ambiq

USER'S MANUAL

Nema GFX Benchmark Suite

A-SOCAPG-UMGA05EN v1.0

Legal Information and Disclaimers

AMBIQ MICRO INTENDS FOR THE CONTENT CONTAINED IN THE DOCUMENT TO BE ACCURATE AND RELIABLE. THIS CONTENT MAY, HOWEVER, CONTAIN TECHNICAL INACCURACIES, TYPOGRAPHICAL ERRORS OR OTHER MISTAKES. AMBIQ MICRO MAY MAKE CORRECTIONS OR OTHER CHANGES TO THIS CONTENT AT ANY TIME. AMBIQ MICRO AND ITS SUPPLIERS RESERVE THE RIGHT TO MAKE CORRECTIONS, MODIFICATIONS, ENHANCEMENTS, IMPROVEMENTS AND OTHER CHANGES TO ITS PRODUCTS, PROGRAMS AND SERVICES AT ANY TIME OR TO DISCONTINUE ANY PRODUCTS, PROGRAMS, OR SERVICES WITHOUT NOTICE.

THE CONTENT IN THIS DOCUMENT IS PROVIDED "AS IS". AMBIQ MICRO AND ITS RESPECTIVE SUPPLIERS MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY OF THIS CONTENT FOR ANY PURPOSE AND DISCLAIM ALL WARRANTIES AND CONDITIONS WITH REGARD TO THIS CONTENT, INCLUDING BUT NOT LIMITED TO, ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHT.

AMBIQ MICRO DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT OF AMBIQ MICRO COVERING OR RELATING TO THIS CONTENT OR ANY COMBINATION, MACHINE, OR PROCESS TO WHICH THIS CONTENT RELATE OR WITH WHICH THIS CONTENT MAY BE USED.

USE OF THE INFORMATION IN THIS DOCUMENT MAY REQUIRE A LICENSE FROM A THIRD PARTY UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF THAT THIRD PARTY, OR A LICENSE FROM AMBIQ MICRO UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF AMBIQ MICRO.

INFORMATION IN THIS DOCUMENT IS PROVIDED SOLELY TO ENABLE SYSTEM AND SOFTWARE IMPLEMENTERS TO USE AMBIQ MICRO PRODUCTS. THERE ARE NO EXPRESS OR IMPLIED COPYRIGHT LICENSES GRANTED HEREUNDER TO DESIGN OR FABRICATE ANY INTEGRATED CIRCUITS OR INTEGRATED CIRCUITS BASED ON THE INFORMATION IN THIS DOCUMENT. AMBIQ MICRO RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN. AMBIQ MICRO MAKES NO WARRANTY, REPRESENTATION OR GUARANTEE REGARDING THE SUITABILITY OF ITS PRODUCTS FOR ANY PARTICULAR PURPOSE, NOR DOES AMBIQ MICRO ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT, AND SPECIFICALLY DISCLAIMS ANY AND ALL LIABILITY, INCLUDING WITHOUT LIMITATION CONSEQUENTIAL OR INCIDENTAL DAMAGES. "TYPICAL" PARAMETERS WHICH MAY BE PROVIDED IN AMBIQ MICRO DATA SHEETS AND/OR SPECIFICATIONS CAN AND DO VARY IN DIFFERENT APPLICATIONS AND ACTUAL PERFORMANCE MAY VARY OVER TIME. ALL OPERATING PARAMETERS, INCLUDING "TYPICALS" MUST BE VALIDATED FOR EACH CUSTOMER APPLICATION BY CUSTOMER'S TECHNICAL EXPERTS. AMBIQ MICRO DOES NOT CONVEY ANY LICENSE UNDER NEITHER ITS PATENT RIGHTS NOR THE RIGHTS OF OTHERS. AMBIQ MICRO PRODUCTS ARE NOT DESIGNED, INTENDED, OR AUTHORIZED FOR USE AS COMPONENTS IN SYSTEMS INTENDED FOR SURGICAL IMPLANT INTO THE BODY, OR OTHER APPLICATIONS INTENDED TO SUPPORT OR SUSTAIN LIFE, OR FOR ANY OTHER APPLICATION IN WHICH THE FAILURE OF THE AMBIQ MICRO PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR. SHOULD BUYER PURCHASE OR USE AMBIQ MICRO PRODUCTS FOR ANY SUCH UNINTENDED OR UNAUTHORIZED APPLICATION, BUYER SHALL INDEMNIFY AND HOLD AMBIQ MICRO AND ITS OFFICERS, EMPLOYEES, SUBSIDIARIES, AFFILIATES, AND DISTRIBUTORS HARMLESS AGAINST ALL CLAIMS, COSTS, DAMAGES, AND EXPENSES, AND REASONABLE ATTORNEY FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PERSONAL INJURY OR DEATH ASSOCIATED WITH SUCH UNINTENDED OR UNAUTHORIZED USE, EVEN IF SUCH CLAIM ALLEGES THAT AMBIQ MICRO WAS NEGLIGENT REGARDING THE DESIGN OR MANUFACTURE OF THE PART.

Revision History

Rev	Date	Description
1.0	January 2026	Initial Release

Reference Documents

Document ID	Description
A-SOCAPG-UMGA01EN	Nema DC API Library User's Manual
A-SOCAPG-UMGA02EN	Nema GFX API Library User's Manual
A-SOCAPG-ANGA01EN	Nema DC MiP Panels Configuration Application Note
A-SOCAPG-ANGA02EN	Nema GFX Extensions TSVG Supported Elements List Application Note
A-SOCAPG-SGGA01EN	Nema Pixpresso Starting Guide
A-SOCAPG-ANGA03EN	Nema GFX API Debugging Application Note
A-SOCAPG-UMGA03EN	Nema GFX Extension Vector Graphics User's Manual
A-SOCAPG-UMGA04EN	Nema GUI Builder User's Manual
A-SOCAPG-UMGA06EN	Nema Pico Graphics Processing Unit User's Manual
A-SOCAPG-UMGA07EN	Nema Pico Platform Drivers User's Manual

@ambiq | all rights reserved

Table of Contents

SECTION

1

Overview

Nema GFX is Think Silicon's graphics API. It has been designed to accelerate high quality Graphics User Interface (GUI) development for embedded and wearable devices. Furthermore, it was designed from scratch to significantly reduce application development by providing a minimal yet very powerful set of functions.

Nema GFX Benchmark Suite offers a series of benchmarks implemented to run on top of Think Silicon's Nema GPUs, using Nema GFX. Each of these benchmarks is designed to perform continuously a single drawing operation for a predefined time frame, stressing different modules of the GPU. The main purpose of this suite is to measure the GPU rendering performance of each drawing operation. Evaluating the reported performance, one can afterwards decide whether the examined GPU along with the Nema GFX API can meet specific application requirements.

In addition, Nema GPUs are configurable at design time and therefore by running these benchmarks on different GPU configurations one can identify the optimal GPU configuration for specific applications. Performance is measured in MPixels/sec, indicating how many pixels the Nema GPU is able to draw per second.

SECTION

2

Benchmarks

The benchmarks implemented in this suite are the following:

1. **Fill Triangle**: Single color triangle rendering. Each fragment drawn leads to one memory write operation.
2. **Fill Triangle Blend**: Fills translucent single color triangles and blends them with the framebuffer. Each fragment drawn leads to one memory read (framebuffer) and one memory write operation.
3. **Fill Rectangle**: It renders single color rectangles. As in the Fill triangle benchmark, a single memory write operation is needed per fragment.
4. **Fill Rectangle Blend**: Fills translucent single color rectangles and blends them with the framebuffer. As in the Fill Triangle Blend benchmark, two memory operations are needed per fragment.
5. **Fill Quad**: Renders single color quadrilaterals.
6. **Fill Quad Blend**: Renders translucent color quadrilaterals and blends them with the framebuffer.
7. **Draw String**: Draws strings into the framebuffer. Results for this benchmark are measured in KChars/sec.
8. **Draw Line**: Draws single color lines. Each fragment drawn leads to one memory write operation.
9. **Draw Line Blend**: Translucent color lines are drawn and blended with the framebuffer. An additional memory operation is needed in order to read framebuffer for blending.
10. **Draw Rectangle**: Renders rectangle outlines. As in the Draw Line benchmark, a single memory write operation is needed per fragment.

11. **Draw Rectangle Blend**: Draws translucent rectangle outlines blending them with the framebuffer. An additional memory operation is needed in order to read framebuffer for blending.
12. **Blit**: Renders a texture by using the GPU Texture Map Unit to render textures into the framebuffer. Two memory operations are needed per fragment in order to read the textures from memory and write them back into the framebuffer.
13. **Blit Colorize**: The same as the previous except that textures are now modulated by a constant color.
14. **Blit Blend:** Textures are blended with the framebuffer before writing them back into the framebuffer. An additional read memory operation is needed (framebuffer) per fragment.
15. **Blit Blend Colorize**: The same as the previous, with the difference that textures are modulated by a constant color.
16. **Blit 90**: This benchmark rotates the textures 90 degrees clockwise. As in the Blit benchmark, two memory operations are needed per fragment in order to read the textures from memory and write them back into the framebuffer.
17. **Blit 180**: In this case, textures are rotated 180 degrees.
18. **Blit 270**: Texture are rotated 270 degrees clockwise.
19. **Blit Vertical Flip**: This benchmark flips the textures vertically before blitting them.
20. **Blit Horizontal Flip**: Unlike the previous benchmark, textures are flipped horizontally.
21. **Blit SRC Colorkeyed**: This benchmark reads textures from memory and displays them into the framebuffer after having applied source color keying. Two memory operations are needed (read/write).
22. **Blit DST Colorkeyed**: The same as before but applying destination color keying instead.
23. **Stretch Blit PS**: This benchmark renders textures into the framebuffer, stretching them using Point Sampling. As in the Blit benchmark, each fragment drawn leads to one memory write operation.
24. **Stretch Blit Blend PS**: The same as before, with the difference that textures are blended with the framebuffer. As in the Blit Blend benchmark, an additional memory operation is needed per fragment.
25. **Stretch Blit BL**: In this case, stretching is done using Bi-linear filtering method.
26. **Stretch Blit Blend BL**: The same as before and additionally textures are blended with the framebuffer.
27. **Stretch Blit Rotate**: Textures are rendered using Point Sampling filtering method and rotated by a specific angle.

@ambiq | all rights reserved

28. **Stretch Blit Rotate BL**: The same as before except for textures are rendered using Bi-linear Filtering.
29. **Textured Triangle PS**: This benchmark renders triangles with textures from memory using Point Sampling. Each fragment drawn leads to one memory read and one memory write operation.
30. **Textured Triangle Blend PS**: The same as before but textured triangles are additionally blended with the framebuffer. An additional read memory operation is needed (framebuffer) per fragment in this case.
31. **Textured Triangle BL**: Triangles with texture are rendered using Bi-linear Filtering.
32. **Textured Triangle Blend BL**: The same as before but textured triangles are additionally blended with the framebuffer.
33. **Textured Quad PS**: Renders quadrilaterals with textures from memory using Point Sampling.
34. **Textured Quad Blend PS**: The same as before but textured quadrilaterals are additionally blended with the framebuffer.
35. **Textured Quad BL**: In this case, quadrilaterals are rendered using Bi-linear Filtering.
36. **Textured Quad Blend BL**: The same as the previous, with the difference that textured quadrilaterals are additionally blended with the framebuffer.

@ambiq | all rights reserved

SECTION

3

Platforms

Nema GFX Benchmark Suite supports different platforms along with Nema GFX and Nema DC libraries. To this end, platform-dependent file **utils.c** for the desired platforms is implemented. This file contains auxiliary methods for printing results into the screen, parsing input parameters and handling elapsed time, among others. It is located in the **platforms/ PLATFORM/** directory.

SECTION

4

Compiling

Compiling Nema GFX Benchmark Suite is performed by using the GNU makefile located in the root directory. Three parameters should be properly set into the corresponding Makefile. These are the following:

- **PLATFORM** - Indicates the platform that the program will be built for. Platform name should be the same as the platform folder in which utils.c file is located.
- **COMPILER** - Must be set to the gcc backend of the corresponding platform's architecture.
- **NEMAGFX_SDK_PATH** - Indicates the corresponding root directory for the Nema GFX SDK.

Once compiled, the executable file can be found in **nemagfx_benchmarks** directory.

SECTION

5

Running the Nema GFX Benchmark Suite on Linux

After the successful compilation of the Nema GFX Benchmark Suite on a Linux based system, the user will find the executable files in **nemagfx_benchmarks** directory. These are the **nemagfx_benchmarks.static** and the **nemagfx_benchmarks.dynamic** files. Running, the statically linked executable file, is performed in a terminal window as follows (the same holds for the dynamically linked file as well):

```
./nemagfx_benchmarks.static [-m MODE] [-t TESTID] [-h]
```

5.1 MODE

Nema GFX Benchmark Suite implements three different execution modes. These execution modes give hints (if any) to the user regarding the performance of each benchmark and whether it is limited by the CPU or the GPU (CPU vs GPU bound).

CPU-GPU

The default execution mode, where each test uses both CPU and GPU. The CPU prepares different command lists that are submitted time after time to the GPU during the execution of the test. Results show the benchmark performance when both GPU and CPU are active and running simultaneously.

GPU

In this case, the CPU prepares a single command list at the beginning of each test. This very command list is submitted multiple times to the GPU for execution. Since the CPU does not create any additional command lists during the execution of the test, its utilization drops practically to zero. In this way, the performance of the GPU can be measured, without the potential bottlenecks introduced by the CPU.

CPU

The CPU creates command lists with drawing operations as normal, but they are never submitted to the GPU for execution. The CPU utilization is maxed out, while the GPU does not do any processing. As a result, the CPU can be measured and possible bottlenecks in the software stack or the CPU itself can be detected.

5.2 TESTID

Indicates the benchmark, as listed in Section Benchmarks to be executed. If TESTID is zero or not defined, all benchmarks are executed one after the other.

5.3 Help

A list with all the available benchmarks and their corresponding ID can be displayed using -h.

© 2026 Ambiq Micro, Inc. All rights reserved.

6500 River Place Boulevard, Building 7, Suite 200, Austin, TX 78730
www.ambiq.com
sales@ambiq.com
+1 512. 879.2850

A-SOCAPG-UMGA05EN v1.0
January 2026