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

About this document ................................................................. 1
Table of contents ....................................................................... 2

1 Introduction ........................................................................... 3
  1.1 Tool overview ................................................................. 3
  1.2 Supported linker file types ............................................... 3
  1.3 How to get the tool .......................................................... 4
  1.4 User guide ....................................................................... 5

2 GUI explanation ..................................................................... 6
  2.1 Available devices tab ....................................................... 6
  2.2 File tab ............................................................................ 7
  2.3 Analysis tab ..................................................................... 10
    2.3.1 Invalid item ............................................................... 11
    2.3.2 Sort option ............................................................... 12
    2.3.3 Insert stack memory value in RAM ............................. 13
    2.3.4 Search function ........................................................ 14
    2.3.5 Export statistics in CSV format ................................. 15
  2.4 Graphic tab ..................................................................... 16
    2.4.1 Memory overflow ...................................................... 17
  2.5 Additional functions ........................................................ 18

3 Licenses disclaimer .............................................................. 19

Revision history ....................................................................... 20
Disclaimer ............................................................................... 21
1Introduction

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.
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.

![Available devices tab](image-url)
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](image)

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
2 GUI explanation

- **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), I (large), 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 table](image)

**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)

![Analysis tab overview](image-url)
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.

![Memory Analyzer Window](Image)

Figure 9 Invalid object found
2.3.2 Sort option

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

![Sort options - type](image1)

![Sort options - sort by](image2)
### 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`.

![Memory Analyzer](image)

**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.

![Image of Memory Analyzer with search function](image)

**Figure 13 Search object function**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Start address</th>
<th>End address</th>
<th>Size (bytes)</th>
<th>Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXINT1_IRQHandler</td>
<td>FUNC</td>
<td>0x1100014F</td>
<td>---</td>
<td>0</td>
<td>IntFlash</td>
</tr>
<tr>
<td>GPT1_IRQHandler</td>
<td>FUNC</td>
<td>0x1100014F</td>
<td>---</td>
<td>0</td>
<td>IntFlash</td>
</tr>
<tr>
<td>GPT2_IRQHandler</td>
<td>FUNC</td>
<td>0x11000501</td>
<td>0x11000524</td>
<td>36</td>
<td>IntFlash</td>
</tr>
<tr>
<td>HardFault_Handler</td>
<td>FUNC</td>
<td>0x11000525</td>
<td>0x11000526</td>
<td>2</td>
<td>IntFlash</td>
</tr>
<tr>
<td>IFx_FOC_speedLoss</td>
<td>FUNC</td>
<td>0x11000529</td>
<td>0x110005BA</td>
<td>1586</td>
<td>IntFlash</td>
</tr>
<tr>
<td>IFx_MAS_Modulator</td>
<td>FUNC</td>
<td>0x1800013D</td>
<td>0x180003BE</td>
<td>642</td>
<td>IntPSRam</td>
</tr>
<tr>
<td>IFx_MDA_FluxEstm</td>
<td>FUNC</td>
<td>0x180003C1</td>
<td>0x18000528</td>
<td>360</td>
<td>IntPSRam</td>
</tr>
<tr>
<td>IFx_MDA_FocContr</td>
<td>FUNC</td>
<td>0x18000529</td>
<td>0x1800075A</td>
<td>562</td>
<td>IntPSRam</td>
</tr>
<tr>
<td>IFx_MHA_Measure</td>
<td>FUNC</td>
<td>0x18000019</td>
<td>0x1800013A</td>
<td>290</td>
<td>IntPSRam</td>
</tr>
<tr>
<td>IFx_MS_FocSolu</td>
<td>FUNC</td>
<td>0x1100059D</td>
<td>0x11000C90</td>
<td>308</td>
<td>IntFlash</td>
</tr>
<tr>
<td>MemManage_Handler</td>
<td>FUNC</td>
<td>0x11000C91</td>
<td>0x11000C92</td>
<td>2</td>
<td>IntFlash</td>
</tr>
<tr>
<td>NMI_Handler</td>
<td>FUNC</td>
<td>0x11000127</td>
<td>0x11000128</td>
<td>2</td>
<td>IntFlash</td>
</tr>
<tr>
<td>PendSV_Handler</td>
<td>FUNC</td>
<td>0x110009C5</td>
<td>0x1100155A</td>
<td>2246</td>
<td>IntFlash</td>
</tr>
<tr>
<td>Reset_Handler</td>
<td>FUNC</td>
<td>0x1110</td>
<td>---</td>
<td>8</td>
<td>IntFlash</td>
</tr>
<tr>
<td>SSCI1_IRQHandler</td>
<td>FUNC</td>
<td>0x1100155D</td>
<td>0x1100156C</td>
<td>16</td>
<td>IntFlash</td>
</tr>
<tr>
<td>SSCI2_IRQHandler</td>
<td>FUNC</td>
<td>0x1100156D</td>
<td>0x110017AC</td>
<td>576</td>
<td>IntFlash</td>
</tr>
<tr>
<td>SVC_Handler</td>
<td>FUNC</td>
<td>0x1100125</td>
<td>0x1100126</td>
<td>2</td>
<td>IntFlash</td>
</tr>
<tr>
<td>SYSTick_Handler</td>
<td>FUNC</td>
<td>0x1100155D</td>
<td>0x1100156C</td>
<td>16</td>
<td>IntFlash</td>
</tr>
<tr>
<td>SystemInit</td>
<td>FUNC</td>
<td>0x1100156D</td>
<td>0x110017AC</td>
<td>576</td>
<td>IntFlash</td>
</tr>
<tr>
<td>UART1_IRQHandler</td>
<td>FUNC</td>
<td>0x110014F</td>
<td>---</td>
<td>0</td>
<td>IntFlash</td>
</tr>
<tr>
<td>UART2_IRQHandler</td>
<td>FUNC</td>
<td>0x110014F</td>
<td>---</td>
<td>0</td>
<td>IntFlash</td>
</tr>
<tr>
<td>UsbGetFault_Handler</td>
<td>FUNC</td>
<td>0x110017AD</td>
<td>0x110017AE</td>
<td>2</td>
<td>IntFlash</td>
</tr>
<tr>
<td>USEuseSemihosting</td>
<td>FUNC</td>
<td>0x11001ED</td>
<td>---</td>
<td>0</td>
<td>IntFlash</td>
</tr>
</tbody>
</table>
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](image)
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.

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.

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:

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

- **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

<table>
<thead>
<tr>
<th>Document version</th>
<th>Date of release</th>
<th>Description of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>01.00</td>
<td>22-10-31</td>
<td>Initial version</td>
</tr>
</tbody>
</table>
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.