tinyMicon MatisseCORE™

mtmake User's Guide

Build tool for tinyMicon MatisseCORE™ mtmake User's Guide

Revision History

Date	Version	Contents
2021/09/19	Rev.001	Describes the contents of mtmake V1.01.00

Table of Contents

1 Technical Terms

Table 1. Technical Terms

Term	Meaning
tinyMicon MatisseCORE™	A compact 8-bit microcomputer developed by ROHM for LSI embedded systems.
build tool	A program that manipulates the compiler to generate an executable file from source files.
Visual Studio Code (VS Code)	An open-source IDE developed by Microsoft Corporation. (https://azure.microsoft.com/ja-jp/products/visual-studio-code/)
IDE	Integrated Development Environment. A tool that integrates the tools necessary for programming into a single environment.
ninja	World-renowned open-source build tool. (https://ninja-build.org/)

2 Overview

mtmake is a build tool for tinyMicon MatisseCORE™.

2.1 Features

mtmake has the following features.

1 All functions of common build tools are supported.
2 High speed build with incremental build function.
3 Integration with VS Code.

The main purpose of mtmake is to work with VS Code. mtmake can be run from VS Code, which greatly reduces the time required for build configuration and build process.

3 Operating environment

The operating environment of mtmake is described below.

3.1 System Requirements

Table 2. System Requirements

OS	Windows 7 (32-bit/64-bit) Windows 10 (32-bit/64-bit)
CPU	Intel Core series or equivalent performance CPU.
memory	4GByte or more installed
HDD/SSD	At least 100 MByte of free space.

3.2 Installation

When you run the mtcc installer (MatisseCCompiler-XX.XX.XX.exe), mtmake will also be installed.

By default, it will be installed in "C:\Program Files\ROHM\Matisse\C\bin".

3.3 Configuring the tool

The configuration of the mtmake is described below.

The actual build process is handled by ninja. mtmake is mainly responsible for generating build scripts for ninja.

mtmake V1.1.0 uses ninja V1.10.0.

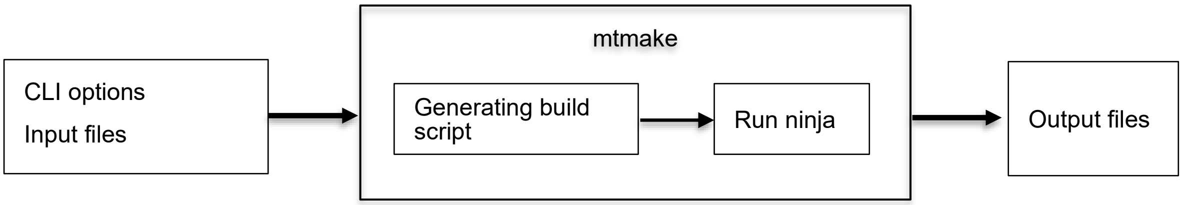

Figure 1. mtmakeSystem Block Diagram

4 How to use

4.1 Running mtmake

You can run mtmake from the command line interface as follows.

```
> mtmake [options]
```

An example of the command line to run mtmake is as follows.

```
> mtmake --vscode-settings=.vscode/settings.json
```

4.2 List of command line options.

A list of command line options for mtmake is as follows.

Table 3. mtmakeList of command-line options

Command line option	Description
--vscode-settings=	Generate a build script for ninja from the specified Visual Studio Code setting file and run the build process with ninja.
Other options	Pass the given command line options to ninja to run ninja.

4.3 Details of command line options

4.3.1 --vscode-settings=

Description

The --vscode-settings= option specifies the Visual Studio Code setting file (settings.json) in the following format.

```
--vscode-settings=[File path to settings.json]
```

When this command is run, mtmake will create a build script file for ninja (build.ninja) in the current directory from the contents of settings.json, and start the build process with ninja.

mtmake will also check the contents in settings.json. See Section 4.4 settings.json for more information about settings and what mtmake checks.

Example of use

```
> mtmake --vscode-settings=.vscode/settings.json
```

4.3.2 Other options

Description

If command line options other than the above are specified, mtmake will pass those command line options to ninja as is and start ninja.

Please refer to the official ninja documentation (https://ninja-build.org/manual.html) for more information on the command line options available to ninja.

Example of use

```
> mtmake -t clean          // This command runs "ninja -t clean"
```

4.4 settings.json

The configuration items in settings.json that mtmake refers to are described below.

4.4.1 Configuration items

The following is a list of configuration items in settings.json that mtmake refers to when generating the build script. It also explains how mtmake checks these items when generating build script.

Table 4. settings.json List of setting items

Configuration item	Description	Check
matisse.C.build.compilerPath	The path to mtcc.	Check if the file exists.
matisse.C.build.buildCommand	The path to mtmake.	Check if the file exists.
matisse.C.build.srcFiles	The list of source files.	Check if the file exists.
matisse.C.build.outputFileName	The file name generated by the build.	Check if the file can be generated.
matisse.C.build.binFileExtension	The extension of the file generated by the build.	Check if the file can be generated.
matisse.C.build.excludeMul	Whether the target has a multiplier.	-
matisse.C.build.programSize	The program memory size of the target.	Check if the total memory size is exceeded.
matisse.C.build.dataSize	The data memory size of the target.	Check if the total memory size is exceeded.
matisse.C.build.exProgramSize	The extended program memory size of the target.	Check if the total memory size is exceeded.
matisse.C.build.enableLto	Whether the link time optimization is enabled.	-
matisse.C.build.optimizationLevel	The optimization level.	-
matisse.C.build.includePath	The list of the directories to load include files from.	Check if the directory exists.
matisse.C.build.libraryPath	The list of the directories to load library files from.	Check if the directory exists.
matisse.C.build.libraryFiles	The list of the library files to be linked.	Check if the file exists.
matisse.C.build.preprocessorDefinitions	The list of the preprocessor macro definitions.	-
matisse.C.build.additionalCompileOptions	The list of additional compiler options.	-
matisse.C.build.mapFilePath	The path to map file generated by the build.	Check if the file can be generated.
matisse.C.build.additionalLinkerOptions	The list of additional linker options.	-
matisse.C.debug.mtProxyPath	The path to the debug server.	Check if the file exists.
matisse.C.debug.debuggerPath	The path to the debugger.	Check if the file exists.
matisse.C.debug.debugPort	The port number used for debugging communication.	Check if the port number is valid.
matisse.C.debug.memoryWindowSettingFile	The path to the setting file for memory window.	Check if the file can be generated.
matisse.C.debug.peripheralWindowSettingFile	The path to the setting file for peripheral window.	Check if the file can be generated.
matisse.C.others.downloaderPath	The path to the downloader.	Check if the file exists.
matisse.C.others.mtCheckerPath	The path to the debug configuration checking tool.	Check if the file exists.

5 Open-source software licenses

This software includes open-source software (hereinafter referred to as "open-source software program") provided under the following license conditions, in addition to software for which ROHM owns or is licensed.

Open-source software programs are subject to their respective license terms, so in the event of a conflict between the license terms of an open-source software program and this material, the license terms of the open-source software program shall prevail.

Included open-source software and their license terms

・Mono.Options(The MIT License)

・Newtonsoft.Json(The MIT License)

6 Trademark notices

"Windows" and "VS Code" are trademarks of Microsoft Group companies.

"Intel" is a trademark of Intel Corporation or its subsidiaries.

"Core™" is a trademark or registered trademark of Intel Corporation or its subsidiaries.

“tinyMicon MatisseCORE™” is a trademark or registered trademark of ROHM Corporation.

Caution

1. The information written in these materials regarding the software and system (hereinafter collectively “Software”) and the contents of the materials are current as of the date of the material’s issuance, and may be changed by ROHM, at any time and for any reason, without prior notice.

2. If you plan to use the Software in connection with any equipment or device (such as the medical equipment, transportation equipment, traffic equipment, aerospace equipment, nuclear power control equipment, vehicle equipment including the fuel control system and/or car accessories, and/or various kinds of safety devices etc.) which require extremely high reliability, and whose breakdown or malfunction relate to the risk of personal injury or death, or any other serious damage (such usage is hereinafter called “Special Usage”), you must first consult with the ROHM’s sales representative. ROHM is not responsible for any loss, injury, or damage etc. incurred by you or any other third party caused by any Special Usage without ROHM’s prior written approval.

3. Semiconductor products may break or malfunction due to various factors. You are responsible for designing, testing, and implementing safety measures in connection with your use of any ROHM products using the Software (such ROHM products are hereinafter called “Product”) Such safety measures include, but are not limited to, derating, reductant design, fire spread prevention, backup, and/or fail safe etc. in order to prevent the accident resulting in injury or death and/or fire damage etc.. ROHM is not responsible and hereby disclaims liability for any damage in relation to your use beyond the rated value, or the non-compliance with any precaution for use.

4. ROHM is not responsible for any direct and/or indirect damage to you, or any third parties, (including the damage caused by loss of intangible asset such as information, data, or program etc., loss and/or interruption of profit) which is caused by the use or impossibility to use of the Software.

5. Since the Software, these materials, and/or the Product contain confidential information of ROHM’, including technical information, and/or trade secrets, you are prohibited from engaging in any of the following acts in whole or part, without ROHM’s prior written approval:
   (i) disclosing any ROHM confidential information to a third party;
   (ii) disassembling, reverse engineering, and/or any other analysis;
   (iii) reprinting, copy, and/or reproduction; or
   (iv) removing the copyright notice included in the Software.

6. When exporting the Software, or the technology and/or confidential information written in these materials, you are required to follow the applicable export control laws and regulations such as “Foreign Exchange and Foreign Tarade Act” and/or “Export Administration Regulations (EAR).”.

7. ROHM disclaims all warranties, statutory or otherwise, and ROHM hereby disclaims any warranty for non-infringement for the Software and/or the information written in these materials. Accordingly, ROHM is not liable to you for any direct or third-party claims of infringement of rights.

8. No license, whether expressly or implied, is granted hereby under any intellectual property rights or other rights of ROHM or any third parties with respect to the Software or Products or the information contained in these materials.

9. You agree to indemnify, defend and hold harmless ROHM and ROHM’s officers and/or employees from responsibility, and hold them harmless, and defend them from any damage, loss, penalty, or cost caused by any claim of liability (including but not limited to the attorney fees) resulting from, or incurred relating to the following acts:
   (1) any alleged infringement of a third party’s rights or the violation of laws caused by reading, download, encryption, summarization, copy, or transfer etc.; or
   (2) violation of these materials.

10. ROHM does not guarantee that these materials or the Software is error free. ROHM shall not be in any way responsible or liable for any damages, expenses, or losses incurred by you or third parties resulting from errors contained in these materials.

ROHM SEMICONDUCTOR

Thank you for using ROHM products.
For inquiries about our products, please contact us.

ROHM Customer Support System

https://www.rohm.co.jp/contactus