AN206375

FR Family, 32-Bit Microcontroller, FR80S/T-series DMA Access Speed

The DMA function transfers data concurrently with the CPU process and it is convenient for shortening the data transfer time and enhancing the processing speed. To obtain the time of data transfer by this DMA function, the number of cycles must be obtained according to the conditions because the data transfer route depends on the transfer source or destination. This document discusses access speeds under various conditions for the DMA function mounted on the FR80S/T series.

Contents

1 Introduction....1
2 Overview of DMA Transfer....2
2.1 DMA Transfer Cycles....2
2.2 Access Cycles Depending on Bus Difference....2
2.3 Disincentive Against DMA Transfer....3
3 Number of DMA Transfer Cycles....4
3.1 Number of Peripheral Function -> DMA Transfer Cycles....4
3.2 Number of DMA -> Peripheral Function Transfer Cycles....4
3.3 Number of RAM -> DMA Transfer Cycles....5
3.4 Number of DMA -> RAM Transfer Cycles....5
3.5 Number of USB -> DMA Transfer Cycles....6
3.6 Number of DMA -> USB Transfer Cycles....6
3.7 Number of External Bus -> DMA Transfer Cycles....7
3.8 Number of DMA -> External Bus Transfer Cycles....8
4 Calculation Examples....10
4.1 Peripheral Function -> DMA -> RAM....10
4.2 RAM -> DMA -> Peripheral Function....10
4.3 Peripheral Function -> DMA -> Peripheral Function....10
5 Timing Chart Examples....11
5.1 Peripheral Function -> DMA -> RAM....11
5.2 RAM -> DMA -> Peripheral Function....13
5.3 Peripheral Function -> DMA -> Peripheral Function....14
5.4 RAM -> DMA -> RAM....15
5.5 USB -> DMA -> RAM....15
5.6 RAM -> DMA -> USB....15
5.7 External bus -> DMA -> RAM....16
5.8 RAM -> DMA -> External Bus....18
5.9 Peripheral Function -> DMA -> External Bus....21
Document History....23

1 Introduction

The DMA function transfers data concurrently with the CPU process and it is convenient for shortening the data transfer time and enhancing the processing speed. To obtain the time of data transfer by this DMA function, the number of cycles must be obtained according to the conditions because the data transfer route depends on the transfer source or destination. This document discusses access speeds under various conditions for the DMA function mounted on the FR80S/T series.

2 Overview of DMA Transfer

2.1 DMA Transfer Cycles

When data is being transferred using the DMA function, data is retrieved from the transfer source to the DMA then data is written from the DMA to the transfer destination. Therefore, the number of DMA transfer cycles can be calculated as the sum of [Transfer source -> DMA cycle] and [DMA -> Transfer destination cycle].

[DMA transfer cycle] = [Transfer source -> DMA cycle] + [DMA -> Transfer destination cycle] … (1)

2.2 Access Cycles Depending on Bus Difference

The data access cycles can be categorized under the Flash memory, RAM, peripheral function, external bus, or USB depending on the data transfer route. (See Figure 1.) If [Transfer source -> DMA cycles] and [DMA -> Transfer destination cycles] can be obtained under these five conditions, the DMA transfer cycles under each condition can be obtained. Since the on-chip bus has the blue and green lines (Figure 1) and two layers (multi-layer bus), DMA transfer can be performed concurrently with CPU access if the access destination is different. The XBS (cross-bar switch bus) can perform DMA transfer to the RAM while the CPU is accessing the Flash memory to give instructions.

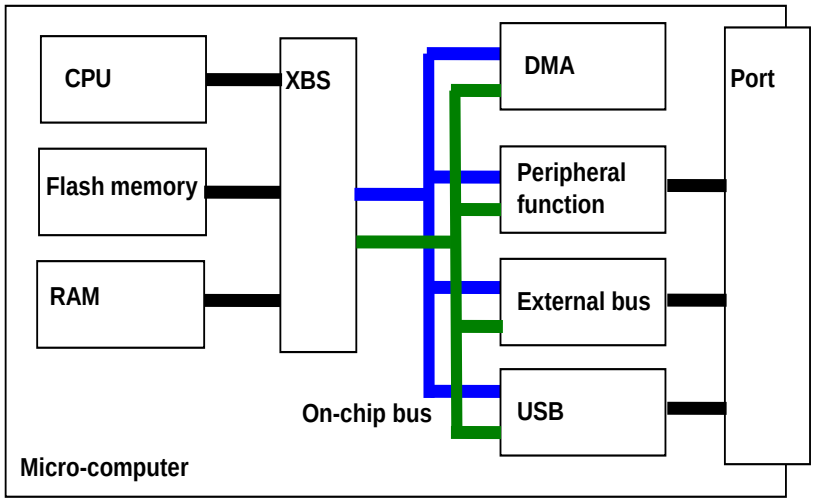


Figure 1 FR80S internal bus pattern

2.3 Disincentive Against DMA Transfer

Although the DMA function can transfer data concurrently with the CPU process, it shares the data transfer bus of the DMA function with the CPU. If both access the same bus, they block transfer to each other. Since the effect of disincentive to DMA transfer depends on instructions executed by the CPU, it is hard to represent the effect in a quantitative manner. Therefore, this document proceeds with the discussion on the number of cycles on the assumption that there is no blocking by the CPU. Described below are specific examples of disincentives against DMA transfer.

2.3.1 Specific Example 1 of Disincentive: Accessing the Flash memory for DMA Transfer

Since the CPU executes the program on the Flash memory, accessing the Flash memory data by the DMA function is blocked. Therefore, this document excludes the Flash memory from the discussion on the number of cycles and proceeds with the discussion under the remaining four conditions.

2.3.2 Specific Example 2 of Disincentive: CPU Accessing the RAM

While the CPU is accessing the data in the RAM, accessing the RAM by the DMA function is blocked.

2.3.3 Special Specific Example 3 Without Blocking: Using Two Layers of On-chip Bus

If the DMA function transfers data from the RAM to a peripheral function while the CPU is accessing an external bus, transfer is not blocked. The reason is that there are two buses; the XBS and the bus connecting the DMA, peripheral function, external bus, or USB. While the CPU is accessing the external bus, the DMA can use the other bus to access the RAM or peripheral function.

CPU access conditions as disincentives against the DMA function are summarized as follows: Mark X in the table below is the blocking condition.

		DMA			
		RAM	**Peripheral function**	**External bus**	**USB**
CPU	**RAM**	X	O	O	O
	Peripheral function	O	X	O	O
	External bus	O	O	X	O
	USB	O	O	O	X

O: The CPU and DMA are concurrently accessible.

X: CPU access blocks DMA access.

3 Number of DMA Transfer Cycles

Described below is the number of DMA transfer cycles under each condition of data transfer routes.

3.1 Number of Peripheral Function -> DMA Transfer Cycles

Minimum value: 1CCLK+2PCLK

Maximum value: 3PCLK

*) CCLK: CPU frequency PCLK: Peripheral function frequency

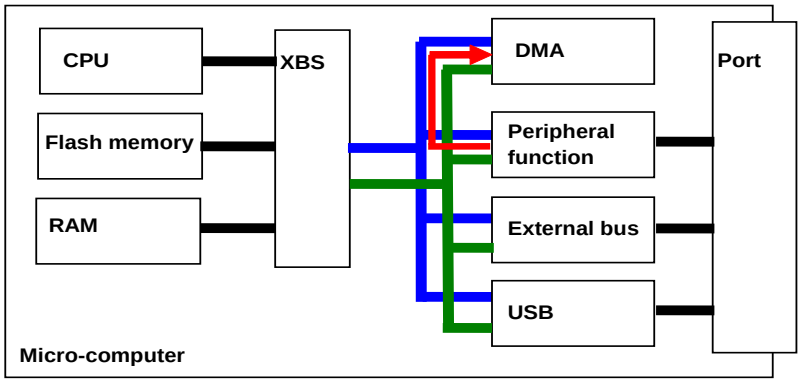


Figure 2 Peripheral function -> DMA transfer

3.2 Number of DMA -> Peripheral Function Transfer Cycles

Minimum value: 1CCLK+2PCLK

Maximum value: 3PCLK

*) CCLK: CPU frequency PCLK: Peripheral function frequency

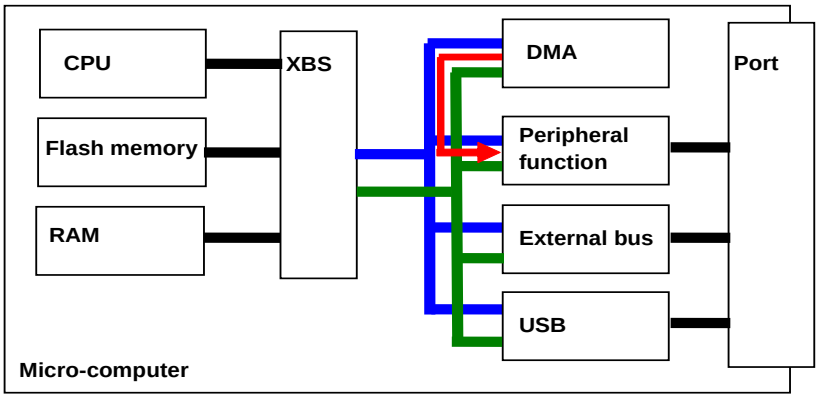


Figure 3 DMA -> Peripheral function transfer

3.3 Number of RAM -> DMA Transfer Cycles

2CCLK

*) CCLK: CPU frequency

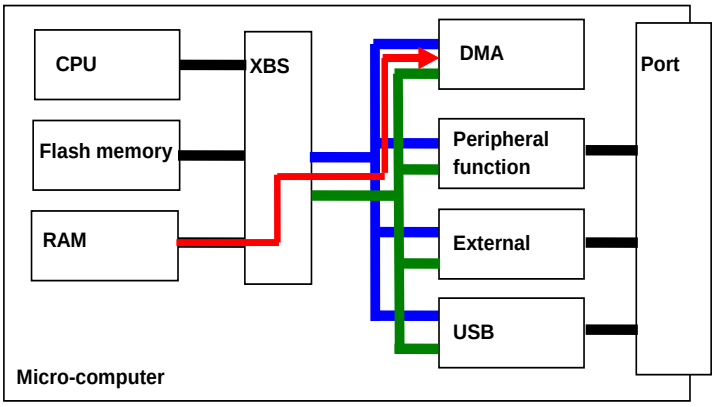


Figure 4 RAM -> DMA transfer

3.4 Number of DMA -> RAM Transfer Cycles

1CCLK

*) CCLK: CPU frequency

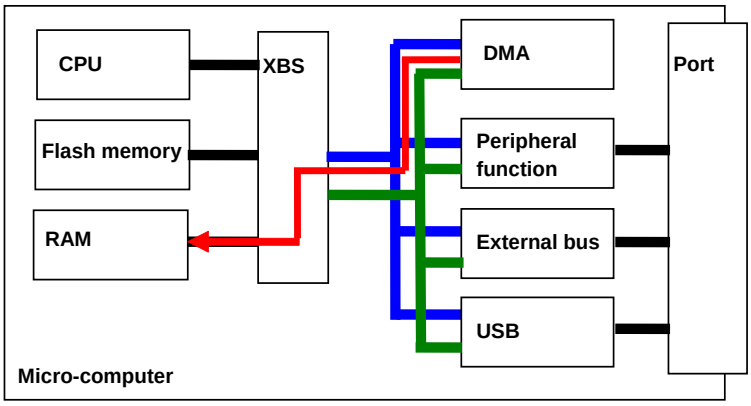


Figure 5 DMA -> RAM transfer

3.5 Number of USB -> DMA Transfer Cycles

Minimum value: 1CCLK

*) CCLK: CPU frequency PCLK: Peripheral function frequency

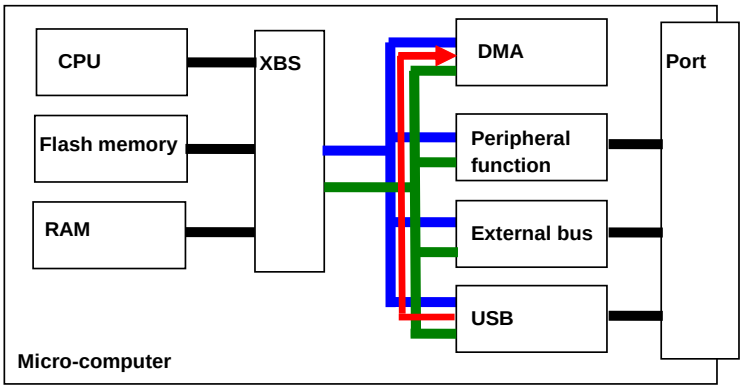


Figure 6 Peripheral function -> DMA transfer

3.6 Number of DMA -> USB Transfer Cycles

Minimum value: 1CCLK

*) CCLK: CPU frequency PCLK: Peripheral function frequency

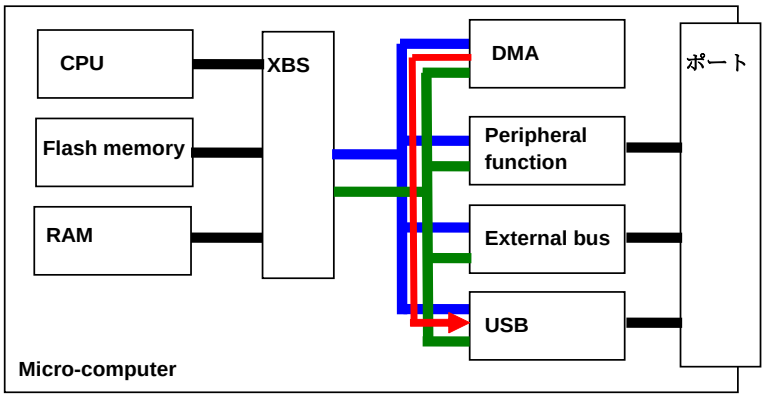


Figure 7 DMA -> USB transfer

3.7 Number of External Bus -> DMA Transfer Cycles

Minimum value: 2TCLK+3CCLK

Maximum value: 3TCLK+2CCLK

*) CCLK: CPU frequency TCLK: External bus frequency

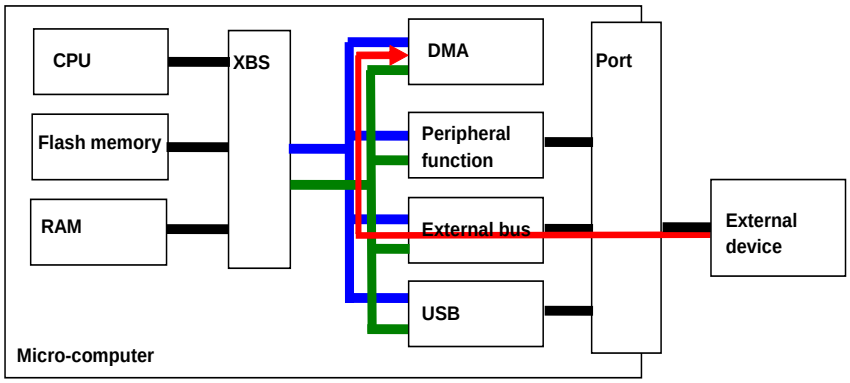


Figure 8 External bus -> DMA transfer

*) For the external bus, one external bus is accessed at 3TCLK.

3.8 Number of DMA -> External Bus Transfer Cycles

a) If 3TCLK – [Transfer source -> DMA cycles] < 0

1CCLK

b) If 3TCLK – [Transfer source -> DMA cycles] = 0

1TCLK

c) If 3TCLK – [Transfer source -> DMA cycles] = 1CCK

Minimum value: 1CCLK Maximum value: 1TCLK+1CCLK

d) If 3TCLK – [Transfer source -> DMA cycles] => 2CCLK

1CCLK or 3TCLK – [Transfer source -> DMA cycles] or 4TCLK – [Transfer source -> DMA cycle]

*) CCLK: CPU frequency TCLK: External bus frequency

*) [Transfer source -> DMA cycles]: Number of cycles for transfer from the transfer source to the DMA For example, if the transfer source is the RAM, the number of transfer cycles is 2CCLK according to the above table.

*) For the external bus, one external bus is accessed at 3TCLK. However, since the external bus has a write buffer, data is written to the write buffer of the external bus at 1TCLK. Then, data written to the write buffer is written to the external device at 3TCLK. However, since there is only one write buffer, writing to the write buffer must wait until access to the external device is completed. Therefore, the number of cycles higher than 1TCLK is required.

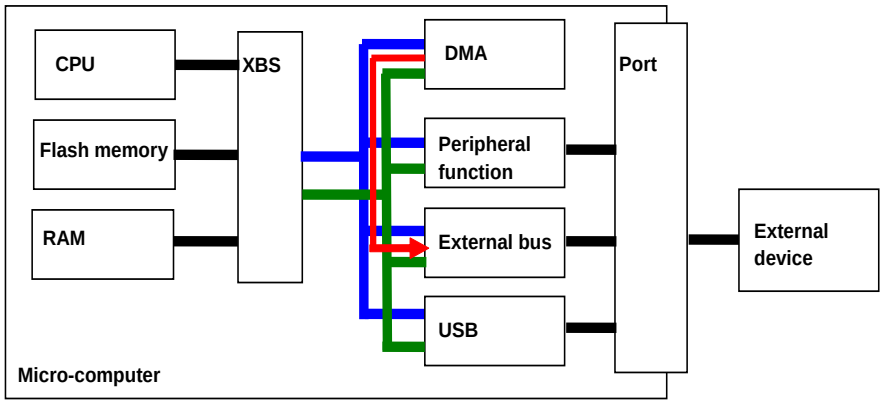


Figure 9 DMA -> External bus transfer

The numbers of transfer cycles described above are summarized under respective conditions:

	Minimum value	Maximum value	Remarks
Peripheral function -> DMA	1CCLK+2PCLK	3PCLK	
DMA -> Peripheral function	1CCLK+2PCLK	3PCLK	
RAM -> DMA	2CCLK		
DMA -> RAM	1CCLK		
USB -> DMA	1CCLK		
DMA -> USB	1CCLK		
External bus -> DMA	2TCLK+3CCLK	3TCLK+2CCLK	
DMA -> External bus	1CCLK		If 3TCLK – [Transfer source -> DMA cycles] < 0
	1TCLK		If 3TCLK – [Transfer source -> DMA cycles] = 0
	1CCLK	1TCLK+1CCLK	If 3TCLK – [Transfer source -> DMA cycles] = 1CCLK
	1CCLK		If 3TCLK – [Transfer source -> DMA cycles] => 2CCLK (For details, see the timing chart example.)
	3TCLK – [Transfer source -> DMA cycles]		
	3TCLK – [Transfer source -> DMA cycles]		

CCLK: CPU frequency

PCLK: Peripheral function frequency

TCLK: External bus frequency

[Transfer source -> DMA cycles]: Number of cycles for transfer from the transfer source to the DMA For example, if the transfer source is the RAM, the number of transfer cycles is 2CCLK according to the above table.

4 Calculation Examples

The maximum and minimum values are obtained using the number of transfer cycles under each condition above and expression (1) in 2.1.

4.1 Peripheral Function -> DMA -> RAM

The maximum value is calculated as follows:

[DMA transfer cycles] = [Transfer source -> DMA cycles] + [DMA -> Transfer destination cycles]

= 3PCLK + 1CCLK

The minimum value is calculated as follows:

[DMA transfer cycles] = [Transfer source -> DMA cycles] + [DMA -> Transfer destination cycles]

= 1CCLK + 1CCLK + 2PCLK

= 2CCLK + 2PCLK

4.2 RAM -> DMA -> Peripheral Function

The maximum value is calculated as follows:

[DMA transfer cycles] = [Transfer source -> DMA cycles] + [DMA -> Transfer destination cycles]

= 2CCLK + 3PCLK

The minimum value is calculated as follows:

[DMA transfer cycles] = [Transfer source -> DMA cycles] + [DMA -> Transfer destination cycles]

= 2CCLK + 1CCLK + 2PCLK

= 3CCLK + 2PCLK

4.3 Peripheral Function -> DMA -> Peripheral Function

The maximum value is calculated as follows:

[DMA transfer cycles] = [Transfer source -> DMA cycles] + [DMA -> Transfer destination cycles]

= 3PCLK + 3PCLK

= 6PCLK

The minimum value is calculated as follows:

[DMA transfer cycles] = [Transfer source -> DMA cycles] + [DMA -> Transfer destination cycles]

= 1CCLK + 2PCLK + 1CCLK + 2PCLK

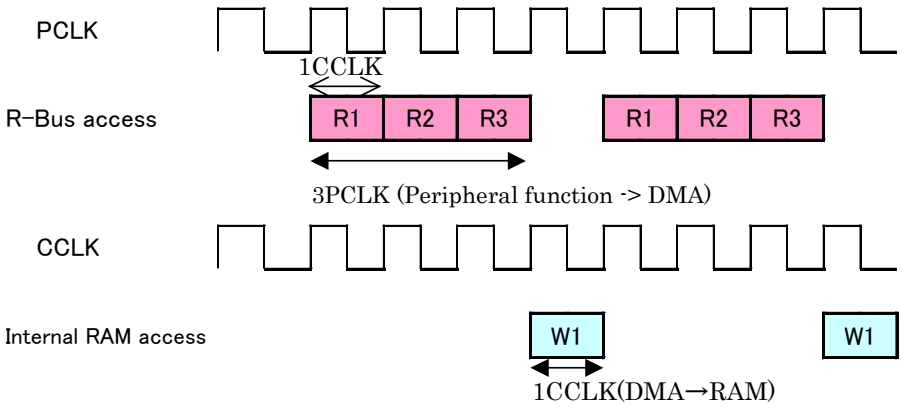
= 2CCLK + 4PCLK

5 Timing Chart Examples

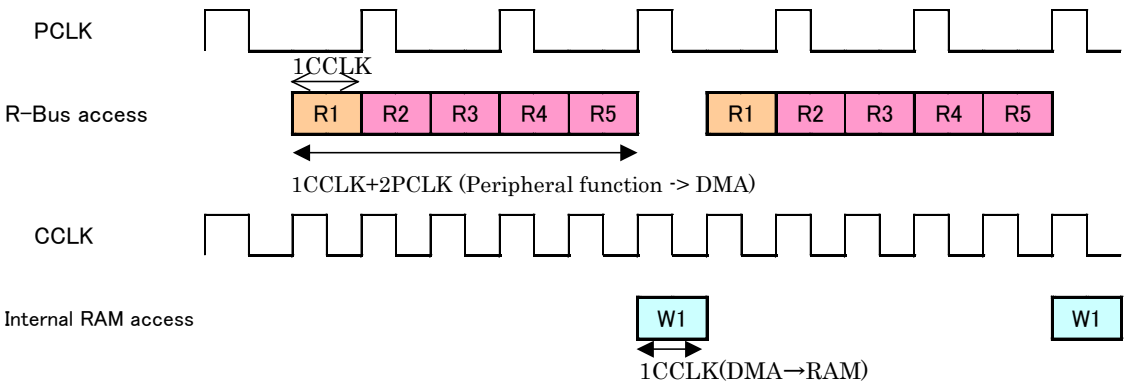
Examples of the number of transfer cycles are shown below using the timing chart. However, these are examples of the number of transfer cycles, so use “3. Number of Transfer Cycles” above to obtain the number of transfer cycles. Since the contents are detailed, you need not read them. The audience are those who want to get deeper understanding on the DMA transfer cycles including the internal operation.

5.1 Peripheral Function -> DMA -> RAM

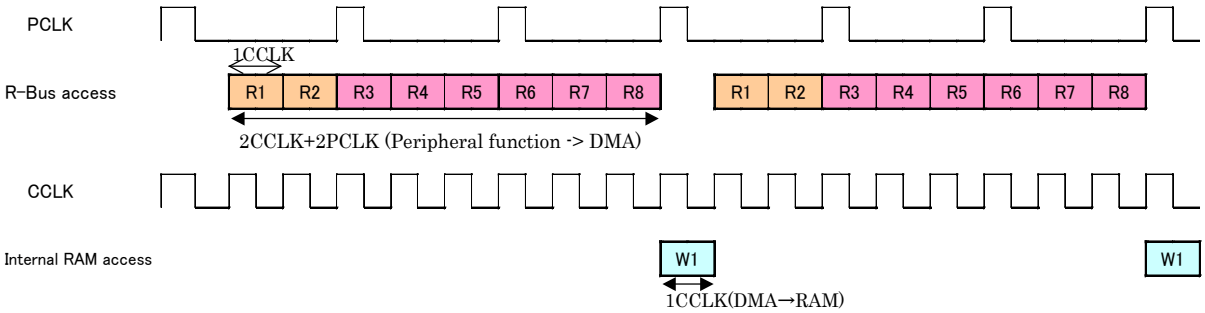
■If CCLK:PCLK=1:1



■If CCLK:PCLK=1:2

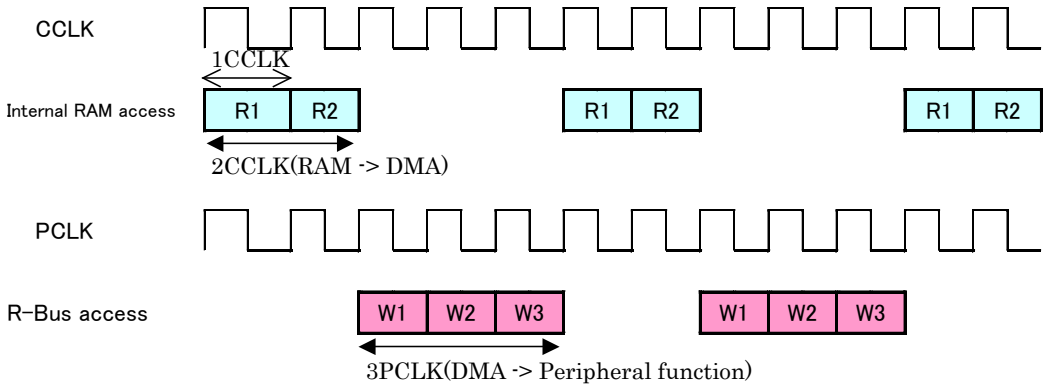


■If CCLK:PCLK=1:3

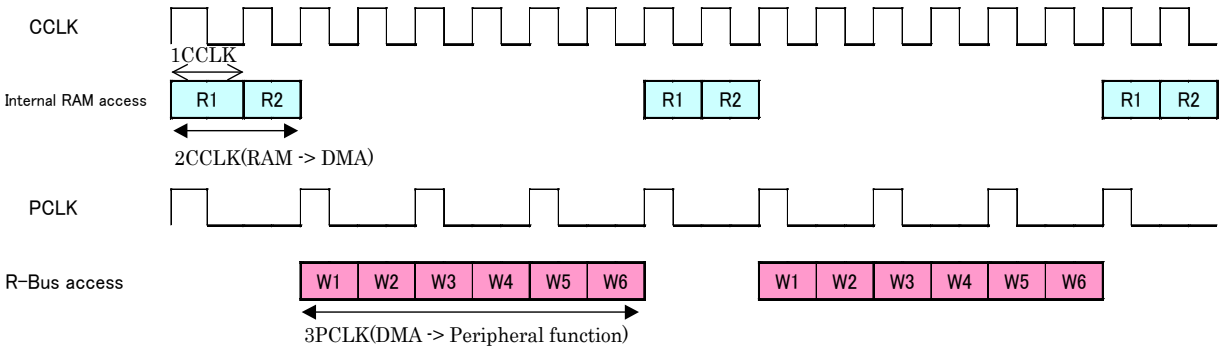


5.2 RAM -> DMA -> Peripheral Function

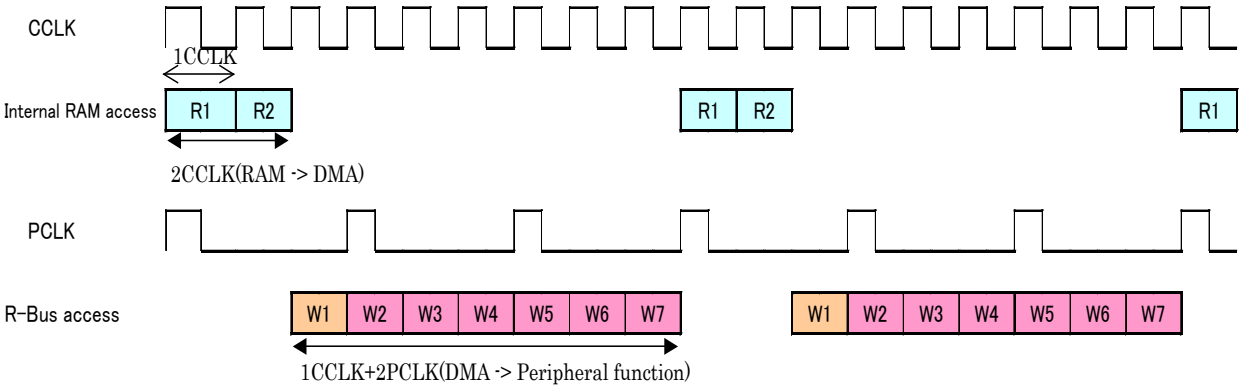
■ If CCLK:PCLK=1:1



■ If CCLK:PCLK=1:2

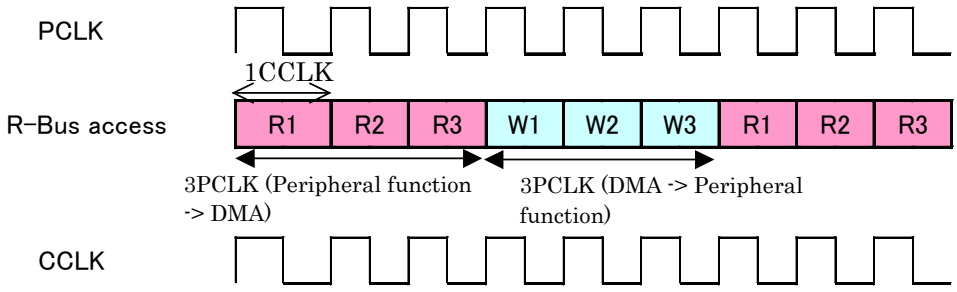


■ If CCLK:PCLK=1:3

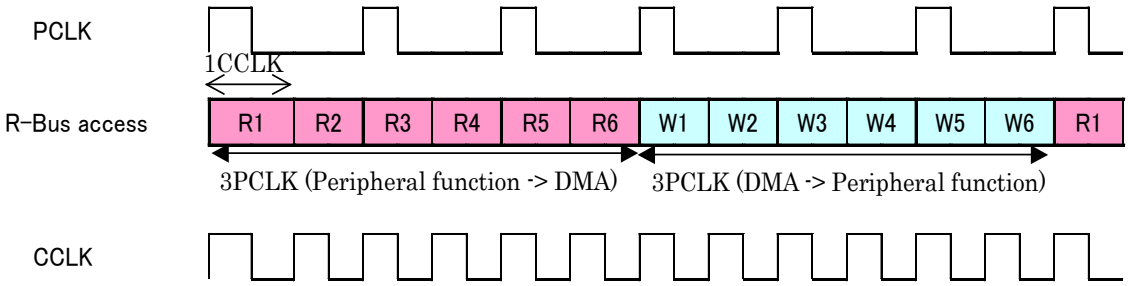


5.3 Peripheral Function -> DMA -> Peripheral Function

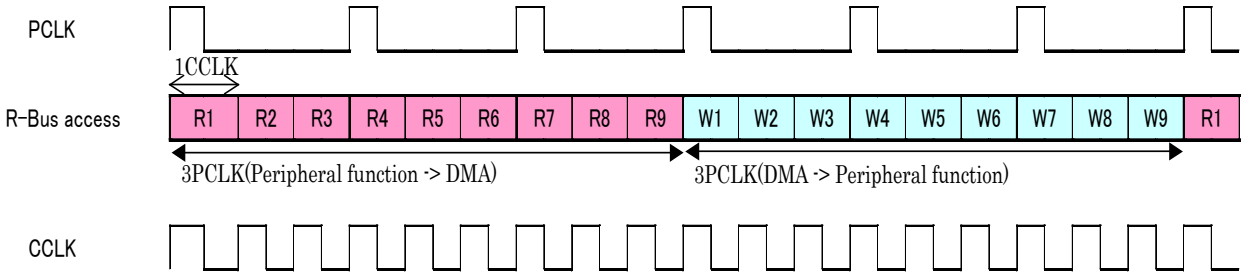
■ If CCLK:PCLK=1:1



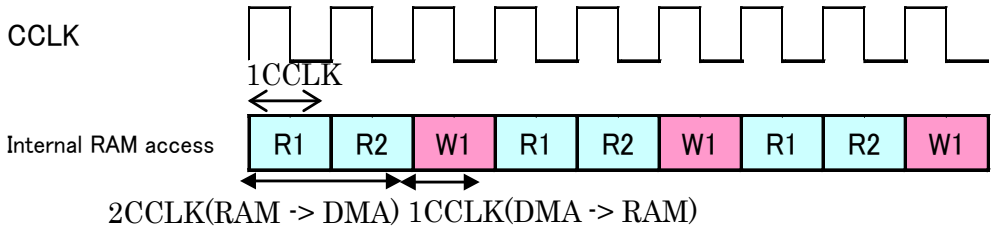
■ If CCLK:PCLK=1:2



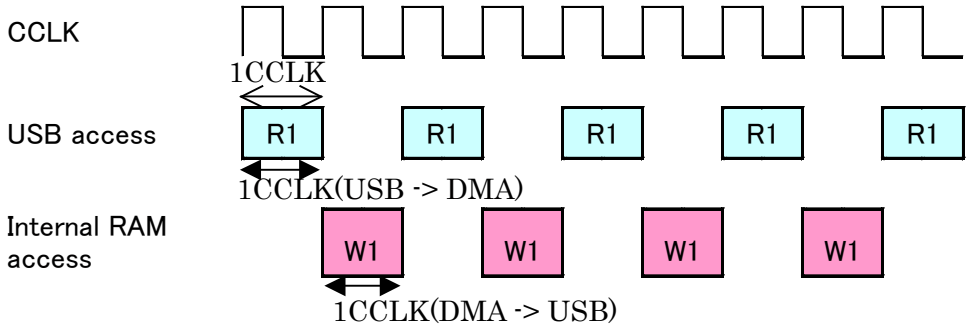
■ If CCLK:PCLK=1:3



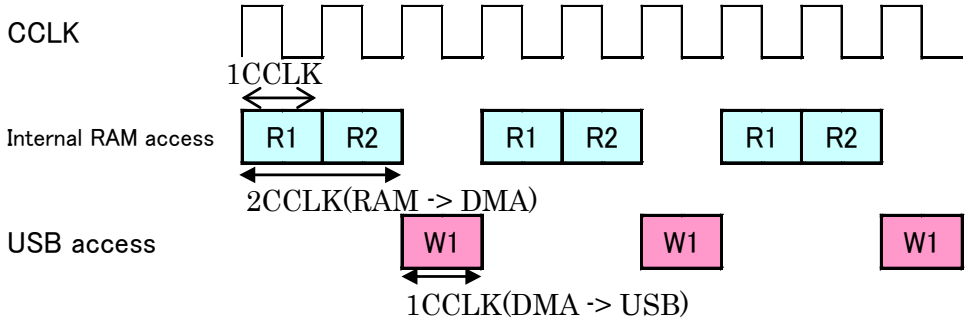
5.4 RAM -> DMA -> RAM



5.5 USB -> DMA -> RAM

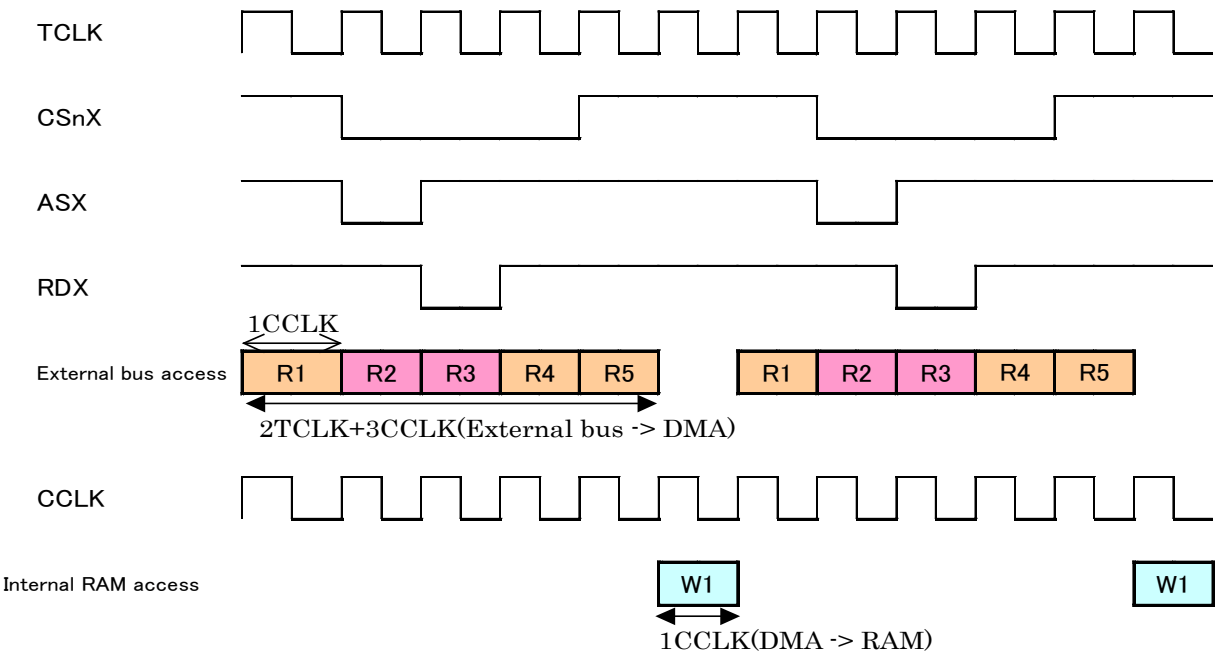


5.6 RAM -> DMA -> USB

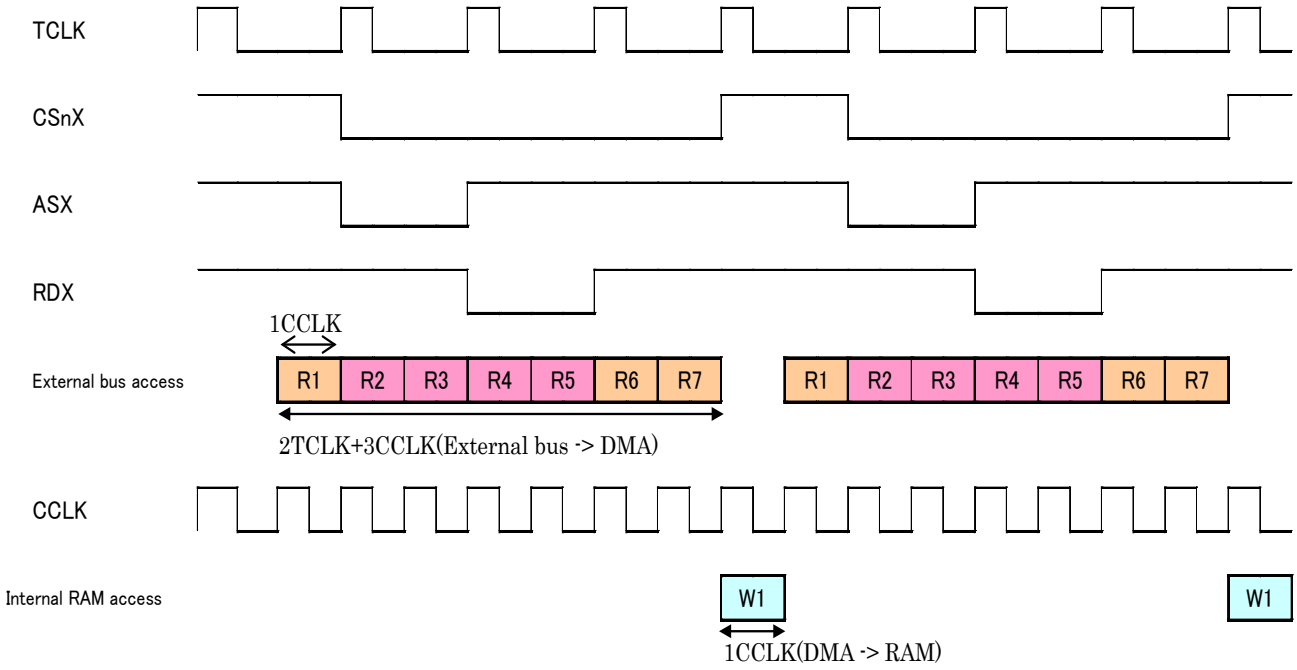


5.7 External bus -> DMA -> RAM

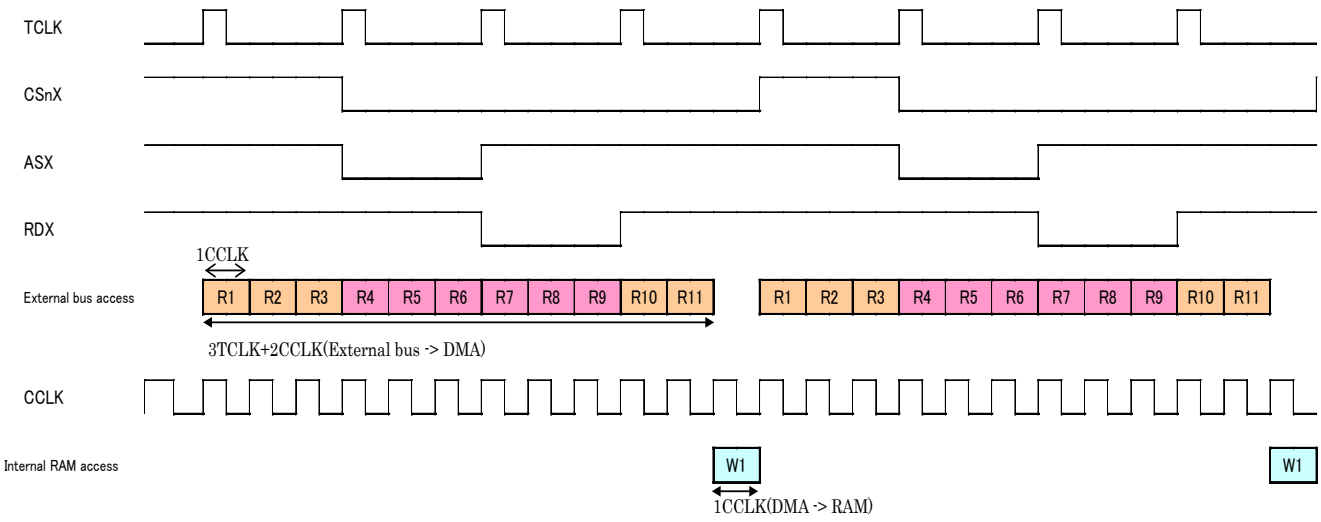
■ If CCLK:TCLK=1:1



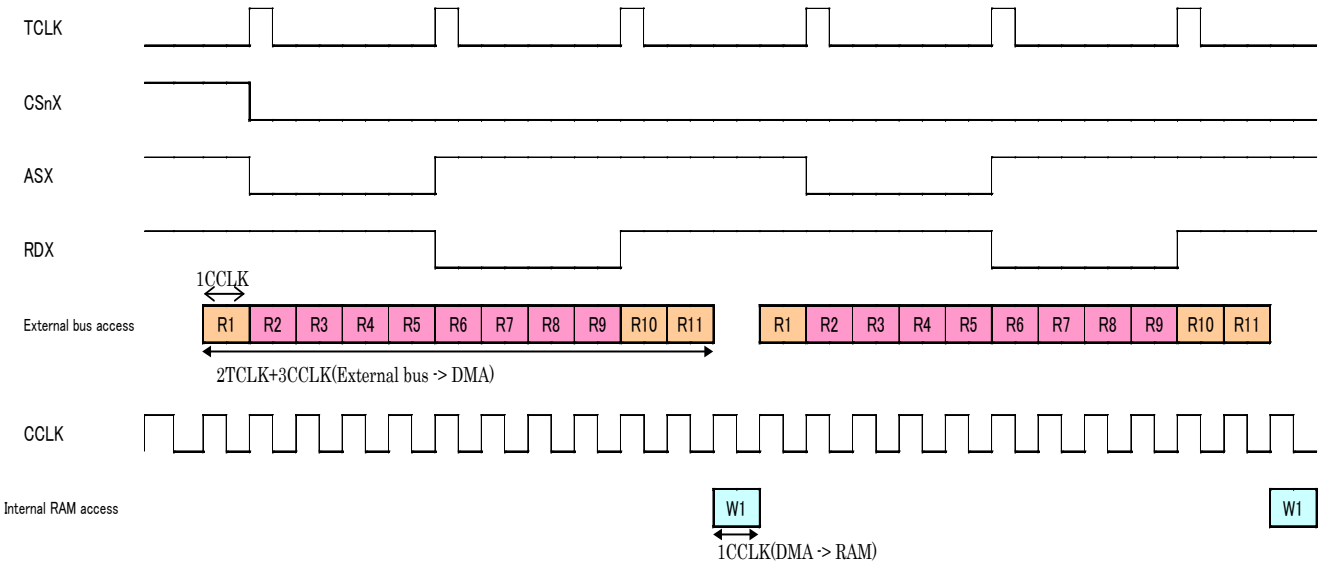
■ If CCLK:TCLK=1:2



■ If CCLK:TCLK=1:3

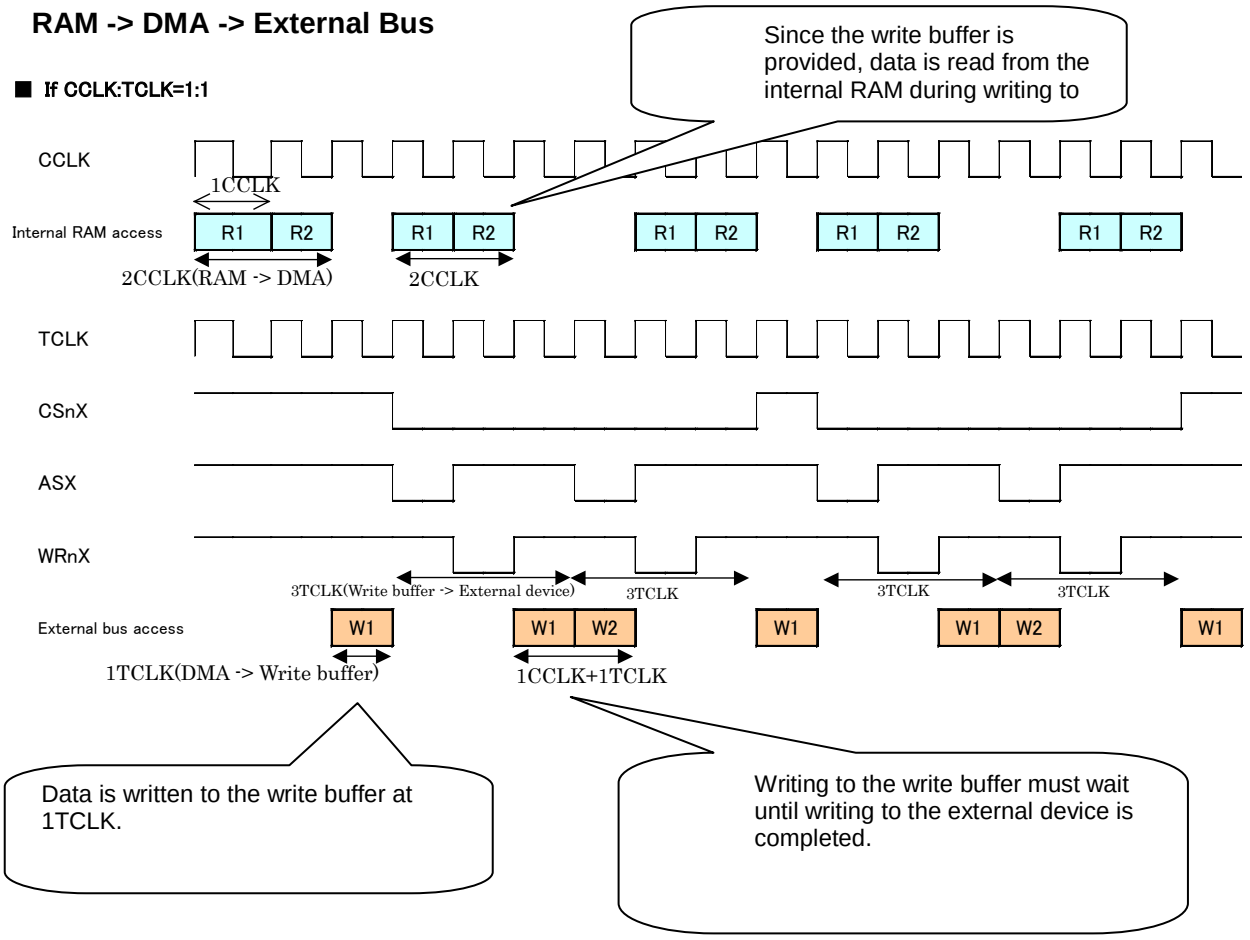


■ If CCLK:TCLK=1:4



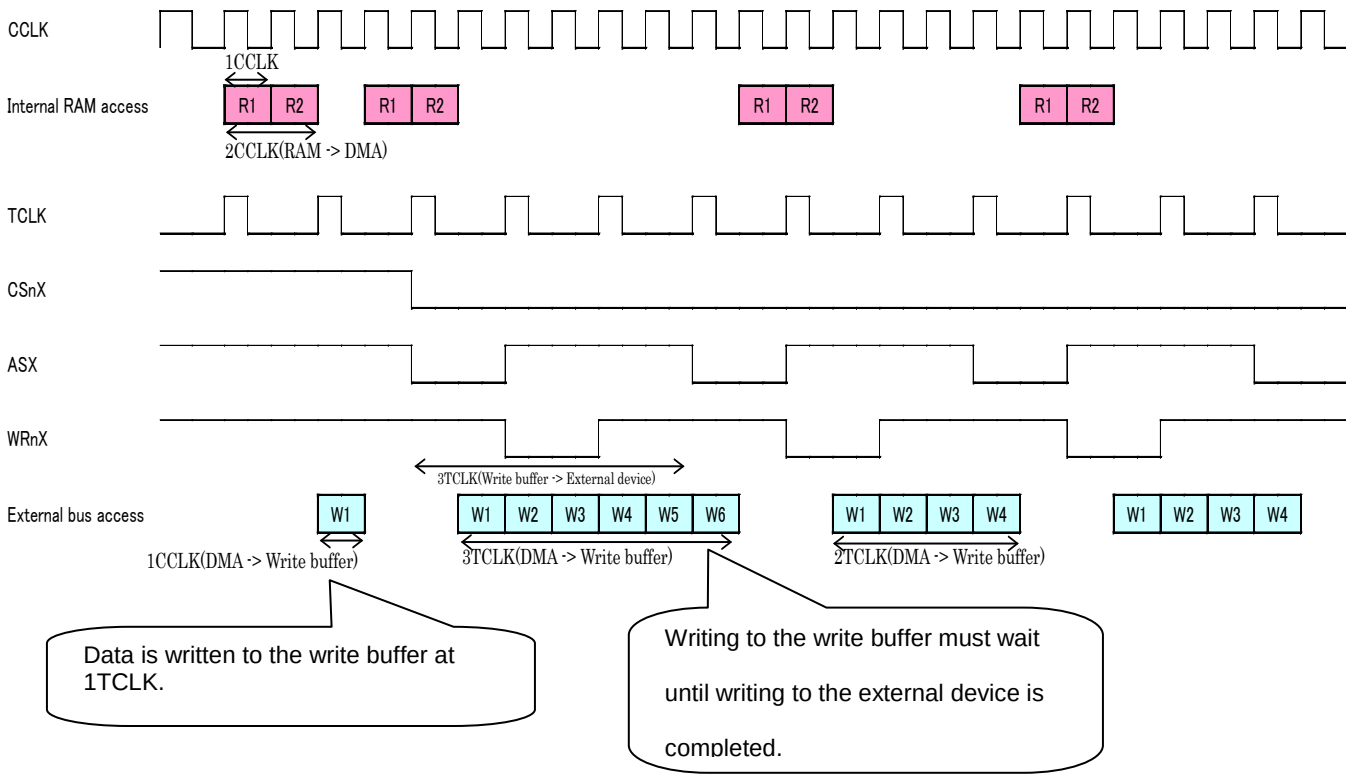
5.8 RAM -> DMA -> External Bus

■ If CCLK:TCLK=1:1



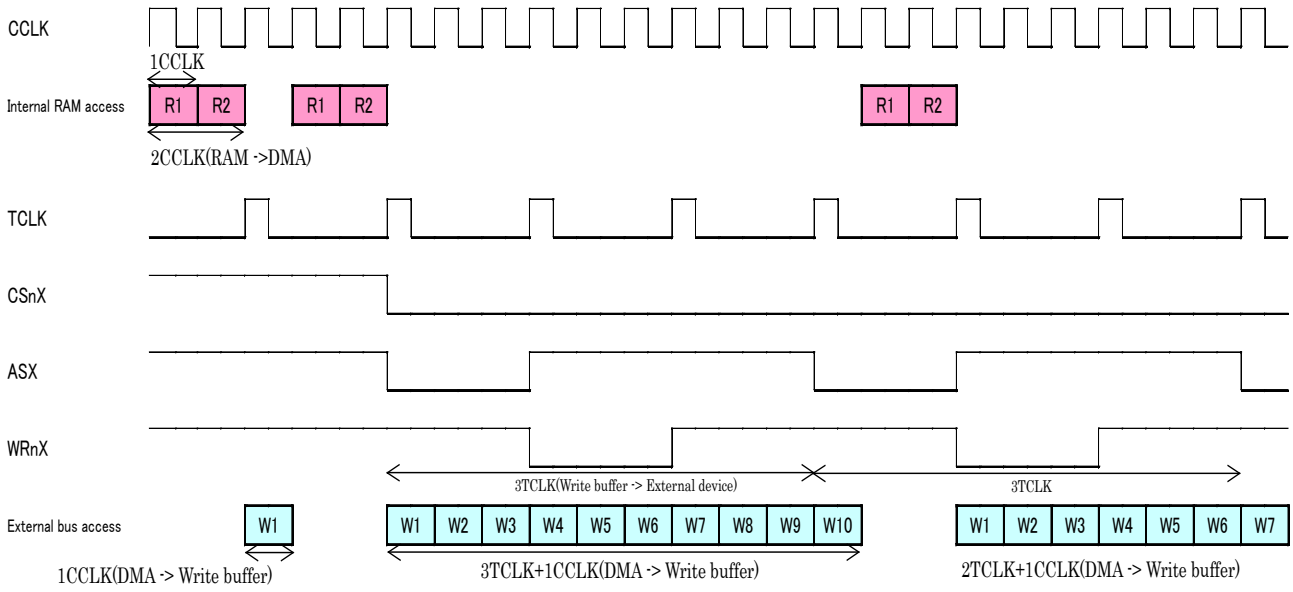
If 3TCLK - [Transfer source -> DMA cycles] = 1CCLK, writing takes "1CCLK" or "1CCLK+1TCLK".

■ If CCLK:TCLK=1:2

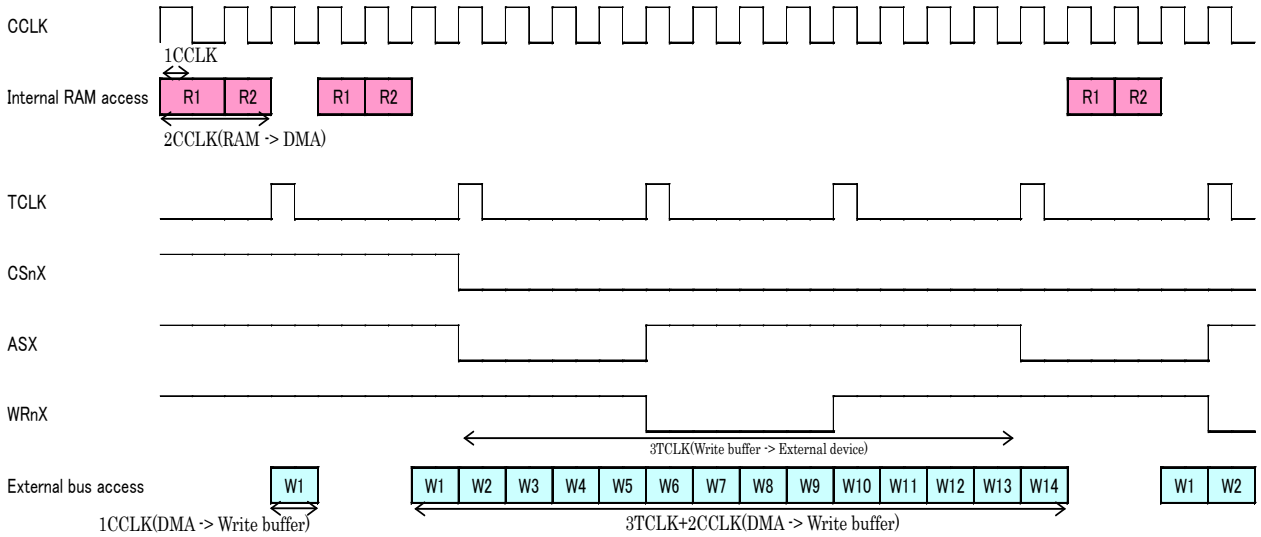


If 3TCLK - [Transfer source -> DMA cycles] => 2CCLK,
writing takes 1TCLK or "3TCLK - [Transfer source -> DMA cycles]" or
"4TCLK - [Transfer source -> DMA cycles]".

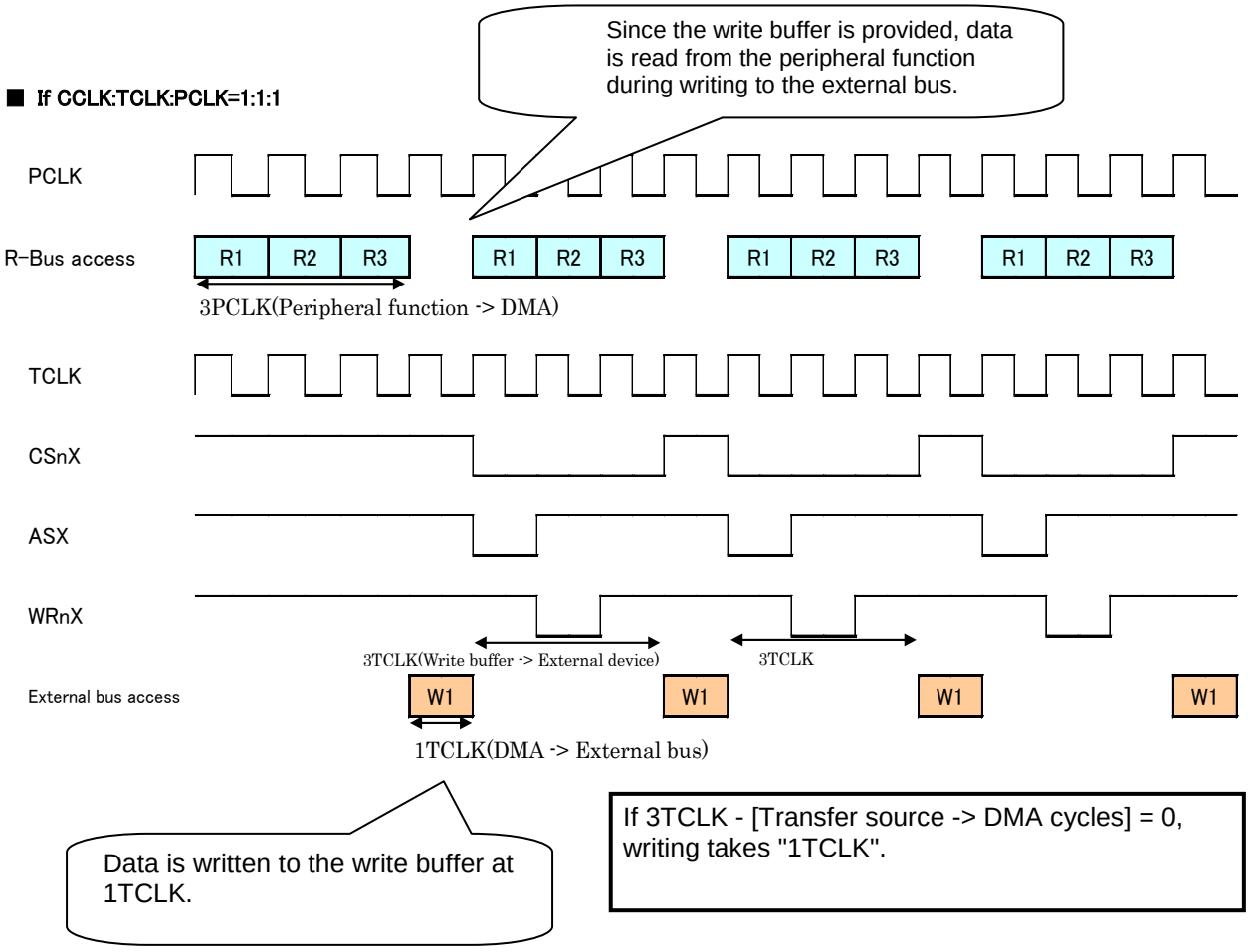
■ If CCLK:TCLK=1:3



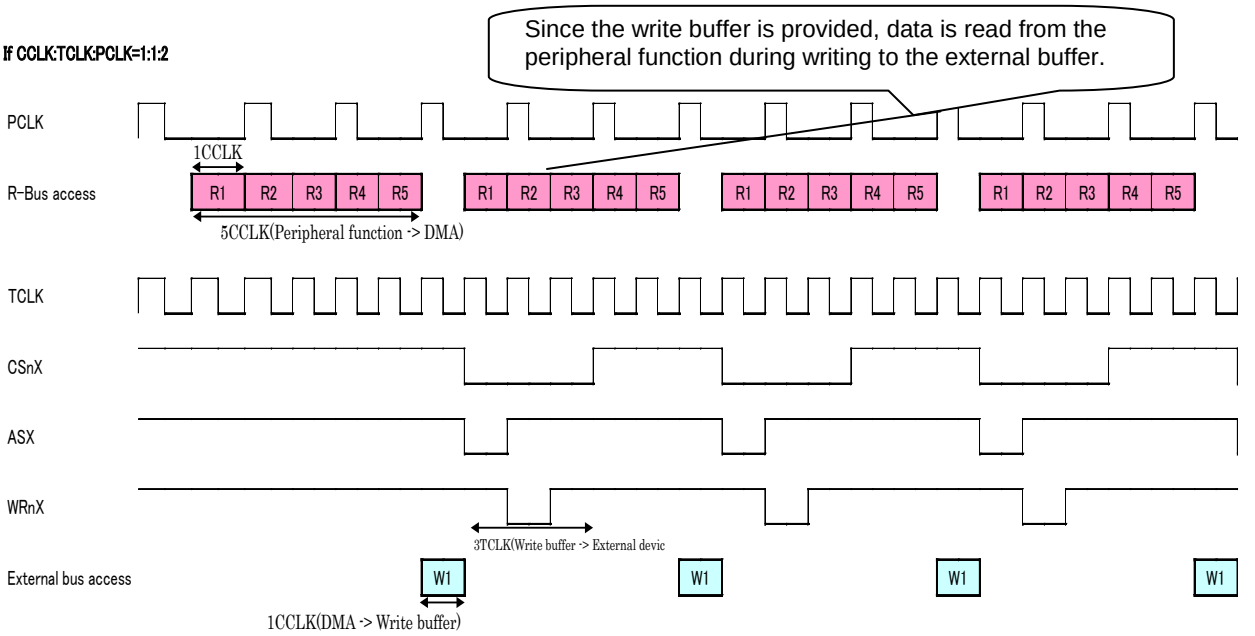
■ If CCLK:TCLK=1:4



5.9 Peripheral Function -> DMA -> External Bus



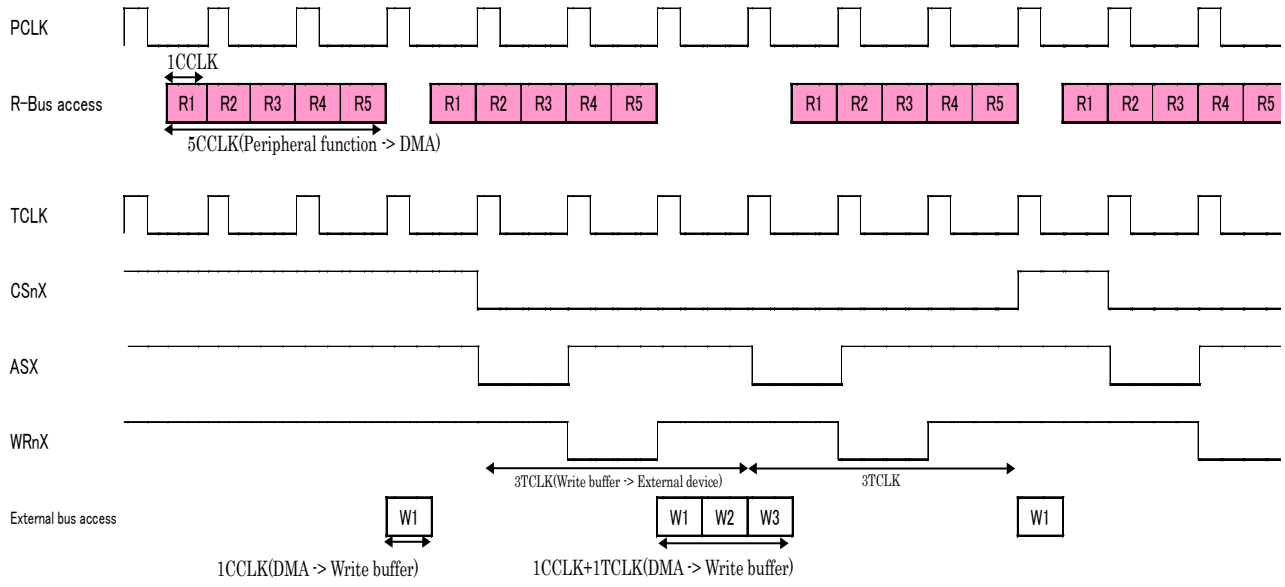
■ If CCLK:TCLK:PCLK=1:1:2



Data is written to the write buffer at 1CCLK.

If 3TCLK - [Transfer source -> DMA cycles] < 0, writing takes "1CCLK".

■ If CCLK:TCLK:PCLK=1:2:2



If 3TCLK - [Transfer source -> DMA cycles] = 1CCLK, writing takes "1CCLK" or "1CCLK+1TCLK".

Document History

Document Title: AN206375 - FR Family, 32-Bit Microcontroller, FR80S/T-series DMA Access Speed

Document Number: 002-06375

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	YUIS	11/21/2008	First Edition
*A	5293700	YUIS	06/02/2016	Migrated Spansion Application Note “AN07-00156-1E” to Cypress format.

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,2008-2016. 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.