Application note

DA14580 Using Timer0

AN-B-025

Abstract

This document is a programming guideline describing a minor limitation when using Timer0 of the DA14580 and how to overcome it.

Contents

Contents 2

1 Terms and definitions 3

2 References 3

3 Introduction 4

4 Interrupt Execution in DA14580 4

5 Faulty Timer0 Handler Execution 4

6 Workaround 4

7 Revision history 6

1 Terms and definitions

BTLE	Bluetooth Low Energy
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read Only Memory
NVIC	Nested Vector Interrupt Controller
OTP	One Time Programmable memory
SysRAM	System RAM
RetRAM	Retention RAM

2 References

3 Introduction

Timer0 is a 16-bit general purpose timer with PWM output capability. Timer0 can be programmed to generate a periodic interrupt (SWTIM). The Systick timer of the ARM Contrex-M0 and the Timer0 are the only timers available to the application for periodic interrupt generation. The use of SWTIM by an application comes with a minor limitation which will be explained in this document. The problem appears when the System RAM is not remapped to address zero.

4 Interrupt Execution in DA14580

DA14580 provides a mechanism for mapping to address zero (0x00000000) one of the ROM, OTP, SysRAM or RetRAM memories. This mechanism is controlled by the SYS_CTRL_REG and during reset the ROM region occupies the memory area beginning at address zero. Since the interrupt vector table is fixed at address zero, the interrupt vector table used by the processor is the one of the BootROM code if the System RAM is not remapped.

To allow the application to set up its own interrupts, the BootROM code provides a generic interrupt handler for the execution of the majority of interrupts. This interrupt handler will read the interrupt vector table of the application and jump to the corresponding interrupt service routine. This procedure is followed for all interrupts except for two: the SWTIM and the RESET.

Not allowing the RAM version of the Reset Vector to be executed, ensures that the DA14580 will be able to properly perform a “warm” reset and boot from the BootROM even if the RAM contents are corrupted. For this reason, it is strongly suggested not to remap the System RAM to address zero.

Timer0 is used in the BootROM code to generate timeouts and, therefore, a SWTIM interrupt service routine exists in ROM. To allow the application to use Timer0, a small piece of BootROM code checks if we are booting or not and selects the appropriate interrupt handler.

This check is performed by looking at the code address that has been interrupted when the SWTIM interrupt was issued. If the Program Counter was pointing at the BootROM code, it is assumed that the chip is booting and the BootROM SWTIM interrupt service routine is selected for execution; otherwise, the application’s SWTIM handler is executed.

5 Faulty Timer0 Handler Execution

The logic used to distinguish which Timer0 service routine will be executed, has a minor pitfall when an application is running in System RAM and the SWTIM interrupt has a higher priority over another interrupt.

If the lower-priority interrupt hits then the BootROM code will be executed to read the interrupt vector table of the application and branch to the proper interrupt service routine, as was described above. If the SWTIM interrupt is issued while the BootROM code executes this procedure for the lower priority interrupt, then the program counter will be pointing in the BootROM generic interrupt handler and the SWTIM‘s interrupt service routine of the BootROM code will be executed instead of the application’s.

The BootROM’s SWTIM interrupt service routine decrements and checks a variable. When it becomes zero, it stops the Timer0. If no action is taken, the user will experience missing ticks, memory corruption or sudden stops of the Timer0.

6 Workaround

Suggested solution is to set the SWTIM interrupt to the lowest possible priority, which ensures that it will never stop the execution of another interrupt. The following piece of code must be used in the setup of the SWTIM before enabling the SWTIM interrupt:

```
NVIC_SetPriority(SWTIM_IRQn, 3);
```

This is properly handled by the SWTIM driver.

Another option is to remap System RAM to zero, in which case the BootROM interrupt service routines are never called. This is not recommended for the reasons previously mentioned in this document.

7 Revision history

Revision	Date	Description
1.0	15-Jul-2014	Initial version.
1.1	23-Jul-2014	NVIC_SetPriority macro parameter changed to the lowest priority value.

Status definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

Disclaimer

Information in this document is believed to be accurate and reliable. However, Dialog Semiconductor does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information. Dialog Semiconductor furthermore takes no responsibility whatsoever for the content in this document if provided by any information source outside of Dialog Semiconductor.

Dialog Semiconductor reserves the right to change without notice the information published in this document, including without limitation the specification and the design of the related semiconductor products, software and applications.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. Dialog Semiconductor makes no representation or warranty that such applications, software and semiconductor products will be suitable for the specified use without further testing or modification. Unless otherwise agreed in writing, such testing or modification is the sole responsibility of the customer and Dialog Semiconductor excludes all liability in this respect.

Customer notes that nothing in this document may be construed as a license for customer to use the Dialog Semiconductor products, software and applications referred to in this document. Such license must be separately sought by customer with Dialog Semiconductor.

All use of Dialog Semiconductor products, software and applications referred to in this document are subject to Dialog Semiconductor's Standard Terms and Conditions of Sale, unless otherwise stated.

RoHS Compliance

Dialog Semiconductor complies to European Directive 2001/95/EC and from 2 January 2013 onwards to European Directive 2011/65/EU concerning Restriction of Hazardous Substances (RoHS/RoHS2).
Dialog Semiconductor's statement on RoHS can be found on the customer portal https://support.diasemi.com/. RoHS certificates from our suppliers are available on request.

Contacting Dialog Semiconductor

Germany Headquarters
Dialog Semiconductor GmbH
Phone: +49 7021 805-0

United Kingdom
Dialog Semiconductor (UK) Ltd
Phone: +44 1793 757700

The Netherlands
Dialog Semiconductor B.V.
Phone: +31 73 640 8822

Email:
enquiry@diasemi.com

North America
Dialog Semiconductor Inc.
Phone: +1 408 845 8500

Japan
Dialog Semiconductor K. K.
Phone: +81 3 5425 4567

Taiwan
Dialog Semiconductor Taiwan
Phone: +886 281 786 222

Web site:
www.dialog-semiconductor.com

Singapore
Dialog Semiconductor Singapore
Phone: +65 64 849929

China
Dialog Semiconductor China
Phone: +86 21 5178 2561

Korea
Dialog Semiconductor Korea
Phone: +82 2 3469 8291