# Application Note

**Document No.: AN1132**

**G32R5xx Zidian Application Note**

**Version: V1.1**

# 1 Introduction

This application note introduces G32R5xx Zidian, including a basic overview and software design examples.

# Contents

# 2 Zidian Overview

Zidian is a mathematical extension instruction set designed for the G32R5xx series real-time control MCU. It can effectively improve the performance of mathematical operations.

## 2.1 ICAU and FCAU

On the software side, Zidian is exposed through “driverlib/<device>/driverlib/zidian_cde.h” and “zidian_math.h”. **The header contents for G32R501 and G32R502 are not identical**: floating-point acceleration (FCAU) is shared on both devices; integer-side (ICAU) capability is device-trimmed. Before selecting APIs, check Table 1 and open the headers under the matching device directory to confirm macro names.

**Note:** Table 1 reflects current SDK headers and indicates whether this SDK provides the given wrapper/macro. Silicon instruction details follow the datasheet.

Table 1 Zidian common interface support (by function family)

| No. | Function family | Representative symbols / header | G32R501 | G32R502 |
|---|---|---|---|---|
| 1 | FCAU sine / cosine | “sinpuf32” / “sin” / “cospuf32” / “cos”; under “ZIDIAN_FCAU”: “sinf” / “cosf” | Y | Y |
| 2 | FCAU arctangent | “atan*” / “atan2*” / “quadf32” / “divf32_atan2”; “atanf” / “atan2f” | Y | Y |
| 3 | FCAU sqrt / divide / 2π scale | “sqrtf32” / “divf32” / “mpy2pif32” / “div2pif32”; “sqrtf” | Y | Y |
| 4 | VCX assembly mnemonic macros | “SINPUF32” / “COSPUF32” / “ATANPUF32” / “SQRTF32”, etc. | Y | Y |
| 5 | VCX C short-name wrappers (502 naming) | “sinpu” / “cospu” / “atanpu” / “sqrt” / “div” / “quad”, etc. | — | Y |
| 6 | CRC (“zidian_cde.h” macros) | “crc8l/h”, “crc16p1l/h”, “crc16p2l/h”, “crc32l/h” | Y | Y |
| 7 | CRC (“zidian_math.h” inline wrappers) | “crc8l32” … “crc32h32” | — | Y |

| No. | Function family | Representative symbols / header | G32R501 | G32R502 |
|---|---|---|---|---|
| 8 | 64-bit division | “div64re” / “div64uu” / “div64iu” / “div64ii”; “div6464” | — | Y |
| 9 | CX2 status / negate | “tas” / “rstatus” / “wstatus” / “neg” | Y | — |
| 10 | CX3 shift-add/sub | “sh1add*” / “sh1sub*” (12 symbols) | Y | — |
| 11 | CX3D butterfly / max | “hstas” / “hstsa” / “lstas” / “lstsa” / “max” | Y | — |
| 12 | CX3DA bit select | “hbitsel” | Y | — |
| 13 | FFT / butterfly assembly macros | “VITBM2” / “VITBM3” / “VITDHADDSUB” / “VITDL” / “VITLSEL” / “VTRACE”, etc. | Y | — |
| 14 | Complex add/MAC assembly macros | “VCADD” / “VCSUB” / “VCMPY” / “VCMAC” / “VCDADD16” / “VCDSUB16”, etc. | Y | — |

## 2.2 G32R501 / G32R502 differences

- **FCAU (floating-point) is shared:** after defining “ZIDIAN_FCAU” and including “zidian_math.h”, “sinf” / “cosf” / “atanf” / “atan2f” / “sqrtf” can be redirected to Zidian-accelerated implementations.
- **G32R501 integer focus is SIMD / FFT / complex:** “zidian_cde.h” provides CX3/CX3D (for example “sh1add”, “hstas”, “max”) and assembly macros such as “VCMPY”, “VCADD”, and “VITBM” for FFT and complex workloads.
- **G32R502 current SDK does not provide FFT / complex macro wrappers:** the 502 “zidian_cde.h” has **no** “sh1add”, “hstas”, or “max”, and no “VCMPY” / “VCADD” / “VITBM” symbols. For SIMD FFT or complex math, **do not** call those 501-style macros; use ordinary software algorithms, or confirm whether another library is available.
- **G32R502 integer focus is CRC wrappers and 64-bit division:** both devices expose “crc*” macros in “zidian_cde.h”; **only 502** further wraps “crc*32” in “zidian_math.h” and provides “div64*” / “div64*64”.
- **Use device-specific paths:** G32R501 uses “driverlib/g32r501/driverlib/zidian_*.h”; G32R502 uses “driverlib/g32r502/driverlib/zidian_*.h”. Do not mix them.

In G32R5xx software, extensions are commonly grouped as CDE and FPCDE.

For CDE extensions, instructions operate on general-purpose registers to improve integer performance. This capability set is the integer computation acceleration unit (ICAU). **Exposure differs by device:**

- G32R501: CRC macros plus SIMD-oriented shift-add/sub, butterfly, and FFT/complex wrappers (see Table 1 rows 9–14).
- G32R502: CRC macros and 64-bit division wrappers; **without** the FFT / complex / CX3·CX3D application macros in Table 1.

For FPCDE extensions, instructions operate on floating-point registers to improve floating-point performance (trigonometric functions, square root, division, and related operations). This capability set is the floating-point computation acceleration unit (FCAU). **Both G32R501 and G32R502 support FCAU** (see Table 1 rows 1–5).

## 2.3 CDE built-in functions

By introducing standardized CDE built-in functions as part of the Arm C language extension, Tables 2 and 3 list the built-in functions supported by Zidian.

Table 2 Built-in functions supported by ICAU

| Instruction type | Built-in functions |
|---|---|
| CX2 | uint32_t __arm_cx2(int coproc, uint32_t n, uint32_t imm); |
| CX2A | uint32_t __arm_cx2a(int coproc, uint64_t acc, uint32_t n, uint32_t imm); |
| CX2DA | uint64_t __arm_cx2da(int coproc, uint64_t acc, uint32_t n, uint32_t imm); |
| CX3 | uint32_t __arm_cx3(int coproc, uint32_t n, uint32_t m, uint32_t imm); |
| CX3D | uint64_t __arm_cx3d(int coproc, uint32_t n, uint32_t m, uint32_t imm); |
| CX3DA | uint64_t __arm_cx3da(int coproc, uint64_t acc, uint32_t n, uint32_t m, uint32_t imm); |
| CX2 | uint32_t __arm_cx2(int coproc, uint32_t n, uint32_t imm); |

Table 3 Built-in functions supported by FCAU

| Instruction type | Built-in functions |
|---|---|
| VCX2 | uint32_t __arm_vcx2(int coproc, uint32_t n, uint32_t imm); |
| VCX3 | uint32_t __arm_vcx3(int coproc, uint32_t n, uint32_t m, uint32_t imm); |

Through these built-in functions, G32R5xx renames the instructions in Zidian. Table 4 uses VCX3 as an example and lists renamed VCX3 functions in Zidian.

Table 4 VCX3 renamed functions

| Instruction | Zidian function |
|---|---|
| VCX3 0, Sd, Sn, Sm, #0x0 | DIVF32 Sd, Sn, Sm |
| VCX3 0, Sd, Sn, Sm, #0x1 | QUADF32 Sd, Sn, Sm |
| VCX3 0, Sd, Sn, Sm, #0x2 | DIVF32_ATAN2 Sd, Sn |

# 3 Using Zidian to accelerate computation

## 3.1 zidian_math.h

In “zidian_cde.h”, built-in functions are renamed through macros. To use Zidian for regular mathematical operations in driverlib, include “zidian_math.h”. See the "Combining driverlib with Zidian" section for details.

“zidian_math.h” repackages commonly used mathematical functions. Table 5 lists the mathematical functions packaged in “zidian_math.h”.

Table 5 zidianmath functions

| Zidian function | Description |
|---|---|
| sinpuf32 | Calculate the sine value |
| sin | Normalize to the [0, 2π) interval, then calculate sine |
| cospuf32 | Calculate the cosine value |
| cos | Normalize to the [0, 2π) interval, then calculate cosine |
| atanpuf32 | Calculate the arctangent value |
| atan | Calculate the arctangent value |
| mpy2pif32 | Single-precision floating-point multiply |
| div2pif32 | Single-precision floating-point divide |
| sqrtf32 | Square root of a single-precision floating-point number |
| divf32 | Floating-point division |
| quadf32 | Calculate the quadrant values of X and Y |
| divf32_atan2 | Calculate the ratio of X to Y |
| atan2puf32 | atan2 |
| atan2 | atan2 |

## 3.2 Compiler support

The SDK currently provides MDK-ARM and IAR environments. Enabling Zidian support differs slightly by compiler.

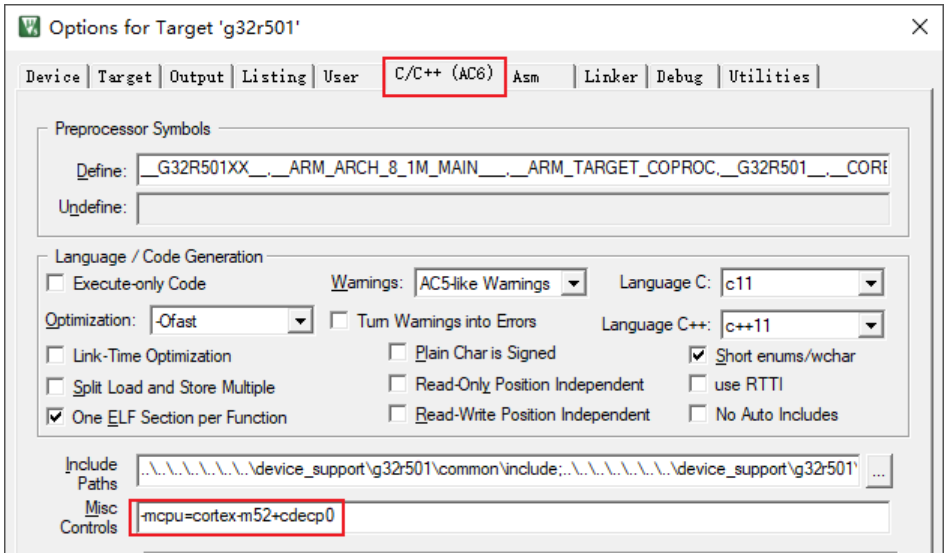
**Note:** The figures in this section use G32R501 as the example.

### 3.2.1 MDK-ARM (AC6)

Under C/C++ (AC6) → Misc Controls, add the following option for the device in use:

- G32R501: “-mcpu=cortex-m52+cdecp0”
- G32R502: “-mcpu=cortex-m52+nomve.fp+nofp.dp+cdecp0”

Figure 1 Enabling CDE in AC6



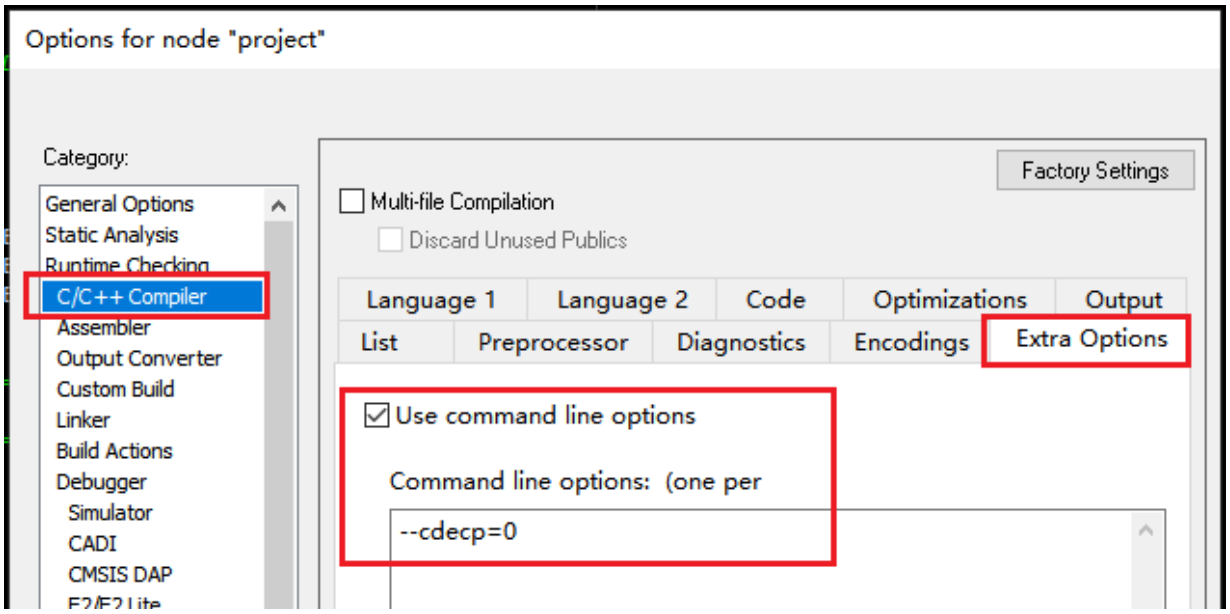

### 3.2.2 IAR (ICC)

IAR Embedded Workbench for Arm support is compilation support.

Enable it under C/C++ Compiler → Extra Options: select Use command line options, then add:

- G32R501 and G32R502: “--cdecp=0”

Figure 2 Enabling CDE compilation in ICC




## 3.3 Software design examples

This section uses the MDK-ARM environment as an example of accelerating computation with Zidian.

### 3.3.1 Combining driverlib with Zidian

Steps for using Zidian in an SDK driverlib example:

1. Enable compiler support.
2. Add the “ZIDIAN_FCAU” macro under the project C/C++ options to redirect trigonometric functions supported by Zidian.
3. Include “zidian_math.h” in source files that need Zidian.

This note follows the SDK driverlib example “zidian_ex1_math”.

The example uses Zidian to exercise “sqrt”, “sin”, “cos”, “atan”, and “atan2”. Because these functions are already packaged in “zidian_math.h”, include that header and call them directly:

```
# include "zidian_math.h"

// Main
void example_main(void)
{
    volatile float x, y;
    uint16_t i;
    // Test1 sqrtf
```

```
    x = 0.81f;
    GET_DWT_CYCLE_COUNT(dwtCycleCounts[0], trace1Result[0] = sqrtf(x));
    for (i = 0; i < COUNT; i++)
    {
        trace1Result[i] = sqrtf(x);
    }
    // Test2 sinf
    x = PI / 2;
    GET_DWT_CYCLE_COUNT(dwtCycleCounts[1], trace2Result[1] = sinf(x));
    for (i = 0; i < COUNT; i++)
    {
        trace2Result[i] = sinf(x);
    }
    // Test3 cosf
    x = PI / 4;
    GET_DWT_CYCLE_COUNT(dwtCycleCounts[2], trace3Result[2] = cosf(x));
    for (i = 0; i < COUNT; i++)
    {
        trace3Result[i] = cosf(x);
    }
    // Test4 atanf
    x = 5.0f;
    GET_DWT_CYCLE_COUNT(dwtCycleCounts[3], trace4Result[3] = atanf(x));
    for (i = 0; i < COUNT; i++)
    {
        trace4Result[i] = atanf(x);
    }
    // Test5 atan2f
    x = 5.0f;
    y = 10.0f;
    GET_DWT_CYCLE_COUNT(dwtCycleCounts[4], trace5Result[4] = atan2f(y, x));
    for (i = 0; i < COUNT; i++)
    {
        trace5Result[i] = atan2f(y, x);
    }
    // Loop
    for (;;)
    {
    }
}
```

### 3.3.2 Combining the math library with Zidian

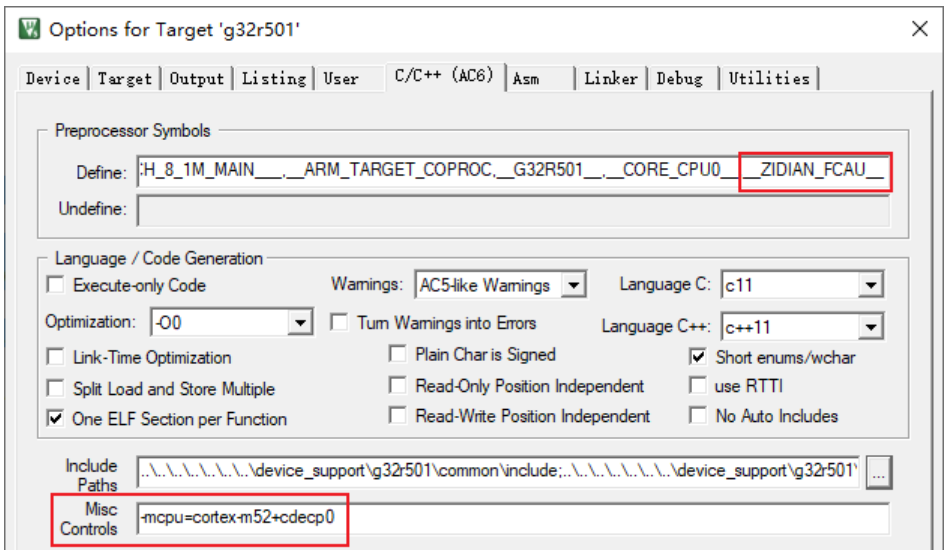
To accelerate math-library calls with Zidian:

1. Enable compiler support.
2. Add the “ZIDIAN_FCAU” macro under the project C/C++ options to redirect math-library functions supported by Zidian.

Take an FPUfastRTS example as reference. Because “sin”, “cos”, and related functions are packaged in FPUfastRTS, calling them without “ZIDIAN_FCAU” uses the FPUfastRTS implementations.

To use Zidian acceleration, add the macro in the corresponding project C/C++ options, then call the target functions.

Figure 3 Using Zidian acceleration in the math library




After enabling compiler support and adding “ZIDIAN_FCAU”, call the functions directly:

```
in.f32 = test_input[i];
// Run the calculation function and measure the DWT cycle count simultaneously
GET_DWT_CYCLE_COUNT(dwtCycleCounts[0], out.f32 = atanf(in.f32));
test_output[i] = out.f32;
```

# 4 Revision

Table 6 Document revision

| Date | Version | Description |
|---|---|---|
| 2025.1 | 1.0 | ● Initial release |
| 2026.8 | 1.1 | ● Listed applicable products as G32R501 and G32R502<br>● Refined the comparison table by function family from “zidian_cde.h” / “zidian_math.h”<br>● Clarified 501-only FFT/complex/CX3·CX3D, 502-only div 64 and CRC wrappers, and shared FCAU plus CRC macros. |

# Statement

This document is formulated and published by Geehy Semiconductor Co., Ltd. (hereinafter referred to as “Geehy”). The contents in this document are protected by laws and regulations of trademark, copyright and software copyright. Geehy reserves the right to make corrections and modifications to this document at any time. Read this document carefully before using Geehy products. Once you use the Geehy product, it means that you (hereinafter referred to as the “users”) have known and accepted all the contents of this document. Users shall use the Geehy product in accordance with relevant laws and regulations and the requirements of this document.

1. Ownership

This document can only be used in connection with the corresponding chip products or software products provided by Geehy. Without the prior permission of Geehy, no unit or individual may copy, transcribe, modify, edit or disseminate all or part of the contents of this document for any reason or in any form.

The “极海” or “Geehy” words or graphics with “®” or “TM” in this document are trademarks of Geehy. Other product or service names displayed on Geehy products are the property of their respective owners.

2. No Intellectual Property License

Geehy owns all rights, ownership and intellectual property rights involved in this document.

Geehy shall not be deemed to grant the license or right of any intellectual property to users explicitly or implicitly due to the sale or distribution of Geehy products or this document.

If any third party’s products, services or intellectual property are involved in this document, it shall not be deemed that Geehy authorizes users to use the aforesaid third party’s products, services or intellectual property. Any information regarding the application of the product, Geehy

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, unless otherwise agreed in sales order or sales contract.

3. Version Update

Users can obtain the latest document of the corresponding models when ordering Geehy products.

If the contents in this document are inconsistent with Geehy products, the agreement in the sales order or the sales contract shall prevail.

4. Information Reliability

The relevant data in this document are obtained from batch test by Geehy Laboratory or cooperative third-party testing organization. However, clerical errors in correction or errors caused by differences in testing environment may occur inevitably. Therefore, users should understand that Geehy does not bear any responsibility for such errors that may occur in this document. The relevant data in this document are only used to guide users as performance parameter reference and do not constitute Geehy's guarantee for any product performance.

Users shall select appropriate Geehy products according to their own needs, and effectively verify and test the applicability of Geehy products to confirm that Geehy products meet their own needs, corresponding standards, safety or other reliability requirements. If losses are caused to users due to user's failure to fully verify and test Geehy products, Geehy will not bear any responsibility.

5. Legality

USERS SHALL ABIDE BY ALL APPLICABLE LOCAL LAWS AND REGULATIONS WHEN USING THIS DOCUMENT AND THE MATCHING GEEHY PRODUCTS. USERS SHALL UNDERSTAND THAT THE PRODUCTS MAY BE RESTRICTED BY THE EXPORT, RE-EXPORT OR OTHER LAWS OF THE COUNTRIES OF THE PRODUCTS SUPPLIERS, GEEHY, GEEHY

DISTRIBUTORS AND USERS. USERS (ON BEHALF OR ITSELF, SUBSIDIARIES AND AFFILIATED ENTERPRISES) SHALL AGREE AND PROMISE TO ABIDE BY ALL APPLICABLE LAWS AND REGULATIONS ON THE EXPORT AND RE-EXPORT OF GEEHY PRODUCTS AND/OR TECHNOLOGIES AND DIRECT PRODUCTS.

6. Disclaimer of Warranty

THIS DOCUMENT IS PROVIDED BY GEEHY "AS IS" AND THERE IS NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

GEEHY'S PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED FOR USE AS CRITICAL COMPONENTS IN MILITARY, LIFE-SUPPORT, POLLUTION CONTROL, OR HAZARDOUS SUBSTANCES MANAGEMENT SYSTEMS, NOR WHERE FAILURE COULD RESULT IN INJURY, DEATH, PROPERTY OR ENVIRONMENTAL DAMAGE.

IF THE PRODUCT IS NOT LABELED AS "AUTOMOTIVE GRADE," IT SHOULD NOT BE CONSIDERED SUITABLE FOR AUTOMOTIVE APPLICATIONS. GEEHY ASSUMES NO LIABILITY FOR THE USE BEYOND ITS SPECIFICATIONS OR GUIDELINES.

THE USER SHOULD ENSURE THAT THE APPLICATION OF THE PRODUCTS COMPLIES WITH ALL RELEVANT STANDARDS, INCLUDING BUT NOT LIMITED TO SAFETY, INFORMATION SECURITY, AND ENVIRONMENTAL REQUIREMENTS. THE USER ASSUMES FULL RESPONSIBILITY FOR THE SELECTION AND USE OF GEEHY PRODUCTS. GEEHY WILL BEAR NO RESPONSIBILITY FOR ANY DISPUTES ARISING FROM THE SUBSEQUENT DESIGN OR USE BY USERS.

7. Limitation of Liability

IN NO EVENT, UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL GEEHY OR ANY OTHER PARTY WHO PROVIDES THE DOCUMENT AND PRODUCTS

"AS IS", BE LIABLE FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE DOCUMENT AND PRODUCTS (INCLUDING BUT NOT LIMITED TO LOSSES OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY USERS OR THIRD PARTIES). THIS COVERS POTENTIAL DAMAGES TO PERSONAL SAFETY, PROPERTY, OR THE ENVIRONMENT, FOR WHICH GEEHY WILL NOT BE RESPONSIBLE.

8. Scope of Application

The information in this document replaces the information provided in all previous versions of the document.