000C4F	0x6248	1264	WA
5	RW_IRAM2	NOBITS	0x18000000	0x1800000F	0x6248	16	WA
6	ER_IROM2	PROGBITS	0x1101EFFC	0x1101EFFF	0x6248	4	A
7	.debug_abbrev	PROGBITS	0x0	0x6C3	0x624C	1732	0
8	.debug_frame	PROGBITS	0x0	0x4D3	0x6910	1236	0
9	.debug_info	PROGBITS	0x0	0x3D0E0	0x6DE4	250081	0
10	.debug_line	PROGBITS	0x0	0x5EF4	0x43EC5	24309	0
11	.debug_loc	PROGBITS	0x0	0x53EB	0x49DBA	21484	0
12	.debug_str	PROGBITS	0x0	0xAE28	0x4F1A6	44585	0

Figure 7 File tab – file sections information

2.3 Analysis tab

This tab contains information about the read files, functions, objects, and symbols. The table shows the content of all entries found in the linker file, the categories are:

- Name
- Type (FILE, FUNCTION, OBJECT, SYMBOL)
- Start address
- End address
- Size (bytes)
- Region (in memory)

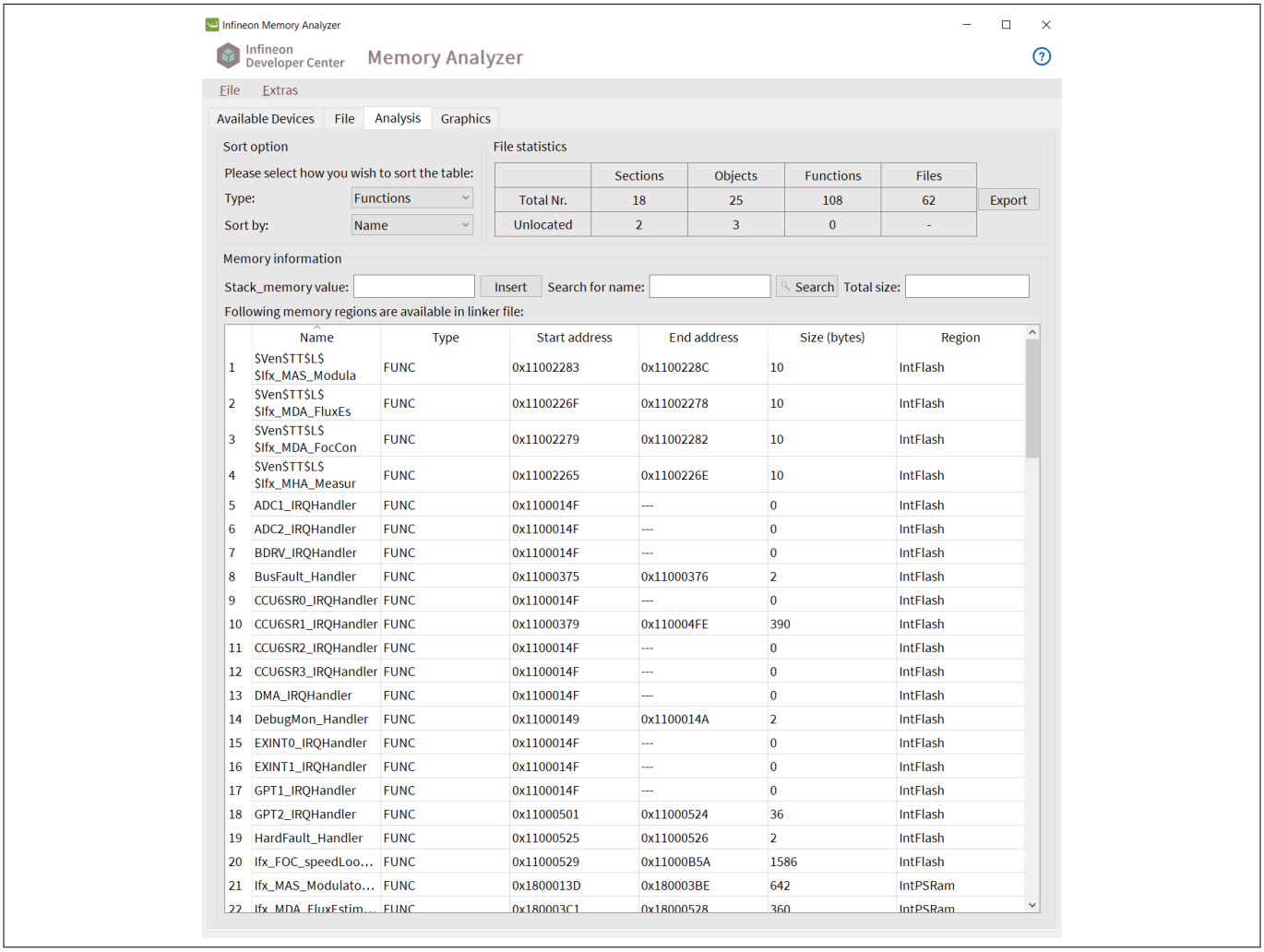


Figure 8 Analysis tab overview

2.3.1 Invalid item

If an object cannot be mapped to a memory region, its row is shown in red font and its region is **<INVALID>** , also the statistics report the number of unlocated objects in the file.

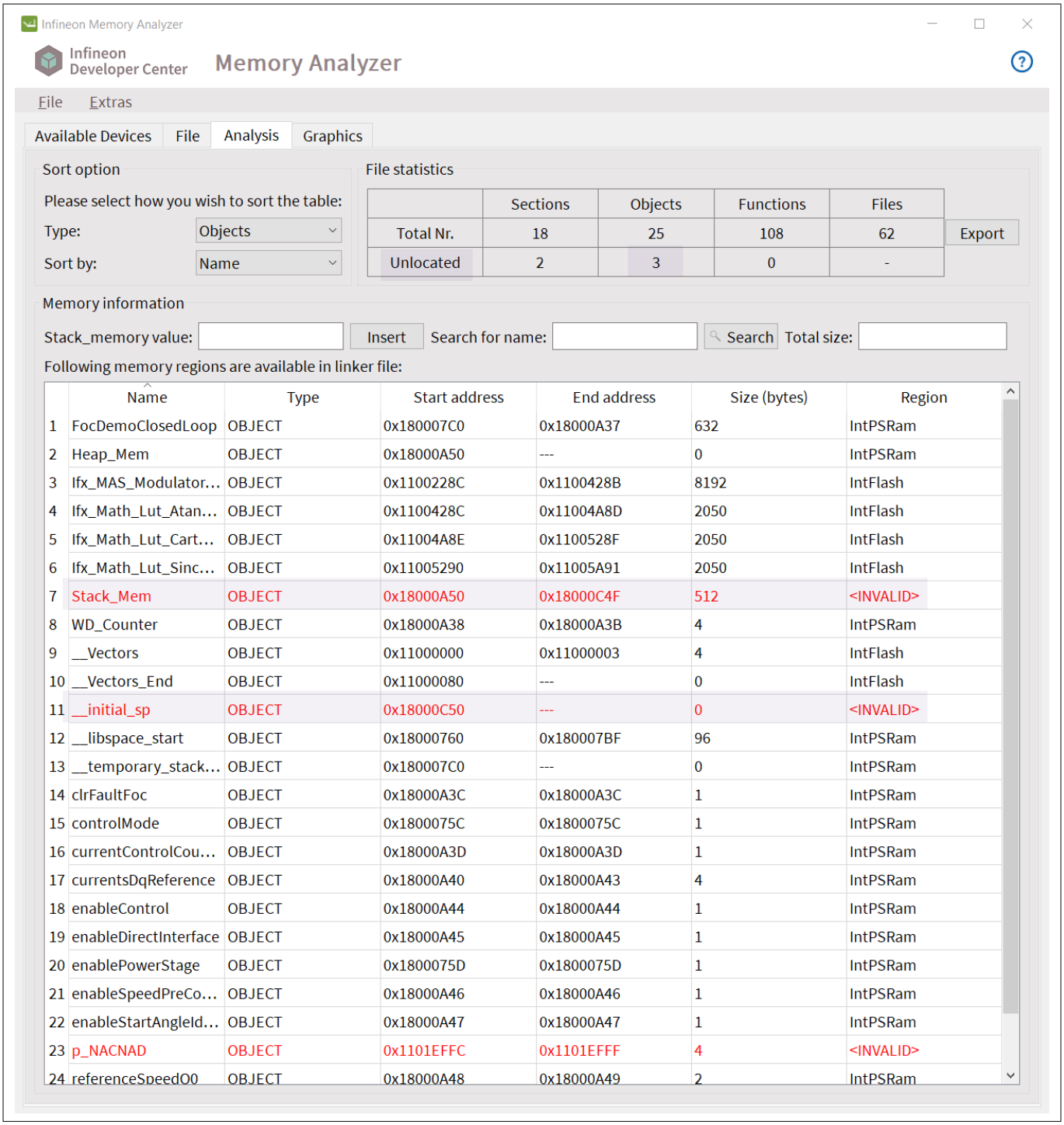


Figure 9 Invalid object found

2.3.2 Sort option

The table can be sorted by using the sorting options in the right corner.

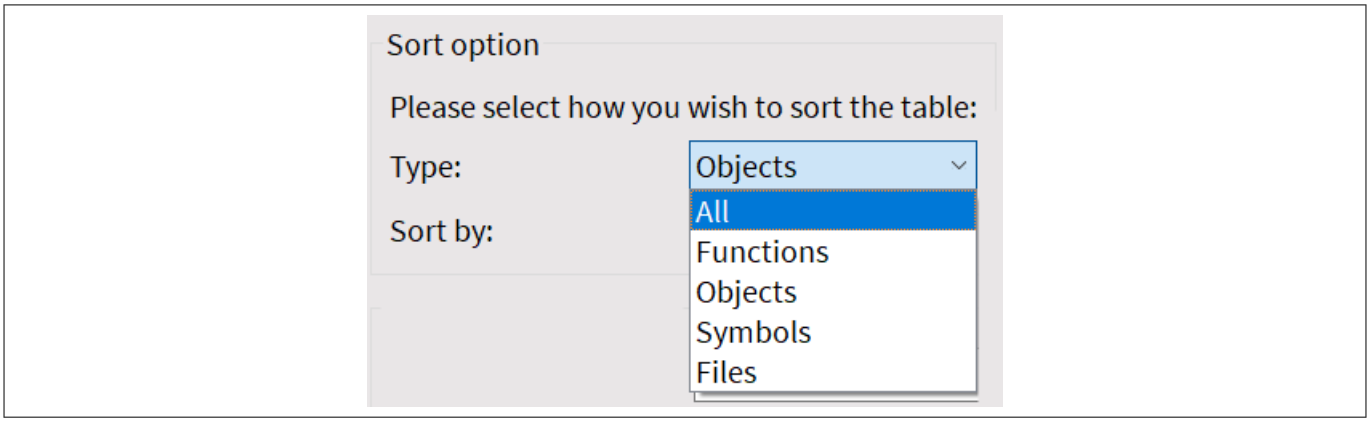


Figure 10 Sort options - type

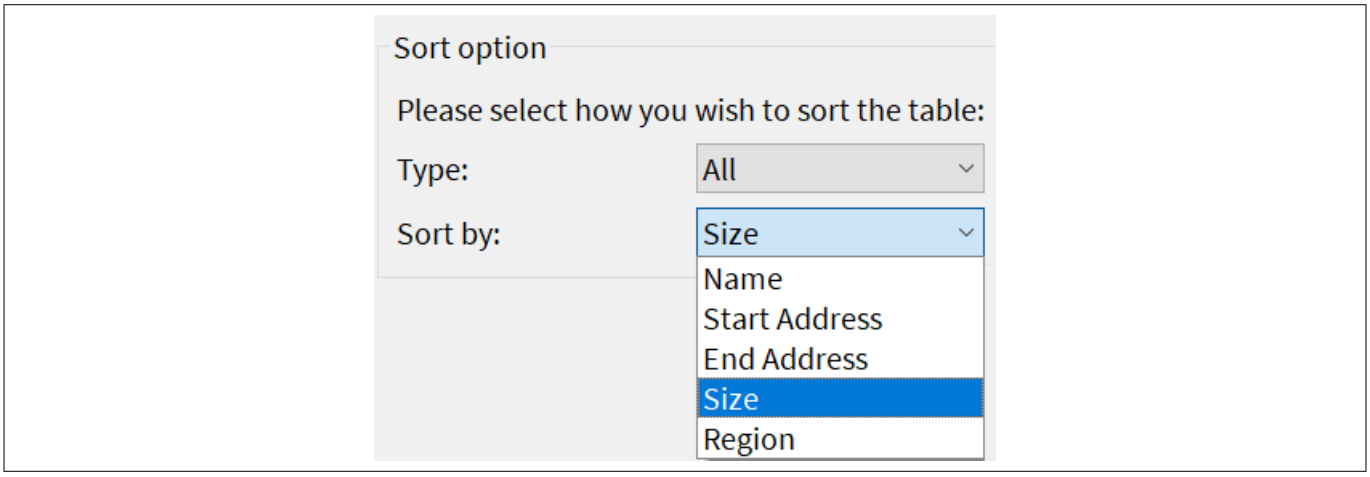


Figure 11 Sort options - sort by

2.3.3 Insert stack memory value in RAM

For simulation purposes, the stack memory value in bytes can be inserted. This value will be added to the table as a new object with the name Stack_Memory and the region IntPSRam.

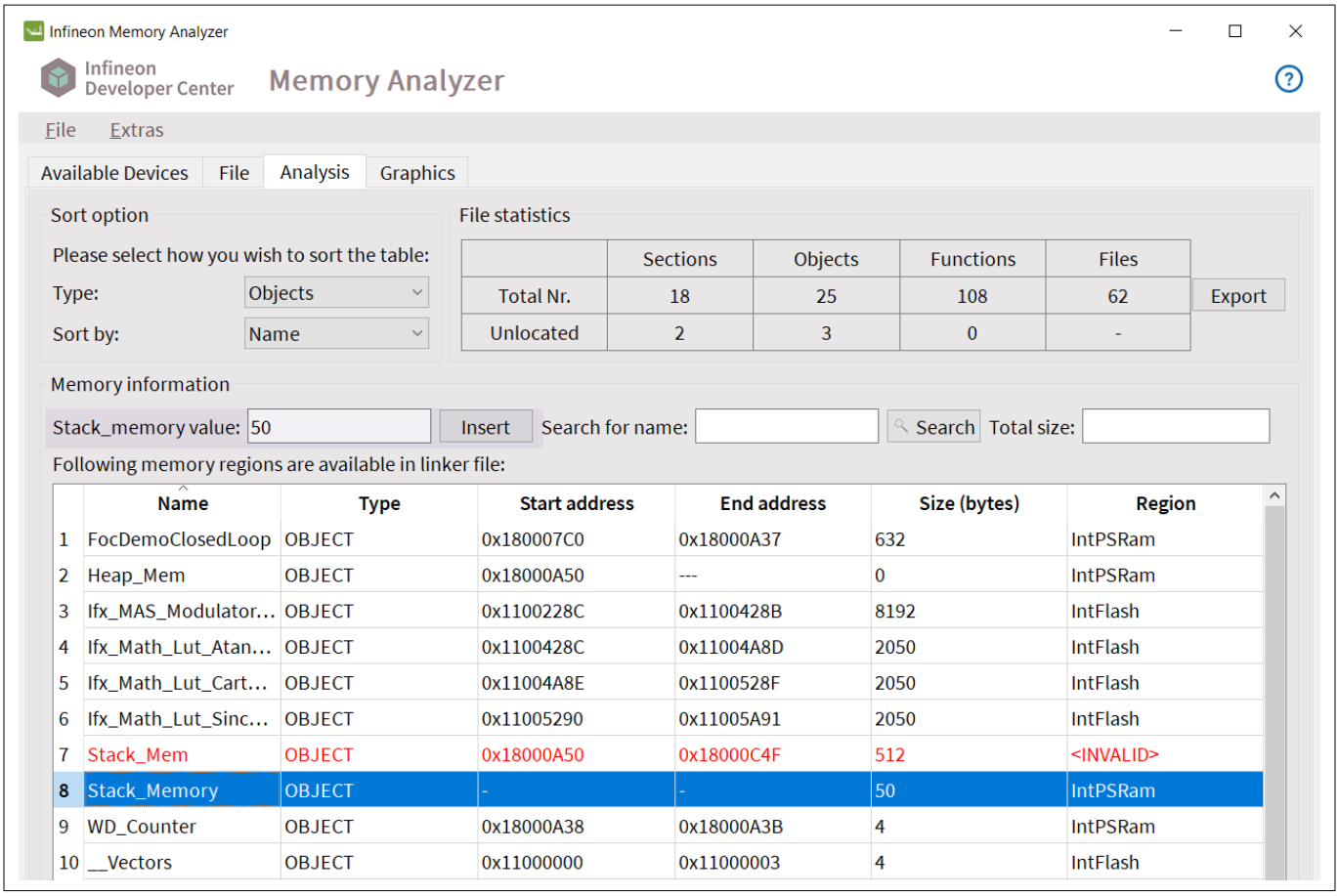


Figure 12 Insert stack memory value

2.3.4 Search function

Any item available in the table can be searched by inserting its name and clicking search. The item(s) will be highlighted and the sum of the sizes of the selected item(s) is shown in total size. An information pop-up window appears, it contains the number of entries found in the file.

Be aware of using the wild card (*) to search content at the beginning or end of the word.

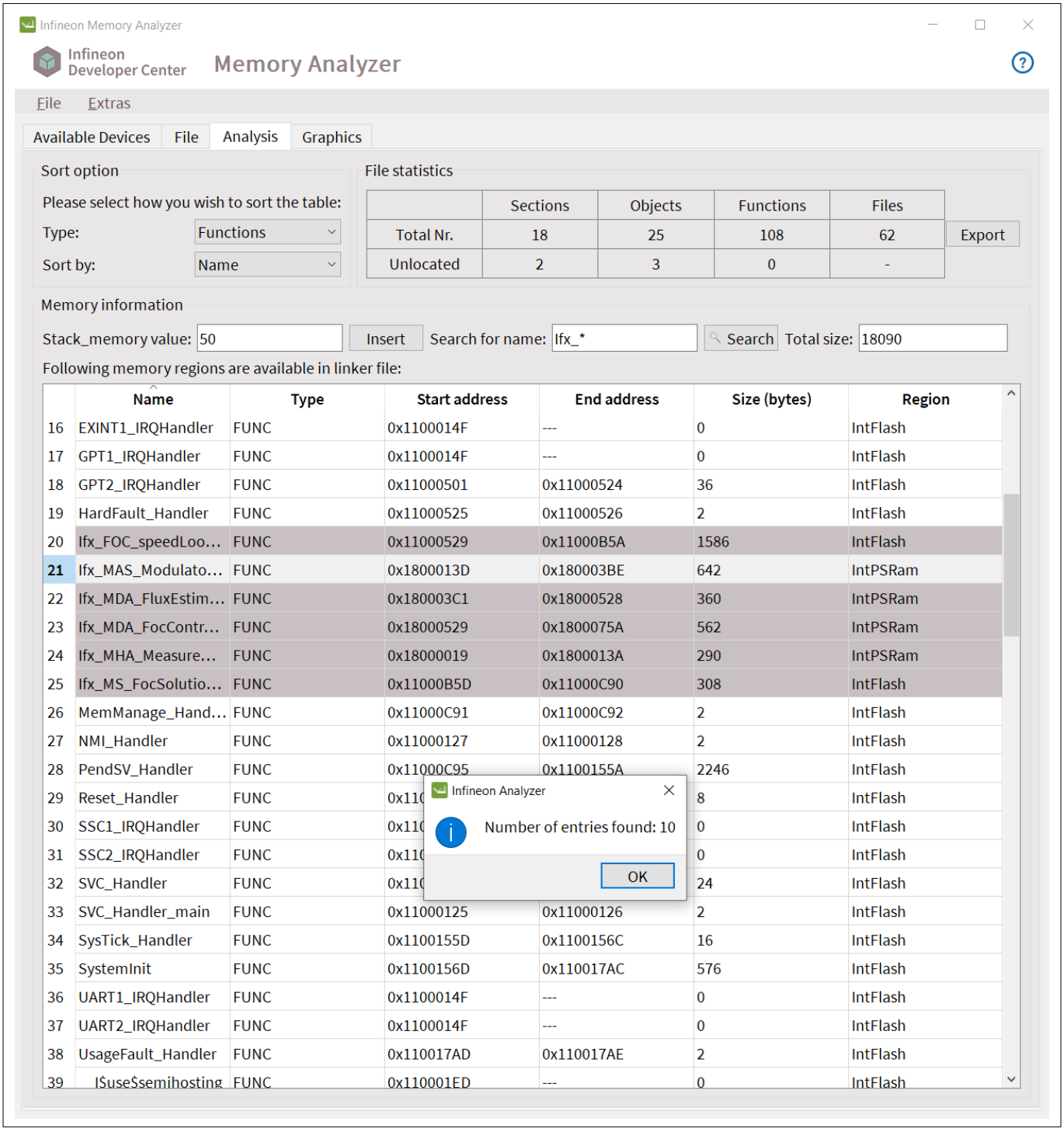


Figure 13 Search object function

2.3.5 Export statistics in CSV format

A general overview of the number of objects can be exported in CSV format. Click export and save the document.

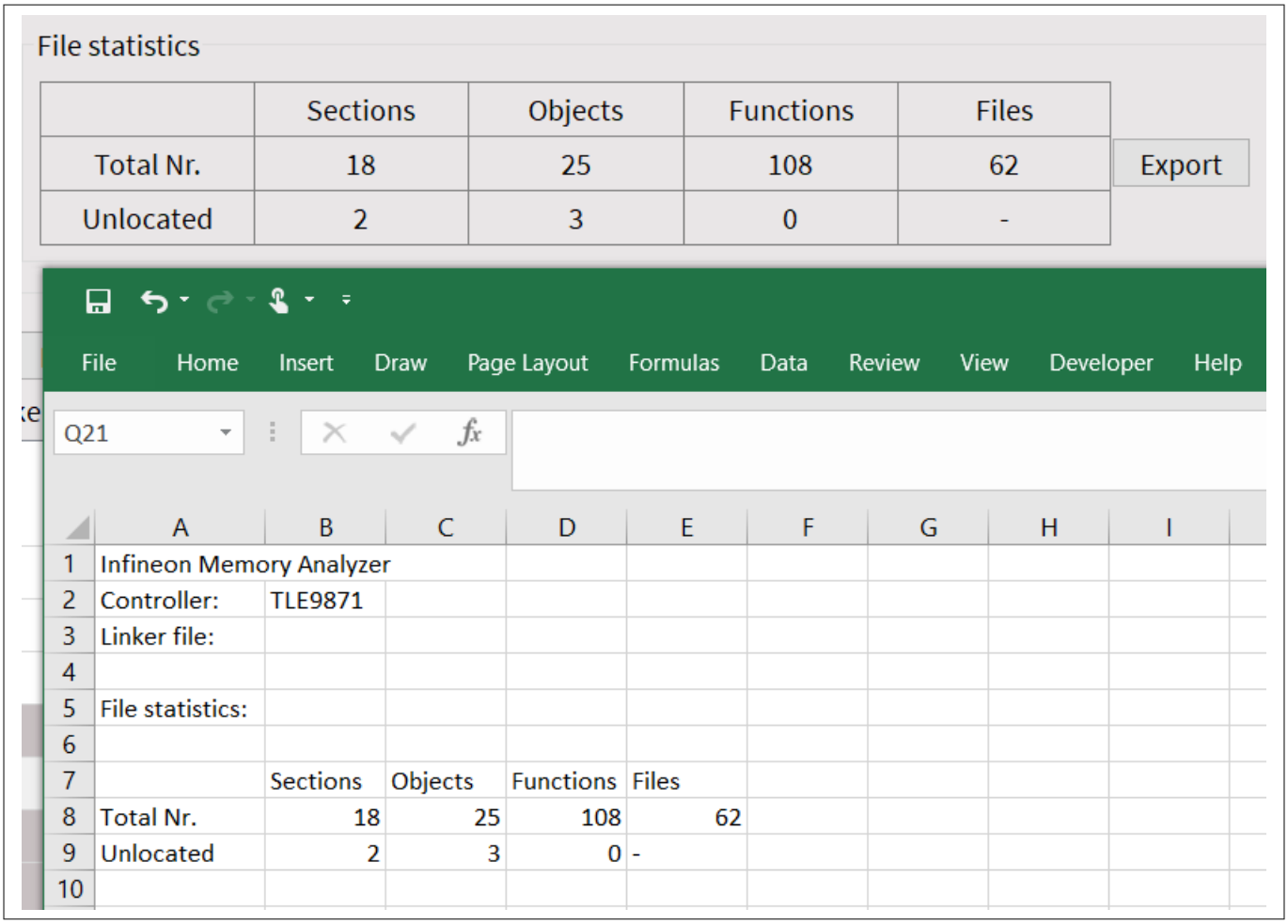


Figure 14 Export statistics in CSV format

2.4 Graphic tab

The content of the **Graphic** tab is available after loading the linker file. The information contained in this tab is similar to the **Analyze** tab, here it is restricted to displaying the size of functions and objects from a region in a graphical manner.

A maximum of four pie diagrams can be shown in parallel. These pies show the memory regions available in the microcontroller, they can be selected from the drop-down menu. Each colored slide corresponds to a function or object, the available memory is shown in green.

The total size of the memory region is also displayed above the chart.

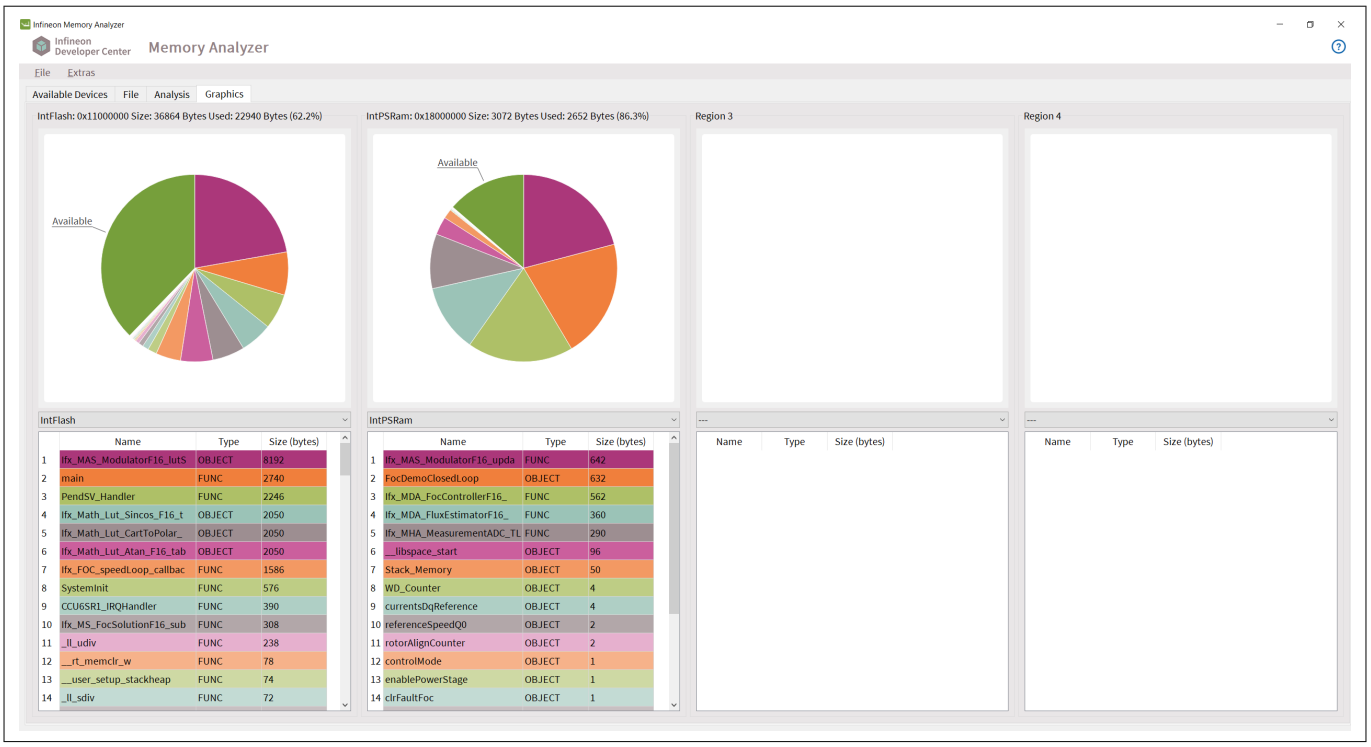


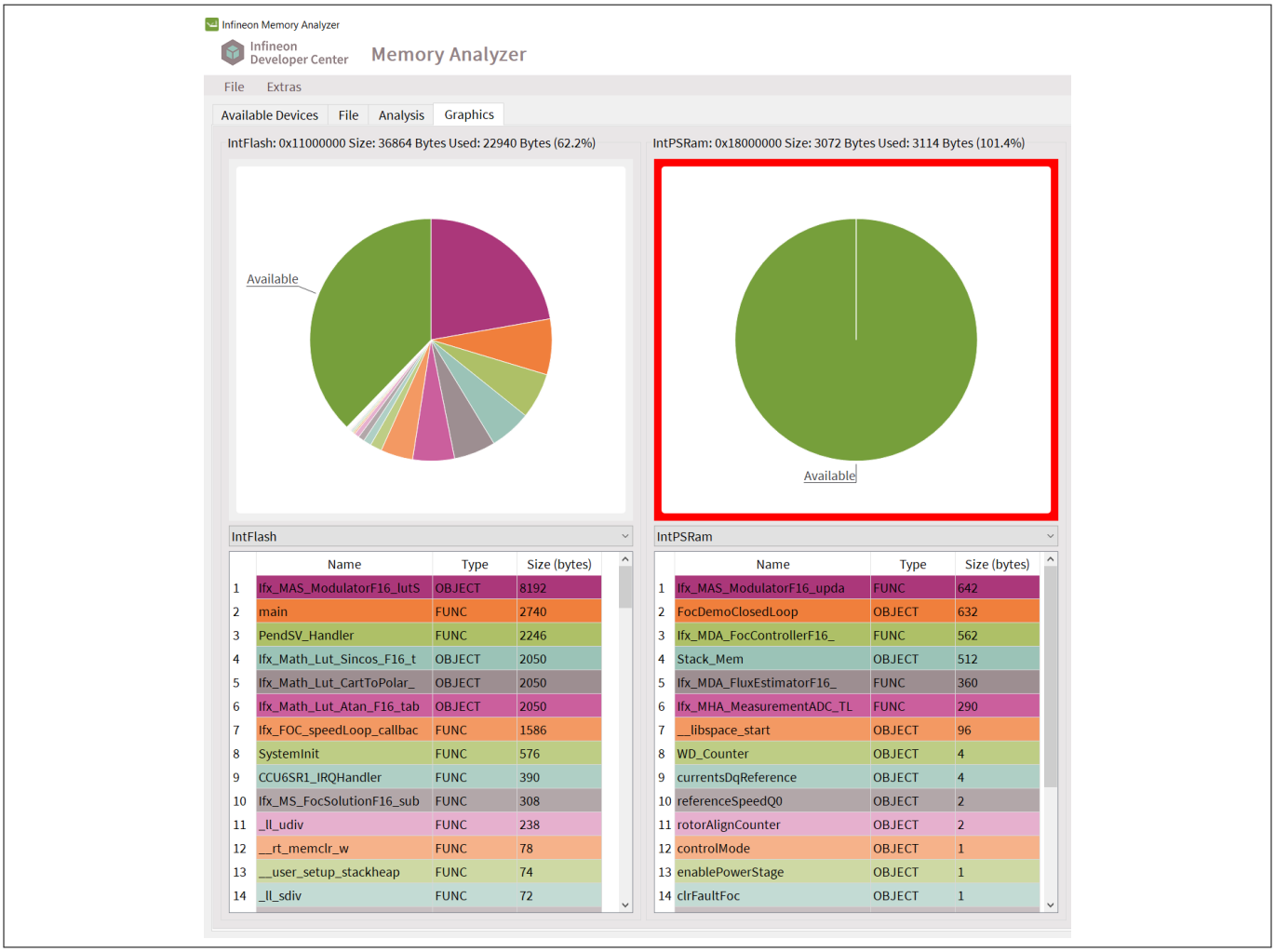
Figure 15 Graphics tab – top 20 objects are colored

For a more precise identification of the 20 largest functions and objects, a fixed color was established to quickly identify them, between the graph and the table.

Selecting a new region from the drop-down list will update the displayed data. If the microcontroller changes in the **File** tab, the list and pie charts are also updated.

2.4.1 Memory overflow

If a memory region overflows, a red frame is displayed around the graph. The selected microcontroller is not suitable for the linker file.



	Name	Type	Size (bytes)
1	Ifx_MAS_ModulatorF16_lutS	OBJECT	8192
2	main	FUNC	2740
3	PendSV_Handler	FUNC	2246
4	Ifx_Math_Lut_Sincos_F16_t	OBJECT	2050
5	Ifx_Math_Lut_CartToPolar_	OBJECT	2050
6	Ifx_Math_Lut_Atan_F16_tab	OBJECT	2050
7	Ifx_FOC_speedLoop_callbac	FUNC	1586
8	SystemInit	FUNC	576
9	CCU6SR1_IRQHandler	FUNC	390
10	Ifx_MS_FocSolutionF16_sub	FUNC	308
11	_ll_udiv	FUNC	238
12	__rt_memclr_w	FUNC	78
13	__user_setup_stackheap	FUNC	74
14	_ll_sdiv	FUNC	72

	Name	Type	Size (bytes)
1	Ifx_MAS_ModulatorF16_upda	FUNC	642
2	FocDemoClosedLoop	OBJECT	632
3	Ifx_MDA_FocControllerF16_	FUNC	562
4	Stack_Mem	OBJECT	512
5	Ifx_MDA_FluxEstimatorF16_	FUNC	360
6	Ifx_MHA_MeasurementADC_TL	FUNC	290
7	__libspace_start	OBJECT	96
8	WD_Counter	OBJECT	4
9	currentsDqReference	OBJECT	4
10	referenceSpeedQ0	OBJECT	2
11	rotorAlignCounter	OBJECT	2
12	controlMode	OBJECT	1
13	enablePowerStage	OBJECT	1
14	clrFaultFoc	OBJECT	1

Figure 16 Graphics tab - memory overflow

2.5 Additional functions

Using the drop-down option **Extras** in the menu bar, the following options can be chosen:

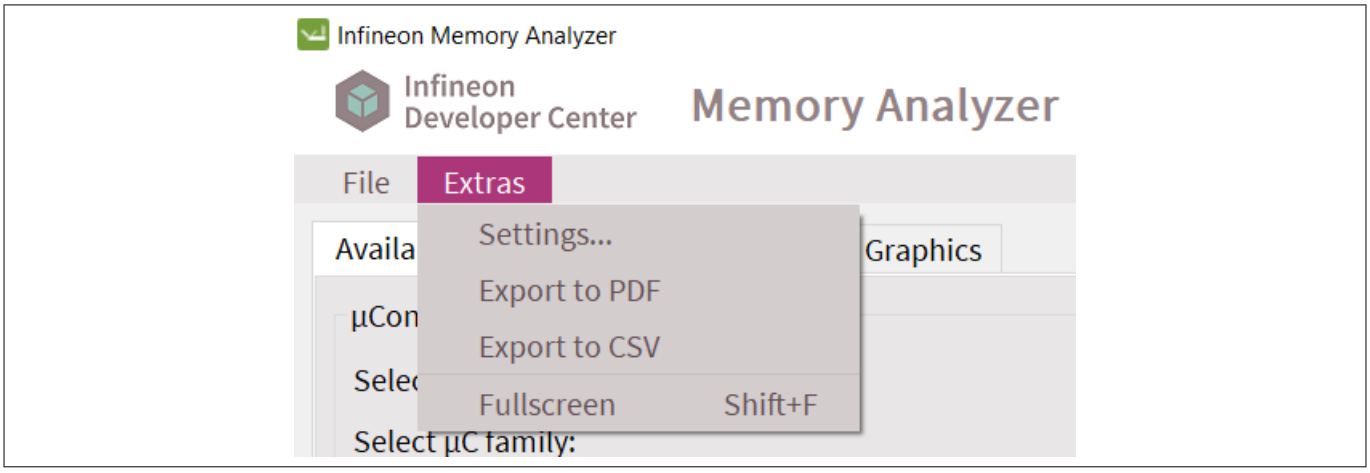


Figure 17 Drop-down option Extras

- Settings:
  in the **Settings** dialog, a working directory (the directory displayed by default when the **Open File** dialog opens) can be entered

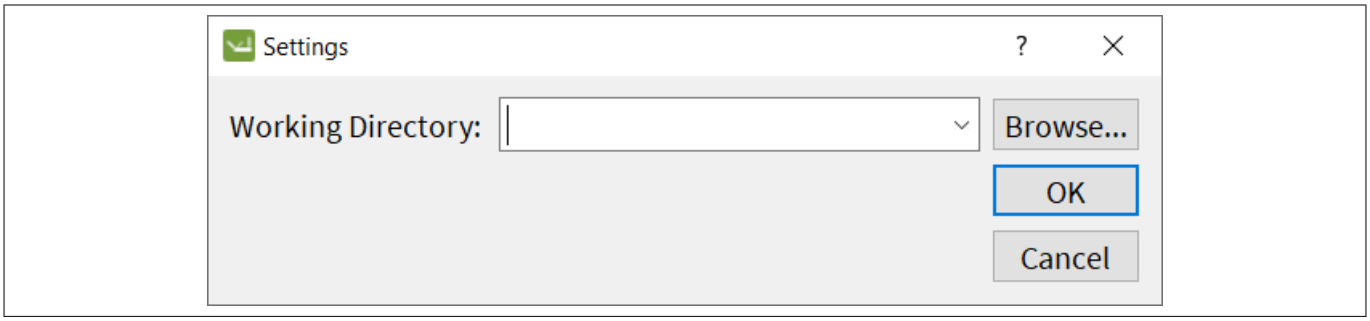


Figure 18 Settings dialog

- Export to PDF:
  writes a screen dump of the tabs to a PDF file
- Export to CSV:
  writes the information about objects into a CSV file
- Full-screen:
  switches the application to full-screen mode by pressing **Shift+F** key on keyboard

3 Licenses disclaimer

The Memory Analyzer is based in part on the work of the Qwt project (http://qwt.sf.net).
The following LGPL/GPLv3 are used in our software and can be found in the license folder:

- QuaZip
- qt 5.12.2
- libiconv 2

Revision history

Document version	Date of release	Description of changes
01.00	22-10-31	Initial version

Trademarks

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

Edition 22-10-31

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-jon1659533810139

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.

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.