AN13579

LPC553x/LPC55S3x CoreMark on Cortex-M33 Porting Guide

Rev. 3 — 23 November 2023 Application note

Document information

Information	Content
Keywords	AN13579, LPC553x/LPC55S3x, CoreMark, Cortex-M33
Abstract	This application note describes how to port CoreMark code to LPC553x/LPC55S3x, which involves setting up software and hardware including memory partitioning, compiler setting, and board setup.

1 Introduction

Embedded Microprocessor Benchmark Consortium (EEMBC) CoreMark is a benchmark that measures the performance of microcontrollers (MCUs) and central processing units (CPUs) used in embedded systems. It contains implementations of list processing (find and sort), matrix manipulation (common matrix operations), state machine (determine if an input stream contains valid numbers), and cyclic redundancy check (CRC) algorithms.

LPC553x/LPC55S3x is an ARM Cortex-M33-based microcontroller for embedded applications. These devices include:

- Up to 128 kB of on-chip SRAM; Up to 256 kB on-chip flash
- FlexSPI with cache and dynamic decryption
- CASPER Crypto/FFT engine
- High-speed and full-speed USB host and device interface with crystal-less operation for full-speed
- One CAN-FD
- One QuadFlash Filter
- One DMIC
- One EZH
- One I3C interface
- Five general-purpose timers, one SCTimer/PWM, one RTC/alarm timer
- One 24-bit Multi-Rate Timer (MRT)
- One OS Timer
- One Micro-tick Timer
- A Windowed Watchdog Timer (WWDT), code Watchdog Timer
- Eight flexible serial communication peripherals (which can be configured as a USART, SPI, high-speed SPI, I2C, or I2S interface)
- Two 16-bit 2.0 Msamples/sec ADCs capable of four simultaneous conversions, four comparators, and two temperature sensors.
- Three 12-bit 1 Msample/sec DACs
- Three OpAmps,
- Two FlexPWM timers
- Two QEIs

The Cortex-M33 offers an 18.2 % performance increase in the same process technology compared to the high embedded performance bars already established by Cortex-M4 processors while improving power efficiency. The Cortex-M33 official CoreMark is 4.02 CoreMark/MHz. The Cortex-M4 official CoreMark is 3.40 CoreMark/MHz.

This application note describes how to port CoreMark code to LPC553x/LPC55S3x, which involves setting up software and hardware including memory partitioning, compiler setting, and board setup. It also describes how to measure CoreMark scores on the Cortex-M33 and the results including CoreMark scores and power consumption in µA/MHz. Separate CoreMark projects for different software development tools (Keil MDK, IAR EWARM, and MCUXpresso IDE) are also included here for reference.

2 Integration of CoreMark library to SDK2.14 framework

The software package associated with this application note contains an SDK2.14 based project framework. It allows developers to drop in the CoreMark library sources and quickly get up and running with benchmarking the LPC553x/LPC55S3x. To get started, follow the link. Click the download link as shown in Figure 1 and follow the instructions on that page.

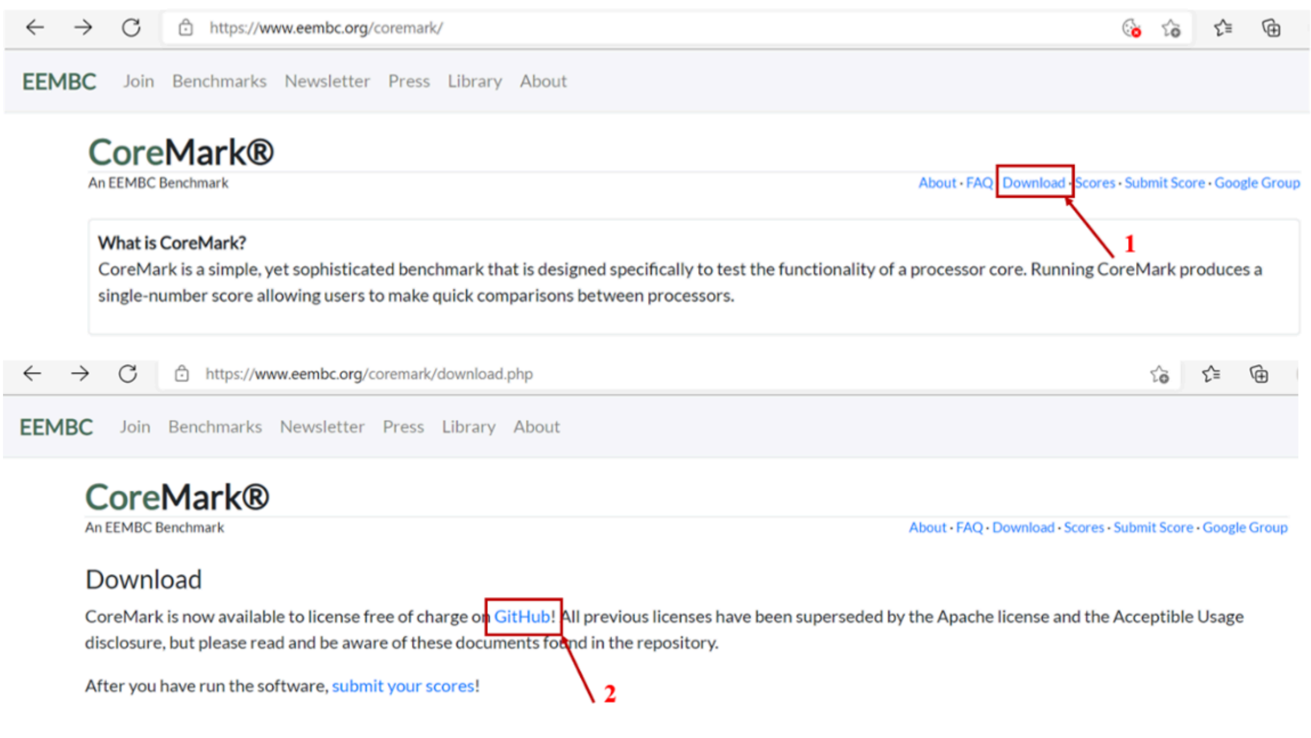


Figure 1. EEMBC CoreMark download link

After reviewing the license terms, look through the readme file. The readme gives step-by-step instructions on unpacking and building the distribution. It also helps with getting familiar with the CoreMark terminology used throughout the application note.

2.1 Port CoreMark library into CoreMark framework

In this application note, there are 4 variants of CoreMark projects for each IDE. There are 2 variants of execution of the CoreMark application: from internal flash and internal SRAMX.

The variants of CoreMark projects:

1. `coremark_score_on_flash` executes the CoreMark application from internal Flash.
2. `coremark_score_on_sramx` executes the CoreMark application from internal RAM.
3. `coremark_uAMHz_on_flash` measures the current when the CoreMark executes on Flash.
4. `coremark_uAMHz_on_sramx` measures current when the CoreMark executes on RAM.

The CoreMark projects are found in the following locations:

Keil MDK IDE:

`lpc55s3x_coremark_mdk\ lpc55s3x_coremark_mdk.uvprojx`

IAR Workbench IDE:

`lpc55s3x_coremark_iar\ lpc55s3x_coremark_iar.eww`

Each of the executing settings has three frequency settings: 12 MHz(FRO), 96 MHz(FRO), 100 MHz(PLL), and 150 MHz(PLL).

Depending on the toolchain, the workspace must look as shown in the following figures. The CoreMark framework requires the addition of the CoreMark files from EEMBC.

2.1.1 CoreMark framework for Keil MDK / IAR EWARM / MCUXpresso IDE

The `lpc55s3x_coremark_xxx` project must be set as active before the CoreMark source code files can be added.

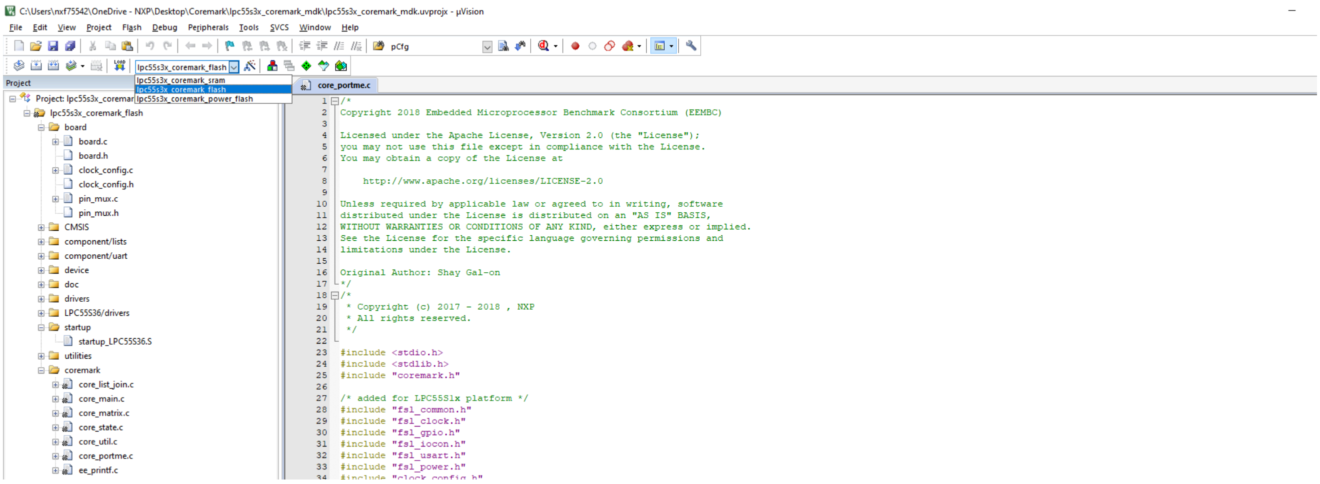


Figure 2. Keil MDK CoreMark project configuration select

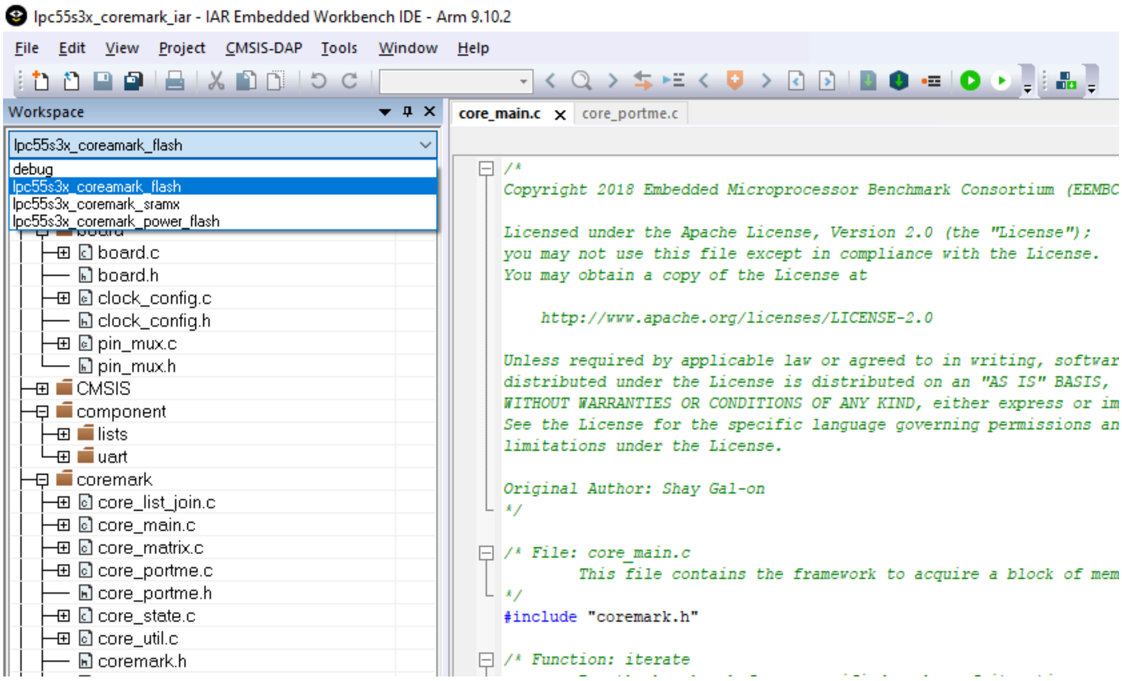


Figure 3. IAR EWARM workspace

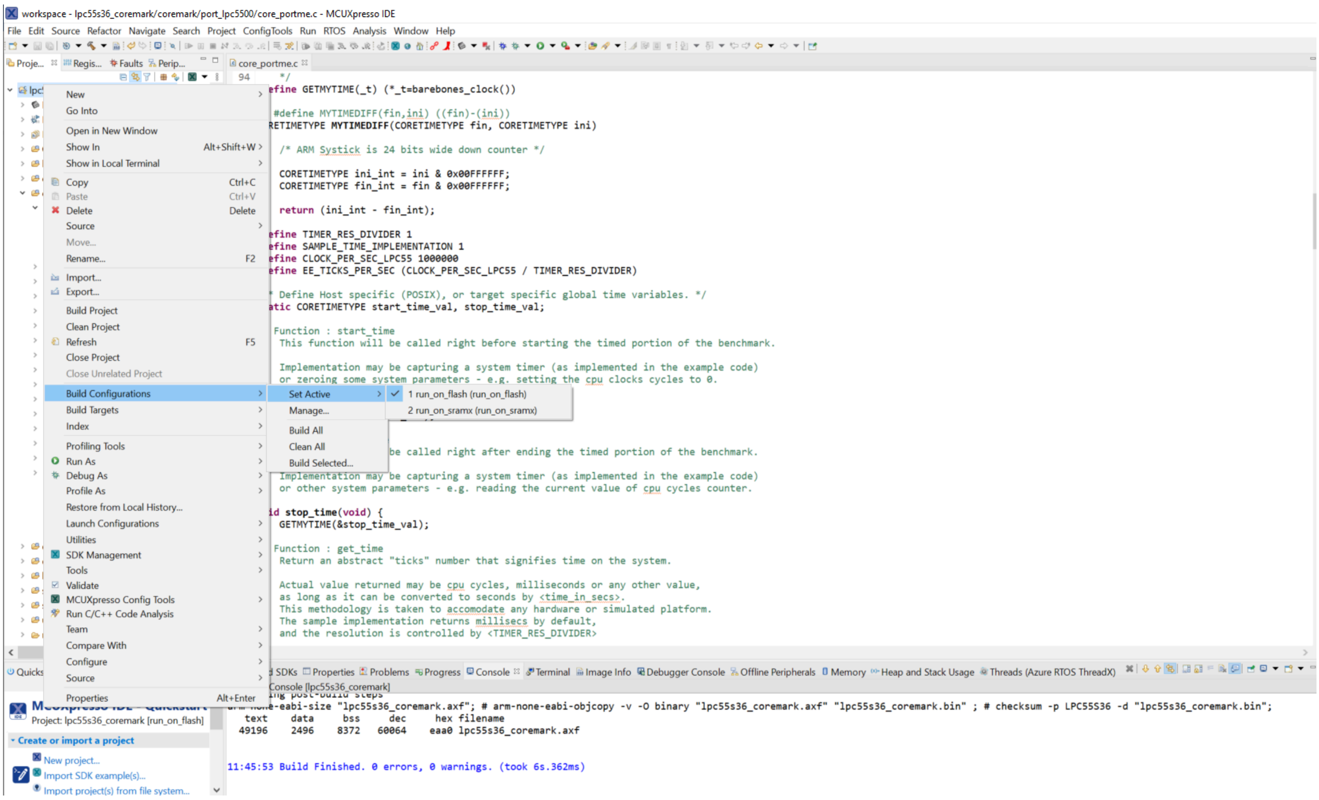


Figure 4. MCUXpresso project configuration select

Copy the following files from the CoreMark package downloaded from EEMBC:

- `core_list_join.c`
- `core_main.c`
- `core_matrix.c`
- `core_state.c`
- `core_util.c`
- `coremark.h`

Copy the following files from the barebones file:

- `core_portme.c`
- `core_portme.h`
- `ee_printf.c`

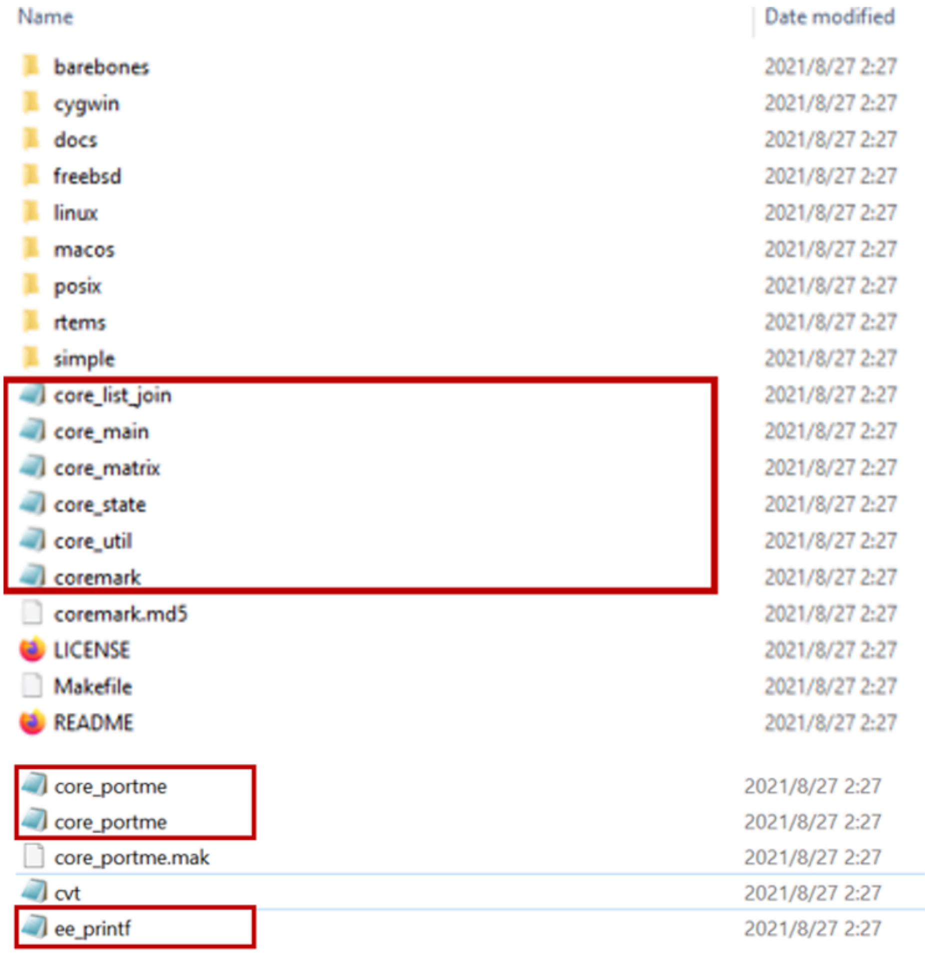


Figure 5. CoreMark files

For the Keil MDK place, these files are in the project directory `lpc55s3x_coremark_mdk\coremark`.

For the IAR Embedded Workbench place, these files are in the project directory `lpc55s3x_coremark_iar\coremark`.

For the MCUXpresso place, these files are in the project directory `lpc55s3x_coremark_mcux\coremark`.

For the KEIL MDK project, right-click the CoreMark folder, select **Add**, then **Add Files**.

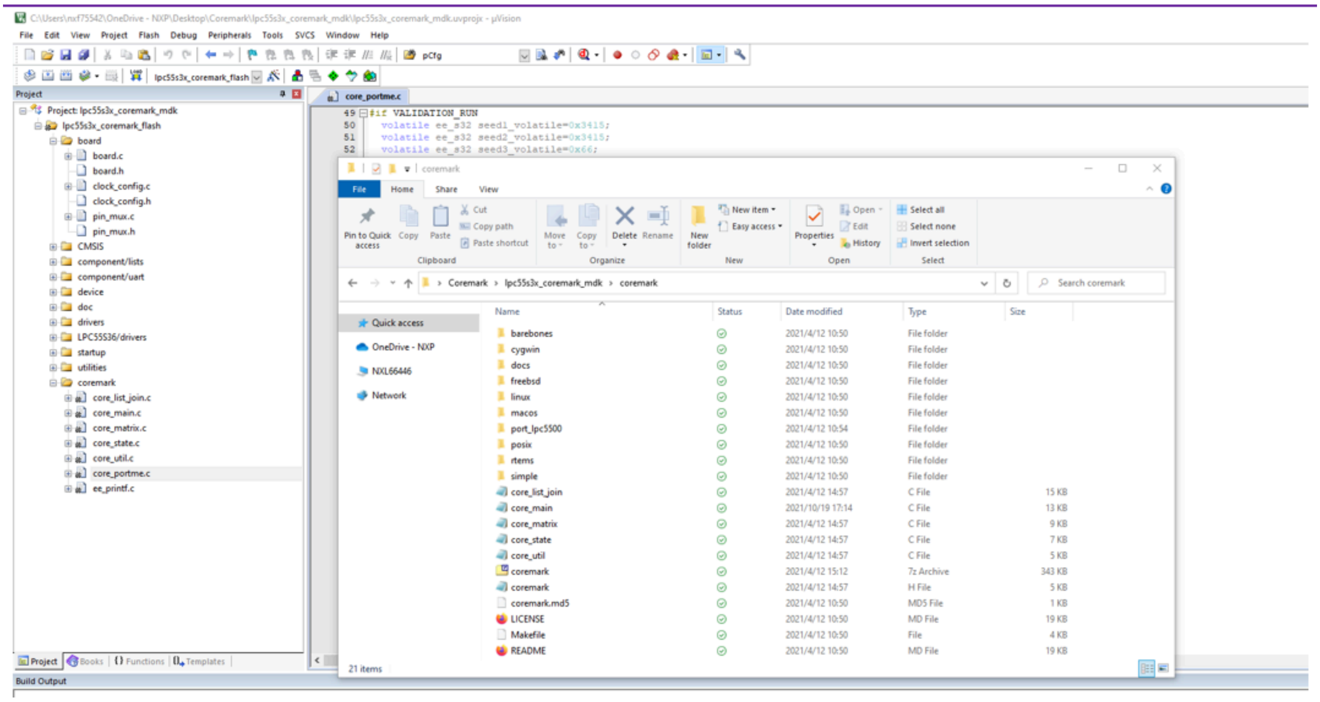


Figure 6. Adding files in Keil MDK

For the IAR Embedded Workbench, right-click the CoreMark folder, select **Add**, then **Add Files**.

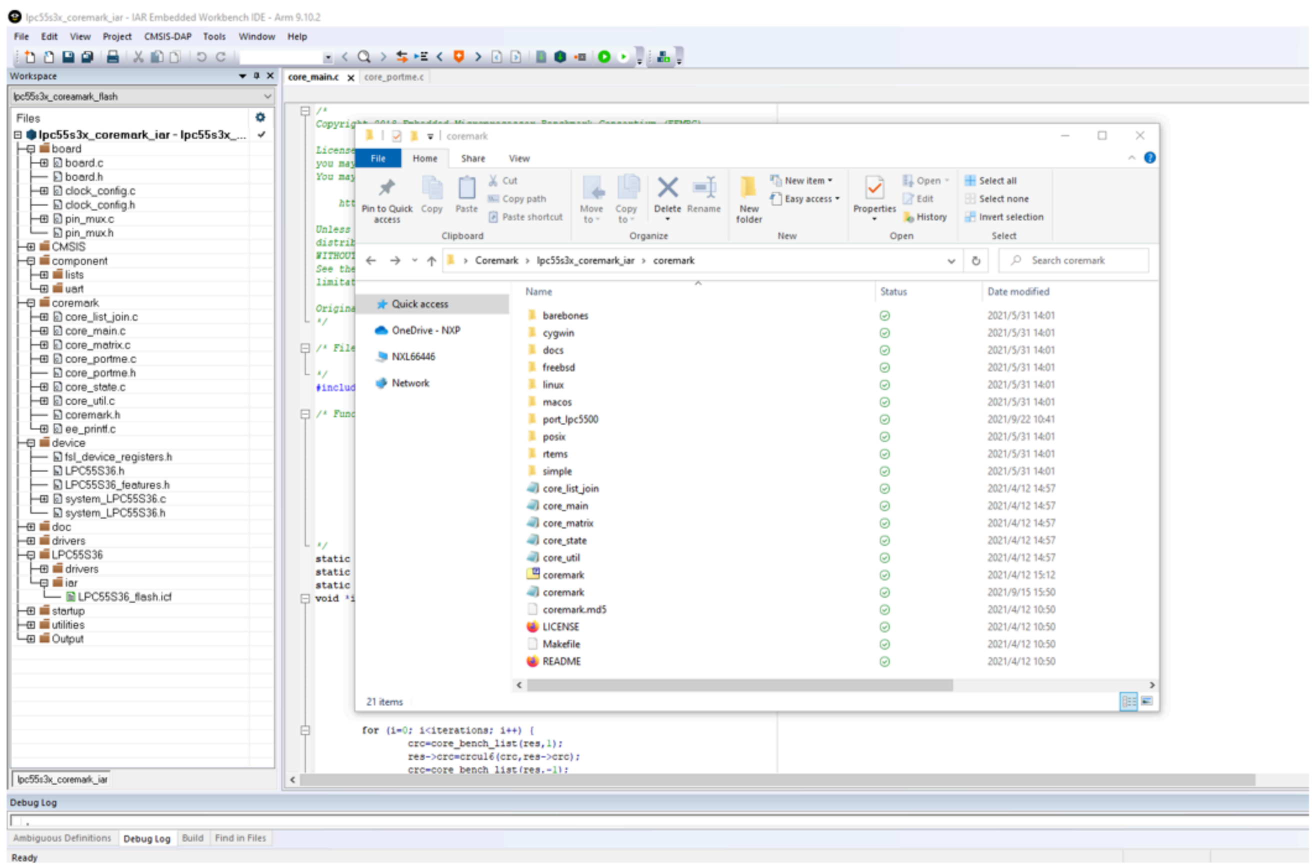


Figure 7. Adding files in the IAR EWARM workspace

For the MCUXpresso project, copy the files into the source folder. Then click **Refresh**, the files are added to the project automatically.

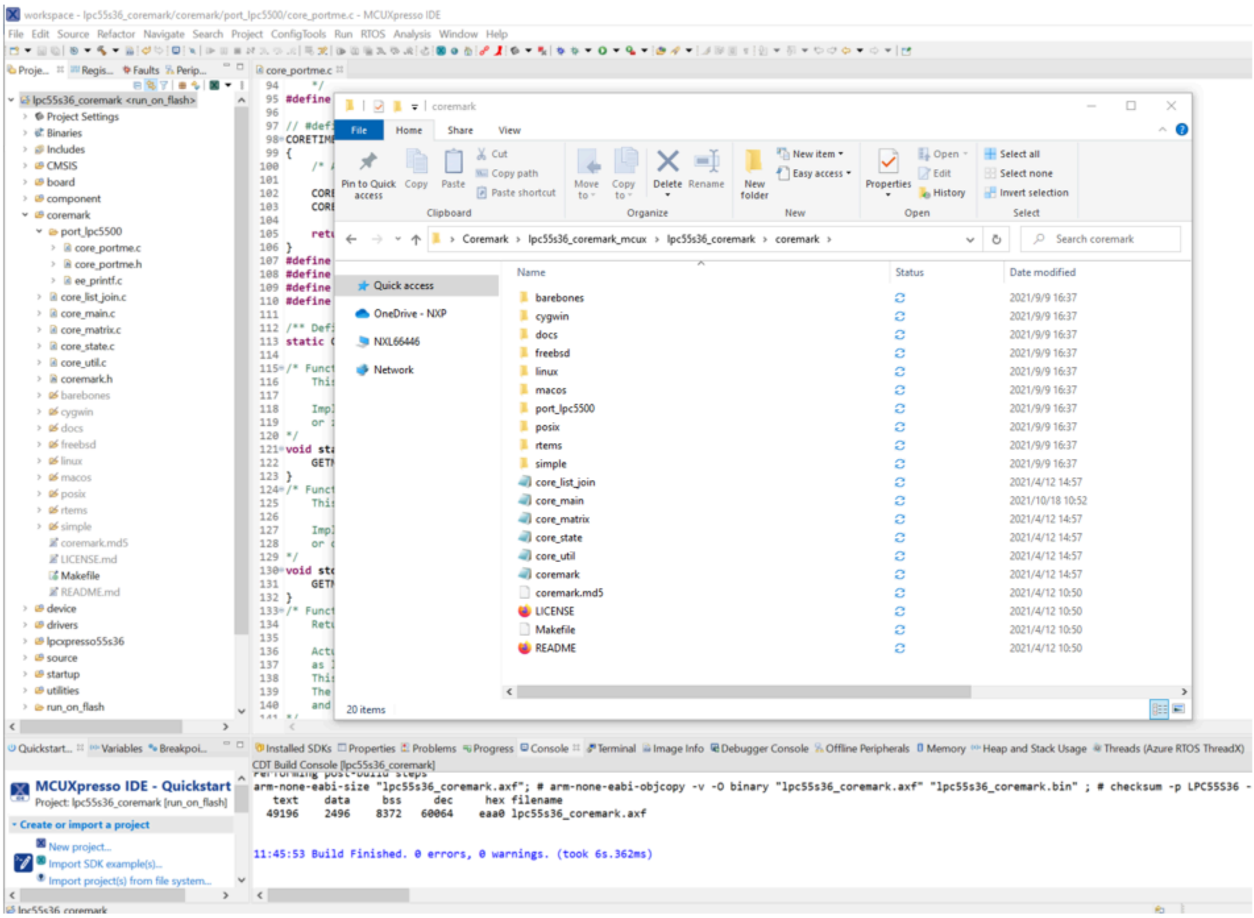


Figure 8. Adding files in the MCUXpresso workspace

Use the `core_portme.c` and `core_portme.h` files provided with the application note and not the one from the EEMBC CoreMark package. For convenience, these files have the required porting changes ready for use.

Copy these files to the CoreMark folder for all three toolchains and add the `core_portme.c` file in the project framework under the source group.

Several files must be modified to support CoreMark. They are described below.

In the project scatter file, change the stack size as 0x1000.

`define symbol __size_cstack__` = 0x1000;

`define symbol __size_heap__` = 0x1000;

To add the path to the header files used in the project, in Keil MDK under Project > Options > C/C++(AC6) tab, click **Include path** and add the following paths that contain the header files.

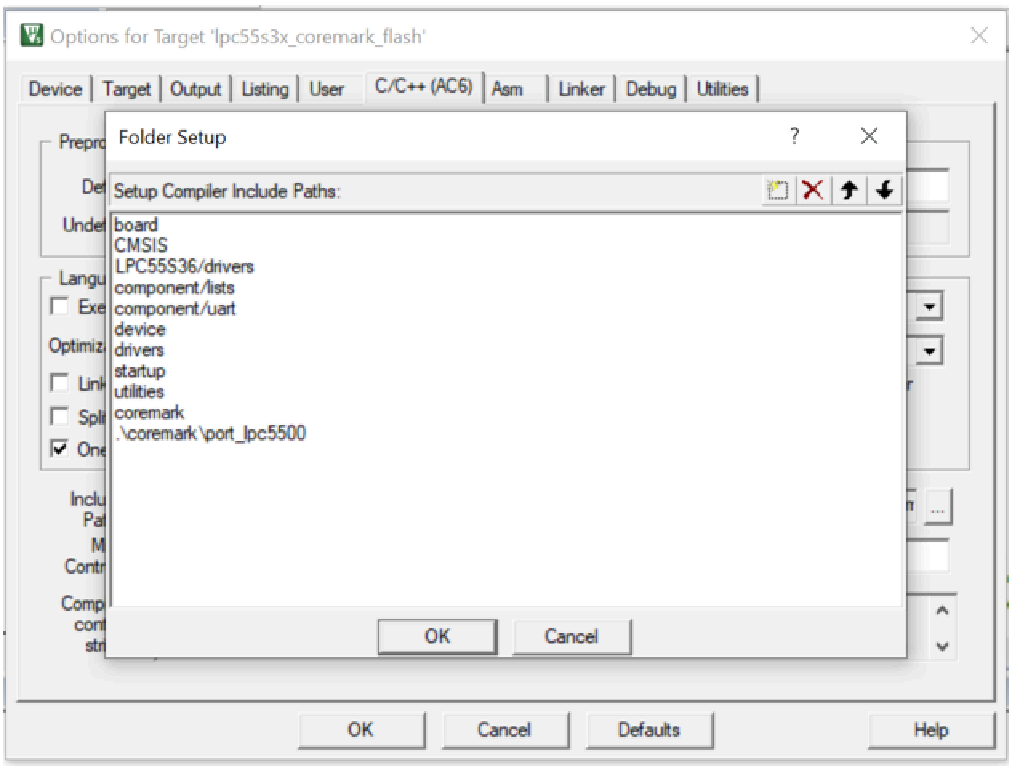


Figure 9. Keil MDK compiler include paths

In IAR, under Project > Options > C/C++ Compiler, click **Preprocessor**, and add the following paths that contain the header files.

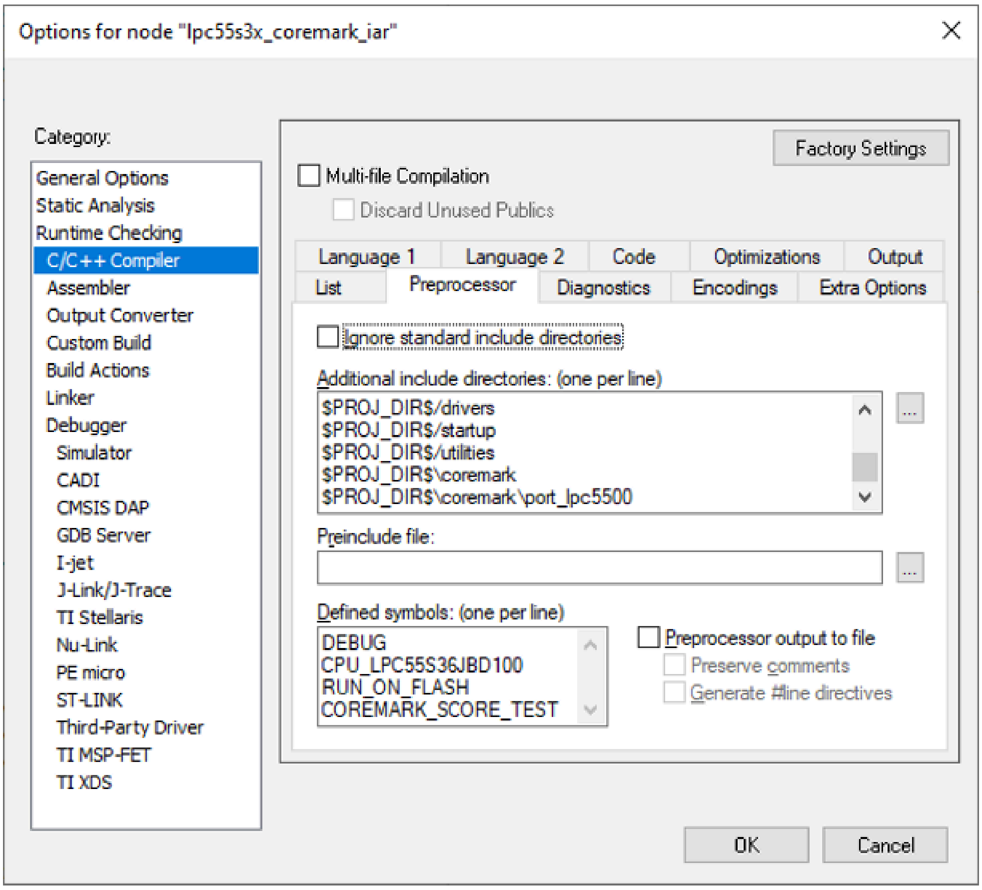


Figure 10. IAR EWARM compiler include paths

The CoreMark files have now been successfully ported into the CoreMark project framework.

In MCUXpresso, under "Properties for xxxx" > C/C++ Build > Settings >, click **Includes** and add the following paths that contain the header files.

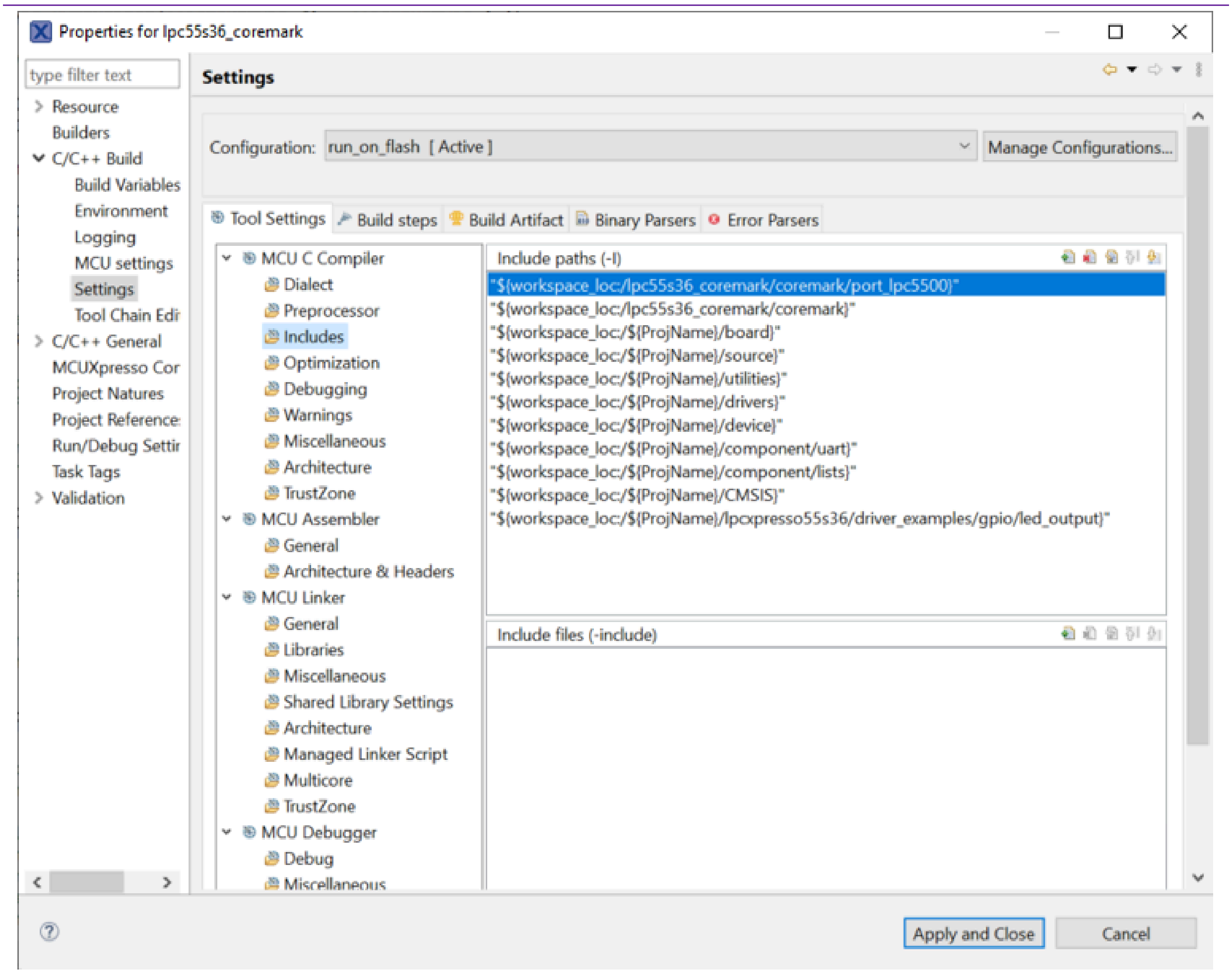


Figure 11. MCUXpresso compiler include paths

The CoreMark files have now been successfully ported into the CoreMark project framework.

2.1.2 CoreMark framework to execute from internal SRAM

The project `lpc55s3x_coremark_sram` executes the CoreMark application from the 16 kB SRAMX memory region.

The files `core_list_join.c`, `core_main.c`, `core_matrix.c`, `core_state.c`, and `core_util.c` are relocated to execute from SRAMX using the linker scripts.

For Keil MDK, the linker script is at: `.\lpc55s3x_coremark_mdk\LPC55s36_sram.scf`.

The linker script setting for the `lpc55s3x_coremark_sram` project is shown in Figure 12:

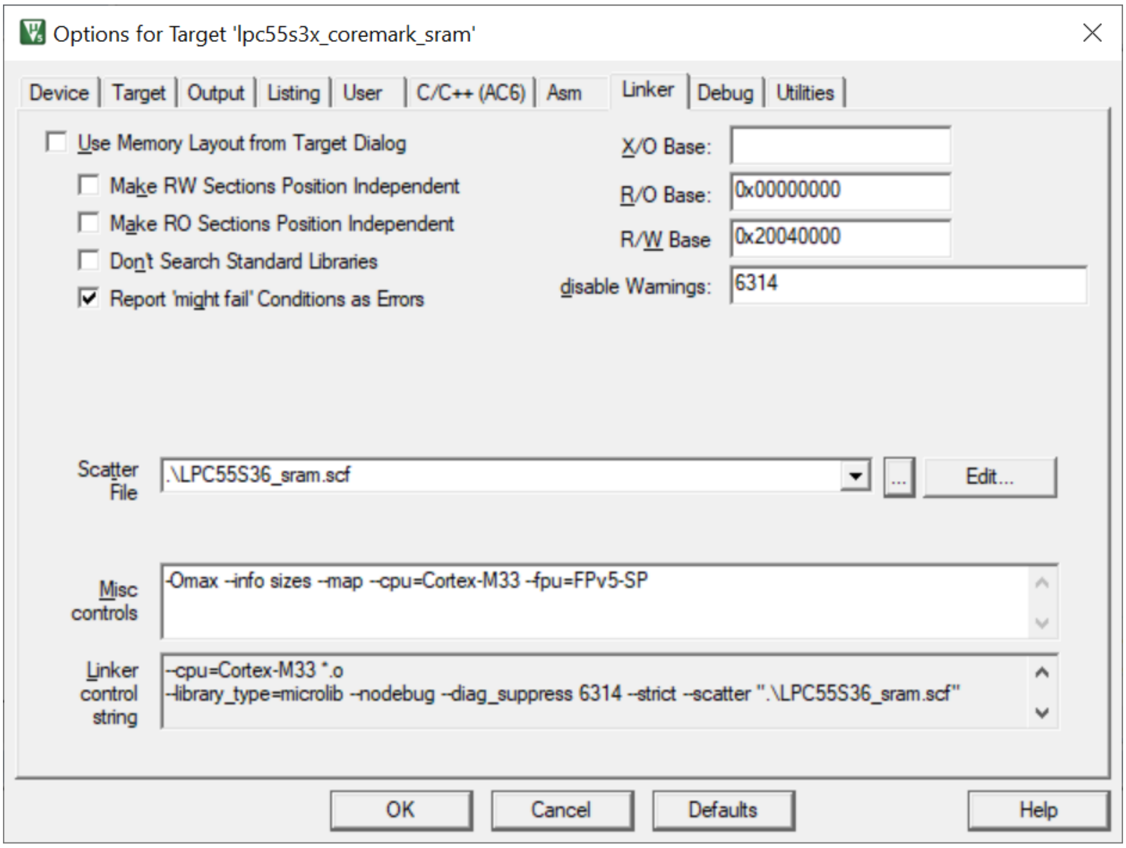


Figure 12. Linker script in Keil IDE

For IAR EWARM IDE, to execute CoreMark in Internal SRAM and to place CoreMark operation codes into the RAM section, add the following lines of code in the `icf` file, as shown in Figure 13:

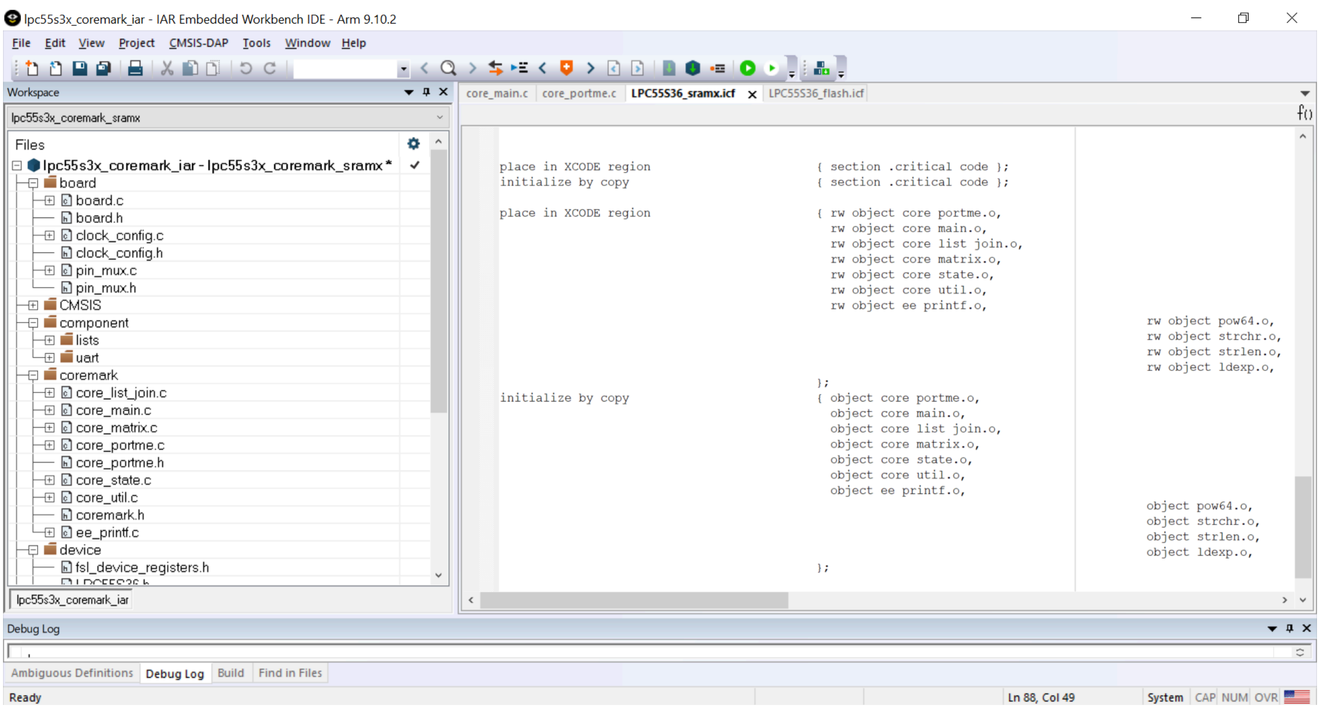


Figure 13. IAR EWARM allocate code to SRAM area

For MCUXpresso to execute CoreMark in Internal SRAM, select the linker file as `LPC55s36_coremark_run_on_sramx.ld` in the Managed Linker script, as shown Figure 14:

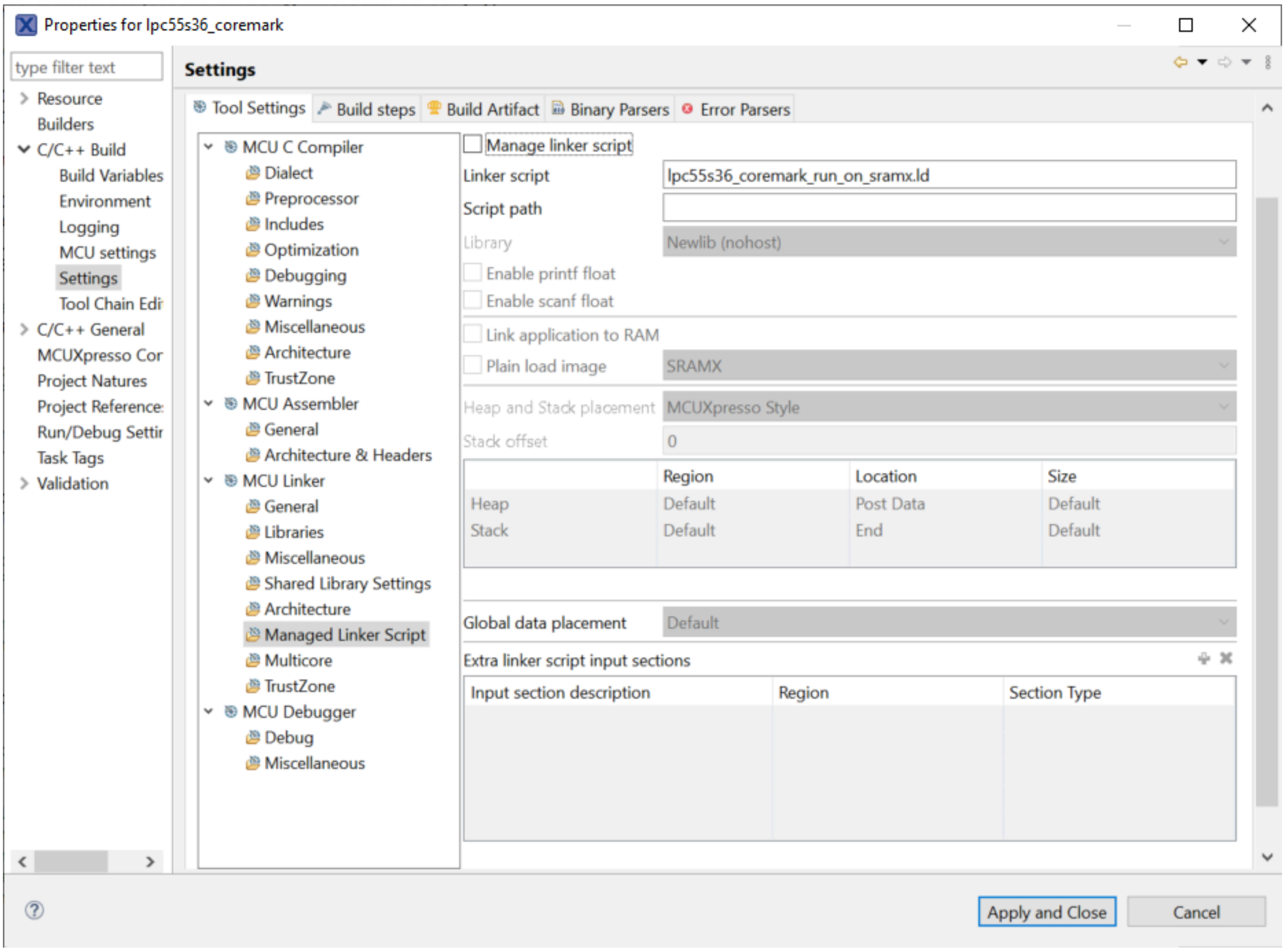


Figure 14. MCUXpresso allocate code to SRAM area

2.2 Optimizing the CoreMark framework

Many factors affecting the CoreMark and µA/MHz score can be optimized. Some of these factors are IDE-dependent optimizations, while others leverage the MCU architecture for better performance. The goal is to be able to produce the best scores from all three IDEs. These IDEs are constantly changing and a different version of a given IDE can add or remove features that can make these optimizations obsolete or ineffective. The following are the IDE versions that are applicable to this application note:

Keil MDK v5.37

IAR EWARM 9.10.2

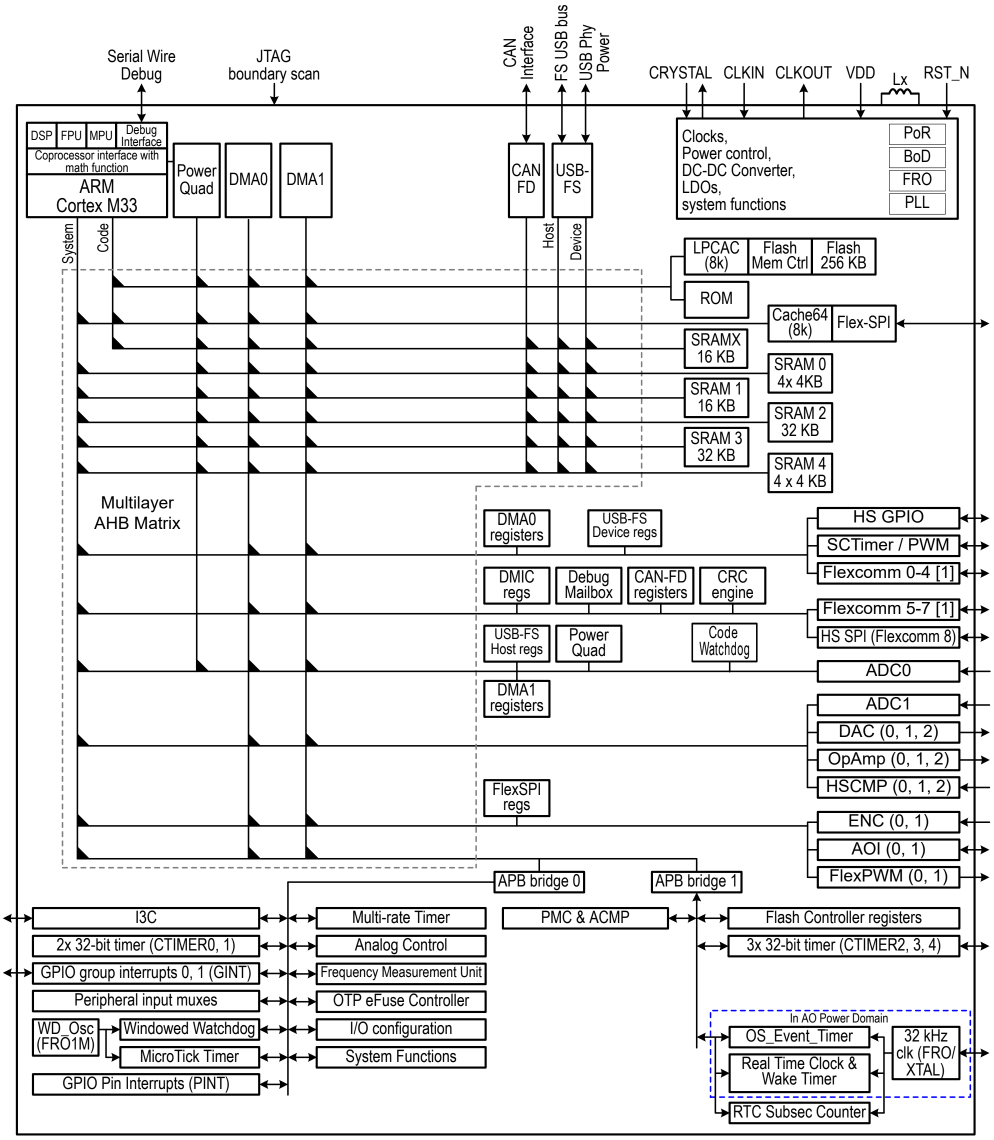
MCUXpresso 11.8.0

2.2.1 Memory considerations

Due to the inherent architecture of SRAM and flash, CoreMark executes faster when running out of SRAM. The LPC553x/LPC55S3x internal memory uses a multilayer AHB matrix system that provides a separate instruction and data bus for Cortex-M33 and SRAMX bank. See Figure 15. SRAM0 to SRAM4 are on the system bus. Placing the CoreMark code and data in different SRAM banks minimizes bus contention and improves instruction and data parallelism.

It is important to minimize the flash wait states according to the MCU frequency to optimize the CoreMark score. In contrast, when performing the µA/MHz test, it is possible to save power by disabling the flash prefetch ability. The LPC553x/LPC55S3x User Manual contains more information on proper flash memory configuration, such as the minimum number of wait states allowed at a given core frequency.

The provided CoreMark framework projects include separate SRAM and flash-based projects that implement various memory optimizations.



Notes: [1] : each FlexComm includes USART, SPI, and I2C. Flexcomms 6 and 7 include 4 channel-pairs of I2S, Flexcomms 0-5 each include 1 channel-pair of I2S.

Figure 15. LPC553x/LPC55S3x AHB matrix

In both the SRAM and flash projects, there is a **COREMARK_SCORE_TEST** macro defined in `core_portme.h` that indicates whether the project is configured to execute the CoreMark benchmark or the µA/MHz test. If this macro is defined, the CoreMark score test runs. If this macro is commented out, µA/MHz test runs. Use this macro to switch between the two benchmark cases.

2.2.2 IDE optimization setting

The following optimizations are compiler-based and therefore IDE dependent. These optimizations apply to both the SRAM and flash-based projects.

2.2.2.1 Keil optimization

Two compiler optimizations can be done to improve CoreMark score and power consumption. In each Coremark source code files' Options and under the C/C++(AC6) tab, the optimization level must be set as – `mcpu=Cortex-m33 --target=arm-arm-none-eabi -Omax -g -mthumb -mfpu=fpv5-sp-d16 -mfloat-abi=hard -fno-common -ffp-mode=fast` in Misc Controls.

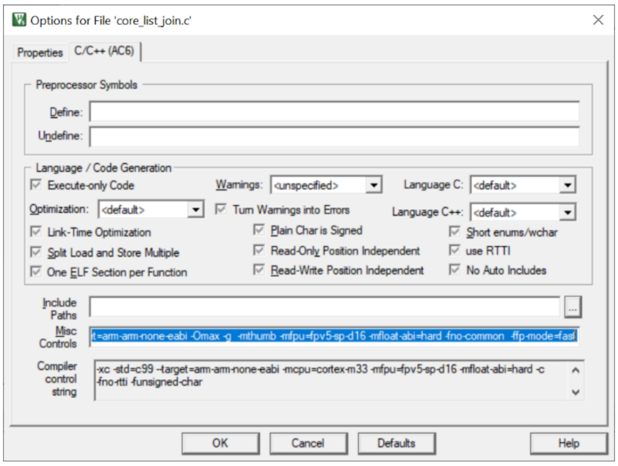


Figure 16. Keil MDK CoreMark score optimization

2.2.2.2 IAR optimization

Two compiler optimizations can be done to improve CoreMark score and power consumption. Set the optimization level to High, select **Speed** from the drop-down menu, and check the **No size constraints** checkbox.

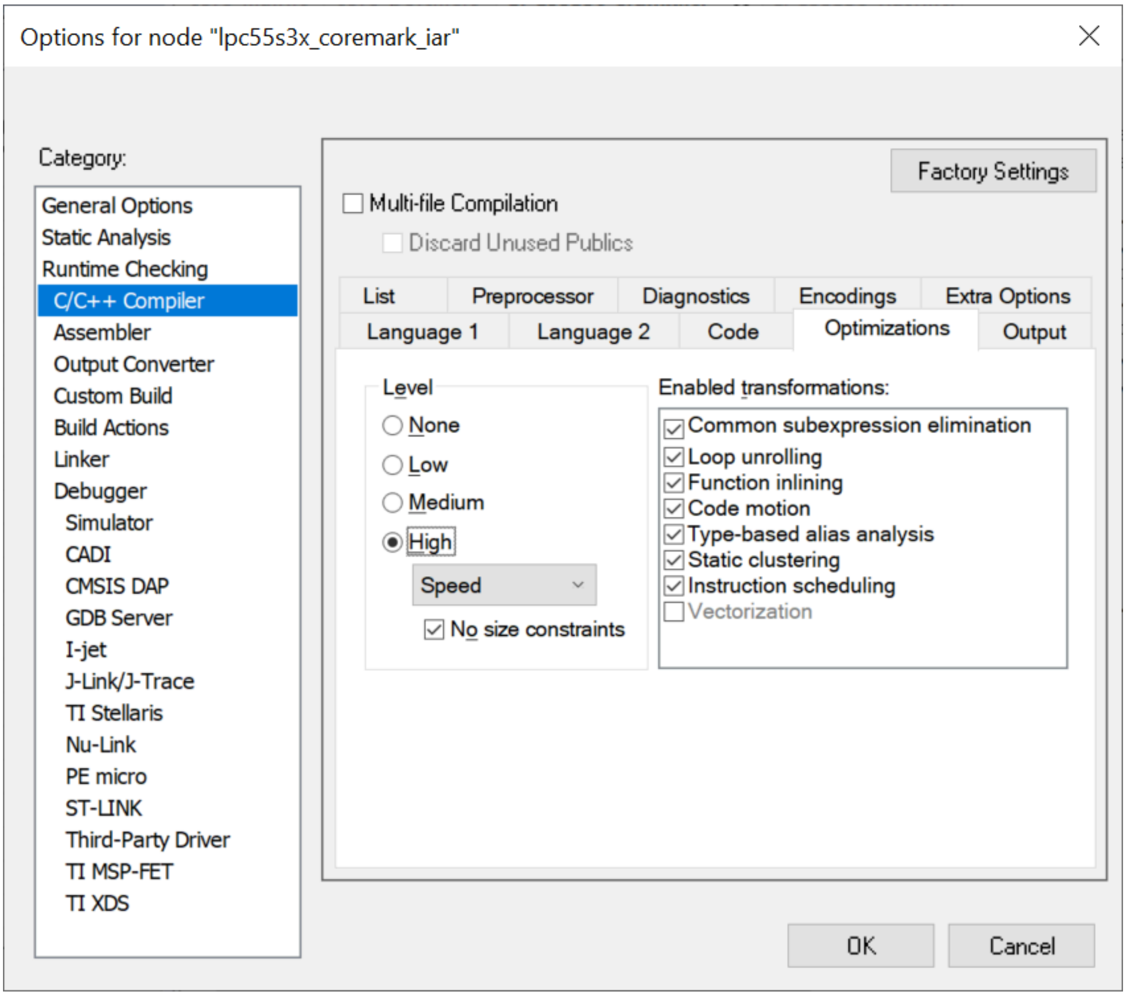


Figure 17. IAR EWARM CoreMark score optimization

2.2.2.3 MCUXpresso optimization

Two compiler optimizations can be done to improve CoreMark score and power consumption. Set the optimization level to -O3. To do this, select **Optimize most(-O3)** from the drop-down menu.

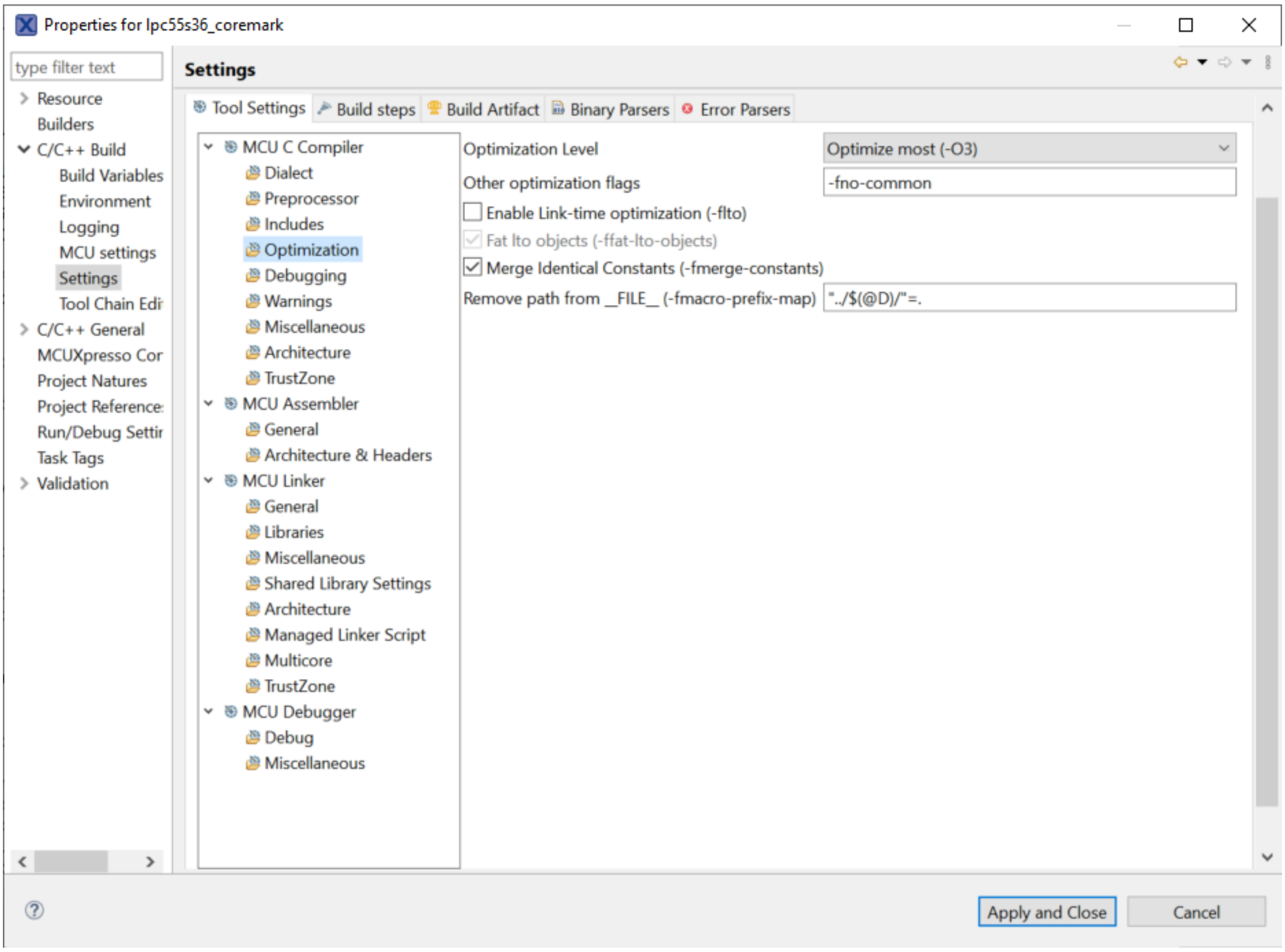


Figure 18. MCUXpresso CoreMark score optimization

3 Measuring CoreMark on board

This section describes the specifics of measuring CoreMark on the LPCXpresso55S36 board.

3.1 LPCXpresso55S36 board

The LPCXpresso55S36 board supports VCOM serial port connection via J1. To observe debug messages from the board, set the terminal program to the appropriate COM port and use the setting '115200-8-N-1-none'. To make the debug messages easier to read, the new line receive setting must be set to automatic.

3.2 Board setup

The LPCXpresso55S36 development board is used for benchmarking.

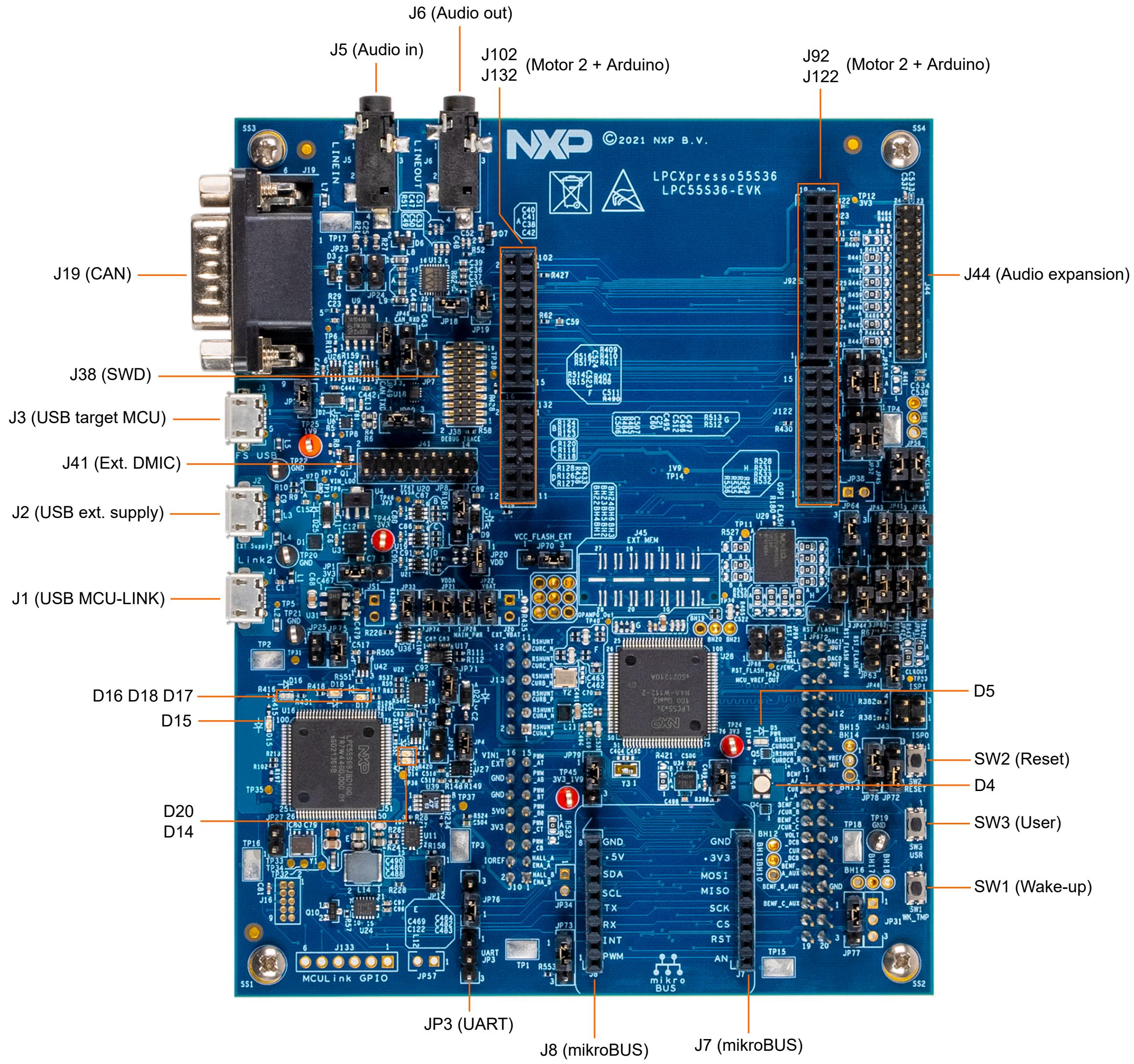


Figure 19. LPCXpresso55S36 development board

The board ships with CMSIS-DAP debug firmware programmed. For more information on CMSIS_DAP debug firmware, see the following link.

For debugging and terminal debug messages, connect a USB cable to the P6 USB connector. Board schematics are available on www.nxp.com.

3.2.1 µA/MHz measurement setup

To measure the LPC553x/LPC55S3x power consumption, remove the JP30 jumper and connect an ammeter across the JP33, as shown in the picture below.

Note: *The current data on EVK can be slightly higher than the data sheet due to more components that can consume more power.*

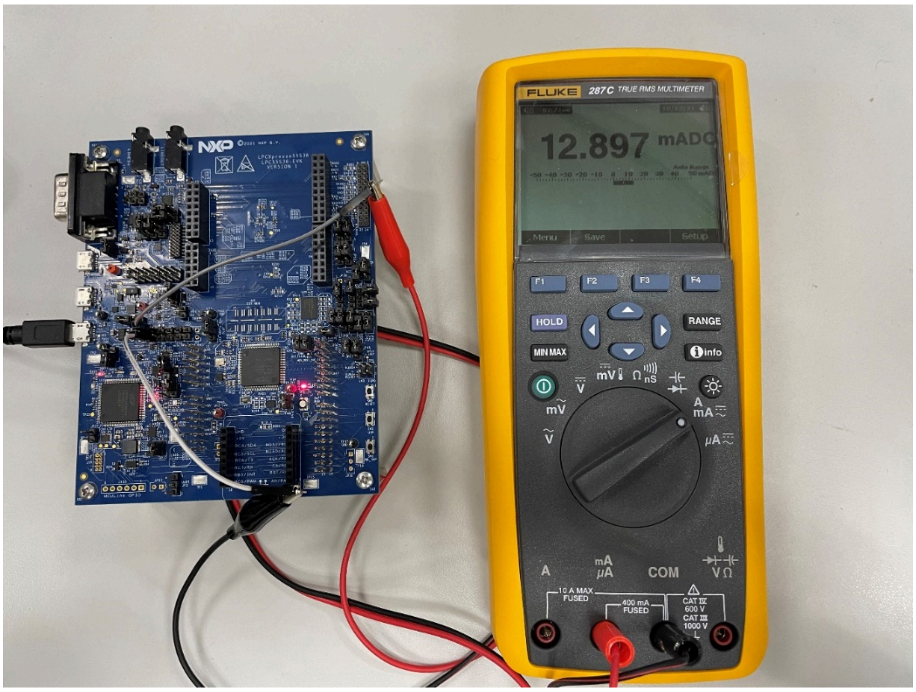


Figure 20. µA/MHz measurement setup

Users can measure the current through JP33 by a multimeter.

When performing the µA/MHz benchmark, use the J2 USB connector to provide power to the board. Also, after the µA/MHz benchmark project has been downloaded, debug the code in the IDE. Make sure to reset the code with the software. Resetting the hardware generates sink current to MCU-Link, which lowers the current measured from JP33 below the chip power consumption.

The core clock frequency can be changed by selecting a different configuration through the shell terminal by MCU UART0.

3.3 Run CoreMark code

The first step to get the CoreMark result is to connect the board's connector J1 to the PC. Then the PC recognizes the onboard MCU-Link debugger with a simulated serial port as shown in Figure 21. If the PC cannot find the serial port driver, refer to the link to update the firmware on your PC.

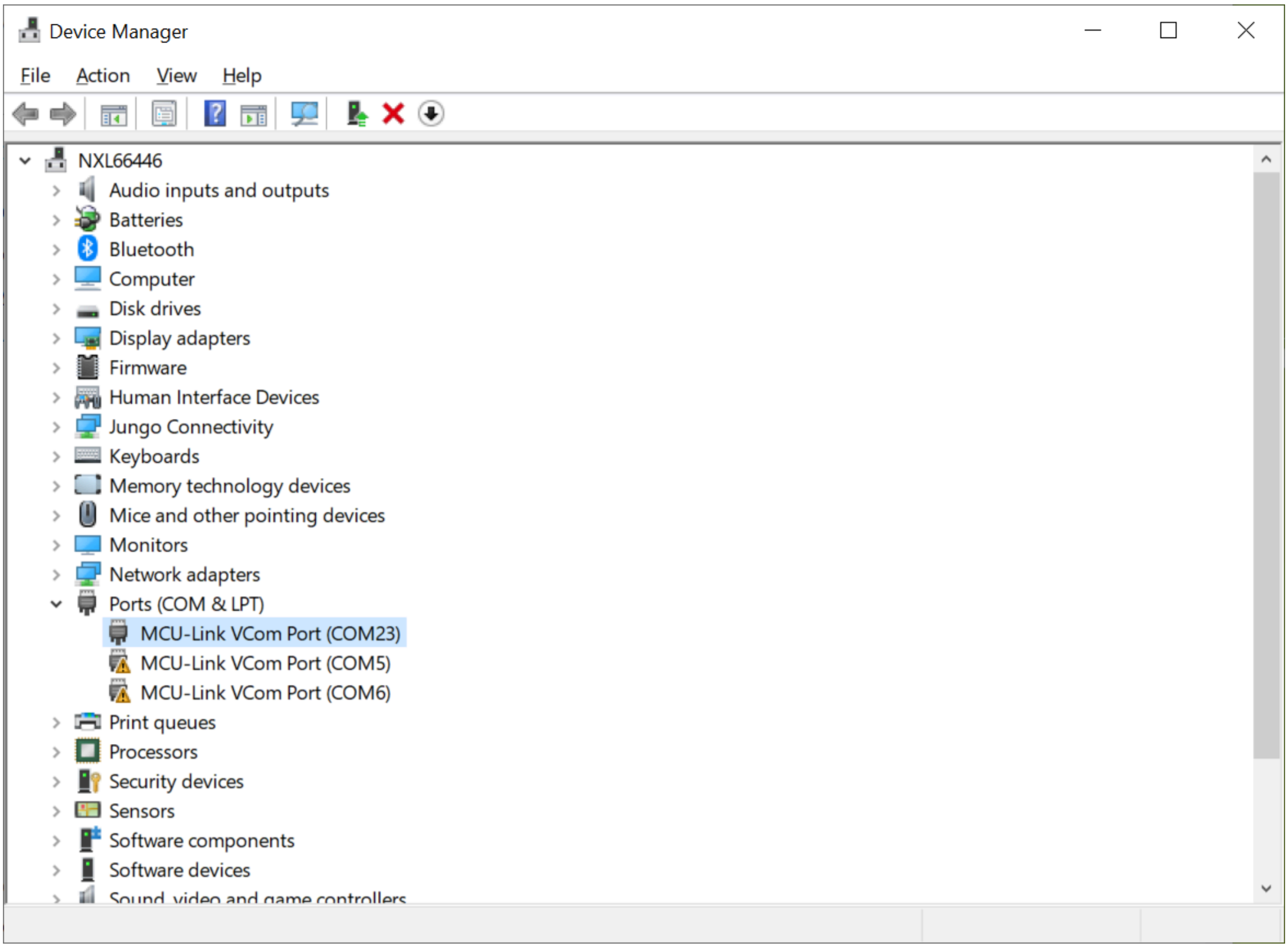


Figure 21. MCU-Link UCom Port

Open a UART debug terminal (Tera Term, putty, and so on) and configure it as 115200, 8 data bits, no parity, 1 stop bit, as shown in Figure 22.

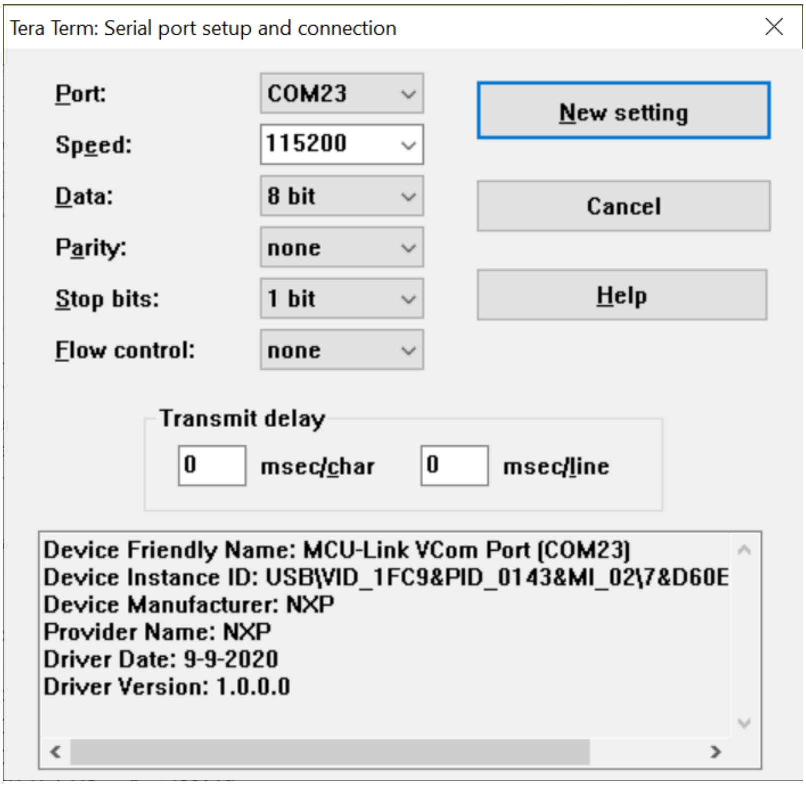


Figure 22. UART debug terminal configuration

Once the necessary CoreMark files are added into the project (by following the Chapter 2.1 instructions), compile the project and download to the LPCXpresso55S36 board:

- Click the **Reset** button as shown in Figure 23.
- The terminal displays the prompt information,
- The user can input ‘1’, ‘2’ from the PC keyboard to select the power source such as DC-DC or LDOcoreHP.
- After inputting a character, the terminal displays the frequency selection information, as in Figure 24.
- The user can input '1', '2', '3', '4' from the PC keyboard to select the frequency.
- After that, the program starts to test CoreMark, then waits 10 seconds or more.
- The CeMark benchmark prints on the terminal after a few seconds, as shown in Figure 25.

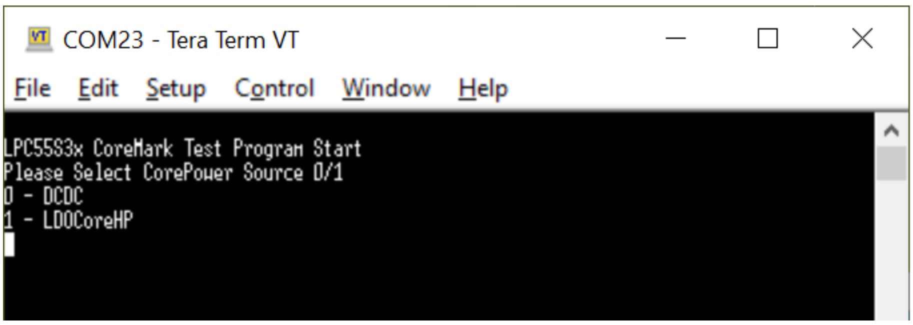


Figure 23. CoreMark test core power source choose menu

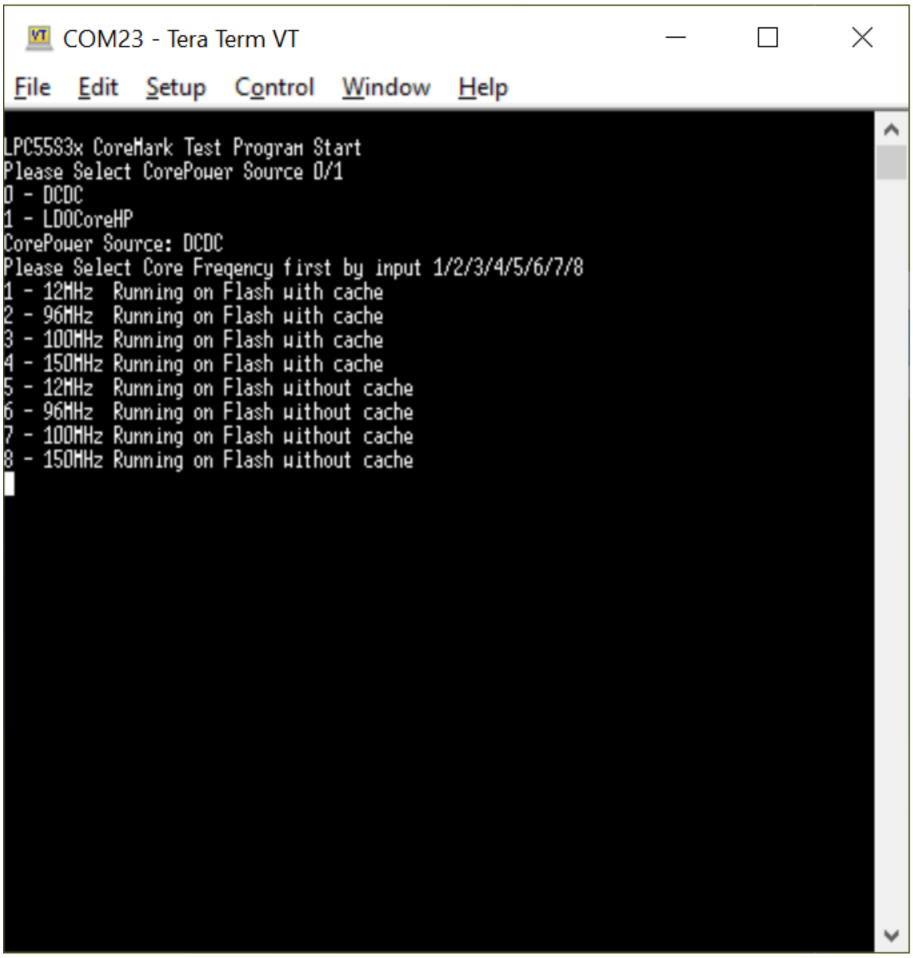


Figure 24. CoreMark test core frequency choose menu

4 Result

Figure 25 and Figure 26 show the CoreMark benchmark result when running LPC553x/LPC55S3x at 12 MHz core frequency in the MCUXpresso IDE. The CoreMark benchmark score is the number of iterations per second. The CoreMark/MHz score executing from an internal flash for this run is 35.573392/12 MHz = 2.964 CoreMark/MHz.

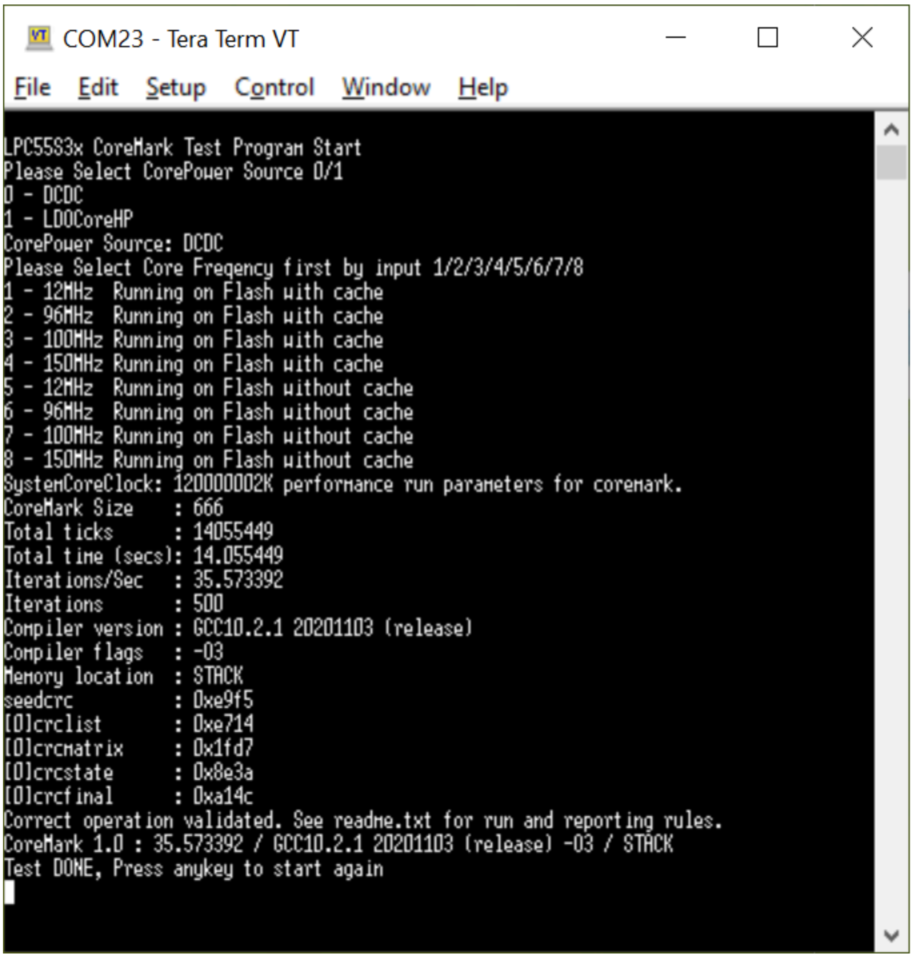


Figure 25. CoreMark result

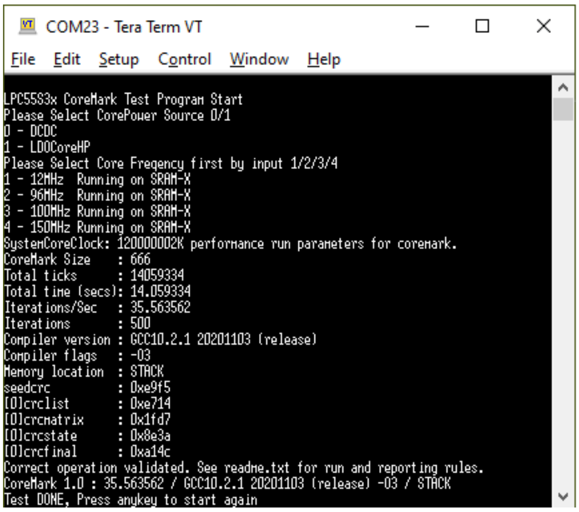


Figure 26. CoreMark result

Table 1 shows the typical CoreMark score when benchmarked on Keil MDK, IAR EWARM, and MCUXpresso IDE running from internal flash and SRAM at 12 MHz core frequency.

Table 1. LPCXpresso55S36 board CoreMark/MHz Score when 12 MHz

IDE		LDO_CORE		DC-DC	
		CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)	CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)
KEIL MDK	**cache enable**	4.076	4.086	4.077	4.085
	cache disable		3.804		3.806
IAR EWARM	**cache enable**	4.016	4.025	4.021	4.025
	cache disable		3.795		3.795
MCUXpresso	**cache enable**	2.957	2.964	2.958	2.957
	cache disable		2.839		2.839

Table 2 shows the typical CoreMark score when benchmarked on Keil MDK, IAR EWARM, and MCUXpresso IDE running from internal flash and SRAM at 96 MHz core frequency.

Table 2. LPCXpresso55S36 board CoreMark/MHz Score when 96 MHz

IDE		LDO_CORE		DC-DC	
		CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)	CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)
KEIL MDK	**cache enable**	3.963	3.986	3.964	3.988
	cache disable		2.255		2.257
IAR EWARM	**cache enable**	4.017	4.023	4.018	4.023
	cache disable		2.471		2.473
MCUXpresso	**cache enable**	2.955	2.956	2.956	2.956
	cache disable		2.024		2.026

Table 3 shows typical CoreMark score when benchmarked on Keil MDK, IAR EWARM, and MCUXpresso IDE running from internal flash and SRAM at 100 MHz core frequency.

Table 3. LPCXpresso55S36 board CoreMark/MHz Score when 100 MHz

IDE		LDO_CORE		DC-DC	
		CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)	CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)
KEIL MDK	**cache enable**	3.949	3.972	3.951	3.974
	cache disable		2.248		2.249
IAR EWARM	**cache enable**	4.003	4.01	4.005	3.863

Table 3. LPCXpresso55S36 board CoreMark/MHz Score when 100 MHz...*continued*

IDE		LDO_CORE		DC-DC	
		CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)	CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)
	cache disable		2.462		2.464
MCUXpresso	**cache enable**	2.945	2.945	2.946	2.946
	cache disable		2.017		2.019

Table 4 shows typical CoreMark score when benchmarked on Keil MDK, IAR EWARM, and MCUXpresso IDE running from internal flash and SRAM at 150 MHz core frequency.

Table 4. LPCXpresso55S36 board CoreMark/MHz Score when 150 MHz

IDE		LDO_CORE		DC-DC	
		CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)	CoreMark/MHz Score(SRAMX)	CoreMark/MHz Score(Flash)
KEIL MDK	**cache enable**	3.873	3.905	3.875	3.906
	cache disable		1.754		1.755
IAR EWARM	**cache enable**	3.999	4.004	4.001	4.007
	cache disable		1.992		1.993
MCUXpresso	**cache enable**	2.942	2.942	2.944	2.943
	cache disable		1.688		1.690

For µA/MHz, the following tables show typical results when running on the LPCXpresso55S36 board in LDO/ DCDC power source mode with VDD = 3.3 V at room temperature.

Note: *The current data on EVK can be slightly higher or lower than in the data sheet, due to EVK having more components can cause an increase in power consumption.*

Note: *The average current in 100MHz&150MHz is higher than other modes, the reason is that those two modes enable PLL, PLL causes an increase in power consumption.*

Table 5. Keil MDK µA/MHz score (LDO_CORE)

Frequency		Power consumption (mA, SRAM X)	Power consumption (mA, Flash)
12 MHz	**cache enable**	1.679	1.967
	cache disable		2.093
96 MHz	**cache enable**	8.4	10.296
	cache disable		9.819
100 MHz	**cache enable**	9.23	11.221

Table 5. Keil MDK µA/MHz score (LDO_CORE)*...continued*

Frequency		Power consumption (mA, SRAM X)	Power consumption (mA, Flash)
	cache disable		10.720
150 MHz	**cache enable**	13.555	16.696
	cache disable		14.812

Table 6. IAR EWARM µA/MHz score (LDO_CORE)

Frequency		Power Consumption (mA, SRAM X)	Power Consumption (mA, Flash)
12 MHz	**cache enable**	1.890	1.884
	cache disable		2.207
96 MHz	**cache enable**	9.090	9.998
	cache disable		9.493
100 MHz	**cache enable**	9.800	10.884
	cache disable		10.360
150 MHz	**cache enable**	14.483	16.298
	cache disable		14.344

Table 7. MCUXpresso µA/MHz score (LDO_CORE)

Frequency		Power Consumption (mA, SRAM X)	Power Consumption (mA, Flash)
12 MHz	**cache enable**	1.715	1.872
	cache disable		1.997
96 MHz	**cache enable**	8.608	9.877
	cache disable		9.435
100 MHz	**cache enable**	9.441	10.78
	cache disable		10.303
150 MHz	**cache enable**	13.877	16.093
	cache disable		14.282

Table 8. Keil MDK µA/MHz score (DC-DC)

Frequency		Power Consumption (mA, SRAM X)	Power Consumption (mA, Flash)
12 MHz	**cache enable**	0.833	0.970
	cache disable		1.032
96 MHz	**cache enable**	3.286	4.148
	cache disable		4.042

Table 8. Keil MDK µA/MHz score (DC-DC) *...continued*

Frequency		Power Consumption (mA, SRAM X)	Power Consumption (mA, Flash)
100 MHz	**cache enable**	3.538	4.446
	cache disable		4.332
150 MHz	**cache enable**	5.799	7.366
	cache disable		6.634

Table 9. IAR EWARM µA/MHz score (DC-DC)

Frequency		Power Consumption (mA, SRAM X)	Power Consumption (mA, Flash)
12 MHz	**cache enable**	0.859	0.910
	cache disable		0.978
96 MHz	**cache enable**	3.485	3.871
	cache disable		3.775
100 MHz	**cache enable**	3.745	4.146
	cache disable		4.046
150 MHz	**cache enable**	6.181	6.951
	cache disable		6.230

Table 10. MCUXpresso µA/MHz score (DC-DC)

Frequency		Power Consumption (mA, SRAM X)	Power Consumption (mA, Flash)
12 MHz	**cache enable**	0.849	0.907
	cache disable		0.966
96 MHz	**cache enable**	3.363	3.827
	cache disable		3.745
100 MHz	**cache enable**	3.617	4.110
	cache disable		4.015
150 MHz	**cache enable**	5.931	6.870
	cache disable		6.196

5 Conclusion

In this application note, three types of CoreMark benchmarking on the LPC553x/LPC55S3x are presented with different IDEs (Keil, IAR, MCUXpresso): CoreMark score, power consumption, and µA/MHz. It also describes how to optimize the benchmark results when running the benchmark from internal SRAM and flash.

The CoreMark results are measured on LPCXpresso55S36. The best CoreMark number is 4.085, achieved by using KEIL MDK (Arm Compiler 6.14) and running CoreMark from flash with cache in DC-DC power source mode. The best CoreMark power consumption in µA/MHz is 35.84. It is achieved by running CoreMark from SRAM in DC-DC power source mode when the core frequency is 100 MHz.

6 Reference

- *CoreMark Benchmarking for ARM Cortex Processors*
- *LPC5411x CoreMark Cortex-M4 Porting Guide (document AN11811)*
- *LPC55S3x User Manual (document LPC553xRM)*

7 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2023 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8 Revision history

Table 11 summarizes the revisions done to this document.

Table 11. Revision history

Revision number	Release date	Description
3	23 November 2023	• Added Section 7 • Updated Figure 15
2	21 November 2023	• Updated SDK package from "SDK2.10" to "SDK2.14" • Updated "Keil MDK v5.34" to "Keil MDK v5.37" • Updated "MCUXpresso 11.4.0 Build[6237]" to "MCUXpresso 11.8.0"
1	26 May 2022	Replaced LPC55(S)3x with LPC553x/LPC55S3x
0	23 February 2022	Initial release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents

Date of release: 23 November 2023
Document identifier: AN13579