XMC4000

XMC4000 Family

POSIF and Multi-Channel Mode

Multi-Channel - Multi-Phase Control

User Guide

Device Guide

<Revision 1.0>, <2012-08-06>

Microcontroller

Edition <2012-08-06>

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.

Revision History	
Page or Item	**Subjects (major changes since previous revision)**
<Revision 1.0>, <2012-08-06>	

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-Channel - Multi-Phase Control ..... Fehler! Textmarke nicht definiert.
1.1 POSIF Multi-Channel Mode Subunit in principle 6

2 POSIF Pin Connections 7
2.1 Multi-Channel Next Pattern Update Set Input 7
2.2 Multi-Channel Pattern Update Synchronization Input 7
2.3 Multi-Channel Pattern Update Request Output 7
2.4 Multi-Channel Pattern Shadow Transfer and Pattern Output 7

3 The POSIF-to-CCU-Slices Multi-Channel Pattern Transfer Synchronization 8

4 POSIFx Multi-Channel Unit Details in Stand-Alone Mode 9
4.1 POSIF Configuration of Next Pattern Update Set Inputs 9
4.2 POSIF Configuration of Pattern Update Synchronization Inputs 9
4.3 POSIF Configuration of Multi-Channel Pattern Update Request Output 10
4.4 POSIF Configuration of Multi-Channel Pattern Shadow Transfer 10

5 POSIFx Multi-Channel Unit Details in Hall-Sensor Mode 11
5.1 POSIF Configuration of Next Pattern Update Set Inputs 11
5.2 POSIF Configuration of Pattern Update Synchronization Inputs 11
5.3 POSIF Configuration of Multi-Channel Pattern Update Request Output 12
5.4 POSIF Configuration of Multi-Channel Pattern Shadow Transfer 12
5.5 Example - Using Multi Channel Control in a BLDC Application 12

6 POSIF Interrupt Registers 13
6.1 POSIF Interrupts/Event requests in Multi-Channel Mode 14

7 Getting Started with POSIF and Multi-Channel Mode 15
7.1 POSIF Global Registers 15
7.2 POSIF Setup when using Stand Alone Multi-Channel Mode 15
7.3 POSIF Setup when using Multi-Channel features in Hall Sensor Mode 15
7.4 POSIF Operation Mode Associated Registers 15
7.4.1 POSIF Associated Configurations of CCU4x/-8x Timers 15
7.4.2 POSIF Synchronized Start of Timers by HW Initialization 15
7.5 Timer Setup 16
7.5.1 Ultimate Initialization Sequence 16
7.5.2 Ultimate Start-Up Enable 16
7.5.3 Setting the Timer Counting Modes 16
7.5.4 Setting Compare Mode Control 16
7.5.5 Setting the Output Pins PASSIVE / ACTIVE Level Control 16
7.5.6 External Events Control Setup Registers 16
7.5.7 Multi Channel Mode Enable 16
7.5.8 Multi Channel Synchronization to Shadow Transfers 17
7.6 Using the POSIF Global and Multi-Channel (MC) Mode Setup Registers 17
7.7 Using the Timer Control Registers for COMPARE Mode 17
7.8 Using the Timer Control Registers for MULTI-CHANNEL Mode 17
7.9 Using the Timer Control Registers for EXTERNAL MODULATION Mode 18
7.9.1 Using the Register Bitfields for EXTERNAL EVENTS CONTROL 18
7.9.2 Using the External Events Control Registers for START 18

List of Figures

Figure 1 POSIFx Multi-Channel Mode Use Cases....6
Figure 2 POSIFx Multi-Channel Mode Subunit in Principle....7
Figure 3 The POSIF-to-CCU-Slices Multi-Channel Pattern Transfer – Profile Example....8
Figure 4 POSIFx Multi-Channel Unit Details in Stand-Alone Mode....9
Figure 5 POSIFx Multi-Channel Unit Details in Hall Sensor Mode....11
Figure 6 Triple-Hall BLDC Motor Commutation Control using CAPCOM units in Multi Channel Mode....13

1 Multi-Channel – Multi Phase Controrl

A POSIFx (x=0-1) is a universal *Position Interface* unit. In conjunction with the CAPCOM units CCU4 or CCU8 it offers powerful solutions for *N-Phase Power Supplies* - or *Motion Control Systems* with various sensors in the feedback loop. The *Multi Channel* is one of the main subunits beside the *Hall* and *Quadrature Decoder* subunits.

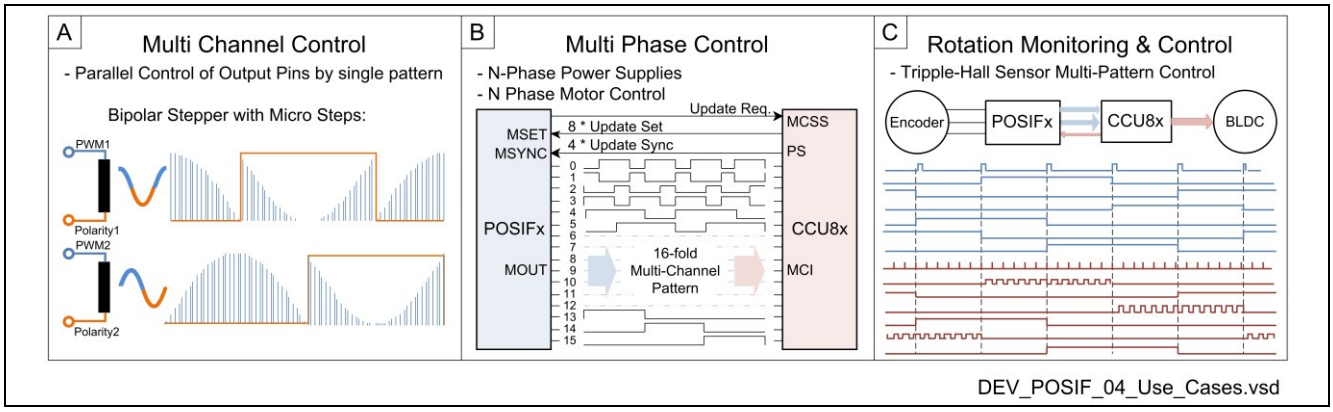

Figure 1 POSIFx Multi-Channel Mode Use Cases

1.1 POSIF Multi-Channel Mode Subunit in principle

See Figure 2.

The Multi-Channel Mode subunit has an input port-pair for the *Next Pattern Update Set* and *Synchronization* – and an output port-pair for the *Pattern Update Request (MCSS)* and *Pattern Out* for upto 16 CAPCOM slices. This HW protocol provides the multi-pattern flow that is needed for an upto 16-fold channel/phase control loop.

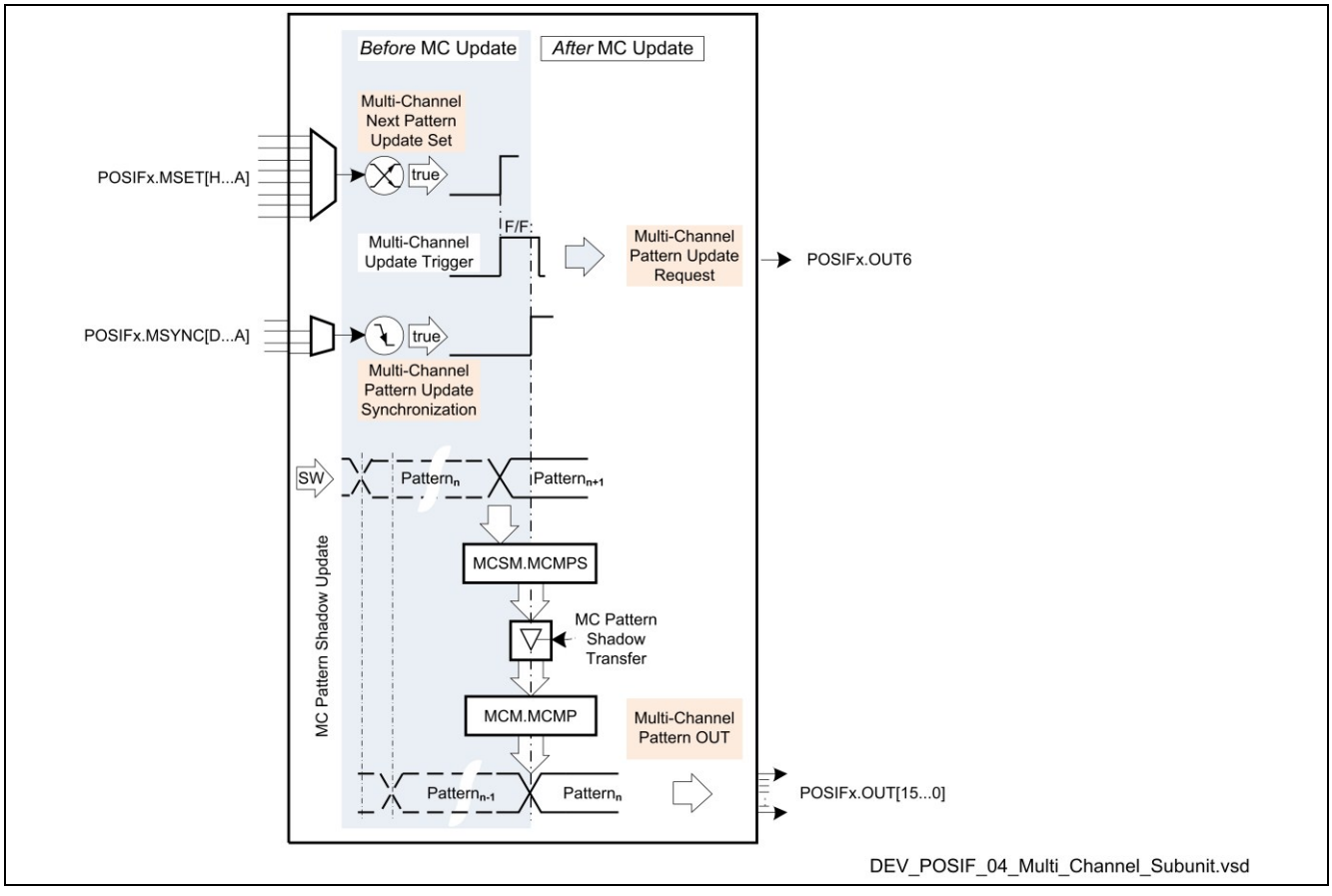

Figure 2 POSIFx Multi-Channel Mode Subunit in Principle

2 POSIF Pin Connections

Sinse a POSIF module has flexible configuration schemes, in order to cover a large universe of *Power* or *Motion* control applications, the *Pin Connections* have *Alternative Functions* - depending on the bitfield **PCONF**.FSEL setup:

00_B: Hall Sensor Mode,

01_B: Quadrature Decoder Mode,

10_B: Stand-alone **Multi Channel Mode**,

11_B: Stand-alone **Multi Channel Mode** & Quadrature Decoder Mode (i.e. a combination mode)

2.1 Multi-Channel Next Pattern Update Set Input

An *Update Set* request by SW or from a selected CAPCOM slice to one of the POSIFx.**MSET[H...A]** inputs will *set,* if enabled, a Flip-Flop (*F/F*) - the MCMF.**MSS** bit. This bit is a *Multi-Channel Update Trigger* that enables a *Set Shadow Transfer Request* for the POSIF *Next Pattern Out* - *and* a '*MCSS*' for each involved CAPCOM slice.

2.2 Multi-Channel Pattern Update Synchronization Input

After a *Next Pattern Update Set* has been asserted by setting the *Multi-Channel Update Trigger* Flip-Flop (*F/F*), the pattern update can be *synchronized* with e.g. a PWM signal. This signal should be mapped to one of the POSIFx.**MSYNC[D...A]** inputs, which will assert "*synchronization event = true*" upon a *falling edge* on this input.

2.3 Multi-Channel Pattern Update Request Output

The POSIFx.**OUT6** output pin asserts a *Multi-Pattern Set Shadow Transfer Request Enable* by its pulse to the *MCSS* inputs of all CAPCOM slices that are selected for *Multi-Channel Pattern Update* in parallel. This pulse is the *Multi-Channel Update Trigger* Flip-Flop (*F/F*) status - and will be *cleared* after a multi-pattern update is done.

2.4 Multi-Channel Pattern Shadow Transfer and Pattern Output

See Figure 3.

Upon a POSIFx.**MSYNC[D...A]** input *falling edge*, a *Multi-Channel Pattern Shadow Transfer* is performed to the POSIFx.**MOUT[15…0]** output. This pattern may control, via the input (**MCI**y), the output (**OUT**y) of all slices that are *Multi-Channel Mode Enabled* (by **MCME**) - **and** *Slice Shadow Transfer Enabled* (globally by GCTRL.**MSE**y).

3 The POSIF-to-CCU-Slices Multi-Channel Pattern Transfer Synchronization

See Figure 3

A slice in *Multi-Channel Mode* prolongs its *status bit* (**CCST**) by 3 module clocks, to be valid even *after* a *Pattern Set* request (**PS**) has been asserted. Thus the **OUT**y operation of such a slice will include the requested *Multi-Channel Input Pattern* (**MCIy**) **AND** (**CCST**) – and so the *POSIF-to-CCU-Slices* pattern transfer will be provided.

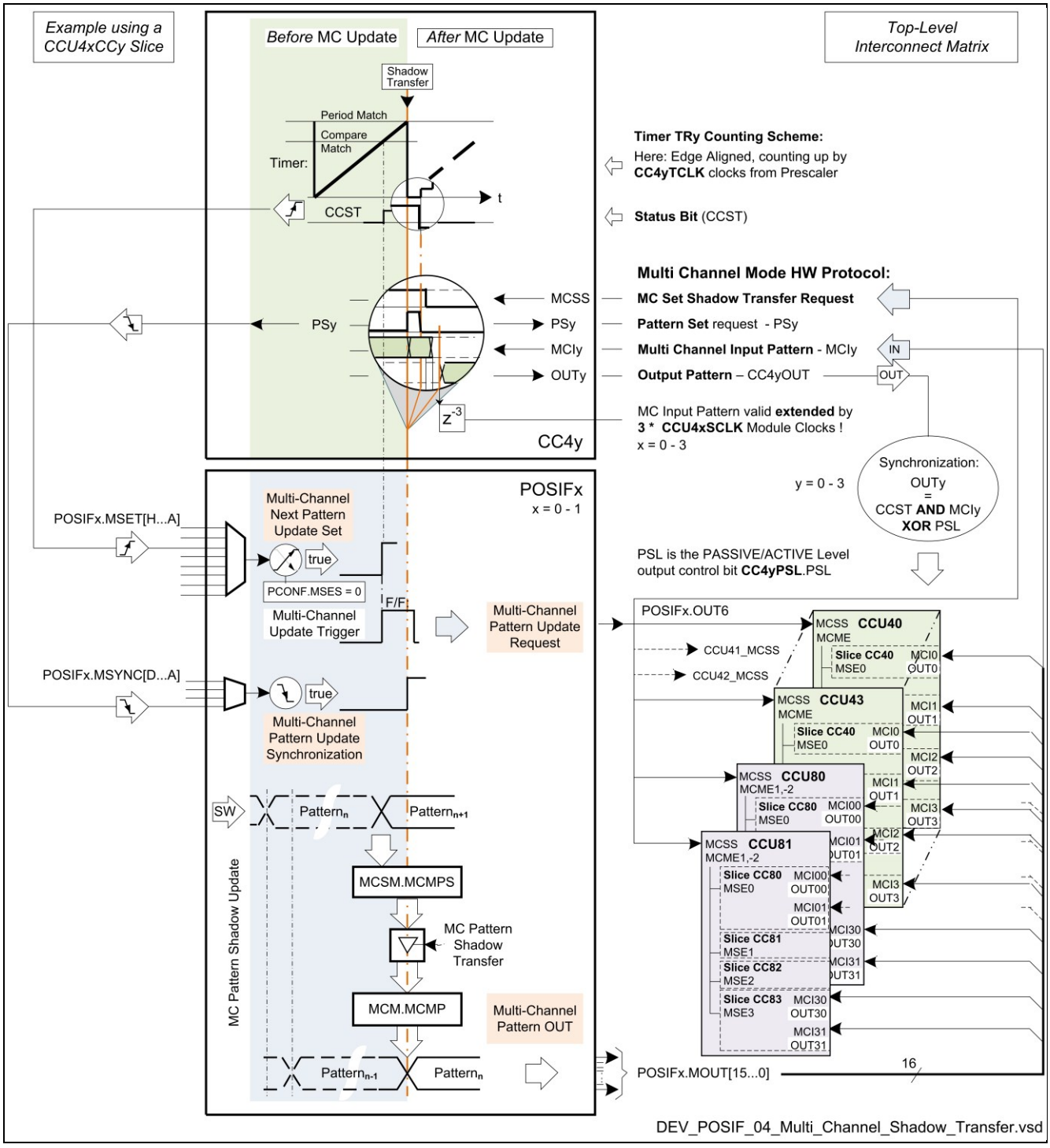

Figure 3 The POSIF-to-CCU-Slices Multi-Channel Pattern Transfer – Profile Example

4 POSIFx Multi-Channel Unit Details in Stand-Alone Mode

See Figure 4.

4.1 POSIF Configuration of Next Pattern Update Set Inputs

A *Multi-Channel Next Update Set* can be asserted either by SW (selected by PCONF.**MCUE**=1_B) - or via one of the POSIFx.**MSET[H...A]** inputs (selected by the PCONF.**MSETS** bitfield) upon a signal transition (selected by PCONF.**MSES**). The MCMF.**MSS** bit (*F/F*) will be set – and cleared by "*synchronization event = true*" or by SW.

4.2 POSIF Configuration of Pattern Update Synchronization Inputs

After the *Multi-Channel Update Trigger* request into Flip-Flop (*F/F*) has been set, the pattern update can be *synchronized* with e.g. a PWM signal. This signal should be mapped to one of the POSIFx.**MSYNC[D...A]** inputs (selected by the PCONF.**MSYNS** bitfield), which will assert a "*synchronization event = true*" upon a *falling edge*.

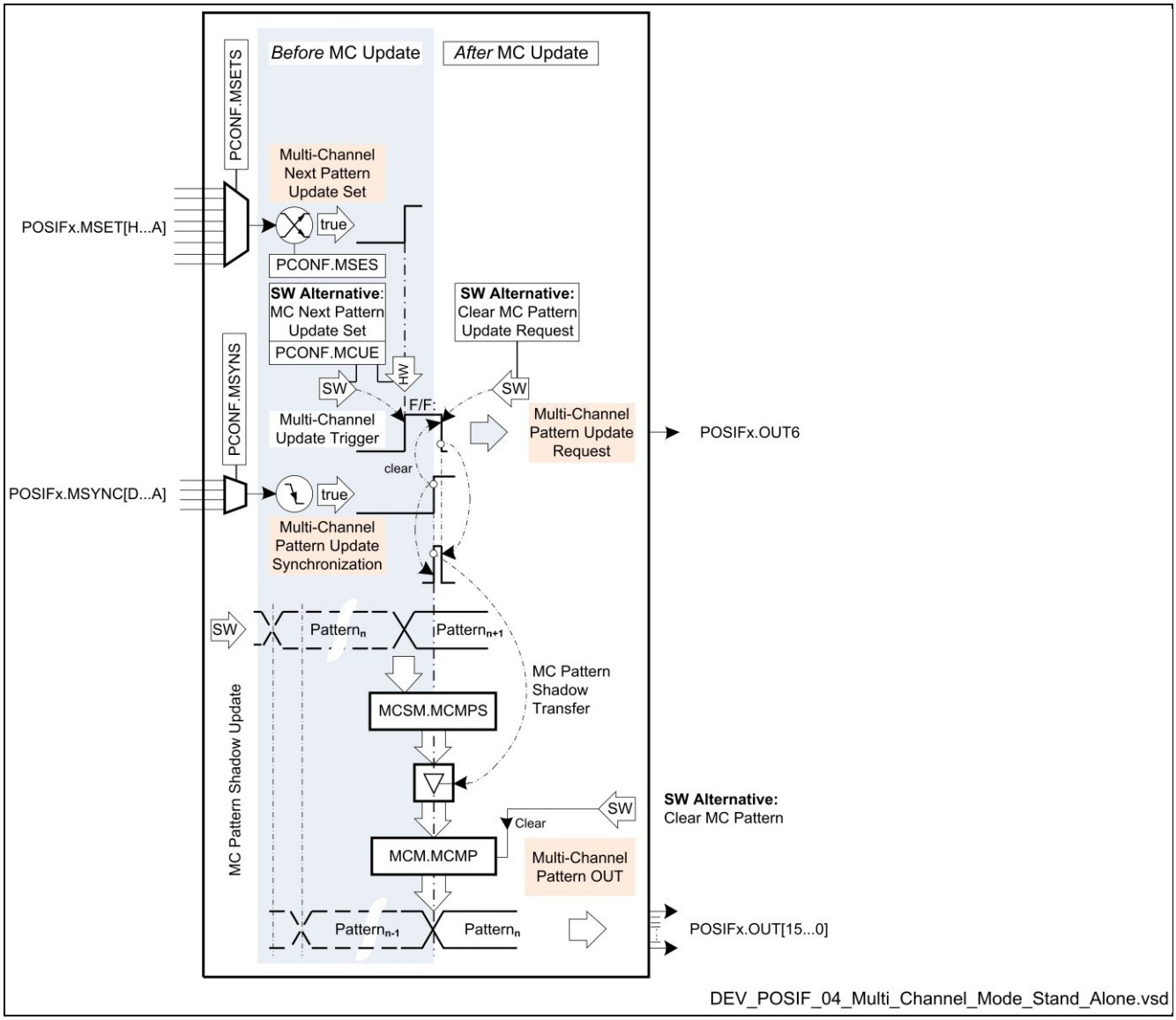

Figure 4 POSIFx Multi-Channel Unit Details in Stand-Alone Mode

4.3 POSIF Configuration of Multi-Channel Pattern Update Request Output

The POSIFx.**OUT6** output pin asserts a *Multi-Pattern Set Shadow Transfer Request Enable* by its pulse to the *MCSS* inputs of all CAPCOM slices that are selected for *Multi-Channel Pattern Update* in parallel. This pulse is the *Multi-Channel Update Trigger* Flip-Flop (*F/F*) status - that will be *cleared* by HW (or SW) on *Update Done*.

4.4 POSIF Configuration of Multi-Channel Pattern Shadow Transfer

Upon a POSIFx.**MSYNC[D...A]** input *falling edge*, a *Multi-Channel Pattern Shadow Transfer* is performed from the MCSM.**MCMPS** shadow register to the POSIFx.**MOUT[15…0]** register output pins. This output register should be maintained by SW. – and may be cleared at any time by SW to stop e.g. PWM generating units.

5 POSIFx Multi-Channel Unit Details in Hall-Sensor Mode

See Figure 5.

5.1 POSIF Configuration of Next Pattern Update Set Inputs

A *Multi-Channel Next Update Set* can be asserted either by SW (selected by PCONF.**MCUE**=1_B) - or via one of the POSIFx.**MSET[H...A]** inputs (selected by the PCONF.**MSETS** bitfield) upon a signal transition (selected by PCONF.**MSES**). However, the *MC Update Trigger* MCMF.**MSS** bit (*F/F*) will just be set at ***Correct Hall Events***!

5.2 POSIF Configuration of Pattern Update Synchronization Inputs

After the *Multi-Channel Update Trigger* request into Flip-Flop (*F/F*) has been set, the pattern update can be *synchronized* with e.g. a PWM signal. This signal should be mapped to one of the POSIFx.**MSYNC[D...A]** inputs (selected by the PCONF.**MSYNS** bitfield), which will assert a "*synchronization event = true*" upon a *falling edge*.

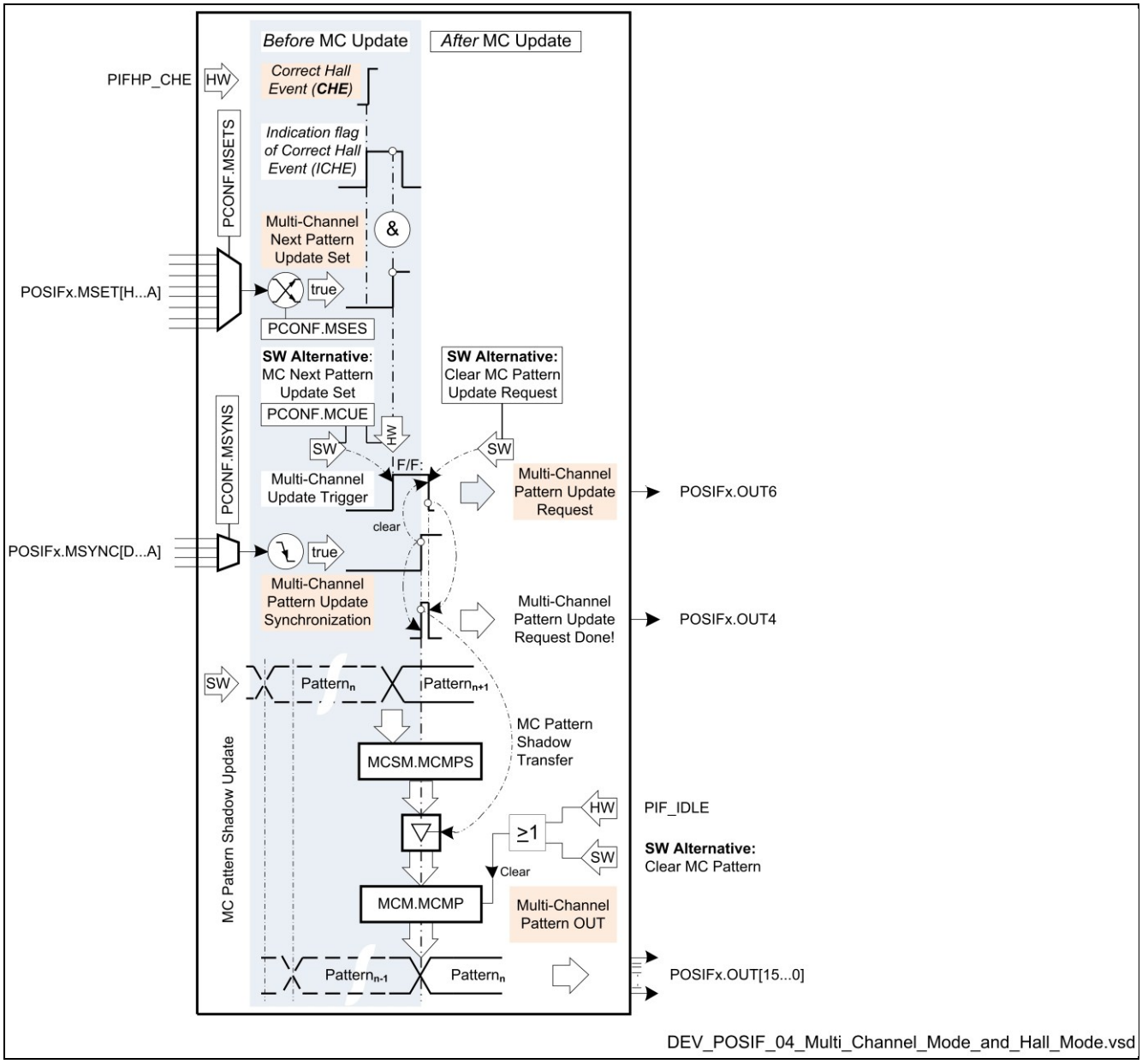

Figure 5 POSIFx Multi-Channel Unit Details in Hall Sensor Mode

5.3 POSIF Configuration of Multi-Channel Pattern Update Request Output

The POSIFx.**OUT6** output pin asserts a *Multi-Pattern Set Shadow Transfer Request Enable* by its pulse to the *MCSS* inputs of all CAPCOM slices that are selected for *Multi-Channel Pattern Update* in parallel. The POSIFx.**OUT4** output pin asserts "*synchronization event* = *true*" - and claims "*Update done"*, when it is cleared.

5.4 POSIF Configuration of Multi-Channel Pattern Shadow Transfer

Upon a POSIFx.**MSYNC[D...A]** input *falling edge*, a *Multi-Channel Pattern Shadow Transfer* is performed from the MCSM.**MCMPS** shadow register to the POSIFx.**MOUT[15…0]** register output pins. The shadow register should be updated by SW. The output can be reset by SW or HW ("*POSIF Idle"*), to stop PWM generating units.

5.5 Example - Using Multi Channel Control in a BLDC Application

See Figure 6.

The **CCU4/8OUT**-pin output level of any slice that has been set in *multichannel mode* (**CC4/8yTC.**MCME = 1_B) can be **&**-*controlled in parallel* by an external level that is present on its Multi Channel Input **MCI**-pin. To get it all synchronized, a Set Shadow Transfer Enable **MCSS** and a Pattern Set **PSy** request complete the HW protocol.

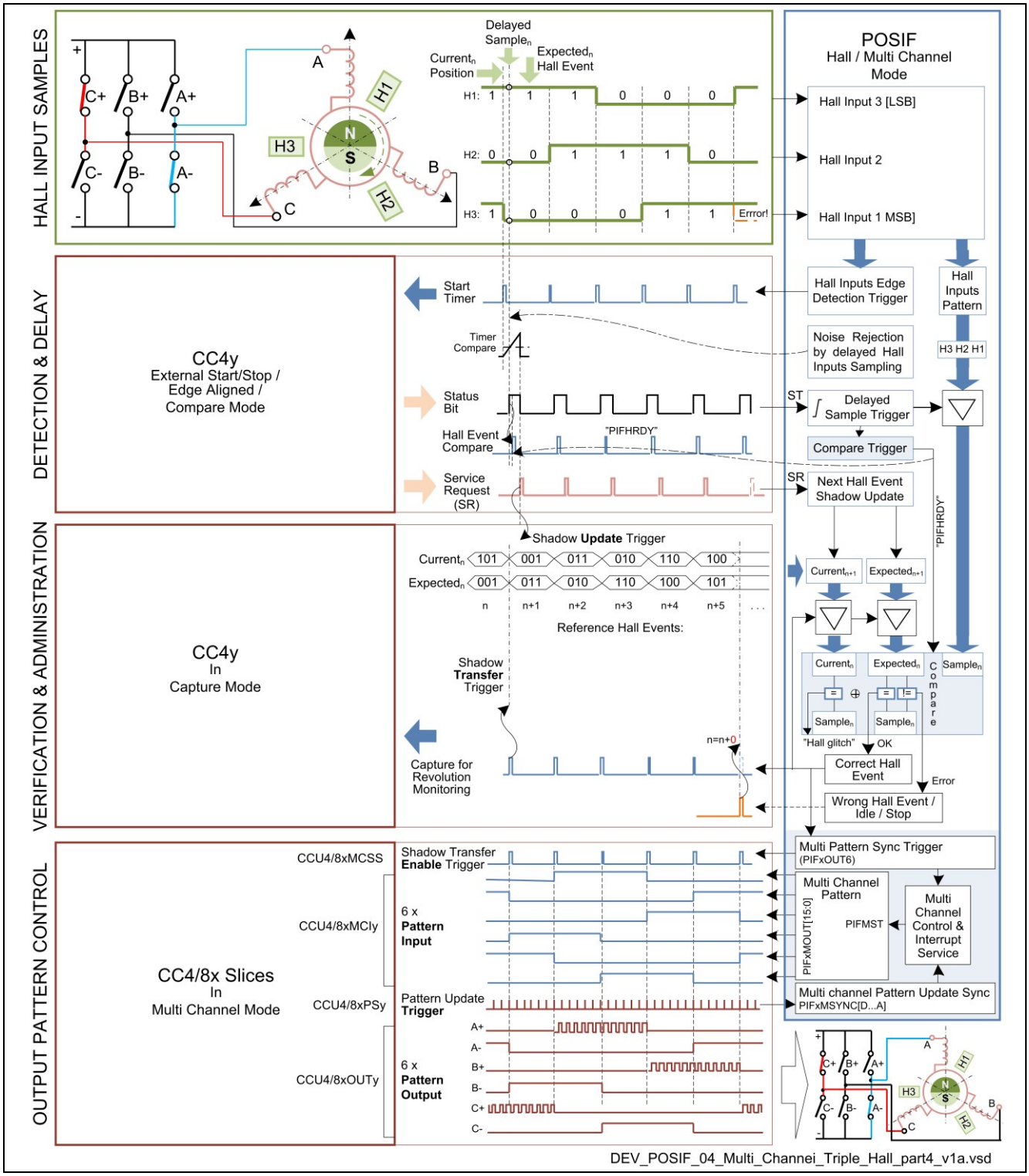

Figure 6 Triple-Hall BLDC Motor Commutation Control using CAPCOM units in Multi Channel Mode

6 POSIF Interrupt Registers

PFLG POSIF Interrrupt Flags
PFLGE POSIF Interrrupts Enable
SPFLG Set POSIF Interrrupt Flag
RPFLG Reset POSIF Interrrupt Flag

6.1 POSIF Interrupts/Event requests in Multi-Channel Mode

The MC Mode interrupt/event request flag (**MSTS**) in register **PFLG** can be enabled by the **PFLGE** register bit *MC Shadow Transfer Enable* (**EMST**). This flag (**MSTS**) status should be mapped to the POSIFx.SR0 /-SR1 *Service Request Line* by the **PFLGE** register bit *MC Pattern Update Event Service Request Selector* (**MSTSEL**).

7 Getting Started with POSIF and Multi-Channel Mode

7.1 POSIF Global Registers

PCONF	POSIF Global **Control**	All modes HW and Protocol Controlling Register
PSUS	Suspend Configuration	Suspend configuration register for *stop* / *sync:d stop*
PRUNS	POSIF run bit **Set**	Via this register it is possible to set the module run bit
PRUNC	POSIF run bit **Clear**	Via this register the run bit or POSIF status can be reset
.		
PRUN	POSIF run bit Status	Run bit status can just be read: 0_B=*IDLE* / 1_B=*running*
MIDR	Module ID register	Module Identification register: *Revision*, *Type*, *Number*

7.2 POSIF Setup when using Stand Alone Multi-Channel Mode

At the activation of a POSIF unit in the system start-up procedures (after hard reset), the POSIF control register **PCONF** should be written to for the Function Selection, i.e. the bitfield **PCONF**.FSEL = 10_B for *Multi Channel Mode* enable - or **PCONF**.FSEL = 11_B for a combined *Stand Alone Multi Channel & Quadrature Decoder Mode*.

7.3 POSIF Setup when using Multi-Channel features in Hall Sensor Mode

The POSIF control function register bitfield **PCONF**.FSEL should be written 00_B for *Hall Sensor Mode* enable – **even if** the *Multi Channel Mode* subunit features are intended to be used for a *Multi-Channel Pattern Update* in the output commutation control flow. The **PCONF**.FSEL should be written within the system start-up procedure.

7.4 POSIF Operation Mode Associated Registers

For *Stand-Alone Multichannel Mode*:	use **MCSM/MCM** for data and **MCMF/MCMS/MCMC** for control
For *Hall Sensor Mode*:	use additionally **HALPS/HALP**
For *Quadrature Decoder Mode*:	use additionally **QDC**

7.4.1 POSIF Associated Configurations of CCU4x/-8x Timers

These modes are mainly controlled by the **CC4yTC** (**CC8yTC**) registers – and the *External Events Control* of e.g. a **CC4y** can be set by **CC4yCMC** and **CC4yINS** registers for *Start*, *Flush/Start*, *Capture*, *Capture/Clear* etc. - See ***Timer Setup***.

7.4.2 POSIF Synchronized Start of Timers by HW Initialization

After the Ultimate Timer Start procedures have been executed - and If the synchronous start function of the POSIF is being used inside the associated Capture/Compare units, then one just needs to set the *Run Bit* of the module, by writing a 1_B into the **PRUNS**.SRB bit.

7.5 Timer Setup

In the following initialization and setup descriptions below there have been chosen the CCU4x CAPCOM units as associated Capture/Compare units for the preceedings of the Getting Started demonstration with the POSIF.

7.5.1 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*.

7.5.2 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!

7.5.2.1 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.

7.5.3 Setting the Timer Counting Modes

The Timer Counting Mode is *Edge Aligned* by default, i.e. **CC4yTC.**TCM=0_B. The *Center Aligned* mode will be set by TCM= 1_B. Single Shot Mode is set by the **CC4yTC.**TSSM bit. *Start* (and even *Stop*) may be controlled by External Events, added by extended functionalities e.g. *Start, Extended Start respective Stop, Extended Stop*.

7.5.4 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.

7.5.5 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.

7.5.6 External Events Control Setup Registers

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 the three *Event[2:0] Input Lines* to any of the 11 *Input Functions*. The *Timer Control* register **CC4yTC** enables different *Mode Alternatives*.

7.5.7 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.

7.5.8 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.

7.6 Using the POSIF Global and Multi-Channel (MC) Mode Setup Registers

The following Multi-Channel (MC) Protocol Controlling Global Registers should be setup:

1. **PCONF**.FSEL Select Mode: 00_B for Hall (incl. MC), 10_B for Stand Alone, 11_B for MC & Quadrature
2. **PCONF**.MCUE Select MC Pattern Update Enable control: "by HW" (default) by 0_B or "by SW" by 1_B
3. **PCONF**.MSETS Select the input POSIFx.**MSET[H...A]** signal that is used to enable a MC pattern update
4. **PCONF**.MSYNS Select the PWM synchronization input POSIFx.**MSYNC[D...A]** for a MC pattern update
5. **PCONF**.MSES Select the PWM synchronization eignal trigger edge: 0_B for *rising* / 1_B for *falling* edge

The following Multi-Channel (MC) Mode Registers should be considered:

6. **MCM**.MCMP MC Mode Pattern Contains *MC Pattern Out* applied to POSIFx.**OUT[15:0]**
7. **MCSM**. MCMPS MC Mode Shadow Pattern This is passed into the **MCM**.MCMP on *Transfer* events
8. **MCMF**.MSS MC Mode **Flag** status *Read Only*: *MC Update Trigger* request bit *F/F* status
9. **MCMS**.<bitfields> MC Mode Control **Set** Perform ***Immidiate*** *Pattern Update* - or just set *Enable*
10. **MCMC**.<bitfields> MC Mode Control **Clear** Clears the *MC Pattern Out*, the **MCMF**.MSS bit - or both

7.7 Using the Timer Control Registers for COMPARE Mode

There is practically just one and only register that is concerned here – namely the **CC4yTC** register bitfields:

11. **CC4yTC.**CMOD CAPCOM Mode selection bit may be used: 0 → *Compare* / 1 → *Capture* mode
12. **CC4yTC.**TCM Timer **Counting Mode selection** bit: 0 → *Edge-Aligned* / 1 → *Center-Aligned*

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!

13. **CC4yPR.**PR Write the timer count cycle upper limit into the *period* register
14. **CC4yCRS.**CRS Write the wanted *Compare Level* value into the compare *shadow* register

Options:

15. **CC4yPSL.**PSL *PASSIVE/ACTIVE* control bit: (*CCU4xOUTy)* = (*CCUxSTy*) xor (*CC4yPSL.PSL*)
16. **CC4yTC.**STRM *Extended Start* on *External Start*. 0_B: *Start (=Set Run Bit)* / 1_B: *Flush* & *Start* Timer
17. **CC4yTC.**ENDM *Extended Stop* on *Ext. Stop*. 00_B: *Stop* / 01_B: *Flush* / 10_B: *Stop* & *Flush* /11_B:reserved
18. **CC4yTIMER** Initialize a timer start value (or *flush* by the **CC4yTCCLR**.TCC bit=1 command)
19. **CC4yTCSET**.TRBS This is the Start Timer by SW command.

7.8 Using the Timer Control Registers for MULTI-CHANNEL Mode

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

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

21. **CC4yTC.**MCME **Enable** *Multi-Channel Mode* (or not = default)
22. **GCTRL.**MSE**y** **Enable** Shadow Transfer Request
23. **CC4yTC.**CMOD **CAPCOM Mode** selection bit should be =0 for *Compare Mode* (default)
24. **CC4yTC.**TCM Set Timer **Counting Mode** selection bit: 0 → *Edge-Aligned* / 1 → *Center-Aligned*
25. **CC4yCRS.**CRS Write the wanted **Compare Value** into the **CRS** compare *shadow* register
26. **CC4yPR.**PR Write the timer count cycle upper limit, **Period Match**, into the *period* register
27. **CC4yPSL.**PSL Set **PASSIVE/ACTIVE** by **0:** *CCU4xOUTy Passive Low*/ **1:** *CCU4xOUTy Active Low*

Optionally:

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

}

31. **Global Start** Map start control to **CCUCON** and *External Events Control* to **CC4yINS** & **CC4yCMC**

7.9 Using the Timer Control Registers for EXTERNAL MODULATION Mode

(See *Using the External Events Control* below.)

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

1. **CC4yCMC.**MOS Input Function Selection and Event Number x=0-2 Input Line Selection bitfield
2. **CC4yINS.**<Event-x bitfields> Event-x Source EVxIS (CC4yINS[P:A]) and Level Mode EVxLM bitfields
3. **CC4yTC**.EMT Timer Control bit for Status Bit Clear or Output Gating Only
4. **CC4yTC**.EMS Timer Control bit for Unsynchronized or Synchronized modulation

}

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 MOS Input Function is always x == <MOS>-1 (when <MOS> != 0).

Note: By "Synchronization" should be understood that the Modulated Signal keeps its pulse width when active (i.e. there is no shortening or cut of any active pulses by the Modulation Signal edges).

7.9.1 Using the Register Bitfields for EXTERNAL EVENTS CONTROL

1. **CC4yCMC.**<function> Input function selection and Event Number (x=0-2) Input Line Selection bitfield
2. **CC4yINS.**<Event-x bitfields> Event-x source EVxIS (CC4yINS[P:A]), edge EVxEM or level EVxLM mode
3. **CC4yTC.**<Mode bitfields> Timer Mode selections e.g. Modes/Extended modes, Counting Schemes etc

7.9.2 Using the External Events Control Registers for START

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

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

Published by Infineon Technologies AG