Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.



THIS SPEC IS OBSOLETE

Spec No: 002-04934

Spec Title: AN204934 - Note on Calculating Reload Values FFMC8L

Replaced by: None

Note on Calculating Reload Values FFMC8L

This application note describes how to generate circular interrupts with a fixed reload time and also shows how to calculate the correct reload value taking in account the produced extra code.

OBSOLETE

1 Introduction

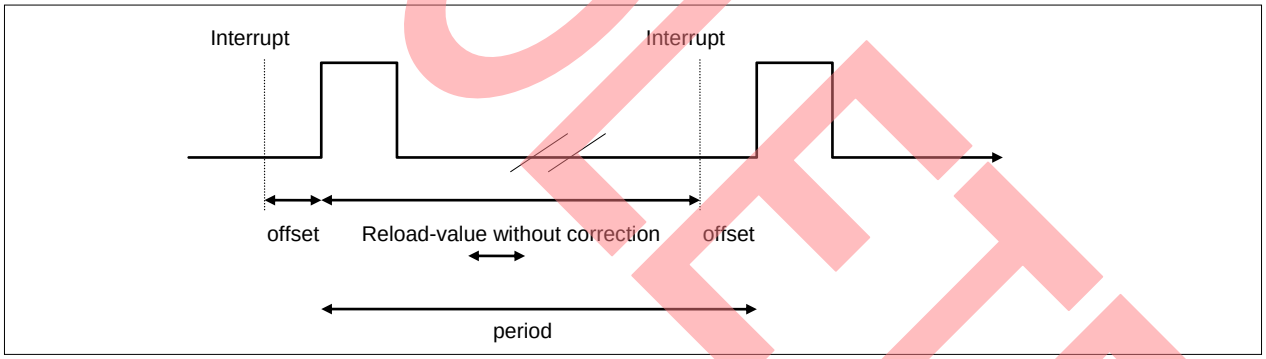
When using timers in reload mode, the main intention is to generate circular interrupts with a fixed reload time. But when interrupt service routines are called, a certain overhead will be produced by the compiler (context save!). This note shows how to calculate the correct reload value taking in account the produced extra code.

2 Calculating Reload Values FFMC8L

In this example, a rectangular signal with a frequency of 1,5 kHz should be produced on pin 40. The reload value has to be calculated to obtain this period. The positive width of the signal is fixed and much smaller than the period itself.

The diagram below shows the signal and the ISR procedure: When the interrupt is signaled and processed immediately, which would assume a certain priority and no other interrupt currently active, the context will be saved on the stack after the ISR-subroutine is called. This context save is produced by the compiler. After that the intended taks can be processed – in this case a signal on pin 40 with a definite width.

Figure 1. Producing a signal on pin 40 using a 16-bit(reload)-timer interrupt



In the produced code, the offset (due to calling ISR and context save) takes 64 cycles (40_{hex}) which must be included in the calculation :

$Reload = FFFF_{hex} - (f_c / (4\ f_{int})) + 40_{hex}$ mit : f_c : *Quartzfreq.*
f_{int} : *Interruptfreq.*

In this example: With f_c = 10 MHz and f_{int}=1500Hz: *Reload*=$F9BD_{hex}$

2.1 Interrupt-Service Routione 16-bit-Timer:

(C-Source-code printed bold)

before calling ISR : finish actual command and save PS and address on stack approx. 10-14 cycles

```
220:  void TC16INT6()
221: {
        C2BA: 40  PUSHW    A                4
        C2BB: 43  XCHW     A,T              2
        C2BC: 40  PUSHW    A                4
        C2BD: F3 MOVW      A,EP             2
        C2BE: 40 PUSHW     A                4
        C2BF: 41 PUSHW     IX               4
        C2C0: F1 MOVW      A,SP        2
        C2C1: E2 MOVW      IX,A        2
        C2C2: 08 MOV       A,R0             3
        C2C3: 10 SWAP                       2
        C2C4: 09 MOV       A,R1             3
        C2C5: 40 PUSHW     A                4
222: TMCR = 0x22;   /* Int enable egain */
        C2C6: 851822 MOV       18,#22       4
223:  TCHR = 0xF6;   /* Reload value */
        C2C9: 8519F6 MOV       19,#F6       4
224:  TCLR = 0x3B;
        C2CC: 851A3B MOV       1A,#3B       4
225:  TCS = 1;       /* start counter again */
        C2CF: A818 SETB     18:00           4
                                            64 cycles
226:  PDR4_0 = 1;    /* show pulse on PIN 40 */
        C2D1: A80F SETB     0F:00
227:  wait(20);
        C2D3: E40014 MOVW      A,#0014
        C2D6: 40 PUSHW     A
        C2D7: 31C04B CALL      \wait
        C2DA: 50 POPW      A
228:  PDR4_0 = 0;
        C2DB: A00F CLRB     0F:00
```

OBSOLETE

```
229:}
        C2DD: 50 POPW     A
        2DE: 49 MOV       R1,A
        2DF: 10 SWAP
        2E0: 48 MOV       R0,A
        2E1: F2 MOVW      A,IX
        2E2: E1 MOVW      SP,A
        2E3: 51 POPW      IX
        2E4: 50 POPW      A
        2E5: E3 MOVW      EP,A
        2E6: 50 POPW      A
        2E7: 43 XCHW      A,T
        2E8: 50 POPW      A
        2E9: 30 RETI
```

OBSOLETE

Document History

Document Title: AN204934 - Note on Calculating Reload Values FFMC8L

Document Number: 002-04934

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	—	WOFR	02/20/2014	Initial release.
*A	5235611	WOFR	04/22/2016	Migrated Spansion Application Note from MCU-AN-389012-E-V10to Cypress format. This AN to be Obsolete, since this is the old 8-bit MCU

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2013-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

OBSOLETE