XMC4000

XMC4500 Family

CCU4 and Multi Channel Mode

Multi Phase Output Pattern Control and Synchronization

User Guide

Device Guide

<Revision 1.0>, <2012-30-03>

Microcontroller

Edition <2012-30-03>

Published by
Infineon Technologies AG
81726 Munich, Germany

© 2012 Infineon Technologies AG
All Rights Reserved.

Legal Disclaimer

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Infineon Technologies 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.

Information

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (**www.infineon.com**).

Warnings

Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office.

Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

Confidential

Revision History	
Page or Item	**Subjects (major changes since previous revision)**
<Revision 1.0>, <2012-30-03>	

Trademarks of Infineon Technologies AG

AURIX™, C166™, CanPAK™, CIPOS™, CIPURSE™, EconoPACK™, CoolMOS™, CoolSET™, CORECONTROL™, CROSSAVE™, DAVE™, EasyPIM™, EconoBRIDGE™, EconoDUAL™, EconoPIM™, EiceDRIVER™, eupec™, FCOS™, HITFET™, HybridPACK™, I²RF™, ISOFACE™, IsoPACK™, MIPAQ™, ModSTACK™, my-d™, NovalithIC™, OptiMOS™, ORIGA™, PRIMARION™, PrimePACK™, PrimeSTACK™, PRO-SIL™, PROFET™, RASIC™, ReverSave™, SatRIC™, SIEGET™, SINDRION™, SIPMOS™, SmartLEWIS™, SOLID FLASH™, TEMPFET™, thinQ!™, TRENCHSTOP™, TriCore™.

Other Trademarks

Advance Design System™ (ADS) of Agilent Technologies, AMBA™, ARM™, MULTI-ICE™, KEIL™, PRIMECELL™, REALVIEW™, THUMB™, µVision™ of ARM Limited, UK. AUTOSAR™ is licensed by AUTOSAR development partnership. Bluetooth™ of Bluetooth SIG Inc. CAT-iq™ of DECT Forum. COLOSSUS™, FirstGPS™ of Trimble Navigation Ltd. EMV™ of EMVCo, LLC (Visa Holdings Inc.). EPCOS™ of Epcos AG. FLEXGO™ of Microsoft Corporation. FlexRay™ is licensed by FlexRay Consortium. HYPERTERMINAL™ of Hilgraeve Incorporated. IEC™ of Commission Electrotechnique Internationale. IrDA™ of Infrared Data Association Corporation. ISO™ of INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. MATLAB™ of MathWorks, Inc. MAXIM™ of Maxim Integrated Products, Inc. MICROTEC™, NUCLEUS™ of Mentor Graphics Corporation. Mifare™ of NXP. MIPI™ of MIPI Alliance, Inc. MIPS™ of MIPS Technologies, Inc., USA. muRata™ of MURATA MANUFACTURING CO., MICROWAVE OFFICE™ (MWO) of Applied Wave Research Inc., OmniVision™ of OmniVision Technologies, Inc. Openwave™ Openwave Systems Inc. RED HAT™ Red Hat, Inc. RFMD™ RF Micro Devices, Inc. SIRIUS™ of Sirius Satellite Radio Inc. SOLARIS™ of Sun Microsystems, Inc. SPANSION™ of Spansion LLC Ltd. Symbian™ of Symbian Software Limited. TAIYO YUDEN™ of Taiyo Yuden Co. TEAKLITE™ of CEVA, Inc. TEKTRONIX™ of Tektronix Inc. TOKO™ of TOKO KABUSHIKI KAISHA TA. UNIX™ of X/Open Company Limited. VERILOG™, PALLADIUM™ of Cadence Design Systems, Inc. VLYNQ™ of Texas Instruments Incorporated. VXWORKS™, WIND RIVER™ of WIND RIVER SYSTEMS, INC. ZETEX™ of Diodes Zetex Limited.

Last Trademarks Update 2011-02-24

Table of Contents

1 Multi Phase Output Pattern Control and Sychronization 5
1.1 Multi Channel Mode Basics 5
1.1.1 The Multi Channel Slice Function 5

2 Multi Channel Synchronization to Shadow Transfers 6
2.1 CCU4 Shadow Transfers for Coherent Signal Pattern Update 6
2.2 The Global Shadow Transfer Set Enable Register 7
2.2.1 Shadow Transfer of Compare Register values 7
2.2.2 Compound Shadow Transfers 7

3 Start Timers 7
3.1 Global Start of CAPCOM Units 8
3.2 Global Start of the CAPCOM Units CCU4 and CCU8 8

4 Getting Started with CCU4 and Multi Channel Mode 9
4.1 Ultimate Initialization Sequence 9
4.1.1 Ultimate Start-Up Enable 9
4.2 Setting Compare Mode Control 9
4.2.1 Setting the Output Pins PASSIVE / ACTIVE Level Control 9
4.2.2 Initialize and Start the Timer 9
4.3 Multi Channel Mode Enable 9
4.3.1 Multi Channel Synchronization to Shadow Transfers 9
4.4 Using the Respective Multi Channel Slices Control Registers 9
4.5 External Events Control Setup Registers 10
4.5.1 Using the External Start Control Register Bitfields 10
4.5.2 Using the External Stop Control Register Bitfields 10
4.5.3 Pseudo Code for the External Start/Stop Control Setup 10

5 Runtime Handling of CCU4 and Multi Channel Mode 12

6 Example Application for CCU4 and Multi Channel Mode 13
6.1 Multi Channel Use in BLDC Motor Control 13

List of Figures

Figure 1 Multi Channel Mode HW Protocol Visualization 5
Figure 2 Multi Channel Timing and Synchronization 6
Figure 3 The Shadow Transfer Mechanism (Center Aligned Mode) 7
Figure 4 Synchronous Global Start of Timer Slices on different CAPCOM units (CCU4 or CCU8) 8
Figure 5 Multi Channel Timing and Synchronization Application in Runtime 12
Figure 6 Using POSIF in Hall Sensor Mode and Control by Multi Channel Interface 13

1 Multi Phase Output Pattern Control and Sychronization

Each CAPCOM4 (CCU40/../43) is a multi purpose timer unit for many use cases. It is designed with repetitive structures with multiple timer slices that have all the same base functionalities. This allows the development of re-usable SW low level drivers. The main function blocks of 1 of the 4 CC4y slices on a CCU4x are shown here. See Figure 1.

1.1 Multi Channel Mode Basics

The output pattern **CC4yOUT** of any slice that has been set in *multichannel mode* (by **CC4yTC.**MCME = 1_B), can be *controlled in parallel* by an external pattern that is present on its Multi Channel Input **MCIy**. To get it all synchronized, a Set Shadow Transfer Enable **MCSS** and a Pattern Set **PSy** request complete the HW protocol.

1.1.1 The Multi Channel Slice Function

In Multi Channel Mode the Status Bit (CCST) is delayed by 3 Module Clocks. This makes the AND-operation between the Status Bit and the requested Multi-Channel Input value MCIy still due during the Set Pattern request PSy. Optionally, an *External Modulation* or *Status Bit Override* on *external events control* can be added.

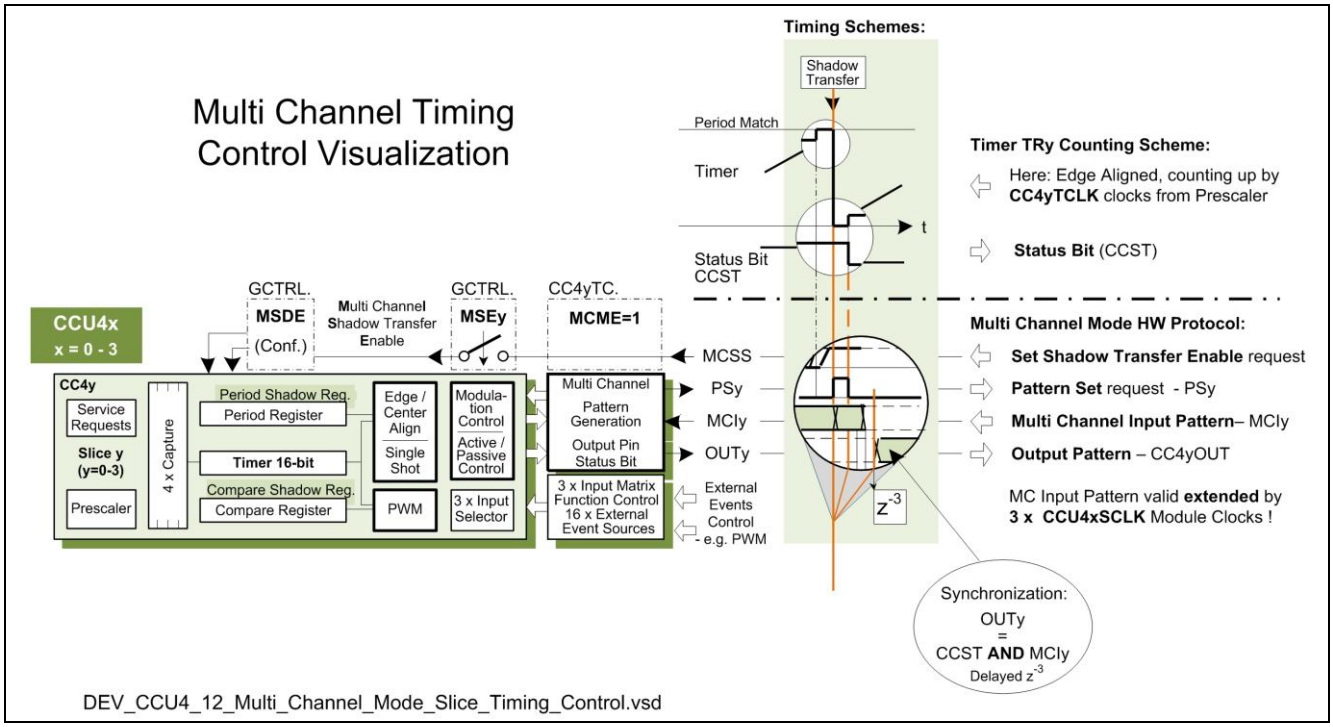

Figure 1 Multi Channel Mode HW Protocol Visualization

2 Multi Channel Synchronization to Shadow Transfers

When it is necessary to perform certain compound shadow transfers synchronously with the update of the multi channel pattern, then the **CCU4xMCSS** trigger can synchronize those scenarios. The synchronization will be performed via the **GCTRL.**MSEy bit field respective the shadow transfers configured on the **GCTRL.**MSDE field.

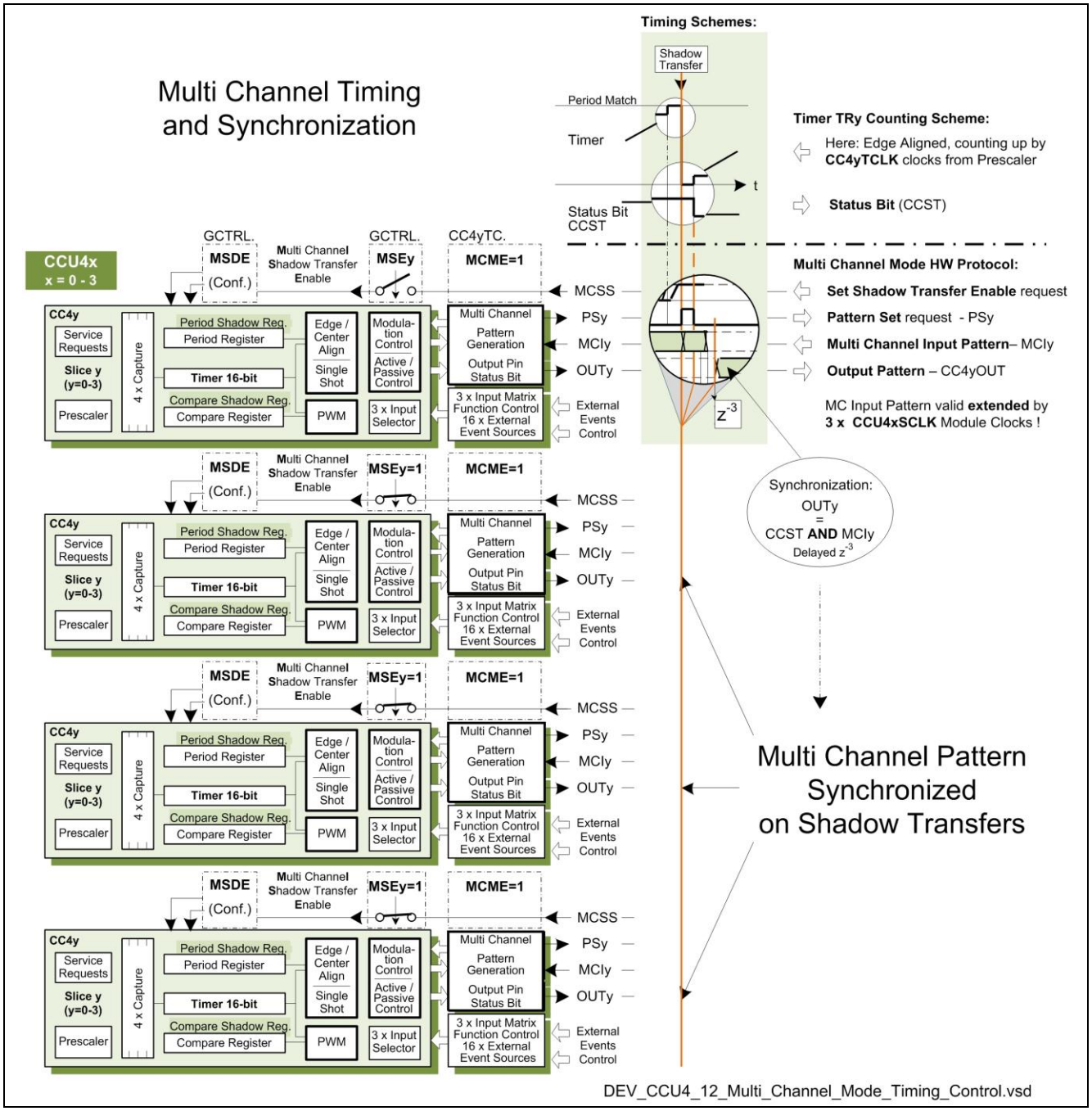

Figure 2 Multi Channel Timing and Synchronization

2.1 CCU4 Shadow Transfers for Coherent Signal Pattern Update

All CAPCOM4 timers, in any slice configuration, no matter how complex or for what signal pattern, are assured *coherent update* by HW of all concerned timer function parameters – by update values in shadow registers that, on *a global* preset request, are transferred simultaneously to all function registers at a **Period-** or **One-Match**.

2.2 The Global Shadow Transfer Set Enable Register

There is one, global register, **GCSS**, carrying *all* enable-flags that have to be preset by SW to selectively activate the targeted Shadow Transfer Requests, which will be cleared by HW after the transfer. By this manner total realtime correctness is achieved also from the logic operations aspect – essential for safe power switching.

2.2.1 Shadow Transfer of Compare Register values

The compare values that are targeted for an update operation have to be written into the **CC4yCRS** shadow registers - *AND the corresponding* **Slice Transfer Set Enable** bits, e.g. **SySE** in **GCSS**, *must be preset – latest within the clock cycle of* **Period Match** (in *Edge Aligned Mode*) or **Period/One Match** (in *Center Aligned Mode*)

2.2.2 Compound Shadow Transfers

Beside the *Compare* (CR) values, there are also the timer *Period* register (PR) and the PWM output *Active/Passive control* bit (PSL) that are updated simultaneously on the **SySE** flag. The *Dithering* or *Floating Prescaler* values (not shown though) may get simultaneous update too by the **SyDSE** & **SyPSE** request flags.

See Figure 3.

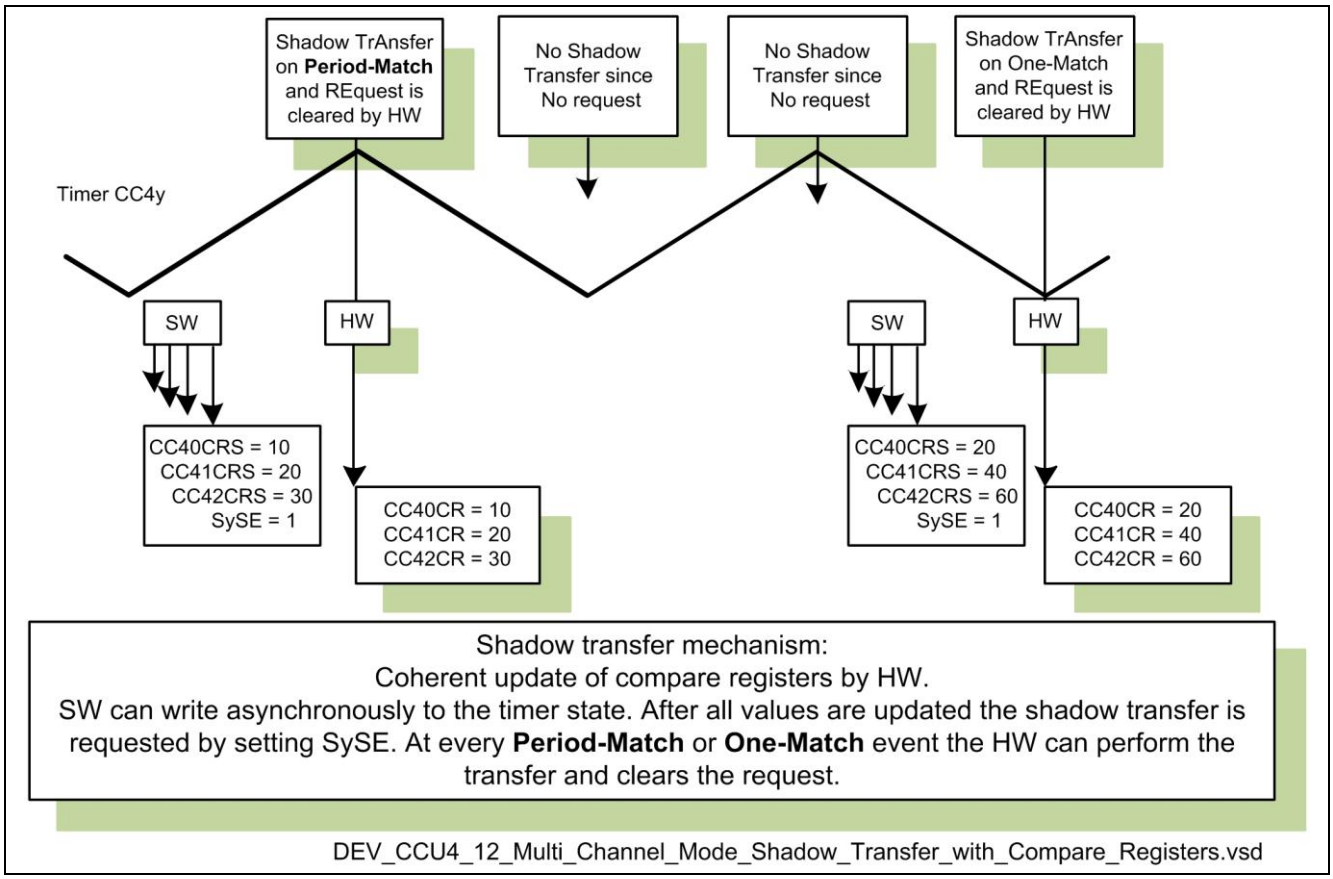

Figure 3 The Shadow Transfer Mechanism (Center Aligned Mode)

3 Start Timers

There are two ways to start a timer, either *directly* by SW setting the Timer Run Bit Set (**TRBS**) - or *indirectly* by HW when a considered event occurs in an external unit that may be selected for this, according to the **Top-Level Connection Matrix** for **External Events Control** – by *CAN, ADC, USIC, IO, CCU4/8, ERU1, POSIF* etc.

3.1 Global Start of CAPCOM Units

There is a way to get a synchronized start of CAPCOM Units, both CCU40/-41/-42,/-43 and CCU80/-81:

- **Global Start by SW**, by the CCUx Global Start Control bits in the Global Start Control register **CCUCON**
- **Global Start by HW**, indirectly, by **External Events Control** using the registers **CC4yINS** and **CC4yCMC**

3.2 Global Start of the CAPCOM Units CCU4 and CCU8

Global start of nearly any number of timers, independently of CAPCOM unit belonging, can to be done! By this manner a *synchronization of the timers* will be achieved and *all timings can be controlled in parallel* – such a output pattern generation of many different kinds – especially in the current topic: *Multi Channel Mode* control.

See Figure 4.

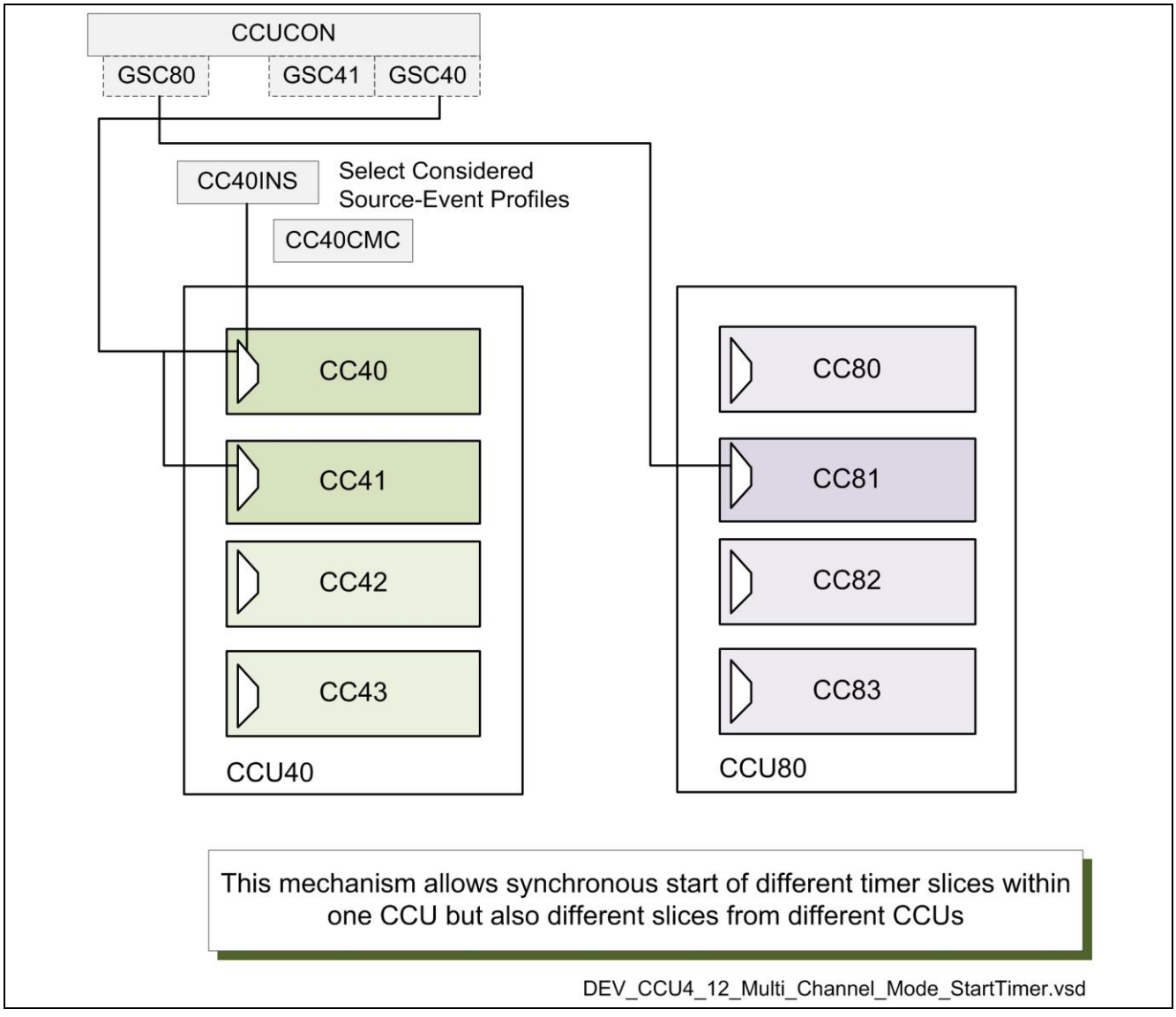

Figure 4 Synchronous Global Start of Timer Slices on different CAPCOM units (CCU4 or CCU8)

4 Getting Started with CCU4 and Multi Channel Mode

4.1 Ultimate Initialization Sequence

Before start and execution of a timer slice SW first time, the CAPCOM4 must have been initialized appropriately according to a specific **CAPCOM4 Initialization Sequence:** 1. *Apply Reset*, 2. *Release Reset*, 3. *Enable Clock*, 4. *Enable Prescaler Block*, 5. *Configure Global Control*, 6. *Configure Slice(s) Functions*, *Interrupts* and *Start-up*.

4.1.1 Ultimate Start-Up Enable

In the last part of the ultimate **CAPCOM4 Initialization Sequence** the startup value(s) for a specific Compare Channel Status of the Timer Slice(s) may be configured by the respective **GCSS.SyTS** bit. After that the default *IDLE mode has to be removed* from the Timer Slice(s) in register **GIDLC** and Start or Global Start can be done!

4.2 Setting Compare Mode Control

Compare Mode is the default state of a CAPCOM timer, i.e. **CC4yTC.**CMOD=0_B. The *compare level* will be set in the compare register **CC4yCR** by the value from the compare *shadow* registers **CC4yCRS** on *shadow transfers - on request within the current period cycle* by the **Slice Transfer Set Enable** bit **GCSS.**SySE.

4.2.1 Setting the Output Pins PASSIVE / ACTIVE Level Control

The PASSIVE/ACTIVE state of a slice *internal* output (i.e. the status bits **CC4yST** is controlled by the compare level and the External Modulation Mode. Whether the *external* output pin **CCU4xOUTy** should be *Passive Low / Active High* or vice versa (e.g. the PWM), is controlled by the **CC4yPSL.**PSL bit.

4.2.2 Initialize and Start the Timer

After the Ultimate Start procedures have been executed, the SW may 1) initialize the slice timer **CC4yTIMER** or just clear it with a **CC4yTCCLR**.TCC bit=1 command and 2) start the timer with a Set Timer Run command in the **CC4yTCSET**.TRBS bit. The timer may be stopped at any time by SW with a **CC4yTCCLR**.TRBC command.

4.3 Multi Channel Mode Enable

A slice has to be set in *multichannel mode* (**CC4yTC.**MCME = 1_B), in order to let its output pattern **CC4yOUT** be *controlled in parallel* by an external pattern present on the Multi Channel Input **MCIy**. Set the multi channel shadow transfer enable bit **GCTRL.**MSEy and configurate the shadow transfers by the **GCTRL.**MSDE bitfield.

4.3.1 Multi Channel Synchronization to Shadow Transfers

When it is necessary to perform certain compound shadow transfers synchronously with the update of the multi channel pattern, then the **CCU4xMCSS** trigger can synchronize those scenarios. The synchronization will be performed via the **GCTRL.**MSEy bit field respective the shadow transfers configured on the **GCTRL.**MSDE field.

4.4 Using the Respective Multi Channel Slices Control Registers

1. **GCTRL.**MSDE **Configurate** transfer for *Compare* / *Compare+Period+Prescaler* / + add even *Dither*

```
for(y=0;y<=3;y++) {
```

2. **CC4yTC.**MCME **Enable** *Multi Channel Mode* (or not = default)
3. **GCTRL.**MSE**y** **Enable** Shadow Transfer Request

4. **CC4yTC.**CMOD **CAPCOM Mode** selection bit should be =0 for *Compare Mode* (default)
5. **CC4yTC.**TCM Set Timer **Counting Mode** selection bit: 0 → *Edge-Aligned* / 1 → *Center-Aligned*
6. **CC4yCRS.**CRS Write the wanted **Compare Value** into the **CRS** compare *shadow* register
7. **CC4yPR.**PR Write the timer count cycle upper limit, **Period Match**, into the *period* register
8. **CC4yPSL.**PSL Set **PASSIVE/ACTIVE** by **0:** *CCU4xOUTy Passive Low*/ **1:** *CCU4xOUTy Active Low*

Optionally:
9. **CC4yTIMER** Initialize a timer start value (or *flush* by the **CC4yTCCLR**.TCC bit=1 command)
10. **CC4yTC.**CLST Enable a shadow transfer each time a clearing action is done by SW or External Event
11. **GCSS.**SySE Set request at least once for a Shadow Transfer to *Initialize the Compare Registers*

}

12. **Global Start** Map start control to **CCUCON** and *External Events Control* to **CC4yINS** & **CC4yCMC**
See Figure 4.

4.5 External Events Control Setup Registers

External Events Control would be useful e.g. for *Start/Stop* or *Trap*. The *Input Selection* register **CC4yINS** maps *External-Event-Sources* and *Edge/Level Mode* conditions to the *Event[2:0] Input Lines*. In turn the *Connection Matrix Control* register **CC4yCMC** maps any *Event[2:0] Input Line* to any of the 11 *Input Functions*, as e.g. *Start*.

4.5.1 Using the External Start Control Register Bitfields

13. **CC4yCMC.**STRTS Input Function Selection and Event Number x=0-2 Input Line Selection bitfield
14. **CC4yINS.**<Event-*x* bitfields> Event-x Source EVxIS (**CC4yINS**[P:A]) and Event Edge Mode EVxEM bitfields
15. **CC4yTC.**<bitfields> Timer Mode selections for e.g. *Start, Extended start, Edge or Center Align, etc*

Note: If External Events Control of Up/Down Counting is used – then the counting scheme is always edge aligned and the Timer Counting Mode bit (TCM) is read-only!

Note: When using the index 'x' in event nomenclature, it should be understood as 'Actual Event[2:0] Number '.

Note: When using the index 'x' in CCU4x nomenclature, it should be understood as 'Actual CCU4 Unit Number'.

Note: The Ultimate Event Number 'x' for the STRTS Input Function is x == <STRTS>-1 (when <STRTS> != 0).

4.5.2 Using the External Stop Control Register Bitfields

Timers can be stopped by External Events Control - even by its own events! The following Control Register Bitfields should be used then:

16. **CC4yCMC.**ENDS Input Function Selection and Event Number x=0-2 Input Line Selection bitfield
17. **CC4yINS.**<Event-*x* bitfields> Event-x Source EVxIS (**CC4yINS**[P:A]) and Event Edge Mode EVxEM bitfields
18. **CC4yTC.**<bitfields> Timer Mode selections for e.g. *Stop, Extended Stop*

Note: The Ultimate Event Number 'x' for the ENDS Input Function is x == <ENDS>-1 (when <ENDS> != 0).

4.5.3 Pseudo Code for the External Start/Stop Control Setup

```
// This is an independent 32-bit based Pseudo Code, regarding that there are no
connections to a certain CAPCOM4 Unit (CCU4x) nor specific Variable Declarations.
// step1
// Make a MASK OPERATION first for bitfield focusing – if not first time – else:
CC4yCMC |= STRTS_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Initialize External Start
// step 2
CC4yINS &=!STRTS_SOURCE_AND_EVENT_MODE_BITFIELDS_MASK; //Focus on STRTS events
CC4yINS |= STRTS_SOURCE_AND_EVENT_MODE_BITFIELDS_PATTERN; //Paste setup pattern
// step 3
CC4yTC &=!STRTS_MODES_AND_EXTENDED_MODE_BITFIELDS_MASK; //Focus on STRTS modes
CC4yTC |= STRTS_MODES_AND_EXTENDED_MODE_BITFIELDS_PATTERN; //Paste setup pattern
// step4
// Make a MASK OPERATION first for bitfield focusing – if not first time – else:
```

```
CC4yCMC |= ENDS_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Initialize External Start
// step 5
CC4yINS &=!ENDS_SOURCE_AND_EVENT_MODE_BITFIELDS_MASK; //Focus on STRTS events
CC4yINS |= ENDS_SOURCE_AND_EVENT_MODE_BITFIELDS_PATTERN; //Paste setup pattern
// step 6
CC4yTC &=!ENDS_MODES_AND_EXTENDED_MODE_BITFIELDS_MASK; //Focus on STRTS modes
CC4yTC |= ENDS_MODES_AND_EXTENDED_MODE_BITFIELDS_PATTERN; //Paste setup pattern
```

5 Runtime Handling of CCU4 and Multi Channel Mode

In this example a part of a *Block Commutation* scheme is demonstrating three timer slices that generate PWM output pattern signals synchronously to an input pattern, controlled by an external source – e.g a POSIF unit. The PWM duty-cycle can be modulated linearly from 0 to 100% (i.e. square-wave) by varying the compare level.

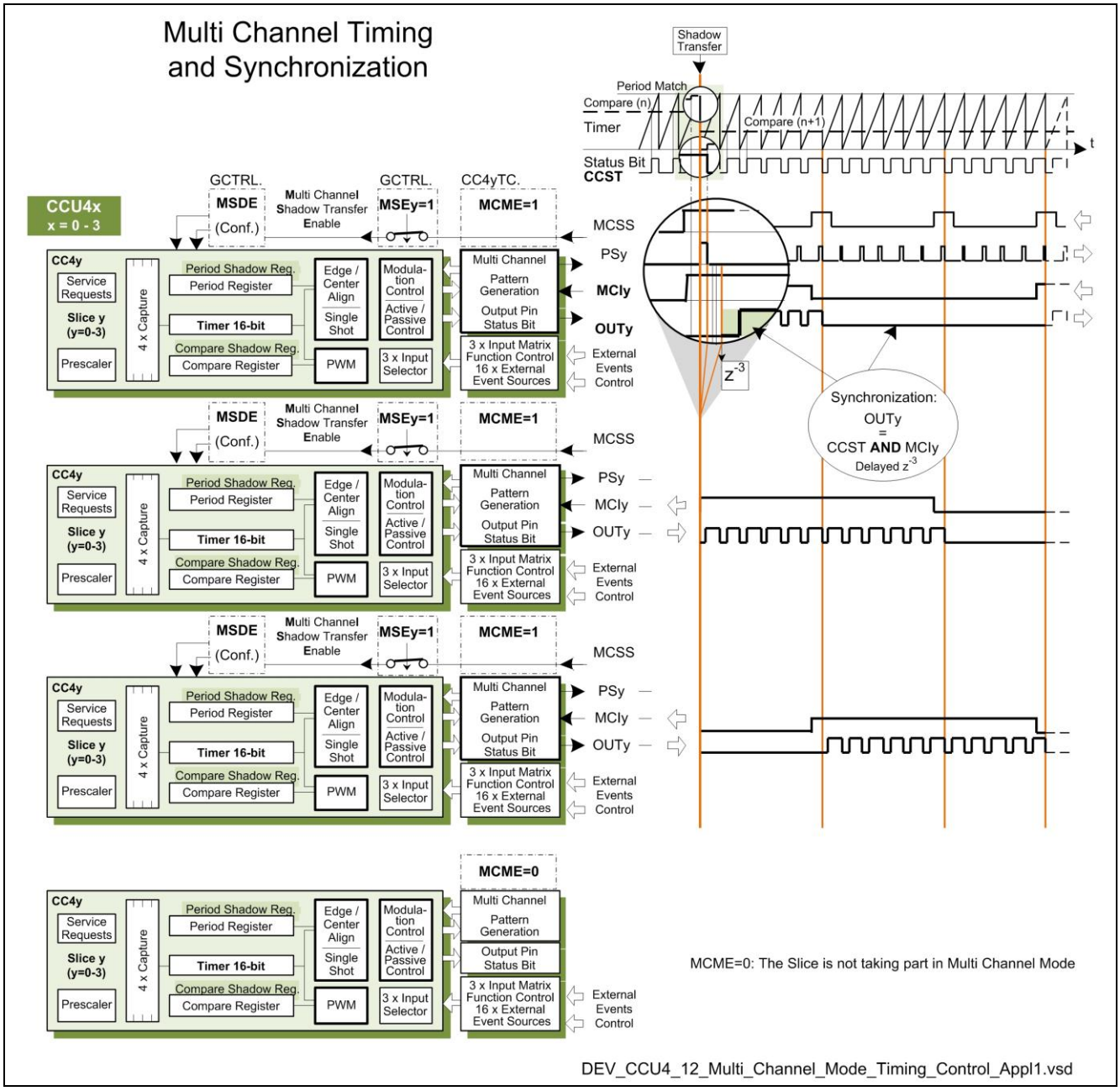

Figure 5 Multi Channel Timing and Synchronization Application in Runtime

6 Example Application for CCU4 and Multi Channel Mode

6.1 Multi Channel Use in BLDC Motor Control

The Position Interface module POSIF in conjunction with a CCU4x unit in multi channel mode enables complete synchronicity between the output state update and the application of a new pattern. With POSIF in Hall Sensor Mode, this can be used for direct control of brushless DC motors, to output the wanted motor control pattern.

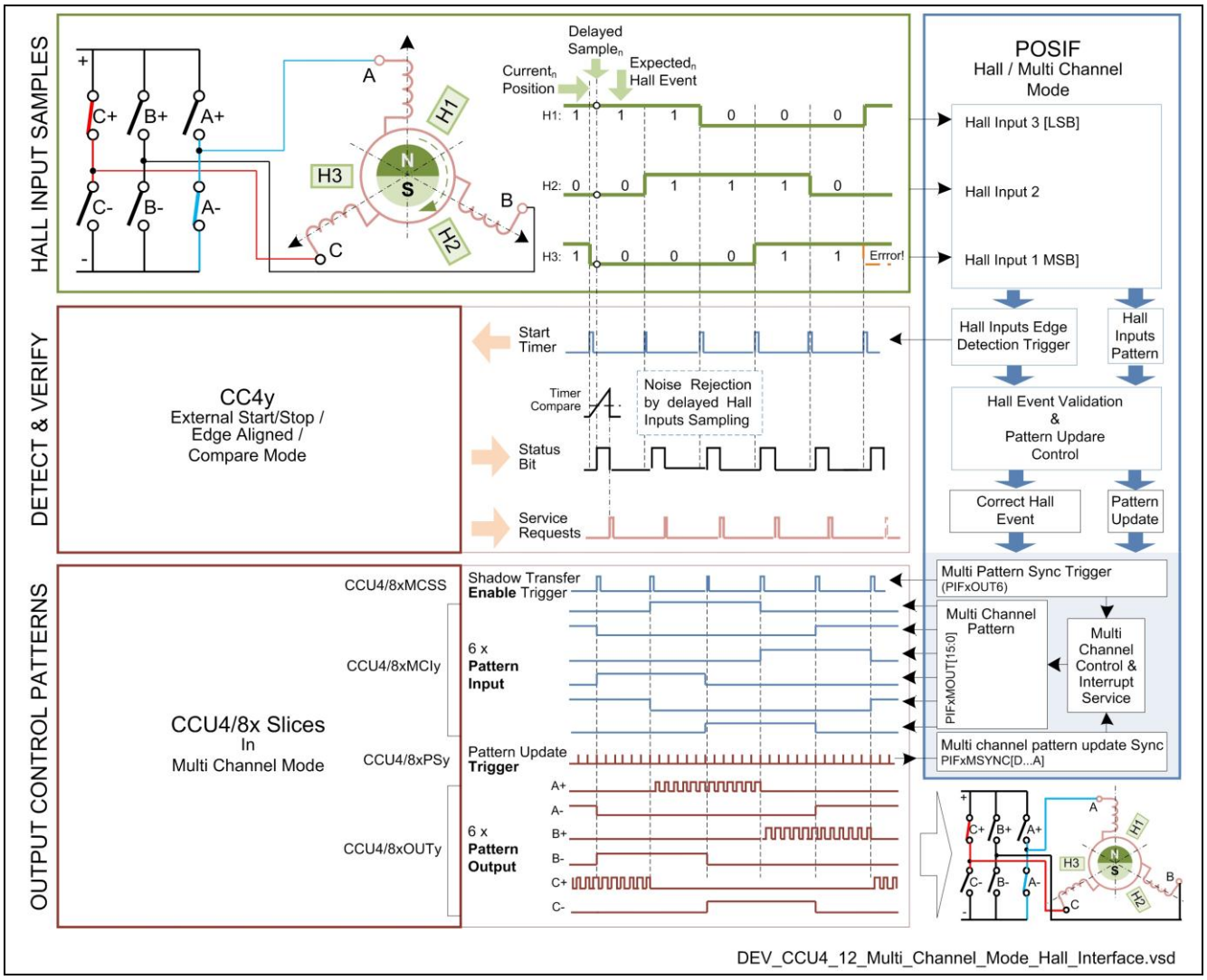

Figure 6 Using POSIF in Hall Sensor Mode and Control by Multi Channel Interface

w w w . i n f i n e o n . c o m

Published by Infineon Technologies AG