Freescale Semiconductor, Inc.
Application Note

Document Number: AN5145
Rev. 0, 06/2015

Use of Memory-Mapped Divide and Square Root (MMDVSQ) Peripheral on Kinetis KV1x

1. Introduction

Many real-time applications require mathematic functions for calculation. However, not all cores are equipped with the instruction set to cover all mathematic functions. In such use cases, the user must emulate the function through time-consuming calculations.

The division and square root functions are not present in the instruction set of the ARM® Cortex®-M0+ core. The KV1x family of Freescale Kinetis microcontrollers compensates for the absence of these instructions by providing Memory-Mapped Divide and Square Root peripheral (MMDVSQ). This peripheral is capable of signed and/or unsigned integer division (32/32) and square root calculation.

2. Peripheral

The peripheral contains several registers which need to be loaded with input values (numerator, denominator, and square). This results in the division or square root calculation of store, configuration, and status registers.

The following sections describe how to use the peripheral for division and square root calculation.

Contents

3. Division

The MMDVSQ_CSR register must be configured. There are several bits that control the mechanism:

- DFS – Disable Fast Start: this bit controls whether the division starts immediately (cleared) after the divisor is stored or starts manually (set) by setting the SRT bit.
- DZE – Divide-by-Zero-Enable: this bit controls the behavior of the division by zero. If cleared, the resulting register content is read after the division by zero. If set, a divide-by-zero signal is generated at the attempt of the result register reading, after division by zero.
- REM – Remainder: this bit controls whether the result after the division is the quotient (cleared) or the remainder (set).
- USGN – Unsigned: this bit sets if the division is signed (cleared) or unsigned (set).
- SRT – Start: writing 1 to this bit initiates division.

When the control bits are properly set, the input registers can be filled by the input values:

- MMDVSQ_DEN – Dividend Register: this 32-bit register stores the numerator of the division.
- MMDVSQ_DSOR – Divisor Register: this 32-bit register stores the denominator of the division. If the DFS bit of the MMDVSQ_CSR register is not set, writing to this register initiates the division. Otherwise, division is started by the SRT bit.

When division has been initiated, check certain flags in the MMDVSQ_CSR register to know when the result is ready:

- BUSY – Busy: if set, this flag indicates that a calculation is in progress and it is necessary to wait.
- DIV – Divide: if set, this flag indicates that the performing operation is division.
- DZ – Divide-by-Zero: if set, this flag indicates that in the last division, the denominator was equal to zero.

As soon as the BUSY flag is cleared, the result is ready in the result register FSLESL_MMDVSQ_RES.

An example of a signed division of two numbers in C-language is shown below. The numerator and denominator are input variables, and the result is the output variable.

```
MMDVSQ_CSR = 0x00000000;
MMDVSQ_DEND = num;
MMDVSQ_DSOR = denom;
```

4. Square root

There are no control bits for the square root algorithm, so it is not necessary to write anything to the MMDVSQ_CSR register.

The square root calculation initiates by writing its radicand into the MMDVSQ_RCND register.

When the calculation has been initiated, certain flags in the MMDVSQ_CSR register should be checked to know that the result is ready:

- BUSY – Busy: if set, this flag indicates that a calculation is in progress and it is necessary to wait.
- SQRT – Square root: if set, this flag indicates that the performing operation is the square root.

As soon as the BUSY flag is cleared, the result is ready in the result register FSLESL_MMDVSQ_RES.

An example of a square root calculation in C-language is shown below. The radicand is the input variable, and the result is the output variable. This example is shown below.

```
MMDVSQ_CSR = 0x00000000;
MMDVSQ_RCND = rad;
res = MMDVSQ_RES;
```

5. Conclusion

This peripheral can save the cycle count for the applications where the division and square root functions are called.

Wherever the peripheral is used, the MMDVSQ peripheral initiates a calculation and generates an interrupt. The calculation process, which is running, aborts, and a new one starts. The application will read an incorrect result because when the application returns from the interrupt, it does not understand that the calculation was previously interrupted. For such use cases, a software state machine should be created.

The MMDVSQ peripheral is used by the Freescale Embedded Software Libraries (FSLESL) available for download on www.freescale.com/fslesl. This library benefits from this peripheral in several algorithms. These algorithms also take care of the previously mentioned interrupt issue.

6. Revision History

This table summarizes revisions to this document.

Table 1. **Revision history**

Revision number	Date	Substantive changes
0	06/2015	Initial release

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AN5145
Rev. 0
06/2015