XMC1000, XMC4000

Microcontroller Series for Industrial Applications

Capture Compare Unit – CCU4

- ✓ Introduction to the Basic Features
- ✓ Output Pattern Generation using the Compare Mode
- ✓ Advanced Signal Measurement using the Capture Mode
- ✓ Single Shot Mode
- ✓ External Events Control
- ✓ Signal Quantification & Quality Enhancement by Dithering
- ✓ Autonomic Signal Measurement Range Adaption using the Floating Prescaler
- ✓ External Modulation
- ✓ Output State Override
- ✓ External Load
- ✓ Counting Scheme
- ✓ Top Level Interconnect
- ✓ Multi-Phase Output control using the Multi-Channel Mode

Device Guide

V1.0 2013-04

Microcontrollers

Edition 2013-04

Published by
Infineon Technologies AG
81726 Munich, Germany

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)**
V1.0, 2013-04	

Table of Contents

Introduction to the Basic Features

1 CCU Basics

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

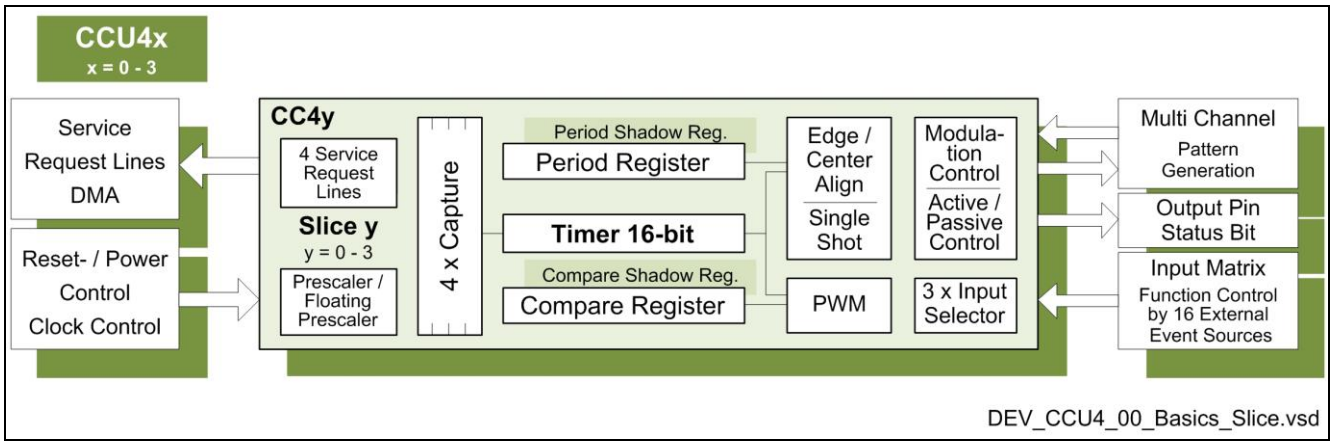


Figure 1 The Timer Slice Block Diagram

Basic Timer Functions

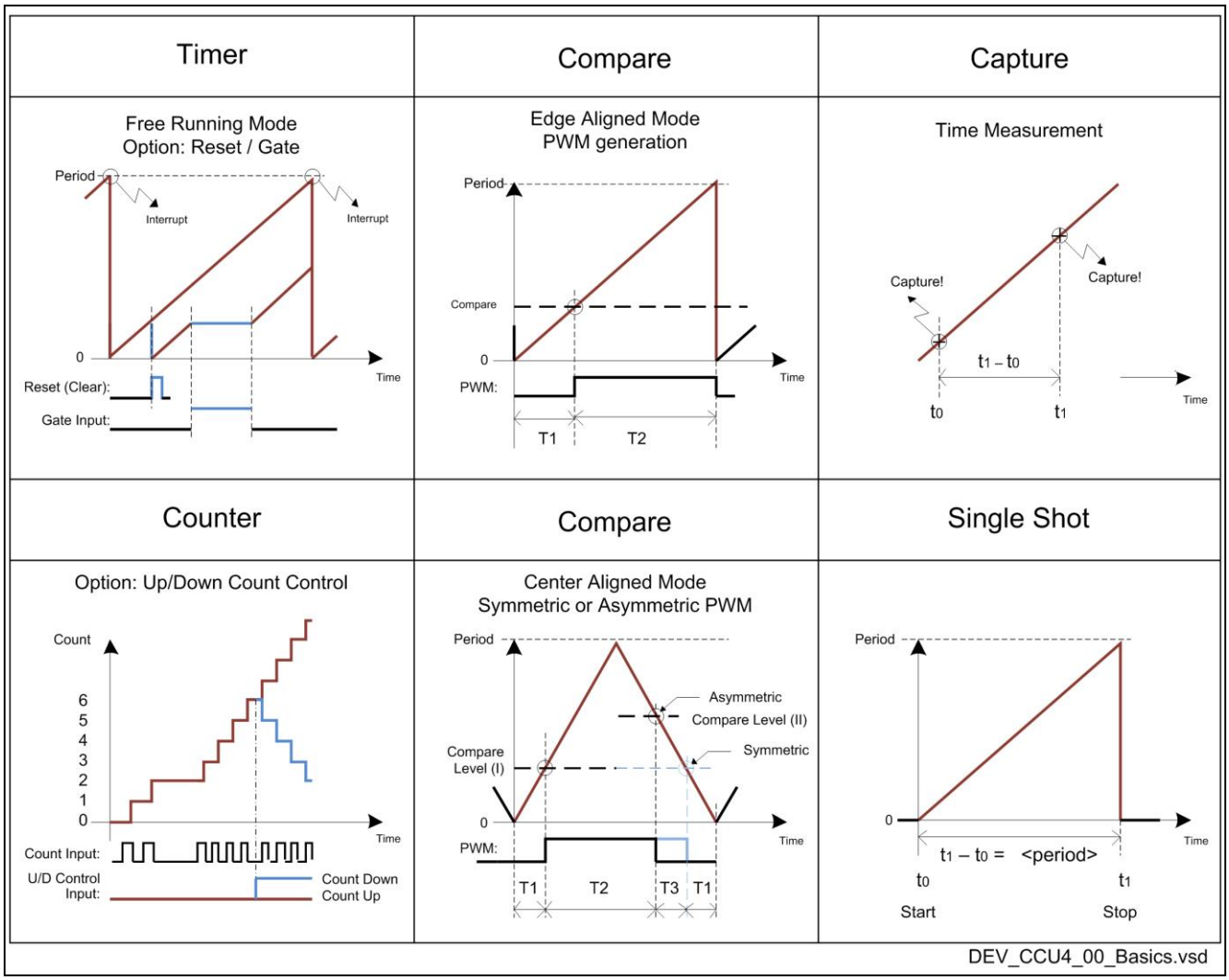


Figure 2 Basic Functions of each Timer Slice

Each timer slice can handle all basic modes and the typical options illustrated in the figure above.

The Compound CAPCOM4 System CCU40 – CCU43

Each CCU4x has four 16-bit timer slices CC4y (y=3-0), which can be concatenated up to 64-bit.

A slice has:

- 1 Timer
- 4 Capture register
- 1 Period register
- 1 Compare register

Both the Period and Compare registers have shadow registers. Each slice can work independently in different modes, but they can also be synchronized, even to other CCU4x slices. They perform multi-channel/multi-phase pattern generation with parallel updates.

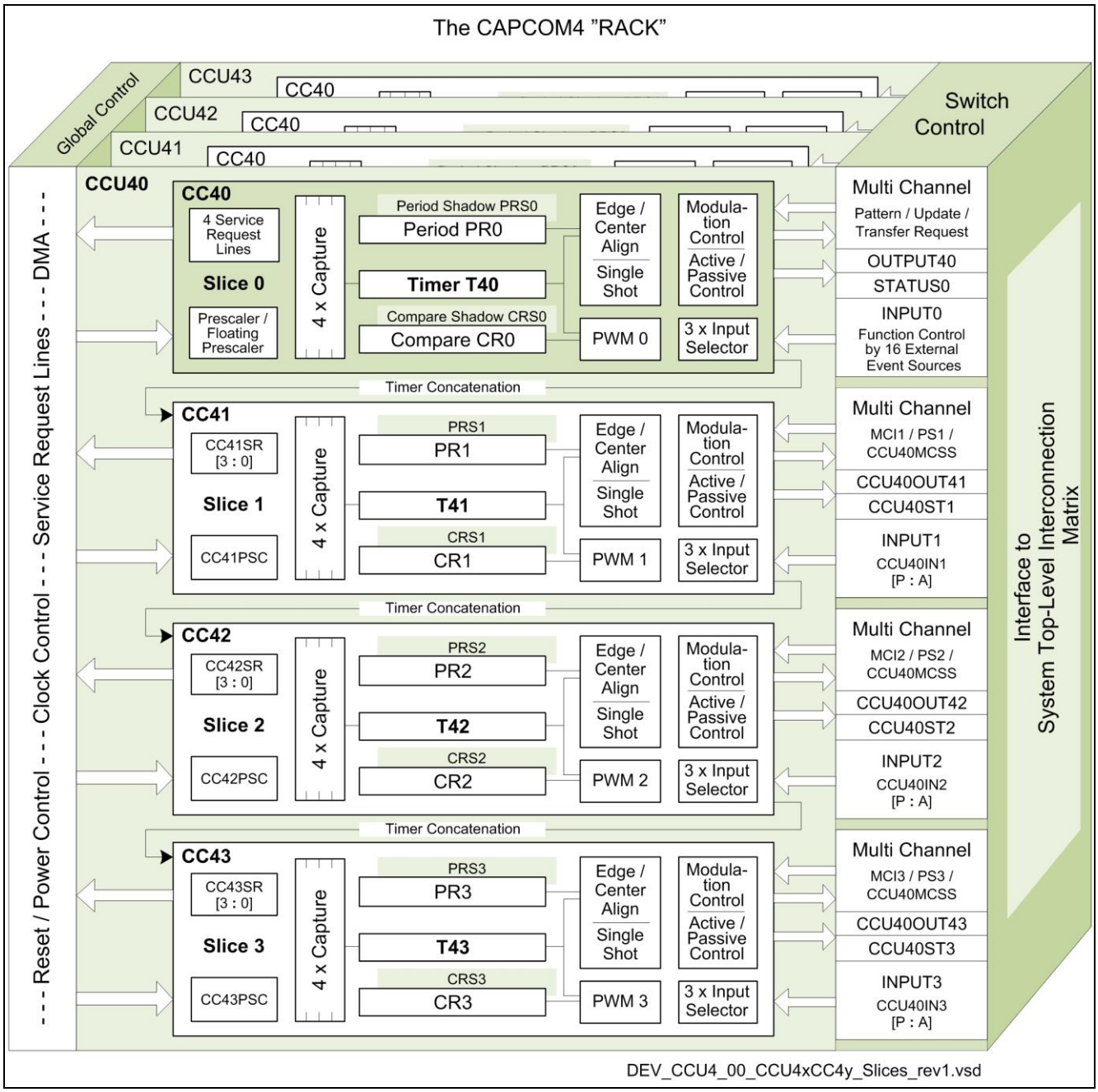


Figure 3 The Four Capture/Compare unit CCU40-CCU43 basic system of CAPCOM4

CCU Use Cases

The various capabilities of the CAPCOM timer slices are illustrated here by typical use cases as shown in the following figure.

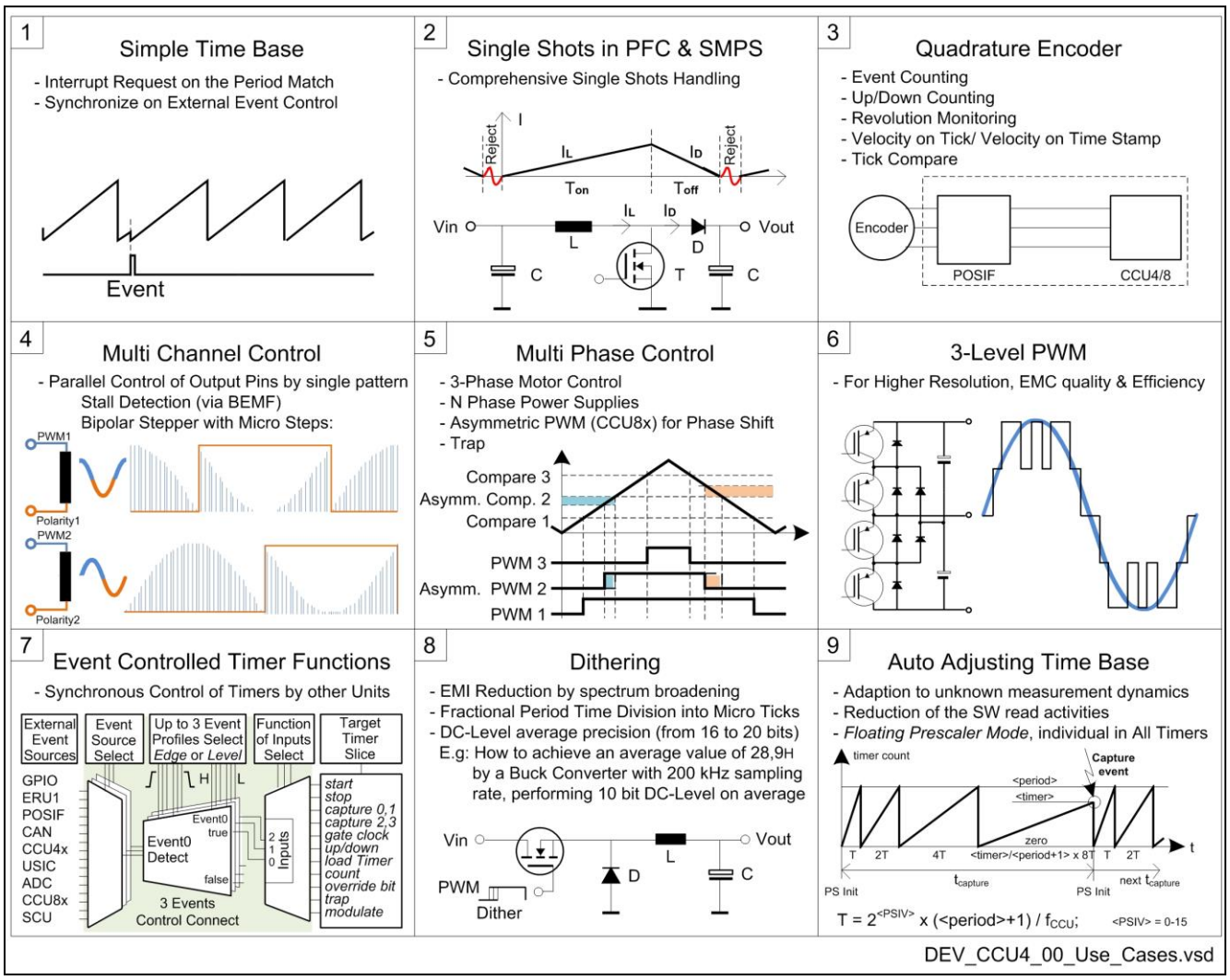


Figure 4 Some Features and Use Cases (1-9) characterizing an CAPCOM Unit (CCU) Features

Typical Use Case Feature of the CAPCOM4/8

1. Simple Time Base with synchronization option by external events control
2. Power Conversion Systems (PFC, SMPS) using Single Shot Mode
3. Event Monitoring and Revolution by Capture, Count and Position Interface facilities (POSIF)
4. Multi-Signal Pattern on Output Pins, created by parallel Multi-Channel Control
5. Drive & Motor Control with Multi-Phase Systems, Phase Adjustment and Trap Handling
6. 3-Level PWM for Inverters and Direct Torque Control (DTC) of AC Motors and High Precision Synchronous Motors
7. External Events Control of Timer Input Functions by requests from external system units
8. Dithering PWM or Period for DC-Level precision, Reduced EMI, Fractional Split of Periods into Micro Steps
9. Auto Adjusting Time Base by Floating Prescaler for adaption of time measurement to a wide range of dynamics

1.1 Additional CCU Features

Single Shot

A slice may be set in Timer Single Shot Mode (TSSM). Both the Timer and its Run Bit (TRB) is cleared by the Period/One match that occurs next to when the TSSM bit was set. As a result the timer stopped running. For example, a certain delay can be determined by the timer preset conditions and may start on any internal or external event.

Timer Concatenation

Any timer may be concatenated with an adjacent slice timer as easy as just by setting the respective TCE bits. Thus you may get one or two 32-bit timers, one of 2 possible 48-bit timers or a whole 64-bit timer this way. The first slice's configuration is propagated throughout the rest and so a complete synchronization from LSB to MSB.

Dithering PWM

Dithering is a hardware option in all the CAPCOM4 timers and may leverage PWM DC-level precision up to 16 times. This special hardware part may create a kind of superimposed jitter upon a timer PWM output signal to such an extent that the affected PWM pattern DC-level (Vout) will converge to an average value with the targeted precision level.

Dithering Period Time

Micro Ticks can be used in the interpolation between for example the sensor pulses from a cog wheel, to achieve position monitoring with higher precision. However, since the divisions of the sensor pulse period intervals seldom become exact multiples of micro tick periods, dithering can optimize the distribution of ticks over time.

Floating Prescaler

By successive halving of the timer clock frequency periodically, as long there is no capture (or compare) event, the dynamic range may be autonomously adapted to any time length, by using the slice's floating prescaler. In compare mode an arbitrary count range can also be setup by using the floating prescaler compare value register.

External Modulation

A slice may have its output pin modulated by external events. This function (and the active level of the event source) should be mapped to one of the 3 inputs of the slice in the CC4yCMC and CC4yINS registers. Modulation alternatives can be selected in CC4yTC for either Status Bit Clear or just Output Gating on events.

Output State Override

An input signal value (or its complement value) from one external source may override a slice's status bit value on an edge event by another external source. Both input sources are selected by the CC4yINS register**,** and the specific function by the matrix control register CC4yCMC. A use case would be output timing control by monitoring.

Multi-Channel Control

The output pin state of slices that are set in multi-channel mode (by the CC4yTC.MCME bit) can be controlled in parallel by a single pattern, according to the connected external signal on each CCU4xMCIy input. This task is requested by the CCU4xMCSS input and synchronized with the update of the status bit and shadow transfers.

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 required motor control pattern.

Timer Load Function

An external source may trigger a Timer Reload transfer from either the compare register (if counting up) or from the Period register (if counting down). The count direction bit CC4yTCST.CDIR can be set by software or by another external source. The input source(s) and function(s) should be mapped to the CC4yINS and CC4yCMC registers.

Trap Function

The trap function forces a PWM output into a predefined state, preset in the active/passive PSL bit. In this way a power device can be safely switched off. The input selection, trap and output functionality should be mapped to the slice's CC4yINS, CC4yCMC and CC4yPSL registers. Outputs are Trap-Enabled by the CC4yTC register.

Asymmetric Compare

Shadow transfers on both Period and One Match in center aligned mode enable asymmetric compare events. However, CAPCOM8 slices offer two compare registers per timer and two corresponding shadow registers, which are useful for applications that require PWM Phase Shift Control facilities.

1.2 CCU4 Input Control

Synchronized Control of CAPCOM Units on External Events

External Events Control distribution to CCUs (including CCU8) starts for advanced applications with synchronized timer control. For example in Motor Drive and Power Control, where 3-Level Inverters might require 12 synchronized PWM. The limits are to the realizable topography or timing pattern complexity range.

External Control Basics

A slice may have its input functions controlled by external sources. The external source(s), active mode(s) and input function(s) should be mapped to the 3 inputs of the slice in the CC4yINS and CC4yCMC registers. Function mode extension alternatives can be added by selections in the CC4yTC Timer Slice Control register.

External Events Control

An External Event Control request is a considered edge or level event signal from a peripheral unit or a GPIO. It can be linked to the CCU4xCC4y slices input selection stages via a comprehensive matrix. A slice that by any of its 3 events setups detects a considered source-event-input profile can be function controlled "remotely" this way.

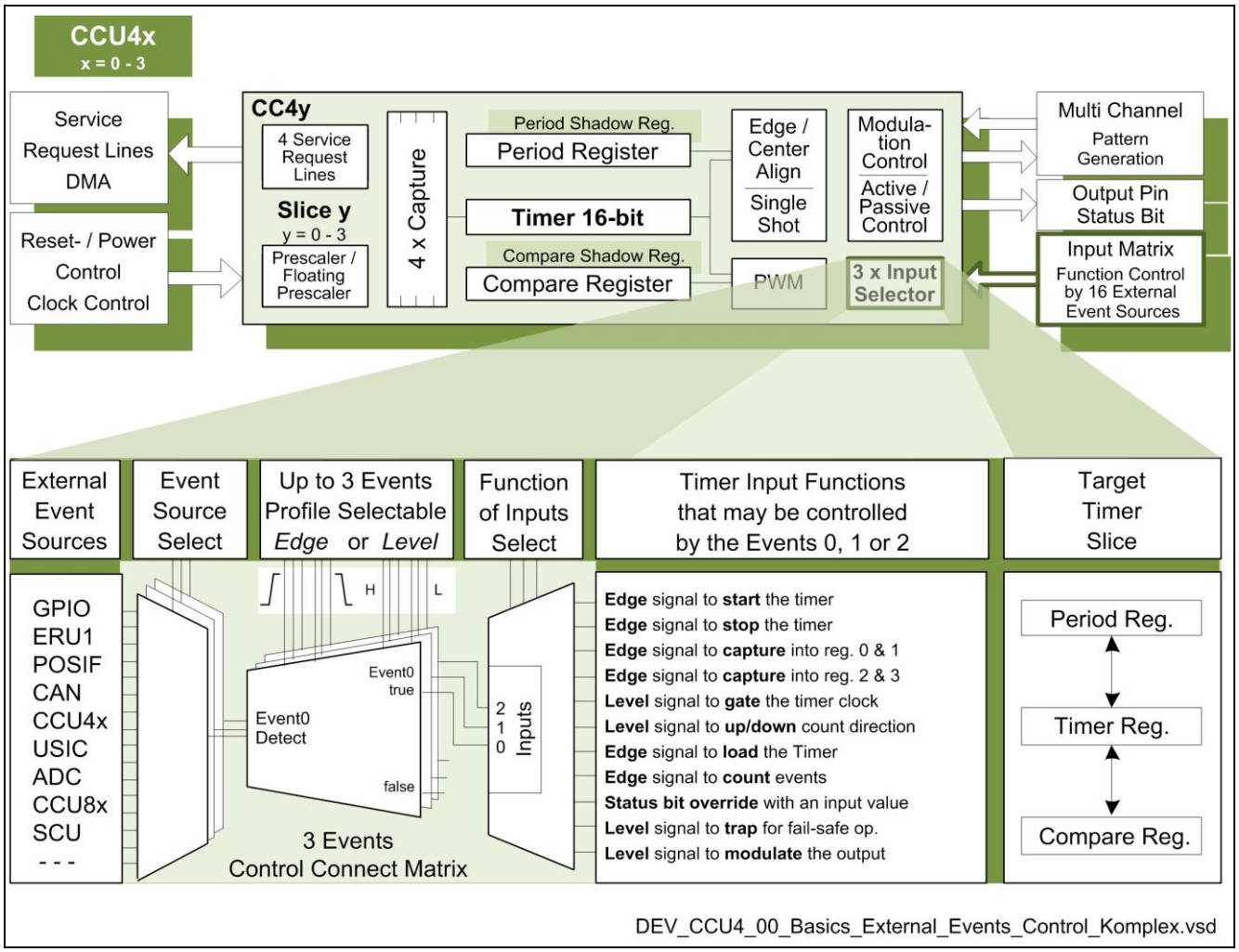


Figure 5 External Control of Timer Input Functions on Events by any External Units

External Event Sources

CCU4xCC4y Input Functions can be linked to external trigger requests from sources such as: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU. Pin Connections are given by the Top-Level Interconnect matrix and the CC4yINS[P:A] Input Select vector. The CC4yCMC register is used for the function selection.

External Events Input Functions

There are 11 Timer Input Functions (such as ‘Start the Timer’ for example), controllable by external events via 3 selectable input lines with configurable source-event profile conditions to the Timer Slices CC4y (y=0-3) of a CCU4x unit for Start, Stop, Capture0-3, Gate, Up/Down, Load, Count, Bit Override, Trap and Modulate Output control.

There are some Extended Input Functions too, in the register CC4yTC for Extended Start, Stop with Flush/Start, Flush/Stop or Flush or Extended Capture Mode which, together with a read access register (ECRD), simplifies administration of capture registers and full-flags, when more than one slice is used in Capture mode.

1.3 Capture Basics

Each CAPCOM4 (CCU4x) has 4 timer-slices. Each slice has 4 capture value registers, split into 2 pairs that will capture on selected event control input: Capt0 or Capt1, according to 2 possible pair

schemes: either as 2 pairs for different events to Capt0 resp. Capt1 or cascaded for the same event via Capt1.

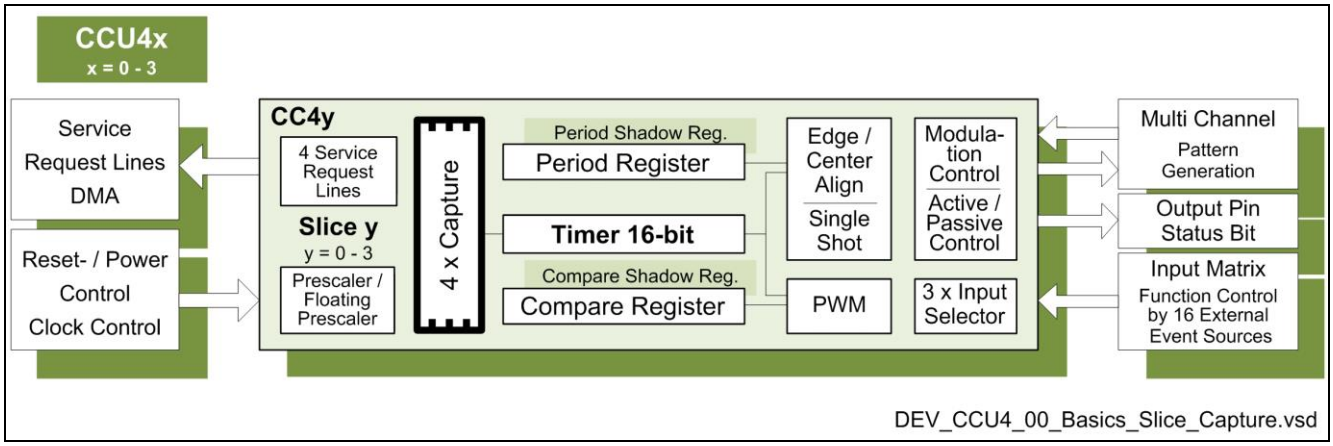


Figure 6 Timer Slice with four Capture Registers

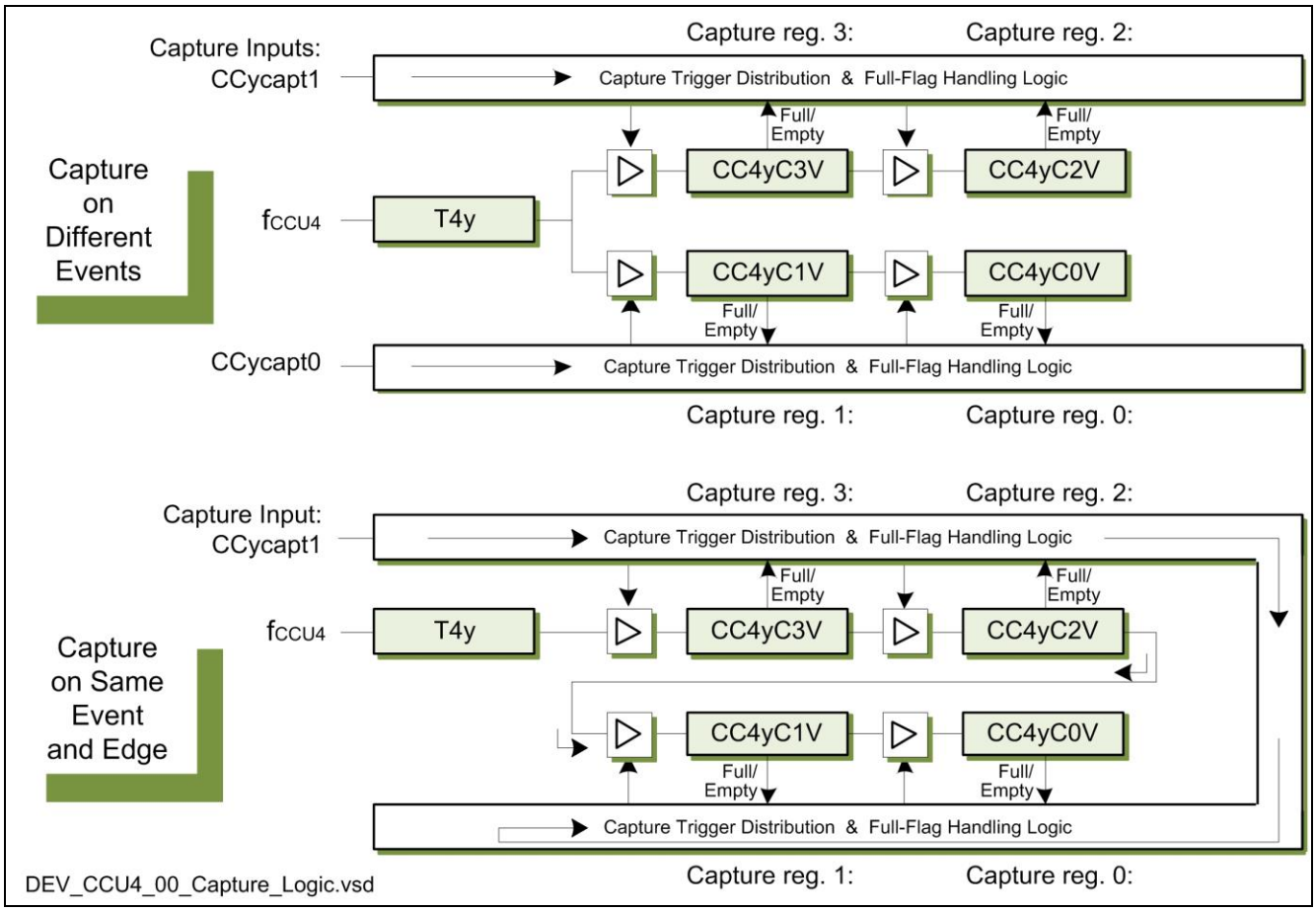


Figure 7 Basic Capture Mechanism – setup in two possible scheme alternatives

1.4 CCU4 Output Control

External Control by Timer Events

A timer event may trigger external actions via the Top-Level Interconnect matrix or on request for an Interrupt. Each CAPCOM4 has four Service Request Lines and each slice has a dedicated output signal CC4ySR[3...0], selectable to a line by CC4ySRS: i.e. timer slice events may request direct peripheral actions or an interrupt.

Top-Level Control of Event Request to/from a Timer Slice

Top-Level control also means conditional control of event requests between a slice and other action providers. The Event Request Unit (ERU1) and the Top-Level Interconnect matrix may combine, control and link event signals according to user defined request-to-action event patterns. For example, invoke I/O states, Time Windowing etc.

1.5 Compare Basics

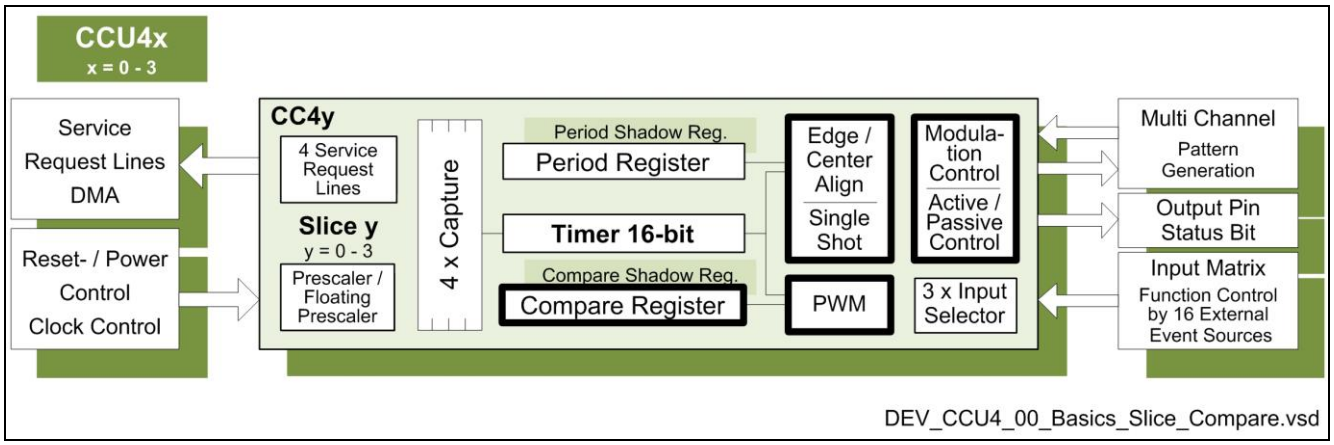


Figure 8 Timer Slice Compare Registers and PWM related Blocks

CCU4 Shadow Transfers

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

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. For example SySE in GCSS**,** must be preset by the latest within the clock cycle of Period Match (in Edge Aligned Mode) or Period/One Match (in Center Aligned Mode)

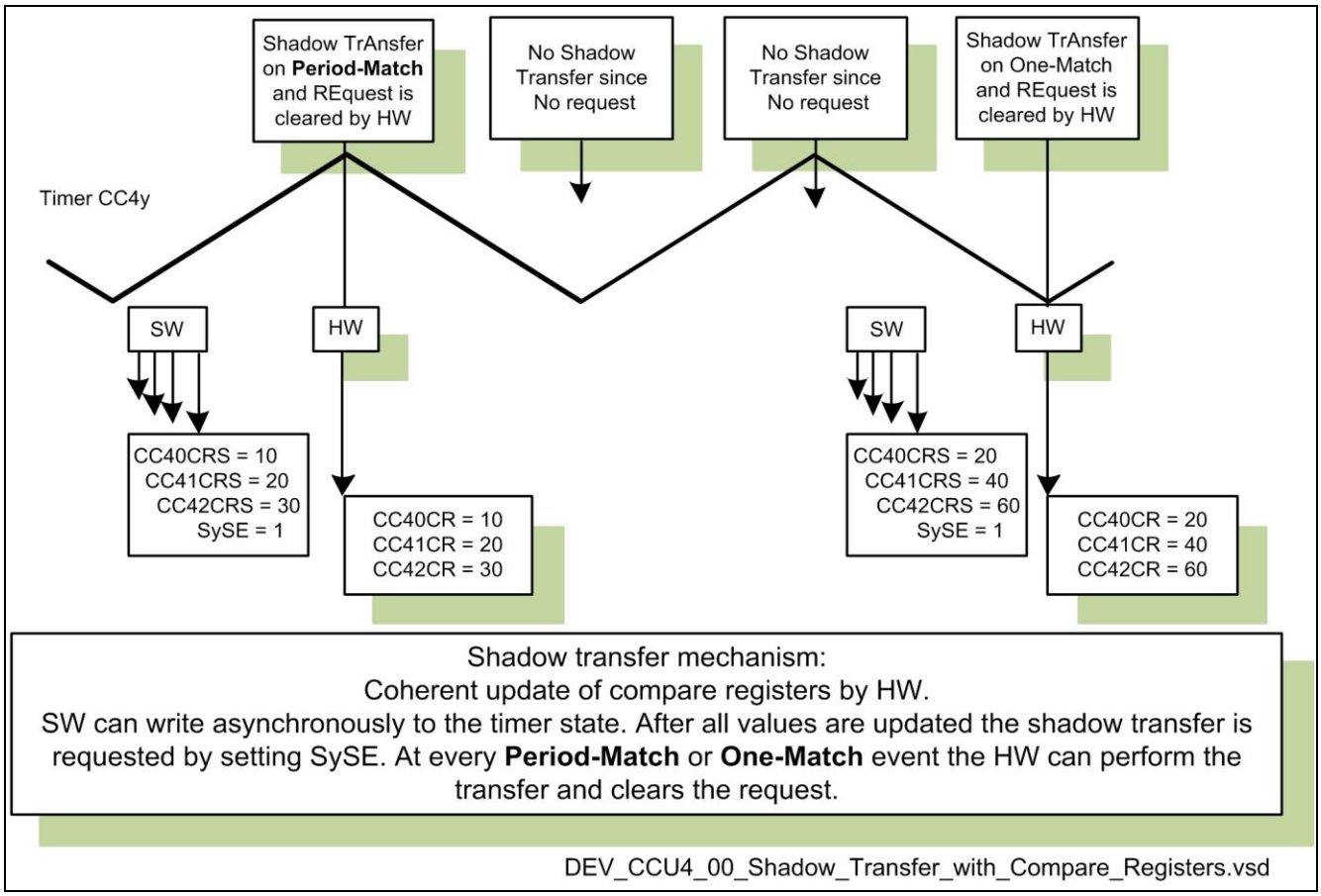


Figure 9 Basic Shadow Transfer Mechanism for Compare Register Values

Asymmetric Compare Events

The benefit of shadow transfers on both Period Match and One Match is to allow asymmetric compare events to be provided in Center Aligned Mode. The real-time conditions are similar to handling shadow value updates in Edge Aligned Mode. An even stronger solution for this is offered by the CCU8 units which has two shadow registers.

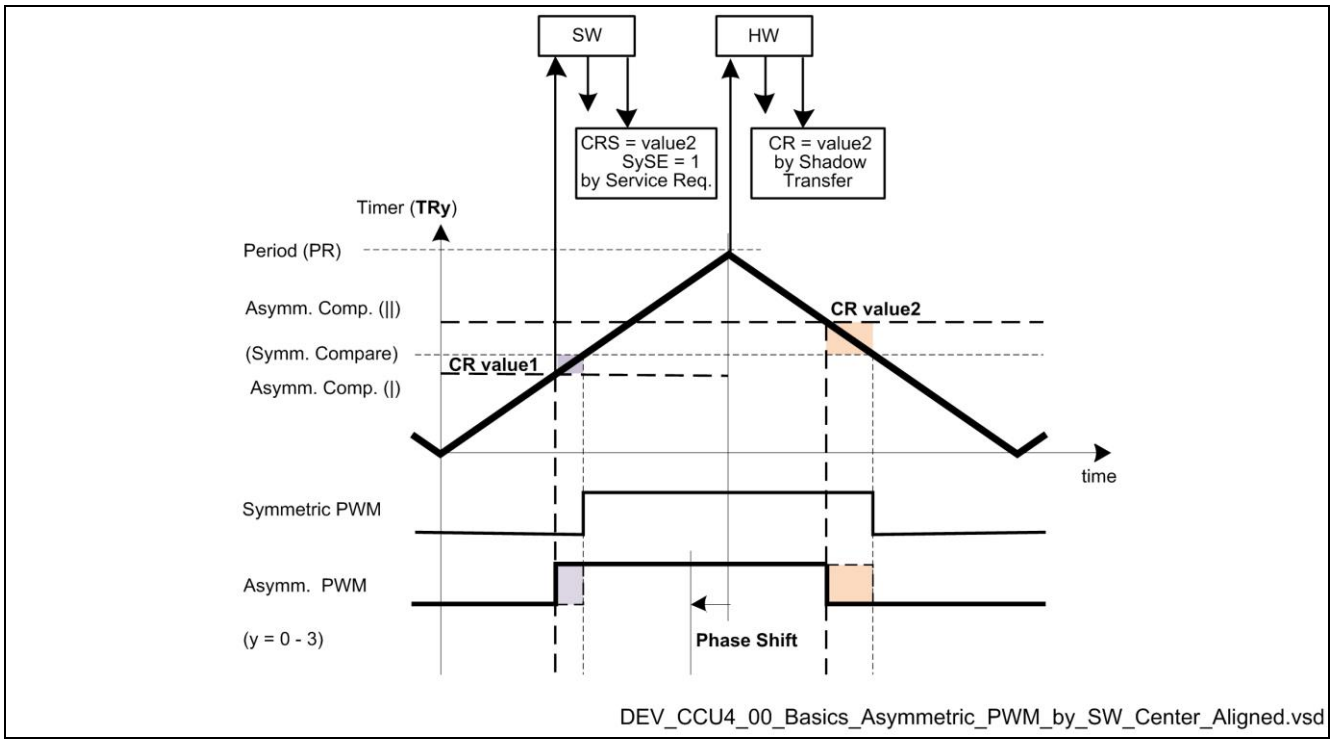


Figure 10 Asymmetric Compare by Shadow Transfers on both Period Match and One Match

Shadow Transfers in General – Compound Shadow Transfers

Beside the Compare (CR) values, there are also the timer Period register (PR) and the PWM Active/Passive control bit (PSL) that may be updated simultaneously on the SySE flag. Dithering or Floating Prescaler values are able to get a simultaneous update via the SyDSE and SyPSE request flags.

CCU4 Output State and Output Pin PASSIVE / ACTIVE Level Control

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

1.6 How to Start a Timer

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 the GIDLC register and then Start or Global Start can be initiated.

Start Timer Running

There are two ways to start a timer:

- Directly by software setting the Timer Run Bit Set (TRBS)
- Indirectly by hardware when a specific event occurs in an external unit as determined by the Top-Level Connection Matrix for External Events Control for CAN, ADC, USIC, IO, CCU4/8, ERU1, POSIF, and so on.

Global Start of CAPCOM Units

There is a way to get a synchronized start of CAPCOM Units, both CCU4x and CCU80x:

To achieve a synchronized start of both CAPCOM Units (i.e. CCU4 and CCU8x) use either

- A global start by software, with the CCUx Global Start Control bits in the CCUCON Global Start Control register
- A global start by hardware, indirectly with External Events Control using the CC4yINS and CC4yCMC registers

Global Start of the CCU4 and CCU8 CAPCOM Units

The Global Start command enables almost unlimited number of timers to be started, independently of the CAPCOM unit they belong to. The global start means that the timers are synchronized and all timing can be controlled in parallel, with many different kinds of output pattern generation.

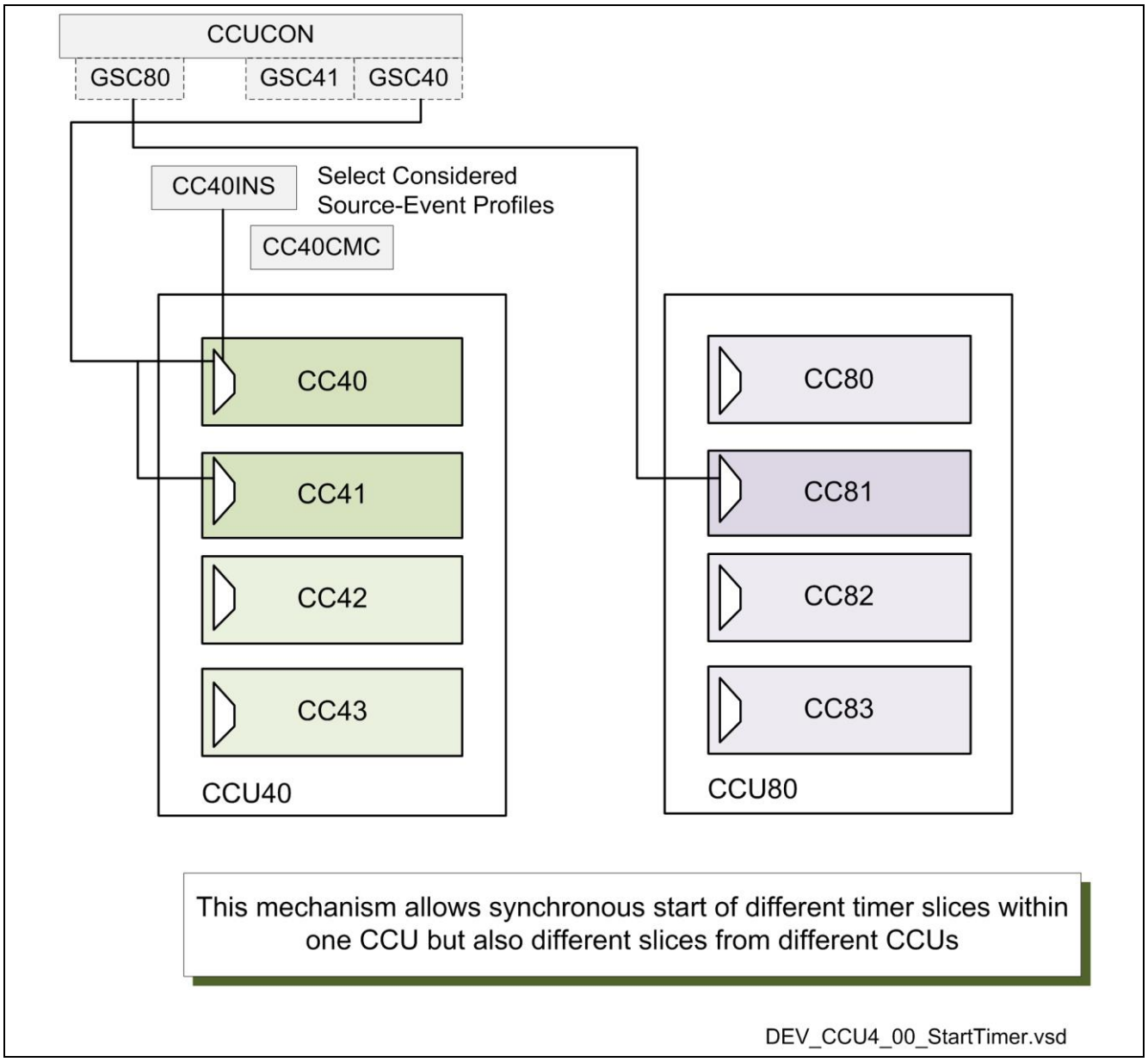


Figure 11 External Event Control with Global Start Command

1.7 Getting Started with Capture Compare Unit - CCU4

In the following examples it is assumed that the CCU4 Initialization Sequence start has already been completed.

Example 1: Start Timer by Software (by Instruction)

```
CC40TCSET |= 0x00000001; //Start Timer CC40 directly by setting the TRBS bit = 1
```

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.

Using the External Start Control Register Bitfields

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

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

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.

```
// Example 2:
// Assume a falling edge transition on GPIO Pin P1.3 as a "considered" external
// event control signal to start Timer CC40 in CAPCOM Unit CCU40.
// Define this event as "Event2". (The alternatives Event0..1 would do as well!):
CC40CMC |= 0x00000003; // Start of Timer CC40 indirectly by External Event 2.

// Setup Timer CC40 External Event Control Input Selection register (CC40INS)
// to activate GPIO Pin P1.3 as selected source for Event 2 in the bitfield EV2S:
EV2S = 0; // where "0" is the input number (from 0-15) for the input
       // "CCU40IN0[A]" and represents Pin P1.3 according to the Top-level
       // connection Matrix.
EV2EM = 2; // set signal active on falling edge on Event 2.
```

1.8 Runtime Handling of Capture Compare Unit – CCU4

Example: Handling Shadow Transfers

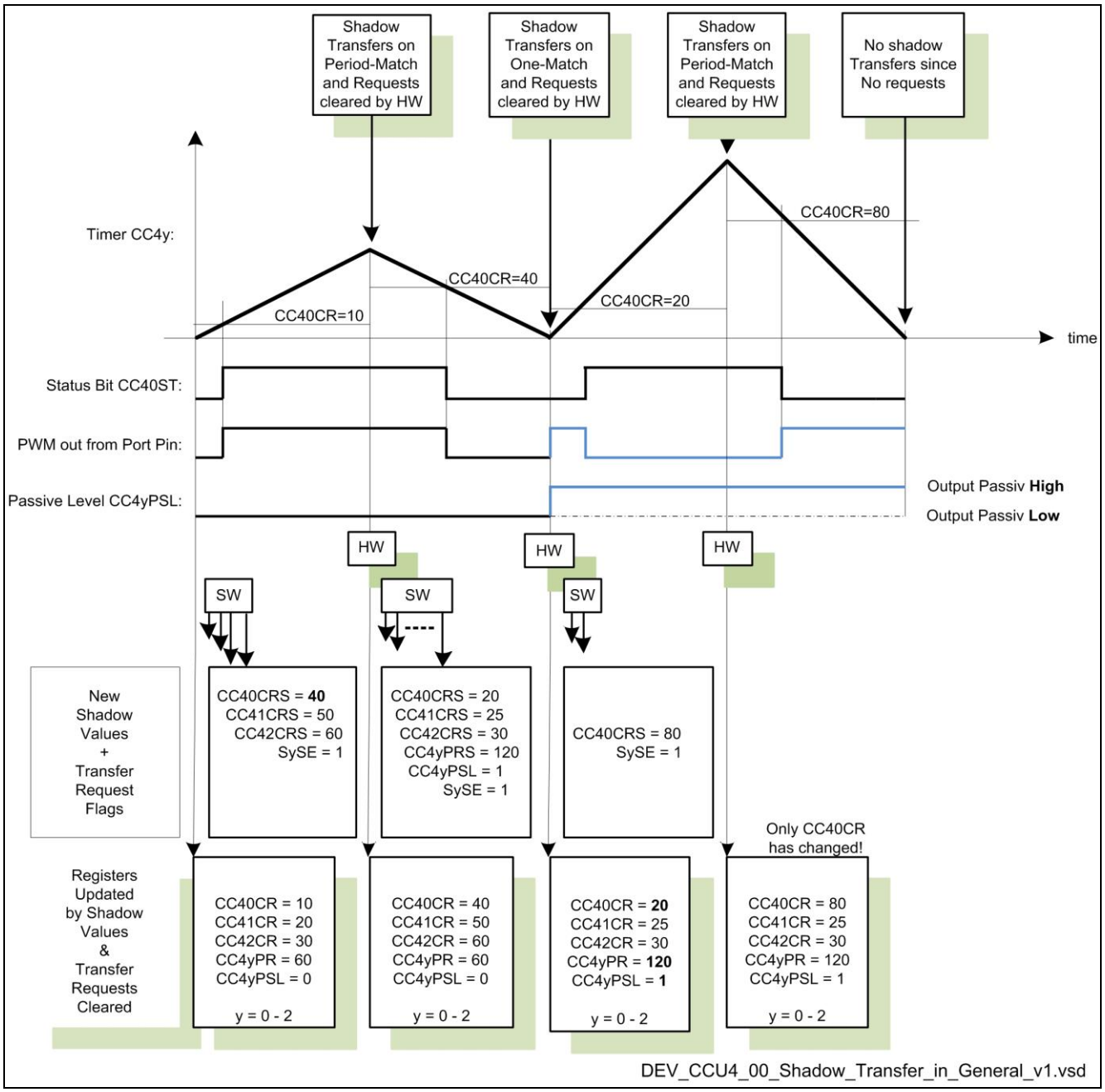


Figure 12 Compound Shadow Transfer Mechanism with Coherent Update of PWM

1.9 Differences between CCU4 and CCU8

The following section will describe the differences between the CCU4 and the CCU8.

Table 1 CCU4 versus CCU8

Feature	CCU4	CCU8
Number of compare channels per slice	1	2
Number of complimentary outputs	Requires 2 slices	1 per compare channel
Number of outputs per CCU	4	16
Dead-time generation	No	2 dead-time values per compare channel
Asymetric PWM	Requires 2 slices (software method)	1 per slice
Output Parity Checker	Yes	No
Number of slices per half-bridge control	2	Half-slice in symmetric 1 slice in asymmetric

Output Pattern Generation using the Compare Mode

2 Output Pattern Generation with CCU4

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

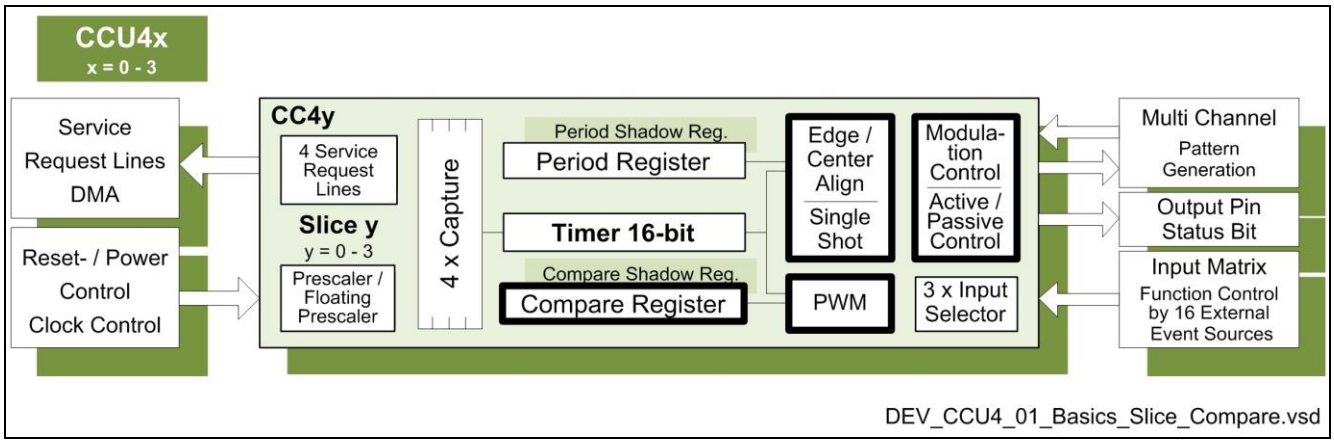


Figure 13 Timer Slice Compare Registers and PWM related Blocks

The Principle Compare Blocks

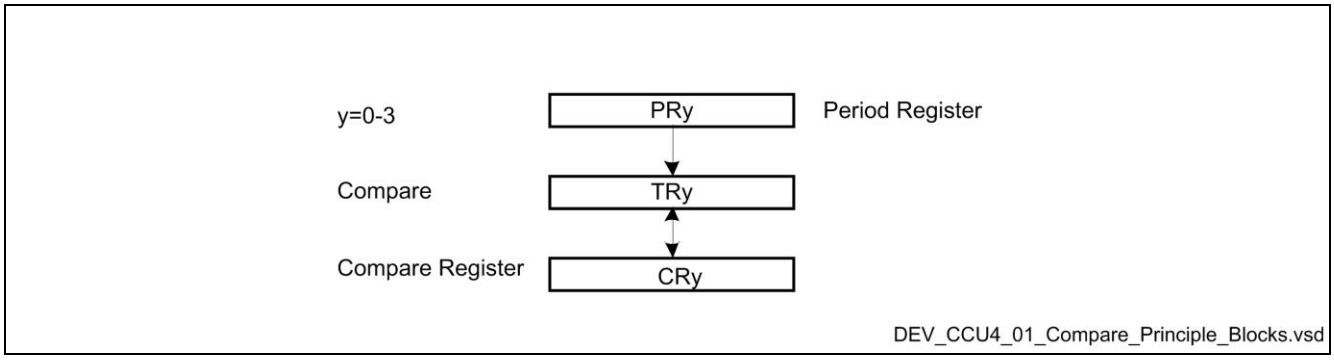


Figure 14 The Compare Blocks

PWM Range 0 – 100% in Up-Count Mode

The Compare Rule in Up-Count Mode is very simple:

As long as the timer register value is equal or greater than the compare register value, the Status Bit (CCST or even named CCU4xSTy) will be set to one (=1). Otherwise it is set to zero (=0).

The dynamic PWM range can be set to any value from 0 up to 100%.

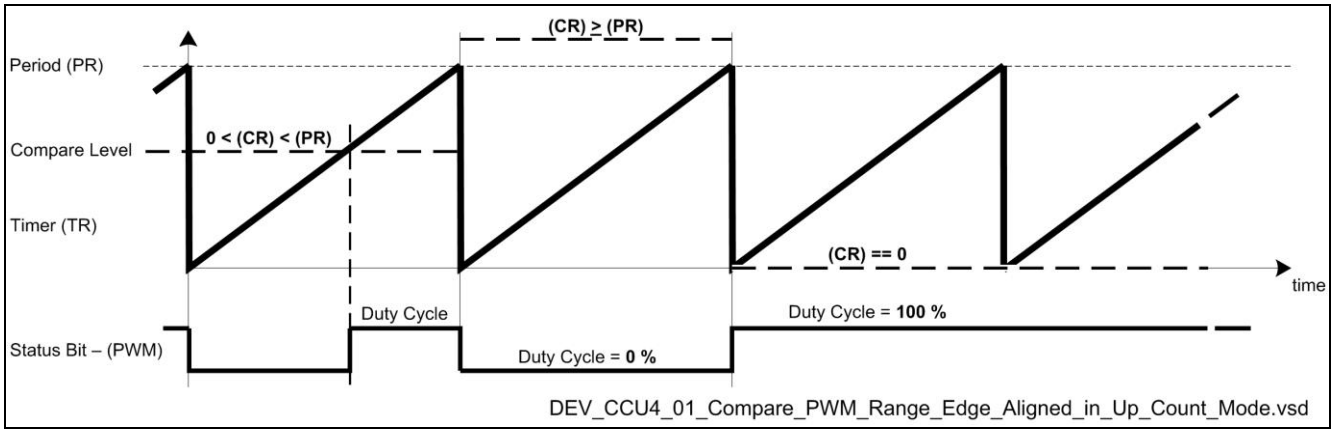


Figure 15 PWM Range in Up-Count Mode

PWM Range 0 – 100% in Down-Count Mode

The Compare Rule in Down-Count Mode is the same as in Up-Count Mode:

When the timer register value is equal or greater than the compare register value, the Status Bit (CCST or CCU4xSTy) will be set to one (=1). Otherwise it is set to zero (=0).

The dynamic PWM range can be set to any value from 0 to 100%.

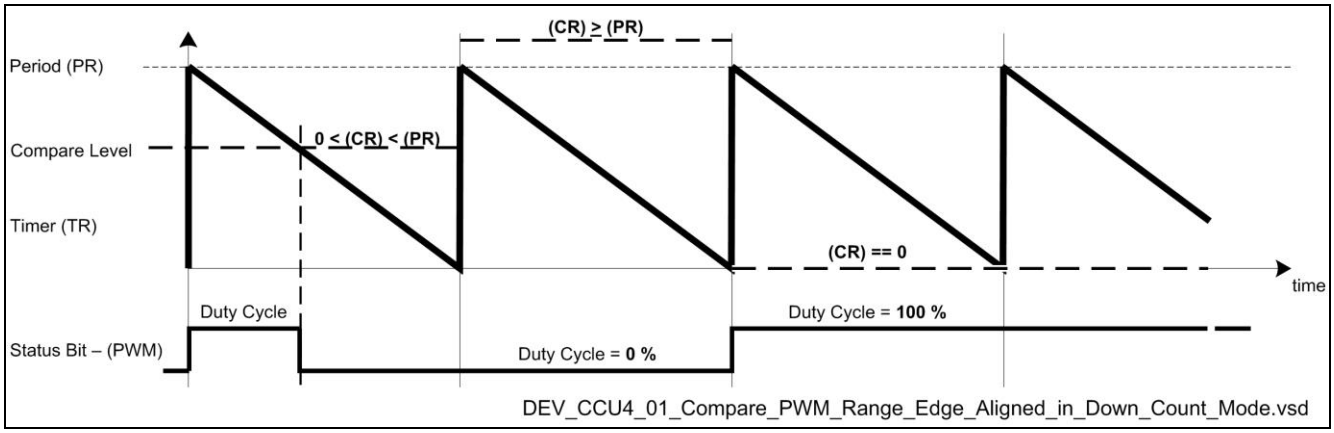


Figure 16 PWM Range in Down-Count Mode

PWM Range 0 – 100% in Center Aligned Mode

The Compare Rule in Center Aligned Mode is the same as in Up- or Down-Count Modes:

When the timer register value is equal or greater than the compare register value, the Status Bit (CCST or CCU4xSTy) will be set to one (=1). Otherwise it is set to zero (=0).

The PWM value may vary from 0 to 100%.

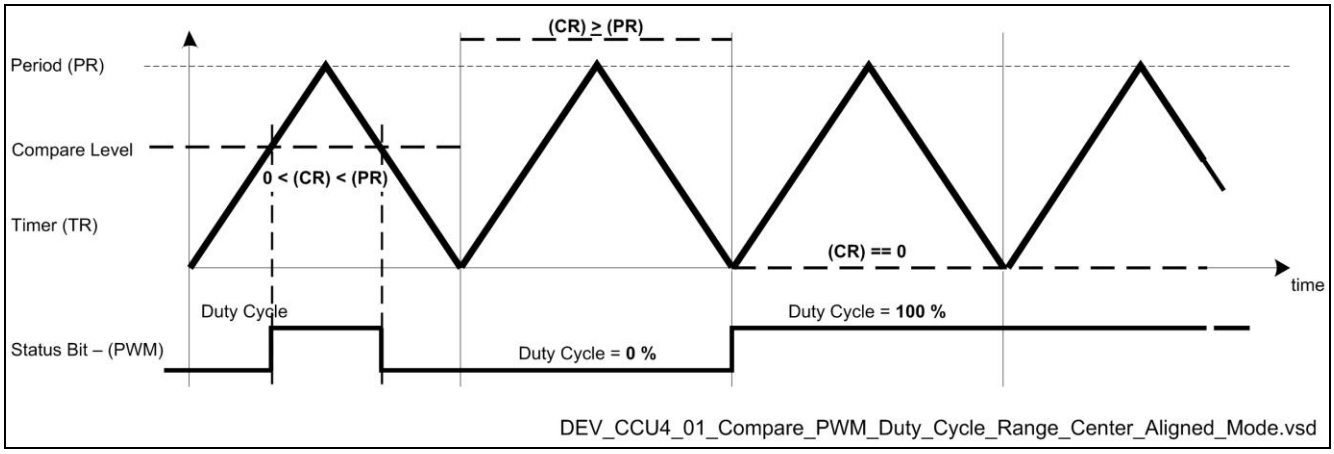


Figure 17 PWM Range 0 – 100% in Center Aligned Mode

2.1 Compare Counting in Detail

Counting Rules

One count cycle after the Counting Direction status bit CC8yTCST.CDIR has been changed, the timer counting direction will be changed at the next count. In Center Aligned Mode this bit will be changed from 0 to 1, one cycle after a Period-Match state is entered, and from 1 to 0, one count cycle after a One-Match is entered.

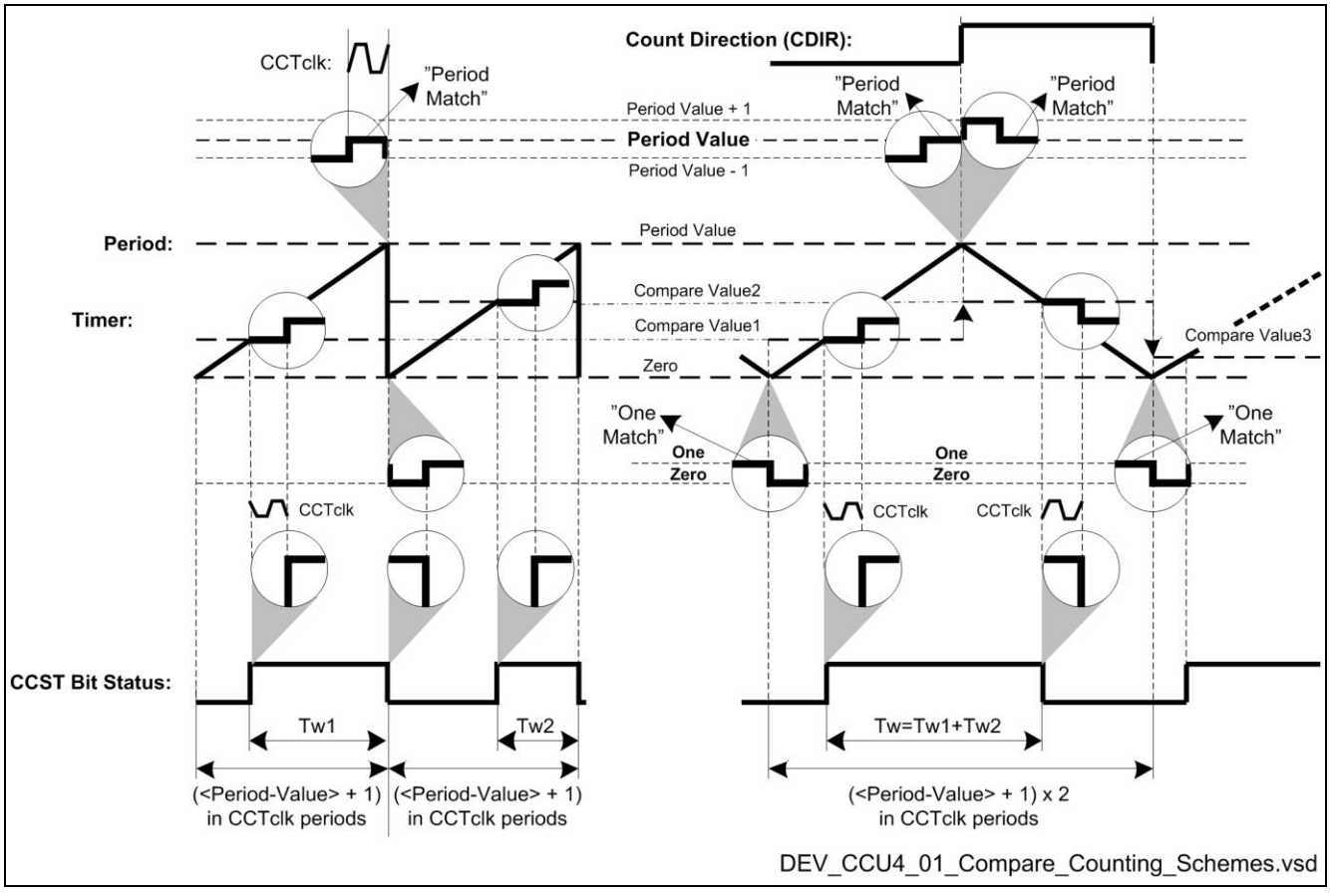


Figure 18 Compare Counting Schemes in Detail

2.2 Compare Reload in Detail

Reload with Shadow Transfer Rules

A reload of registers by Shadow Transfers from the associated shadow registers will occur according to the following rules:

- In the next clock cycle after a Period Match while counting up
- In the next clock cycle after a One Match while counting down
- Immediately if the timer is stopped and a transfer request was triggered

Note: This implies that it is still due time to Set Transfer Enable requests as late on a Period Match event, by using a Period Match Interrupt Service with a shorter latency time than the last clock cycle before the Shadow Transfer occurs. Normally the Compare Event Interrupts are used for this purpose. However, it shows that even a compare register value equal to the period register value is accurate for Transfer Enable request settings. In other words, the entire timer period may be used to Set Transfer Enable bits.

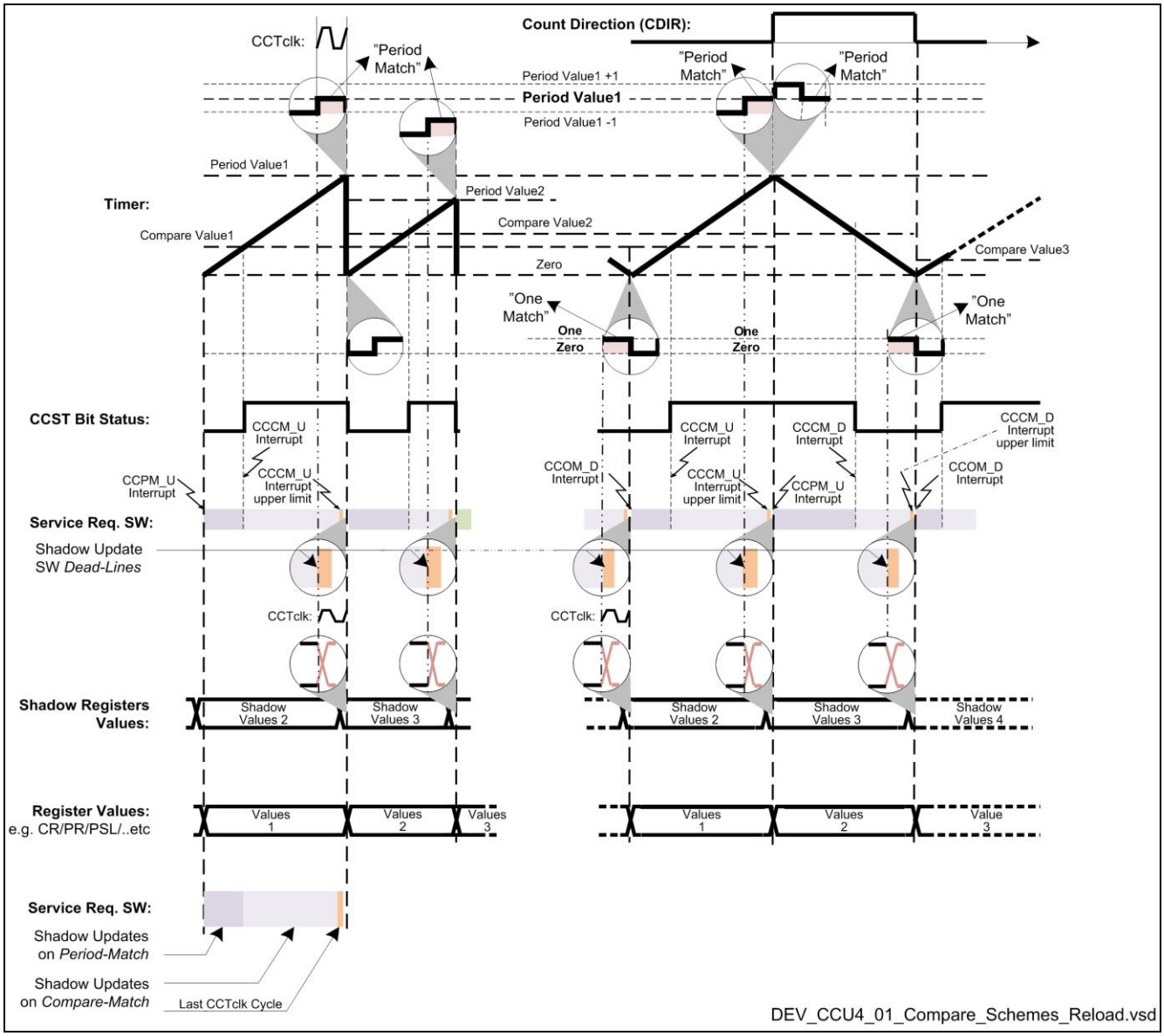


Figure 19 Compare Reload Schemes in Detail

CCU4 Shadow Transfer 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 hardware of all relevant timer function parameters with the update values held in the shadow registers. On a global preset request these values are transferred simultaneously to all function registers at a Period-Match or One-Match.

The Global Shadow Transfer Set Enable Register

There is one global register, GCSS, carrying all enable-flags that have to be preset by software to selectively activate the targeted Shadow Transfer Requests, which will be cleared by hardware after the transfer. The total real-time correctness is therefore achieved from the logic operations aspect, which is essential for safe power switching.

Shadow Transfer of Compare Register values

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

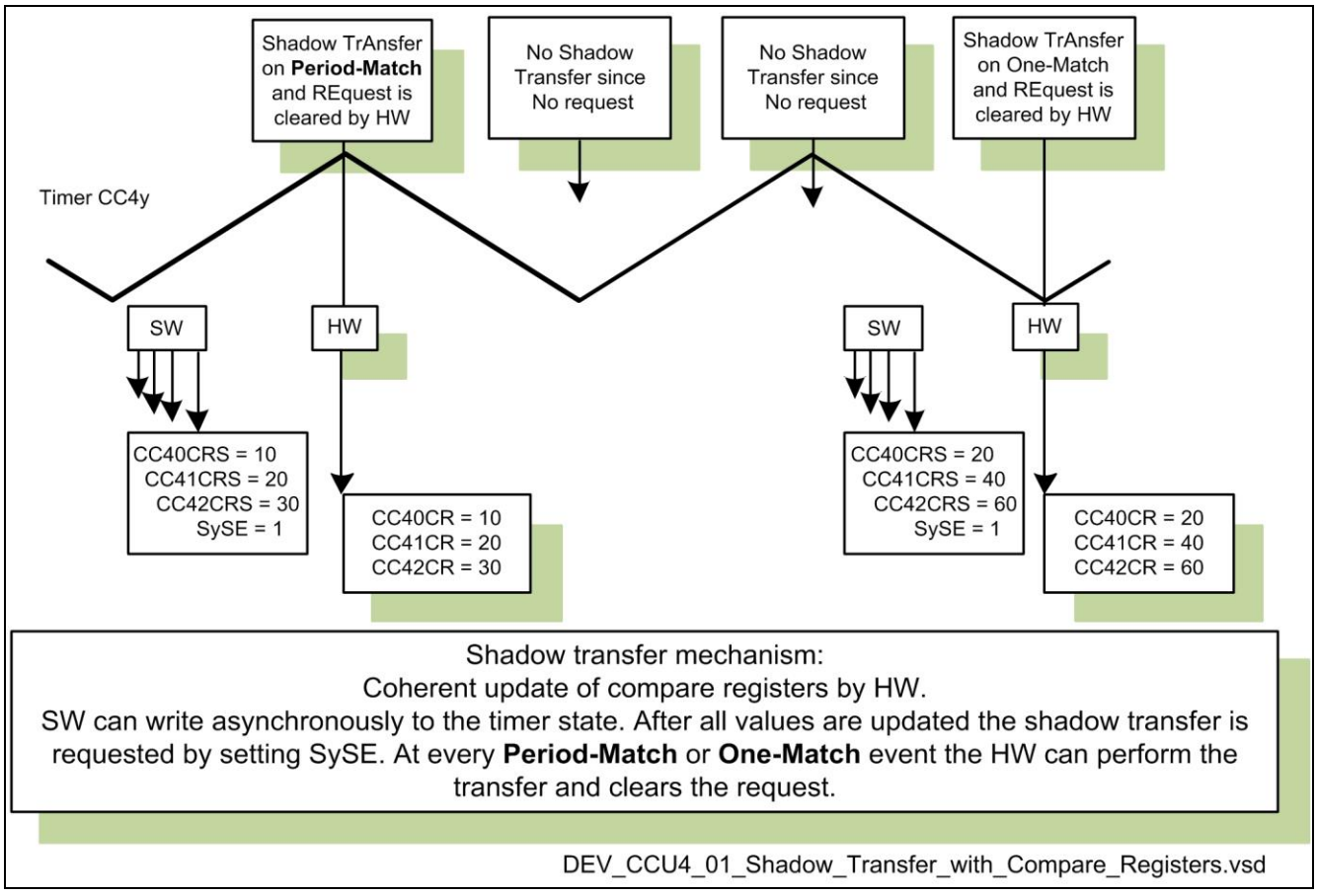


Figure 20 Shadow Transfer Mechanism with Compare Registers

Compound Shadow Transfers

Beside the Compare (CR) values, there are also the timer Period register (PR) and the PWM Active/Passive control bit (PSL) that are updated simultaneously on the SySE flag. Dithering or Floating Prescaler values can also be simultaneously updated by the SyDSE & SyPSE request flags.

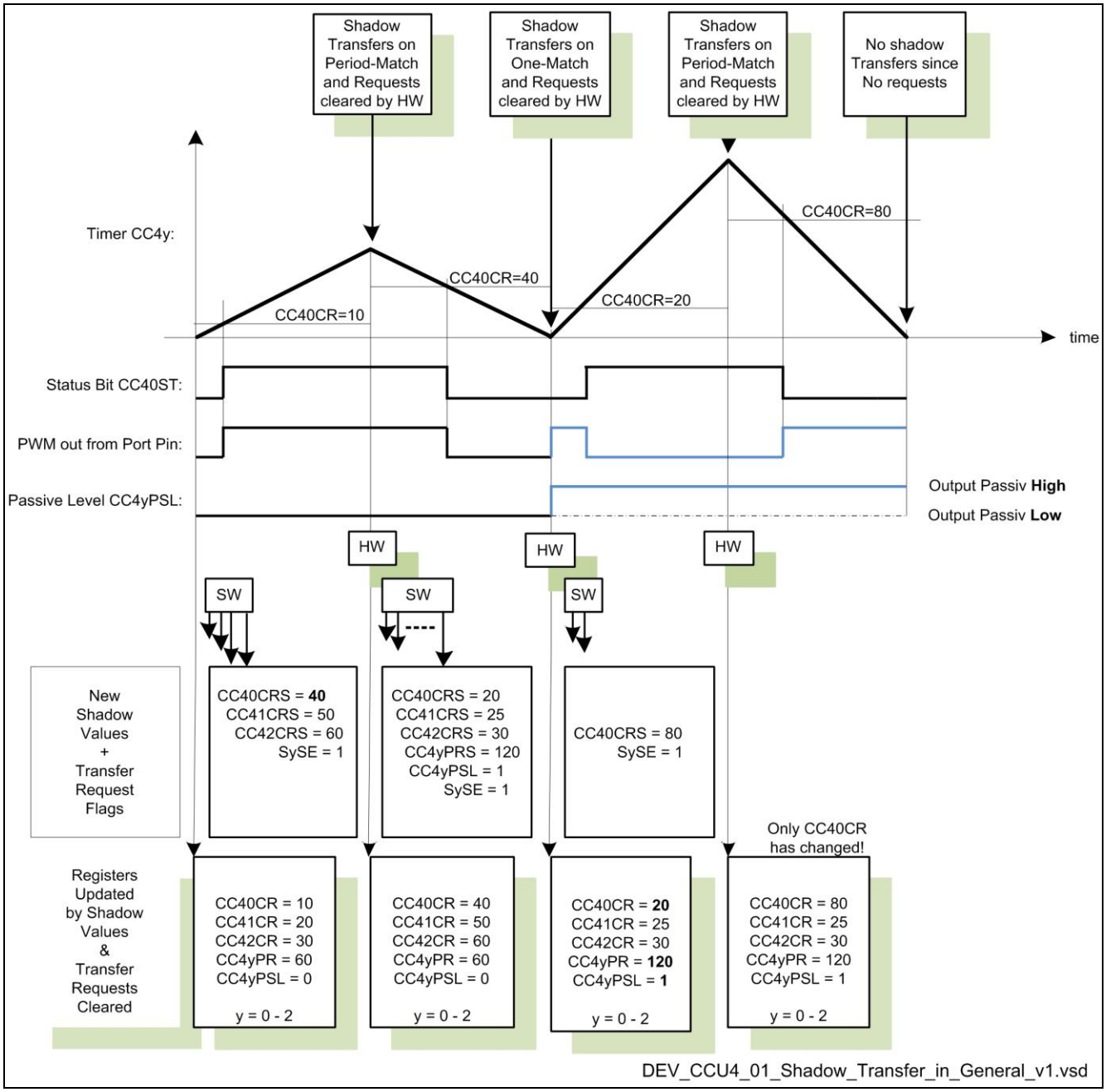


Figure 21 Compound Shadow Transfer Mechanism with Coherent Update of PWM

2.3 CCU4 Output Control

Output State and Output Pin PASSIVE / ACTIVE Level Control

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

External Control by Compare Events

A compare event may trigger external actions via the Top-Level Interconnect matrix or on request for an Interrupt.

Each CAPCOM4 has four Service Request Lines and each slice has a dedicated output signal CC4ySR[3...0], selectable to a specific line by CC4ySRS. For example, compare events may request for immediate ADC actions or for interrupts.

Top-Level Control of Event Requests to/from a Timer in Compare Mode

Top-Level control also means conditional control of events requests between a slice and other action providers. The Event Request Unit (ERU1) can be combined with the Top-Level Interconnect matrix to control and link event signals according to user defined request-to-action event patterns. For example, ADC triggering combined on I/O events.

CCU4 Using Top-Level Interconnect Control with Event Request Unit (ERU)

A slice interface to the ERU1 and to the Top-Level Interconnect matrix can be represented by a simplified scheme. To complete the picture of the possible interaction, this scheme also shows how operations can be extended to involve DMA transfers (by the GPDMA), triggered by a handler (DLR) on the Service Request Lines (SRn).

CCU4 and ERU1 Use Cases

If an application requires ADC conversions to start on timer events under specific conditions. (i.e. if not directly via a Top-Level Interconnect matrix path), then the ERU1 is able to offer an alternative signal path. This may involve dependence on a port pin, a time window from a second timer, or a certain sequence of event patterns.

Example – Using CCU4 and ERU1 for Delayed ADC-Start Controlled by an IO

In this example, the CCU4xCC4y is a single shot delay timer. Its status bit (red) that is delayed and set by the compare event, shall delay an ADC-start when triggered by a PWM timer (blue) on a GPIO state (orange). The ERU1 combines, detects and links it all as a trigger (green) via the delay timer and the Top-Level Interconnect matrix to the ADC.

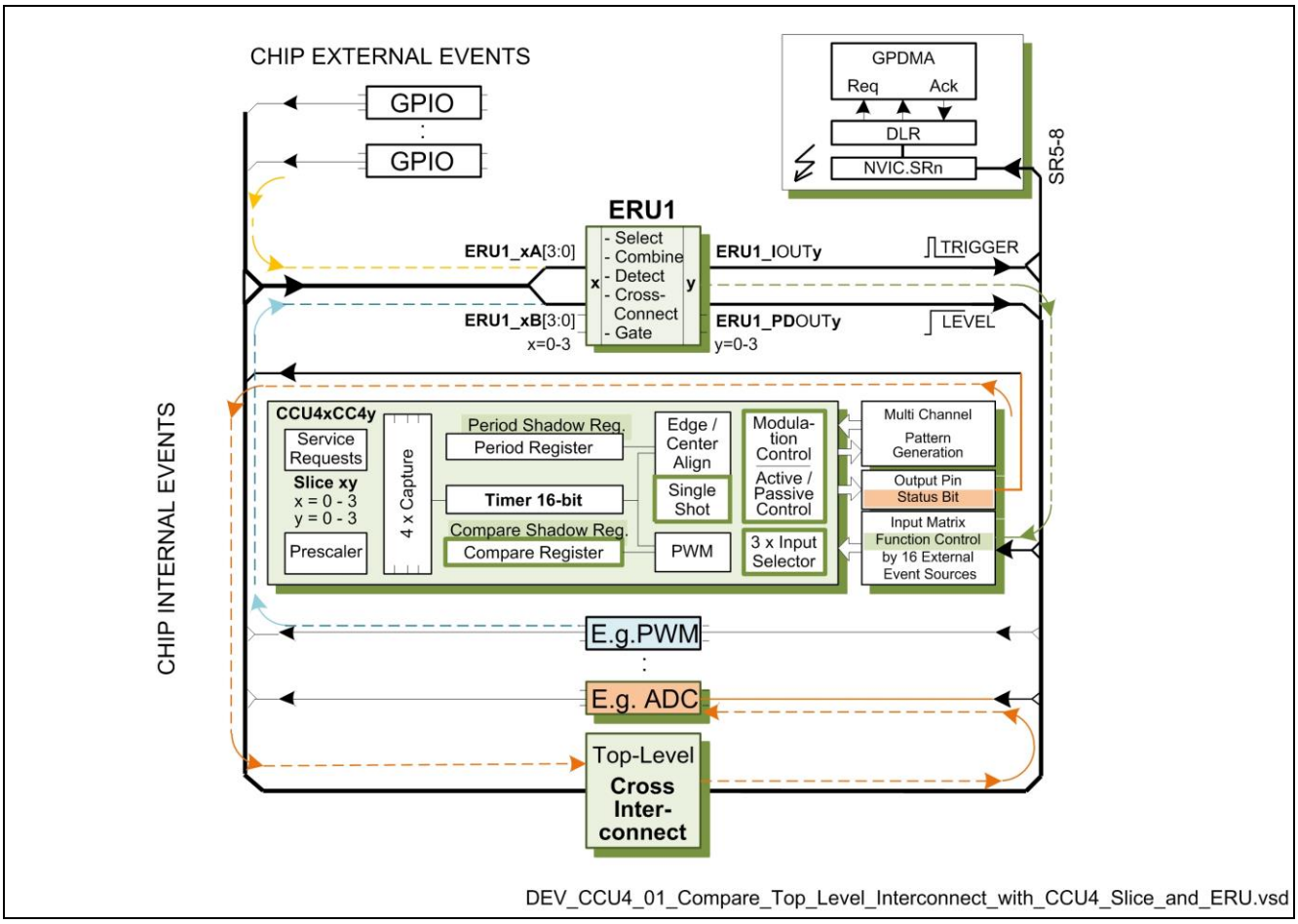


Figure 22 Using CCU4 and ERU1 for Delayed ADC Start Controlled by an IO

2.4 Getting Started with CCU4 and Compare Mode Features

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 the GIDLC register and then Start or Global Start can be initiated.

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 read-only compare register CC4yCR by the value from the compare shadow register CC4yCRS on each shadow transfer, if requested within the current period cycle by the Slice Transfer Set Enable bit GCSS.SySE.

The Shadow Transfer Events for all registers concerned

- In the next clock cycle after a Period Match while counting up
- In the next clock cycle after an One Match while counting down
- Immediately if the timer is stopped and a transfer request triggered

Setting the Output Pin PASSIVE / ACTIVE Level Control

The PASSIVE/ACTIVE state of a slice internal output CCUxSTy (i.e. status bit CC4yST**)** is controlled by the compare level and the External Modulation Mode. Whether the external output pin state CCu4xOUTy (e.g. the PWM) should be Passive Low / Active High or vice-versa, is controlled by the CC4yPSL Passive/Active bit PSL.

Initialize and Start the Timer

After the Start procedures have been executed, the software may:

- Initialize the slice timer CC4yTIMER or clear it with a CC4yTCCLR.TCC bit=1 command
- Start the timer with a Set Timer Run command in the CC4yTCSET.TRBS bit

The timer may be stopped at any time by software with a CC4yTCCLR.TRBC command.

Using the Compare Mode / PWM Control Registers

- CC4yTC.CMOD — CAPCOM Mode selection bit should be set =0 (which is the default value)
- CC4yTC.TCM — Timer Counting Mode selection bit: 0 → Edge-Aligned / 1 → Center-Aligned
- CC4yCRS.CRS — Write the required Compare Level value into the compare shadow register

- CC4yPR.PR Write the timer count cycle upper limit, Period Match, into the period register
- CC4yPSL.PSL PASSIVE/ACTIVE control bit: (CCU4xOUTy) = (CCUxSTy) xor (CC4yPSL.PSL)

Optionally:

- CC4yTIMER Initialize a timer start value (or flush by the CC4yTCCLR.TCC bit=1 command)
- GCSS.SySE Request the Shadow Transfer to Initialize the Compare Register
- CC4yTCSET.TRBS Start Timer by software

External Events Control Setup Registers

External Events Control is useful for Start/Stop options. 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 to Start for example.

Using the External Start Control Register Bitfields

- CC4yCMC.STRTS Input Function Selection and Event Number x=0-2 Input Line Selection bitfield
- CC4yINS.<Event-x bitfields> Event-x Source EVxIS (CC4yINS [P:A]) and Event Edge Mode EVxEM bitfields
- 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 read as 'Actual Event[2:0] Number'.

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

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

Using the External Stop Control Register Bitfields

Besides the Timer Single Shot Mode (selected with the CC4yTC.TSSM bit set), a timer may also be stopped by External Events Control, even by its own events. The following Control Register Bitfields should be used:

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

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

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
```

Advanced Signal Measurement using the Capture Mode

3 Advanced Signal Measurement

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

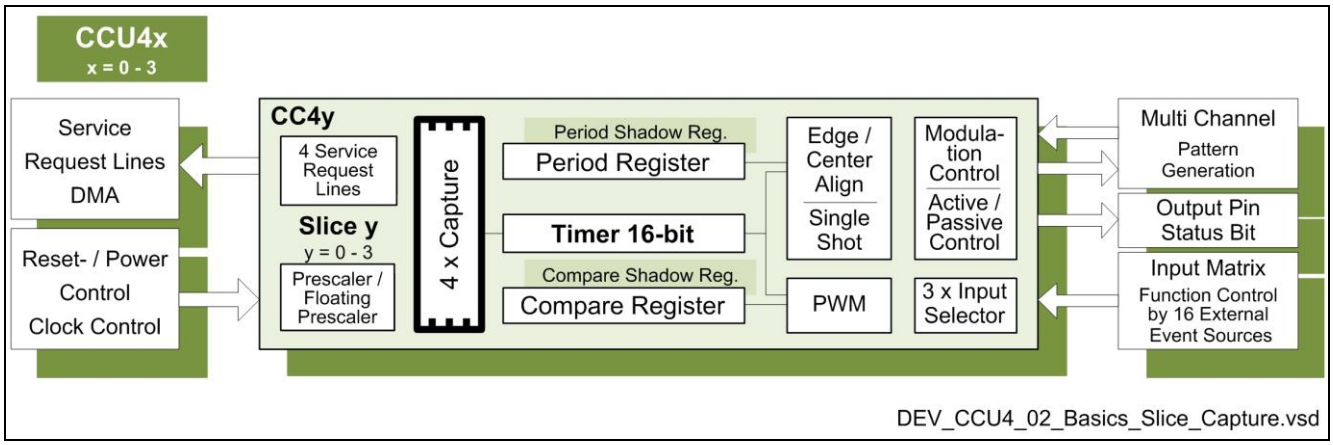


Figure 23 Timer Slice four Capture Registers

Slice Timer Setup in Capture Mode

Each CCU4x has with 4 timer-slices. Each slice has 4 capture value registers, split into 2 pairs that will capture on selected event control input either Capt0 or Capt1, according to 2 possible pair schemes: either as 2 pairs for different events for Capt0 with respect to Capt1, or cascaded for the same event via Capt1.

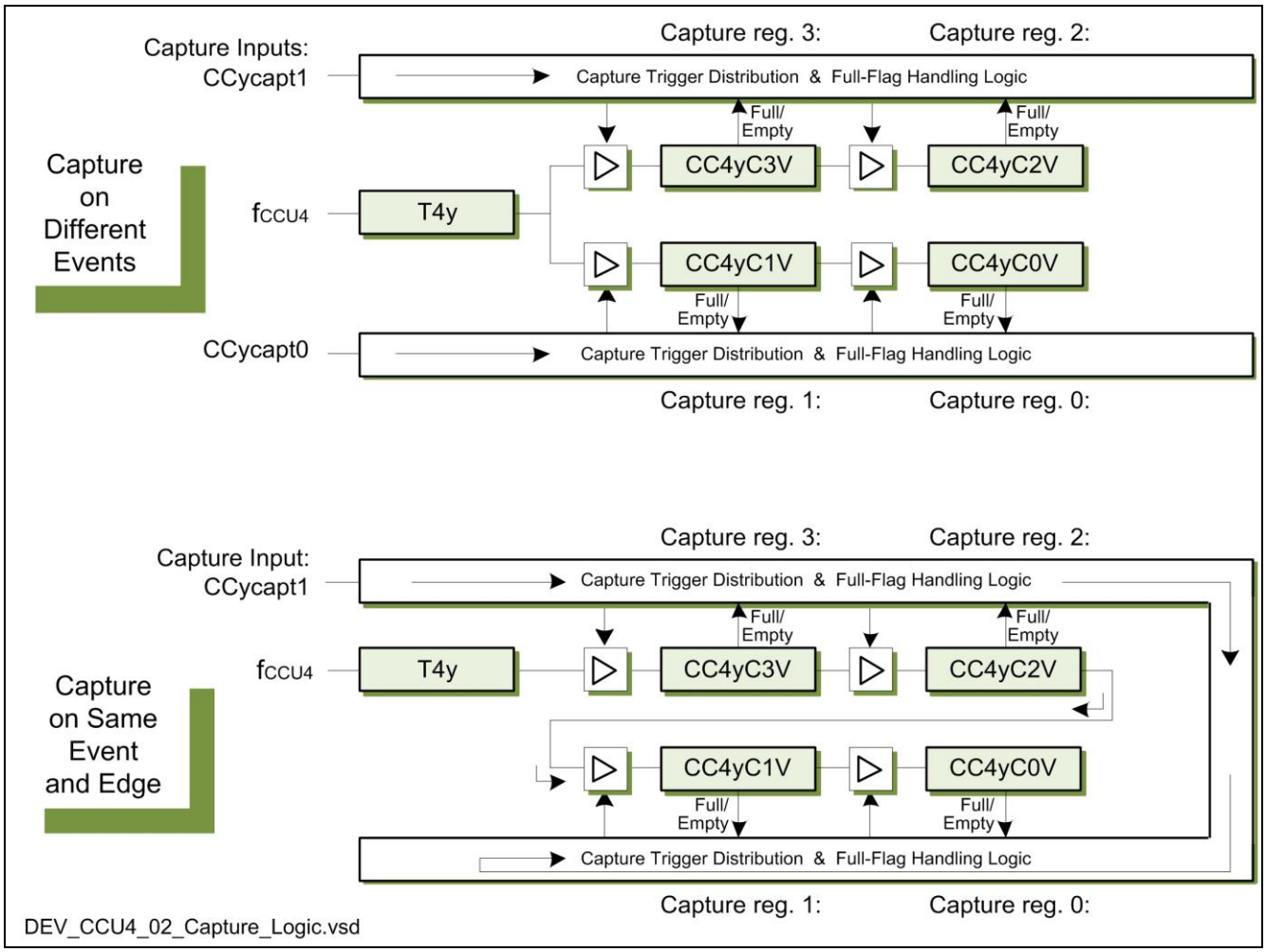


Figure 24 Slice Capture Logic

The Capture Algorithm

Each capture register has a Full-Flag that will be set on a capture-to and cleared on a read-from the register.

At a Capture Input Event (Capt1 or Capt0), each register will capture data from the next higher indexed register IF that higher register is full AND a lower indexed register is empty. The timer is seen as of highest index.

Disabling the Full-Flags

Continuous capturing without any effect from any Full-Flags is set by changing the bit CC4yTC.CCS = 1.

When set, registers will capture data on the Capture Input Events without taking account of the Full-Flags status.

3.1 Capture by External Events Control

This scenario involves linking the Capture0 or the Capture1 register pairs to external trigger requests from any of the following: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU.

A Connection Pin Table is given by the Top-Level Interconnection Matrix. Setup is via the CC4yINS and CC4yCMC registers

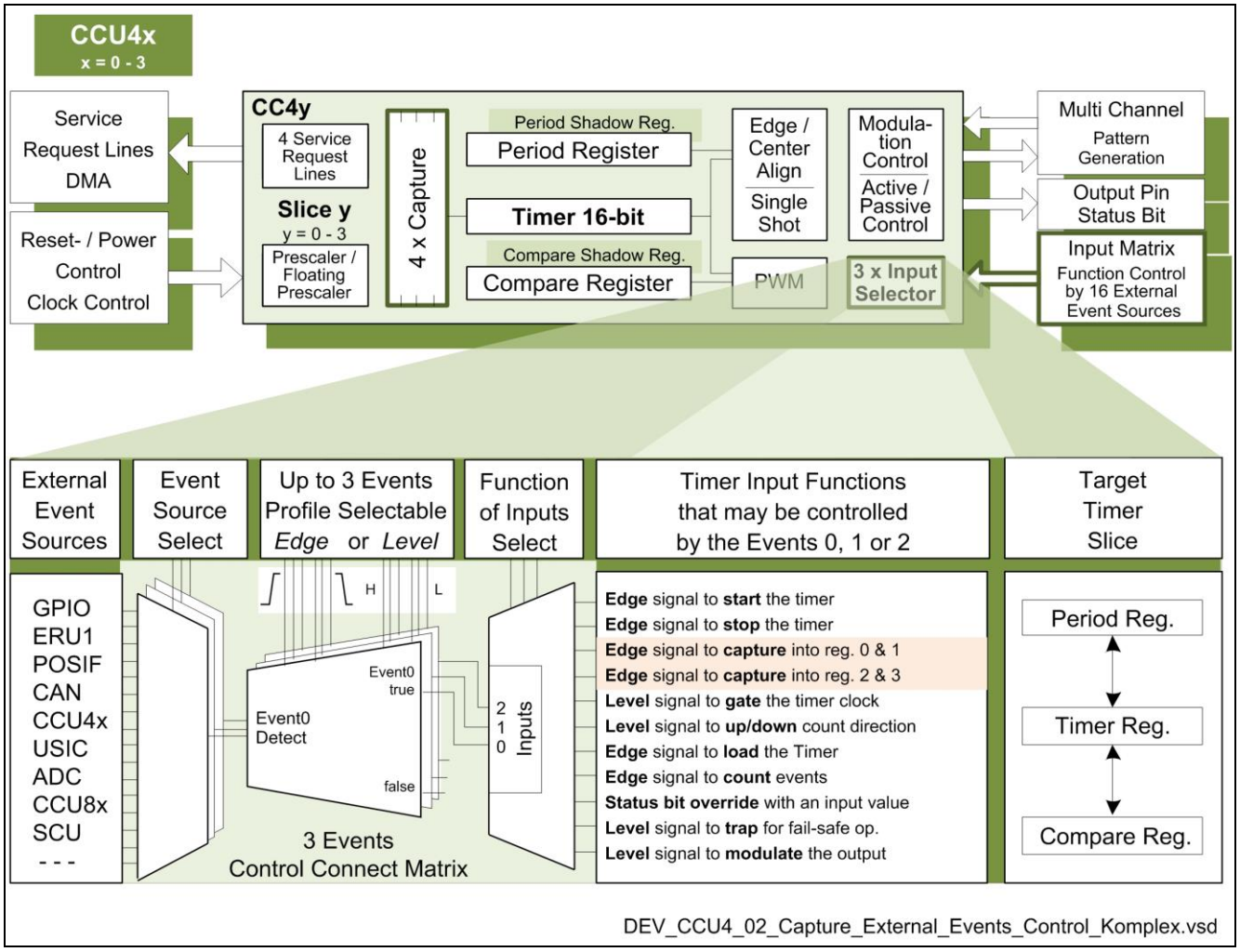


Figure 25 Capture by External Events Control

Timer Inputs for Capture

There are 3 selectable input lines with configurable source-event-condition profiles available for Capture by external event control functions, extendable in the CC4yTC register. There is also a read access register ,ECRD, that simplifies the administration of capture registers and full-flags, when more than one slice is used in Capture mode.

External Control by Capture Events

A capture event may trigger external actions via the Top-Level Interconnect matrix or request for an Interrupt. Each CAPCOM4 has four Service Request Lines and each slice has a dedicated output signal CC4ySR[3...0] selectable to a line by using CC4ySRS. For example, a capture event may request action from some other unit for an interrupt.

Top-Level Control of Event Requests to/from a Timer in Capture Mode

Top-Level control means also conditional control of event requests between a slice and other action providers. The Event Request Unit (ERU1) and the Top-Level Interconnect matrix may combine, control and link event signals according to user defined request-to-action event patterns. For example, capture on timer and other event status.

3.2 Getting Started with CCU4 and Capture Mode Features

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 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 any Event[2:0] Input Line to any of the 11 Input Functions. The registers CC4yTC and ECRD enable Extended Capture Read Mode options.

Using the External Capture Control Register Bitfields

- CC4yCMC.CAP0S Input Capture0 Selection and Event Number x=0-2 Input Line Selection bitfield
- CC4yINS.<Event-x bitfields> Event-x Source EVxIS (CC4yINS[P:A]) and Edge Mode EVxEM bitfields
- CC4yCMC. CAP1S Input Capture1 Selection and Event Input Line Number x=0-2 Selection bitfield
- CC4yINS.<Event-x bitfields> Event-x Source EVxIS (CC4yINS[P:A]) and Level Mode EVxEM bitfields
- CC4yTC.CMOD CAPCOM Mode bit should be set =1. (CMOD=0 is Compare Mode by default)
- CC4yTC.ECM Extended Capture Mode enable bit (=0 by default, if no extended read mode)
- ECRD Extended Capture Read Register (A Global read-only register, written by HW)

Note: The External Capture functions are Edge Sensitive.

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 Event Number 'x' for the CAP0S Function is always x == <CAP0S>-1 (when <CAP0S> != 0).

Note: The Event Number 'x' for the CAP1S Function is always x == <CAP1S>-1 (when <CAP1S> != 0).

Pseudo Code for the Capture0 or Capture1 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 MASK OPERATION first for bitfield focusing – if not first time – else:
CC4yCMC |= CAP0S_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Init External Capture0
// step 2
CC4yINS &= CAP0S_SOURCE_AND_EVENT_MODES_BITFIELDS_MASK; // Focus on bitfields
CC4yINS |= CAPOS_SOURCE_AND_EVENT_MODES_BITFIELDS_PATTERN; //Paste setup pattern
// step3
// Make MASK OPERATION first for bitfield focusing – if not first time – else:
CC4yCMC |= CAP1S_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Init External Capture1
// step 4
CC4yINS &= CAP1S_SOURCE_AND_EVENT_MODES_BITFIELDS_MASK; //Focus on bitfields
CC4yINS |= CAP1S_SOURCE_AND_EVENT_MODES_BITFIELDS_PATTERN; //Paste setup pattern

// step 1 and 3 could naturally be merged into one operation sequence, as follows:
// Make a MASK OPERATIONS first for bitfield focusing – if not first time – else:
CC4yCMC |= CAP01S_FUNCTIONS_AND_EVENTS_NUMBERS_BITFIELD //Init External Capture0&1

// step 2 & 4 could naturally be merged into one operation sequence, as follows:
CC4yINS &= CAP01S_SOURCES_AND_EVENTS_MODES_BITFIELDS_MASK; //Focus on bitfields
CC4yINS |= CAP01S_SOURCES_AND_EVENTS_MODES_BITFIELDS_PATTERN; //Paste both setups!
```

Single Shot Mode

4 Event Trigger Delay by Single Shot

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

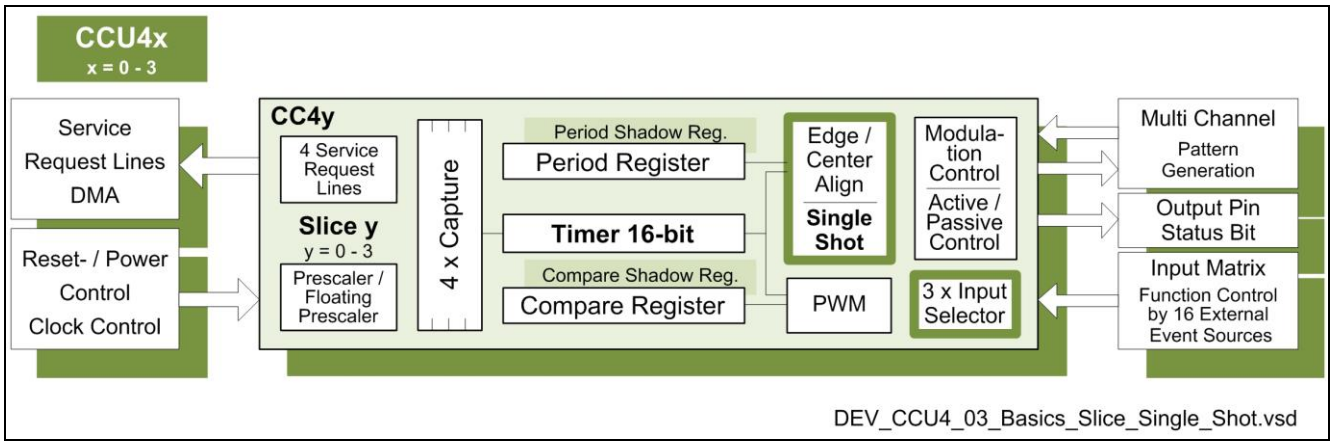


Figure 26 Basic Timer Slice for Single Shot

Use Cases

A timer in Single Shot mode has a specific role for applications where a certain delay has to be invoked between trigger events and operations that should be triggered. For example noise rejection in shunt-current signal measurement. Any CAPCOM4 timer may be setup in this mode and co-operate with other timers, ADC, or other modules.

Timer Setup in Single Shot Mode

A slice can be set into Timer Single Shot Mode (TSSM). Both the Timer and its Run Bit (TRB) is cleared by a Timer Period End that occurs after the TSSM bit is set, and the timer is stopped. A time frame, e.g. a Single Shot Delay, is set by the timer start conditions, selected counting mode and the period (PR) value.

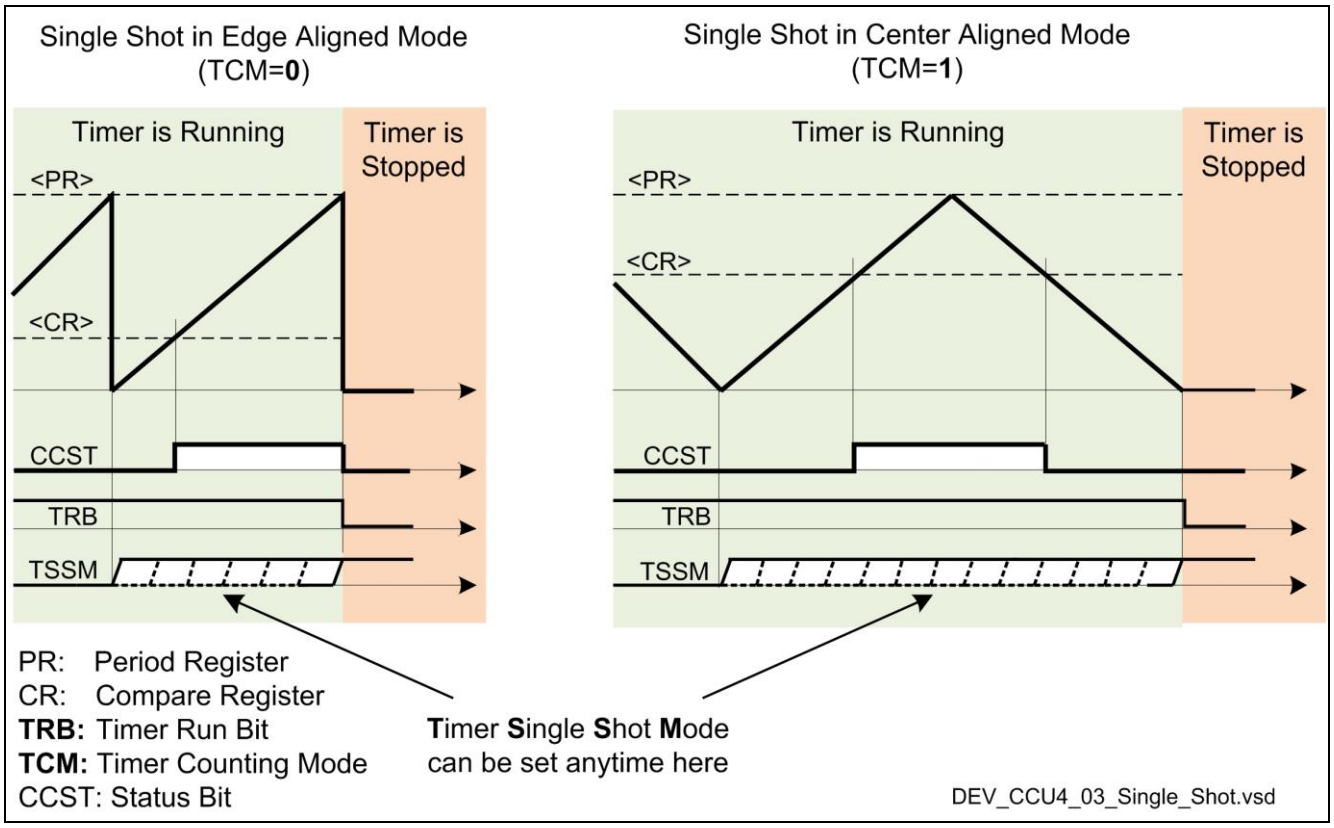


Figure 27 Timer in Single Shot Mode for Edge Aligned or Center Aligned Mode respectively

4.1 Using Timer Single Shot Delay for Noise Rejection

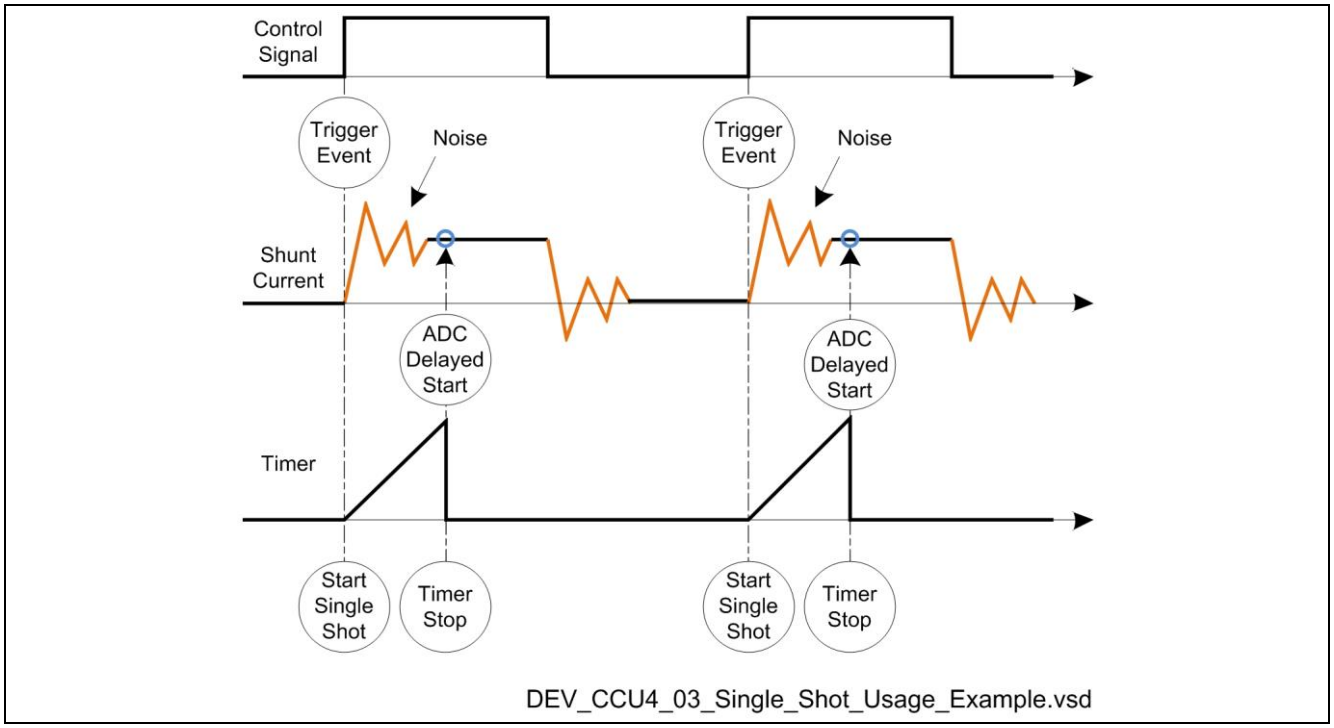


Figure 28 Timer in Single Shot Mode for Noise Rejection in Shunt-Current Measurement

4.2 Timer-Start in Single Shot Mode by External Event Control

A timer-start in Single Shot Mode can be linked to external triggers from sources such as: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU. Pin Connections are given by the Top-Level Interconnect matrix and the CC4yINS [P:A] Input Select vector and Function Select by the CC4yCMC register.

Timer Inputs for Start or Stop Facilities

A timer has 3 selectable function inputs with configurable source-event-condition profiles. These 3 function inputs can each have up to 16 sources for External Events Control, such as for Timer Start or Stop. The Extended functions such as Flush/Start, Flush/Stop or Flush-only may also be added in Single-Shot mode via the CC4yTC register.

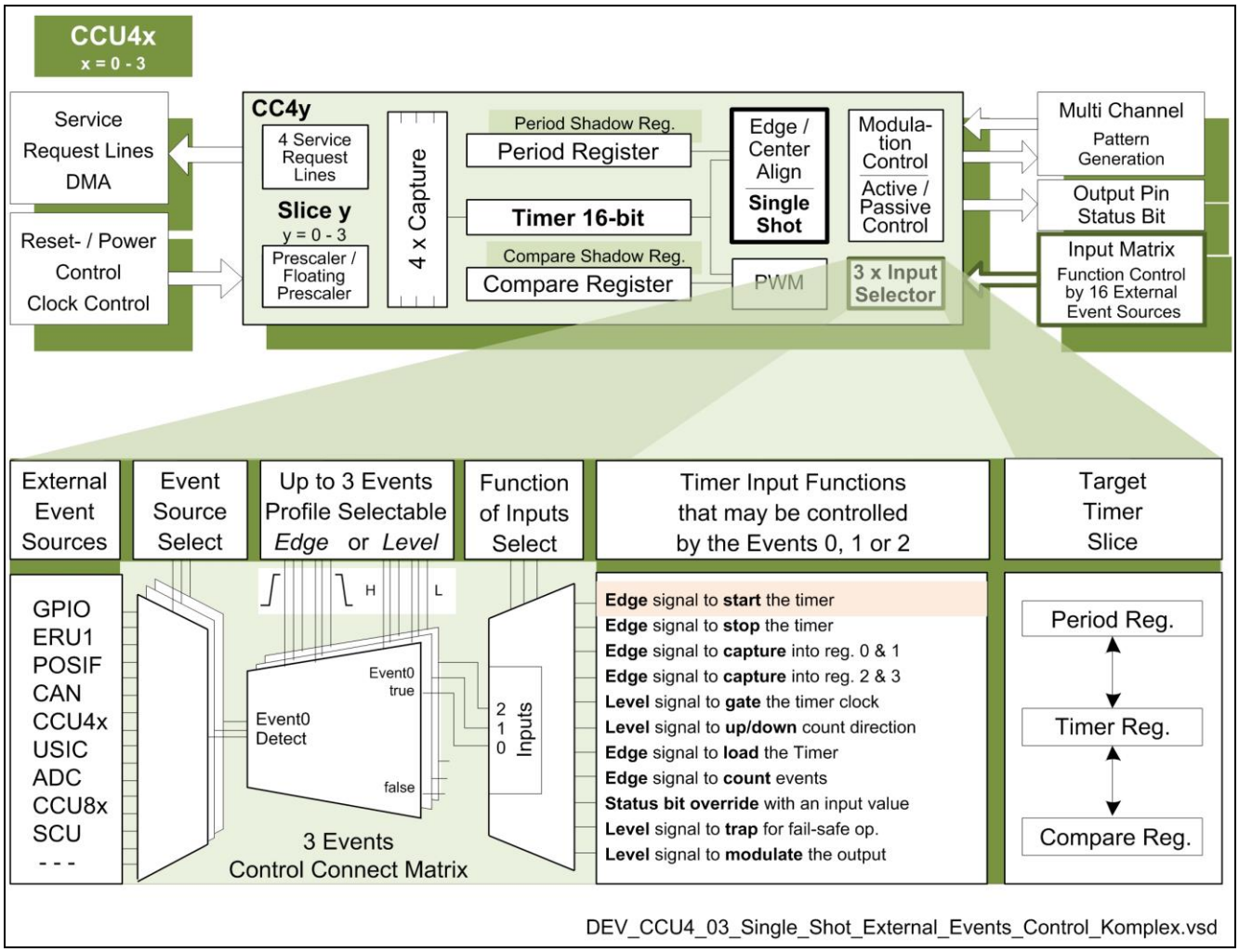


Figure 29 Single Shot Triggering on External Events

External Control by Single-Shot Events

Single-Shot events may trigger external actions via the Top-Level Interconnect matrix or they may request interrupts. Each CAPCOM4 has four Service Request Lines and each slice has a dedicated output signal CC4ySR[3...0] selectable to a line via CC4ySRS. So single-shots may for example act as a delayed trigger for ADC actions or interrupts.

Top-Level Control of Event Request to/from a Timer in Single-Shot Mode

Top-Level control also means conditional control of event requests between a slice and other action providers. The Event Request Unit (ERU1) and the Top-Level Interconnect matrix may combine, control and link event signals according to user defined request-to-action event patterns, such as ADC triggering limited by time windows.

4.3 Getting Started with CCU4 and Single Shot Mode

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 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 read-only compare register CC4yCR to the value from the compare shadow register CC4yCRS on each shadow transfer, if requested within the current period cycle by the Slice Transfer Set Enable bit GCSS.SySE.

Setting the Output Pin PASSIVE / ACTIVE Level Control

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

Initialize and Start the Timer

After the Start procedures have been executed, the software may:

- Initialize the slice timer CC4yTIMER or just clear it with a CC4yTCCLR.TCC bit=1 command
- Start the timer with a Set Timer Run command in the CC4yTCSET.TRBS bit

The timer may be stopped at any time by software with a CC4yTCCLR.TRBC command.

Using the Compare Mode / PWM Control Registers

- CC4yTC.TSSM Set Timer Single Shot Mode =1. (Default=0)
- CC4yTC.CMOD CAPCOM Mode selection bit should be set =0 (which is the default value)
- CC4yTC.TCM Timer Counting Mode Selection bit: 0 → Edge-Aligned / 1 → Center-Aligned
- CC4yCRS.CRS Write the required Compare Level value into the compare shadow register
- CC4yPR.PR Write the timer count cycle upper limit, Period Match, into the period register

Options:

- CC4yPSL.PSL PASSIVE/ACTIVE control bit: (CCU4xOUTy) = (CCUxSTy) xor (CC4yPSL.PSL)

- CC4yTIMER Initialize a timer start value (or flush with the CC4yTCCLR.TCC bit=1 command)
- GCSS.SySE Request Shadow Transfer to Initialize the Compare Register
- CC4yTCSET.TRBS Start Timer by software

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

Using the External Start Control Register Bitfields

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

Note: If the 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 read as 'Actual Event[2:0] Number'.

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

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

Using the External Stop Control Register Bitfields

Besides the Timer Single Shot Mode (when CC4yTC.TSSM bit is set), a timer may also be stopped by External Events Control, or even by its own events. The following Control Register Bitfields should be used:

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

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

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
```

External Events Control

5 Dynamic Control of Timer Functions on External Events

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

Synchronized Control of CAPCOM Units on External Events

The External Events Control distribution to CCUs (i.e. including CCU8) allows for advanced applications with synchronized timer control. For example in Motor Drive and Power Control, such as 3-Level Inverters requiring 12 synchronized PWMs.

5.1 External Control Basics

A slice may have its input functions controlled by external sources. The external source(s), active mode(s) and input function(s) should be mapped to the 3 inputs of the slice in the CC4yINS and CC4yCMC registers. Function mode extending alternatives can be added by selections in the CC4yTC Timer Slice Control register.

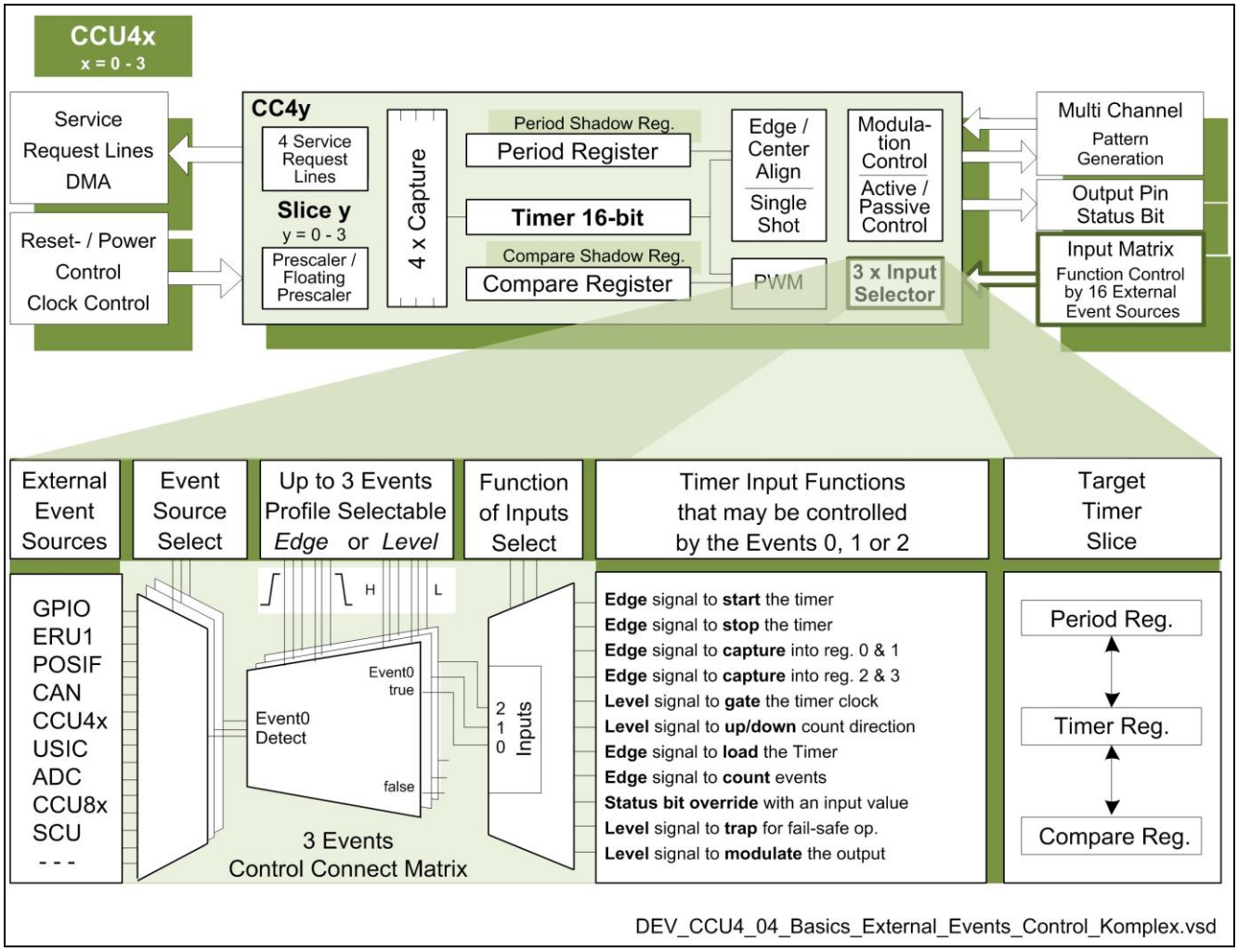


Figure 30 Timer Slice Input Functions Control on External Events via the System Interconnect Matrix

External Events Control of Timer Functions

An External Event Control request is a considered edge or level event signal from a peripheral unit or a GPIO , and is distributed to CCU4xCC4y slices input selection stages via a comprehensive matrix. A slice that by any of its 3 events setups can identify a considered source-event profile will be controlled by a selected input function!

Selection of External Events Control Sources

CCU4xCC4y Input Functions can be linked to external trigger requests from sources such as: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU. Pin Connections are given by the Top-Level Interconnect matrix and the CC4yINS[P:A] Input Select vector - and Function Select by the CC4yCMC register.

A CC4y internal event is also regarded as External Event: i.e. a CC4y can control itself by its own events.

Selection of External Events Control of Input Functions

There are 11 Timer Input Functions (such as Start the Timer), controllable by external events via 3 selectable input lines with configurable source-event profile conditions to the Timer Slices CC4y (y=0-3) of a CCU4x unit (x=0-3) for Start, Stop, Capture0-3, Gate, Up/Down, Load, Count, Bit Override, Trap and Modulate Output control.

The Input Functions are, due to their nature, controlled by either event edge or event level signals.

Extended Slice Input Functions

There are some Extended Input Functions in the CC4yTC register for the options Flush/Start, Flush/Stop or just Flush the timer and for an Extended Capture Mode option that via a read access register (ECRD) setup simplifies administration of capture registers and full-flags, when more than one slice is used in Capture mode.

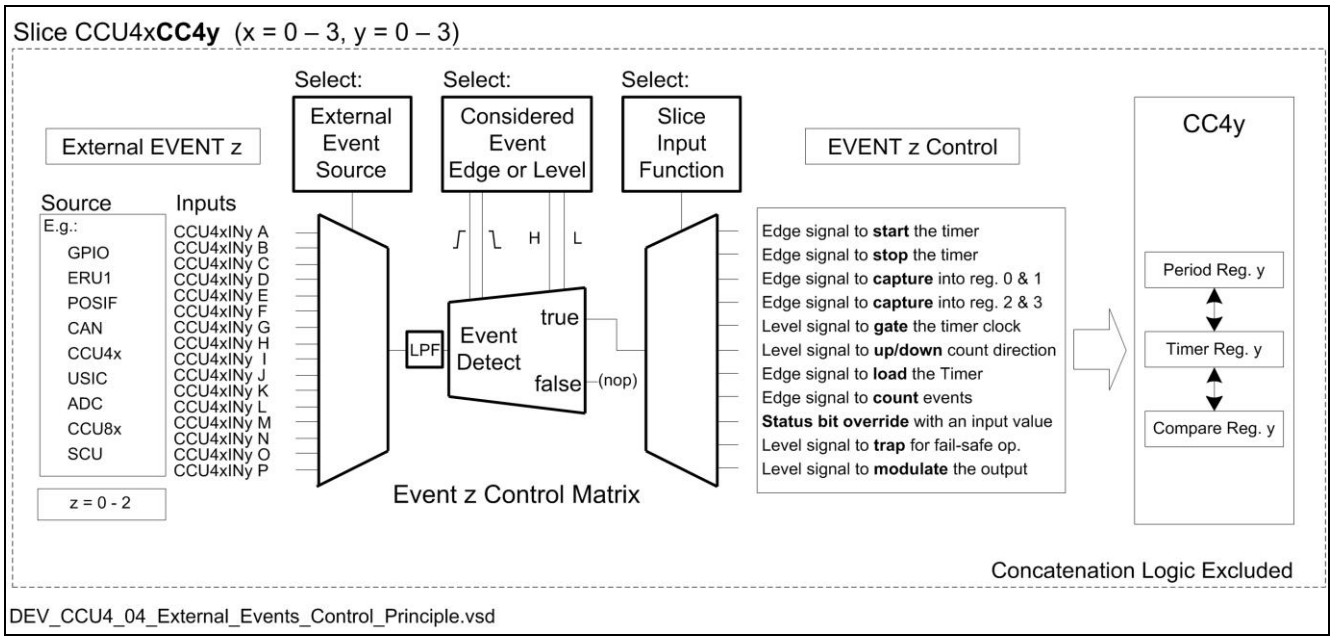


Figure 31 Principle Block Diagram illustrating External Event Control of a CCU4y Timer Slice

External Control by Timer Events

A timer event may either trigger external actions via the Top-Level Interconnect matrix or request for an Interrupt. Each CAPCOM4 has four Service Request Lines and each slice has a dedicated output

signal CC4ySR[3...0] selectable to a line via CC4ySRS, i.e. timer slice events may request for direct peripheral actions or request an interrupt.

Top-Level Control of Event Requests to/from a Timer Slice

Top-Level control also means conditional control of events requests between a slice and other action providers. The Event Request Unit (ERU1) together with the Top-Level Interconnect matrix may combine, control and link event signals according to user defined request-to-action event patterns, such as invoke I/O states, Time Windowing etc.

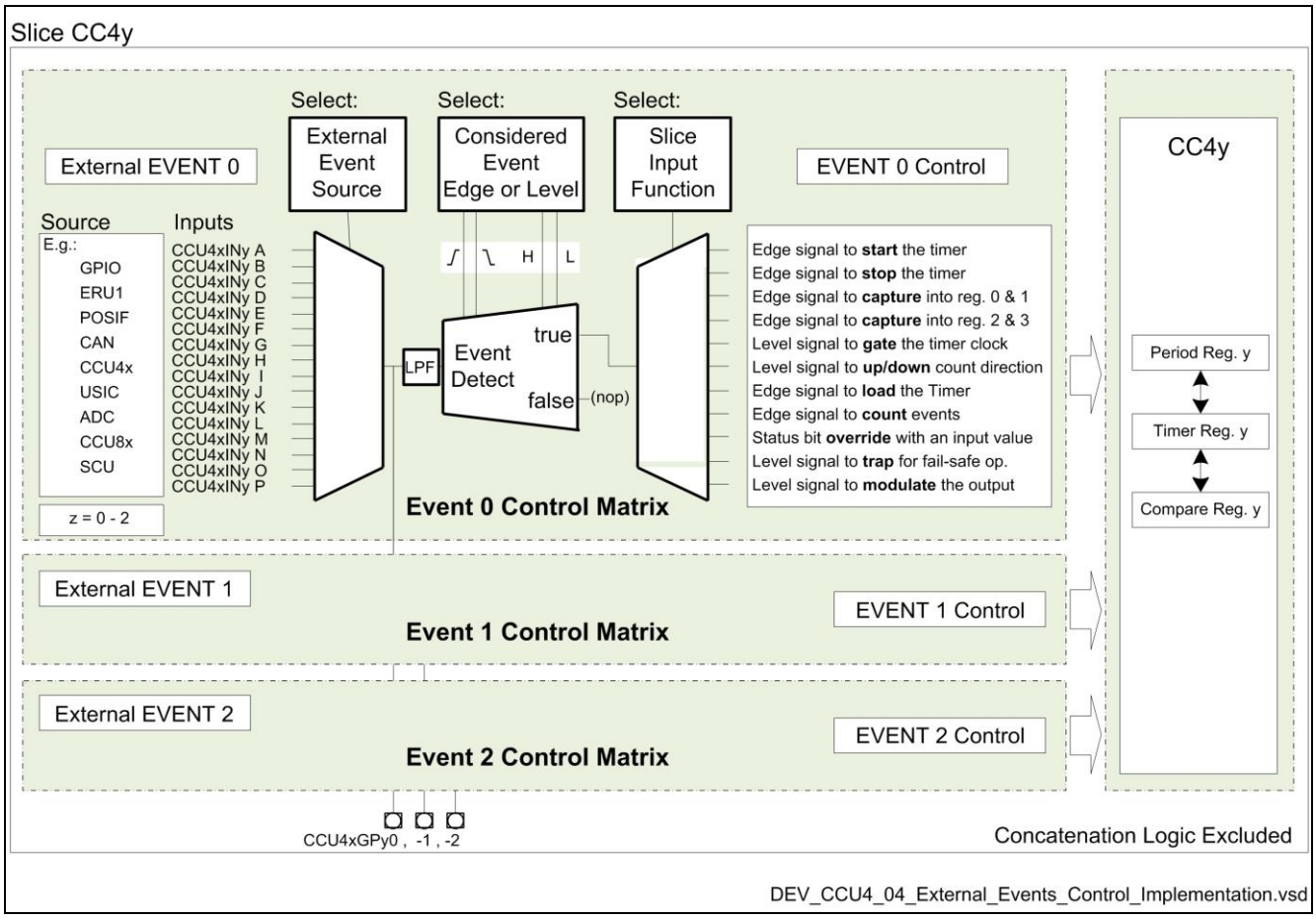


Figure 32 Block Diagram of the External Event Control Implementation

5.2 Getting Started with CCU4 and External Events Control

Getting Started with the CAPCOM4 Unit CCU4x

In the following examples, the CCU4 Initialization Sequence is assumed to have been followed.

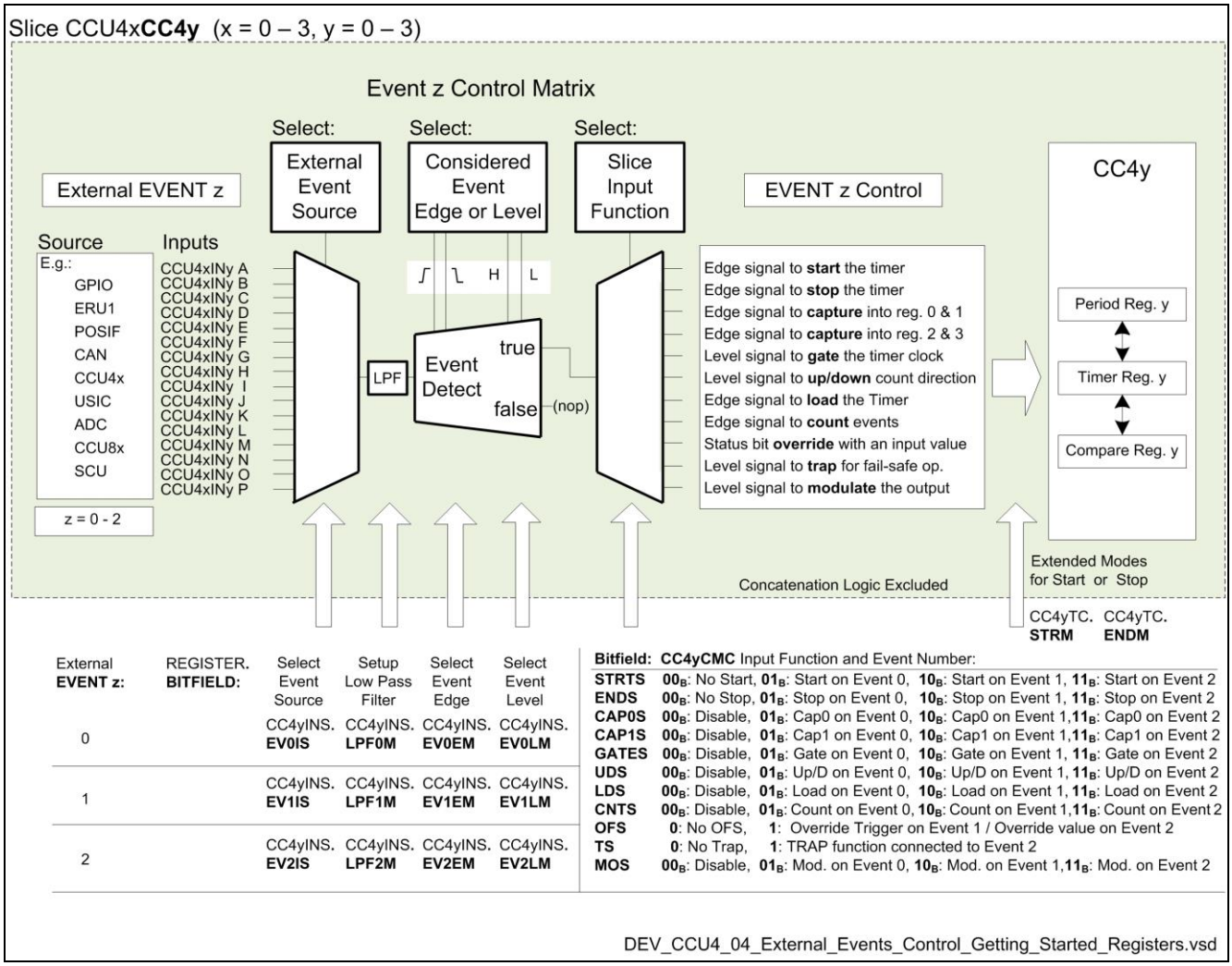
Using the CAPCOM4 External Events Control Setup Registers

There is usually just one Input Selection Register and one Input Function Control Register per CCU4x (x=0-3) to setup the entire functionality of External Event Control of all CC4y Slices (y=0-3). There are some extended Input functions available though in the CC4yTC register.

For example the options Flush/Start or Flush/Stop or just Flush.

- CC4yINS Input Selection Register for considered event profile (i.e. source and event type)
- CC4yCMC Input Function Selection Register

Initializations and functions are determined by writing to the respective register bitfields.



External EVENT z:	REGISTER. BITFIELD:	Select Event Source	Setup Low Pass Filter	Select Event Edge	Select Event Level
0		CC4yINS. EV0IS	CC4yINS. LPF0M	CC4yINS. EV0EM	CC4yINS. EV0LM
1		CC4yINS. EV1IS	CC4yINS. LPF1M	CC4yINS. EV1EM	CC4yINS. EV1LM
2		CC4yINS. EV2IS	CC4yINS. LPF2M	CC4yINS. EV2EM	CC4yINS. EV2LM

Bitfield: CC4yCMC Input Function and Event Number:

STRTS	00_B: No Start, 01_B: Start on Event 0, 10_B: Start on Event 1, 11_B: Start on Event 2
ENDS	00_B: No Stop, 01_B: Stop on Event 0, 10_B: Stop on Event 1, 11_B: Stop on Event 2
CAP0S	00_B: Disable, 01_B: Cap0 on Event 0, 10_B: Cap0 on Event 1,11_B: Cap0 on Event 2
CAP1S	00_B: Disable, 01_B: Cap1 on Event 0, 10_B: Cap1 on Event 1,11_B: Cap1 on Event 2
GATES	00_B: Disable, 01_B: Gate on Event 0, 10_B: Gate on Event 1, 11_B: Gate on Event 2
UDS	00_B: Disable, 01_B: Up/D on Event 0, 10_B: Up/D on Event 1, 11_B: Up/D on Event 2
LDS	00_B: Disable, 01_B: Load on Event 0, 10_B: Load on Event 1, 11_B: Load on Event 2
CNTS	00_B: Disable, 01_B: Count on Event 0, 10_B: Count on Event 1,11_B: Count on Event 2
OFS	**0**: No OFS, **1**: Override Trigger on Event 1 / Override value on Event 2
TS	**0**: No Trap, **1**: TRAP function connected to Event 2
MOS	00_B: Disable, 01_B: Mod. on Event 0, 10_B: Mod. on Event 1,11_B: Mod. on Event 2

Figure 33 Initialization of External Event Control Registers

Using the 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.

External Event Control Register Bitfields

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

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 Event Number 'x' for the Input Function is x == <function>-1 (when <function> != 0). <function> == 0 means "disabled".

Example: Start a Timer by External Event Control

```
// Assume a falling edge transition on GPIO Pin P1.3 as a "considered" external
// event control signal to start Timer CC40 in CAPCOM Unit CCU40.
// Define this event as "Event2". (The alternatives Event0..1 would do as well!):
CC40CMC |= 0x00000003; // Start Timer CC40 by External Event 2 (Bitfield STRTS).

// Setup Timer CC40 External Event Control Input Selection register (CC40INS)
// to activate GPIO Pin P1.3 as selected source for Event 2 in the Bitfield EV2S:
EV2S = 0; // where "0" is the input number (from 0-15) for the input
       // "CCU40IN0[A]" and represents Pin P1.3 according to the Top-level
       // connection Matrix.
EV2EM = 2; // set signal active on falling edge on Event 2, as "considered".
```

Signal Quantification & Quality Enhancement by Dithering

6 Signal Quantification & Quality Enhancement by Dithering

Dithering is a technique used to reduce or remove the correlation between noise and signal in audio or graphic signal processing technique, and it is also used to enhance EMC quality by distributing portions of the signal quantification error in time. This effect can be utilized to enhance PWM DC-level precision or Micro Ticks tuning.

6.1 A simple case of value adding by Dithering PWM

Consider a Buck Converter, from V_{in} to V_{out}, such as shown in the following diagram. In this example, the PWM frequency is 800 kHz and the accuracy (precision) requirement 0.1%. If one assumes a timer clock frequency of 20 MHz, then the timer period would only be able to perform a 6-bit counter length, which implies that a precision of 1.6% would be achievable.

Dithering by Software

Theoretically, it would be possible to control the average value, in-the-long-run, for a 0.1% precision with a 6-bit counter by using a loop control that modified the PWM from one timer period to another in some weighted way (and signal smoothing via an LC-filter). But this is not practical for software because of the timing constraint and CPU load.

Dithering by Hardware

Dithering is a hardware option in all the CAPCOM4 timers and may leverage PWM DC-level precision up to 16 times. This special hardware part may create a kind of superimposed jitter upon a timer PWM output signal, to such an extent that the affected PWM pattern DC-level (V_{out}) will converge to an average value with the targeted precision level.

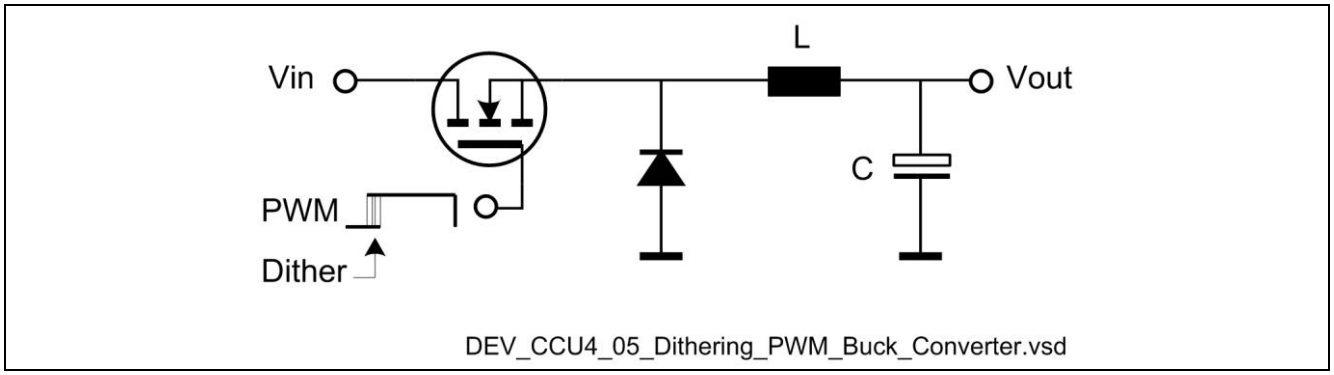


Figure 34 Buck Converter with high precision requirement

6.2 Adding by value by dithering Micro Ticks

Micro Ticks can be used in the interpolation between for example the sensor pulses from a cog wheel, to achieve position monitoring with higher precision. However, since the divisions of the sensor pulse period intervals seldom become exact multiples of micro tick periods, dithering can optimize the distribution of ticks over time.

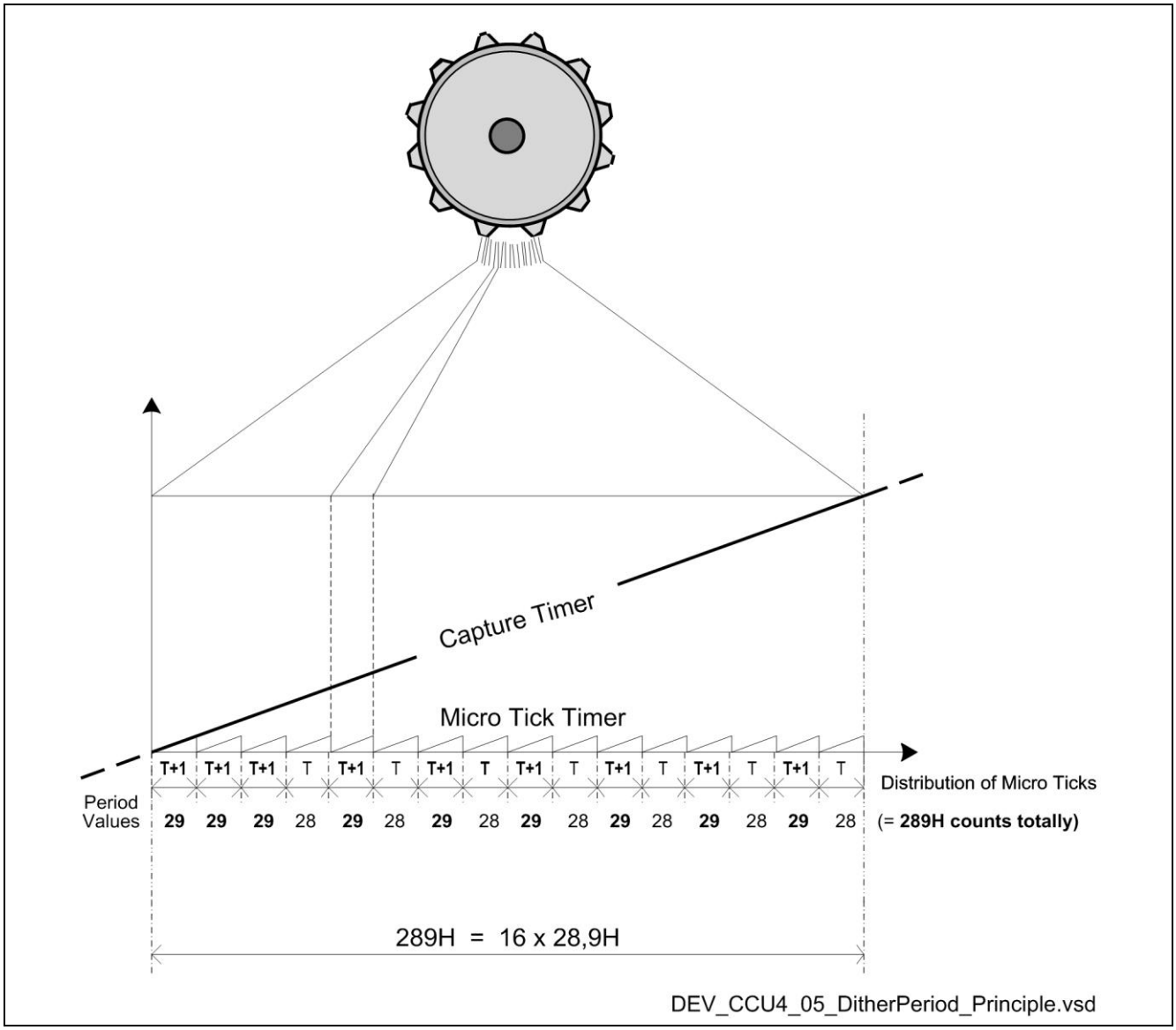


Figure 35 Optimized distribution of Micro Ticks within a certain time-slot by Dithering

6.3 The Dithering Implementation

For each timer there is an extra counter, a so called Dither Counter, which can dither the timer PWM edges. The precision is gained by the same magnitude as the count length of the counter.

In the current implementation there is 4-bit Dither Counter for each CAPCOM4 slice timer, therefore offering a precision leverage by 16 times.

The Dithering Function

Dithering is accomplished by increasing the PWM timer period with 1 bit-time or by delaying the compare event for the pulse width with 1 bit-time (or both these operations) at certain periods of the PWM timer. This is controlled by the dither logic and the algorithms that follows.

The Dithering Range

For example, if the compare event is repeatedly delayed with 1 bit-time in 9 of 16 periods, in a cyclic manner, then the average PWM value would be reduced with 9/16 of a bit-time: i.e. there is a trivial

way of fine tuning the precision of a PWM signal average value within a dynamic range of 1 bit-time length.

The Dithering Algorithm

By utilizing the linear relationship between the average bit-time and the quotient of periods using delayed compare events within a specific cycle length of timer periods, then have the mechanism for precision enhancement by dithering. The precision leverage is proportional to the specific cycle length:

- A 4-bit binary Dither Counter gives the PWM timer a 4-bit precision extension, from 16 to 20 bits.

The Dithering Distribution

A Dither Counter with a binary representation would not give an optimal distribution of dithered periods, with regard to how the quantification errors would be spread in time (for low pass filtering or for EMC), so the Dither Counter values will be split up according to a bit-reverse representation, as demonstrated in the following diagrams.

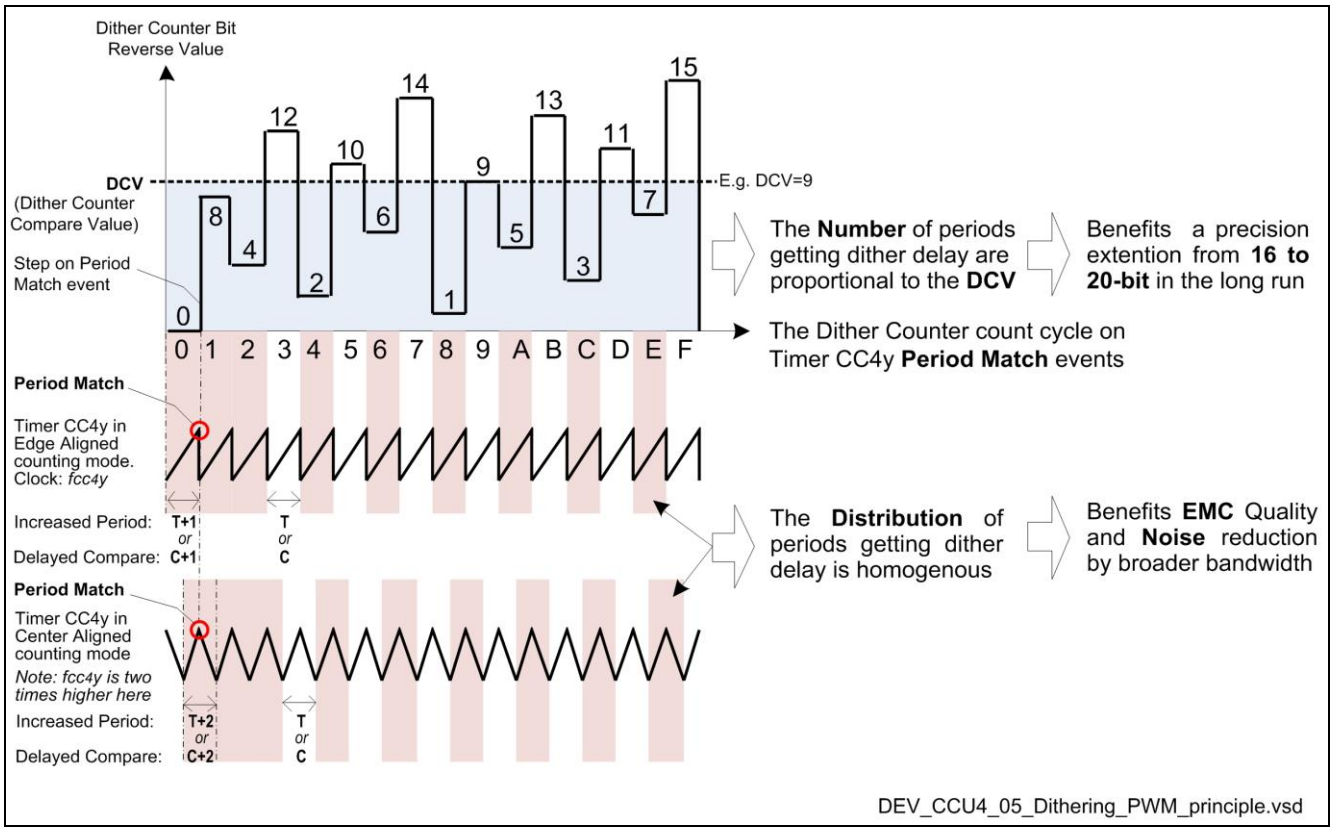


Figure 36 Demonstration of the Dithering Function for Edge or Center Aligned Timer Periods

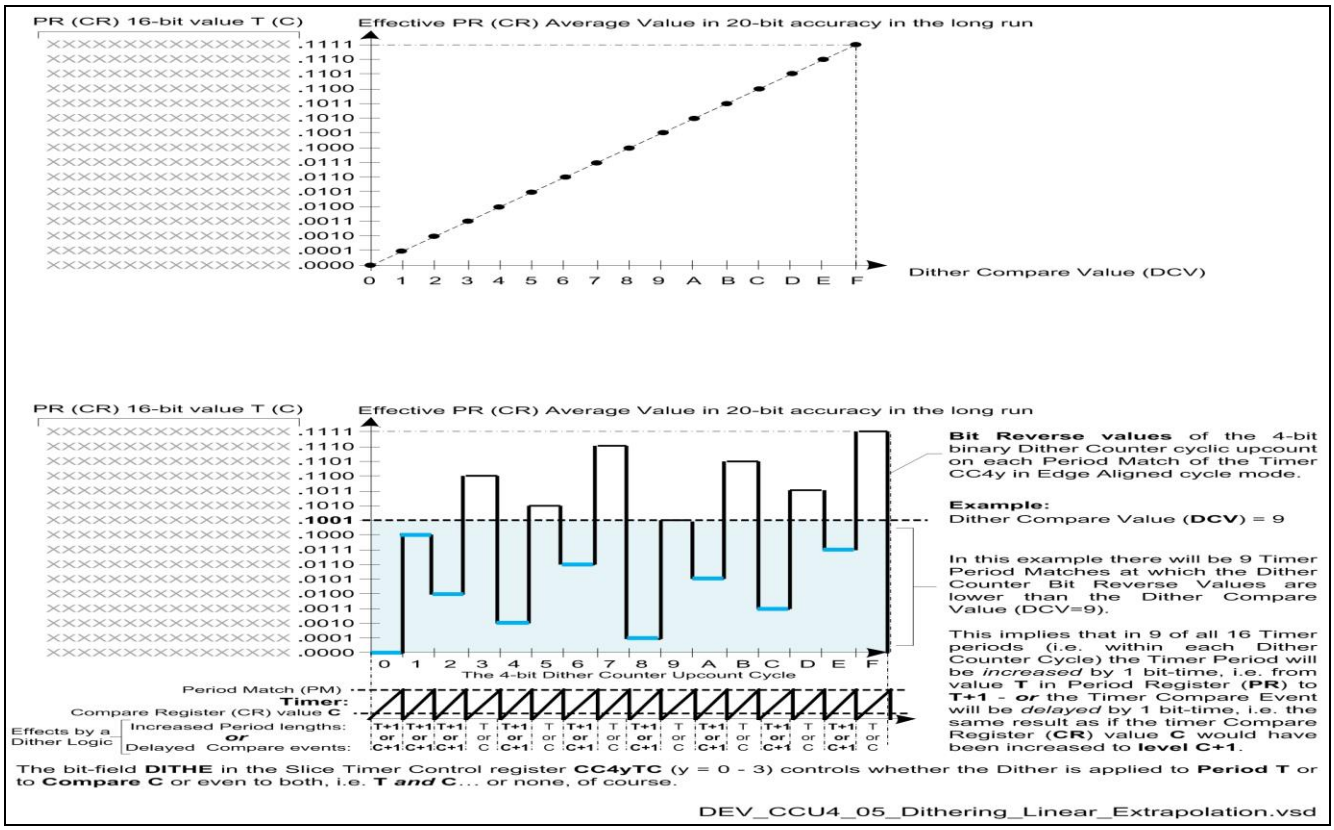


Figure 37 Linear Relationship: Average Bit-Time vs. Quotient of Periods with Delayed Compare Events

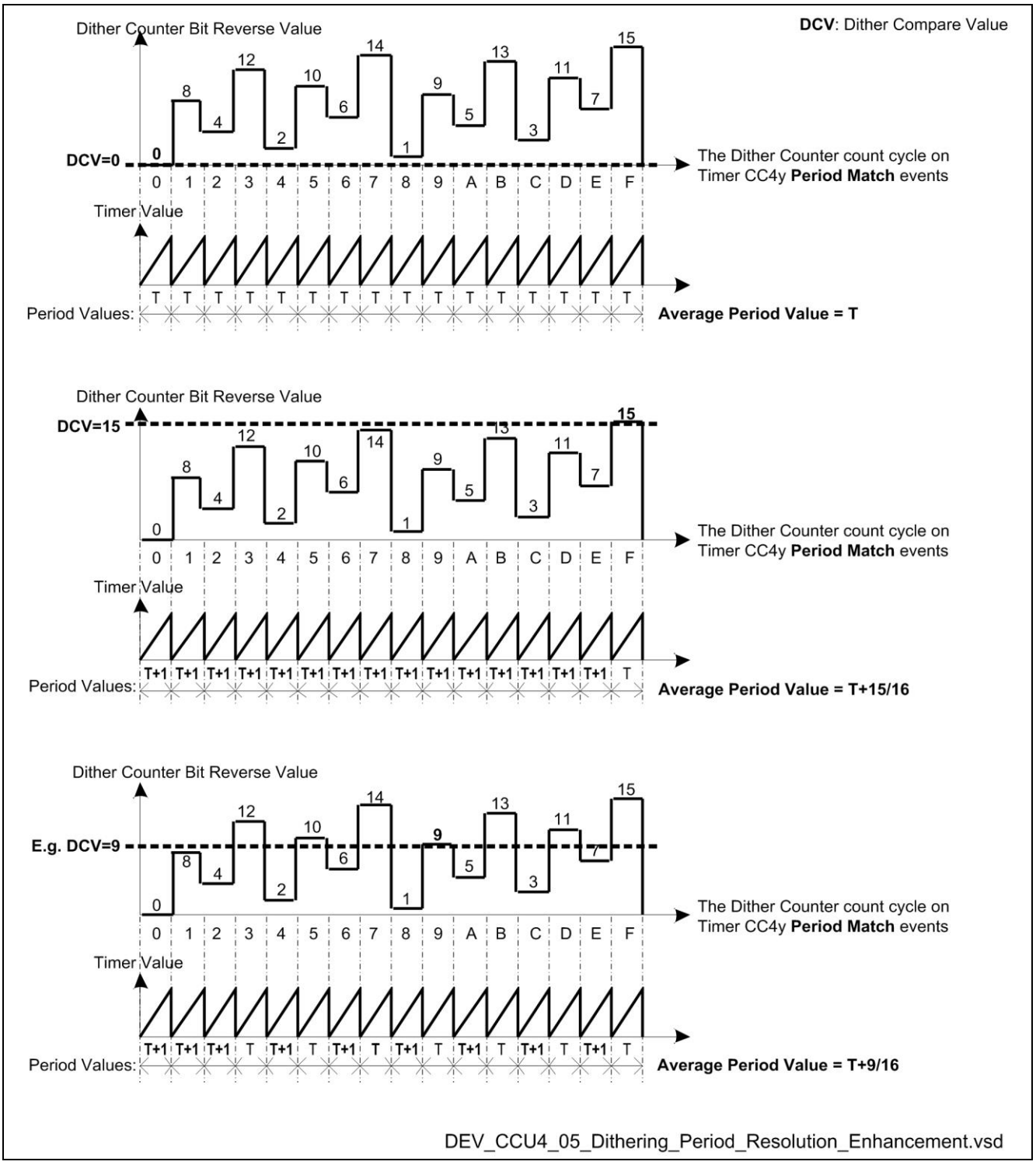


Figure 38 Dithering of Period Width in Edge Aligned Mode

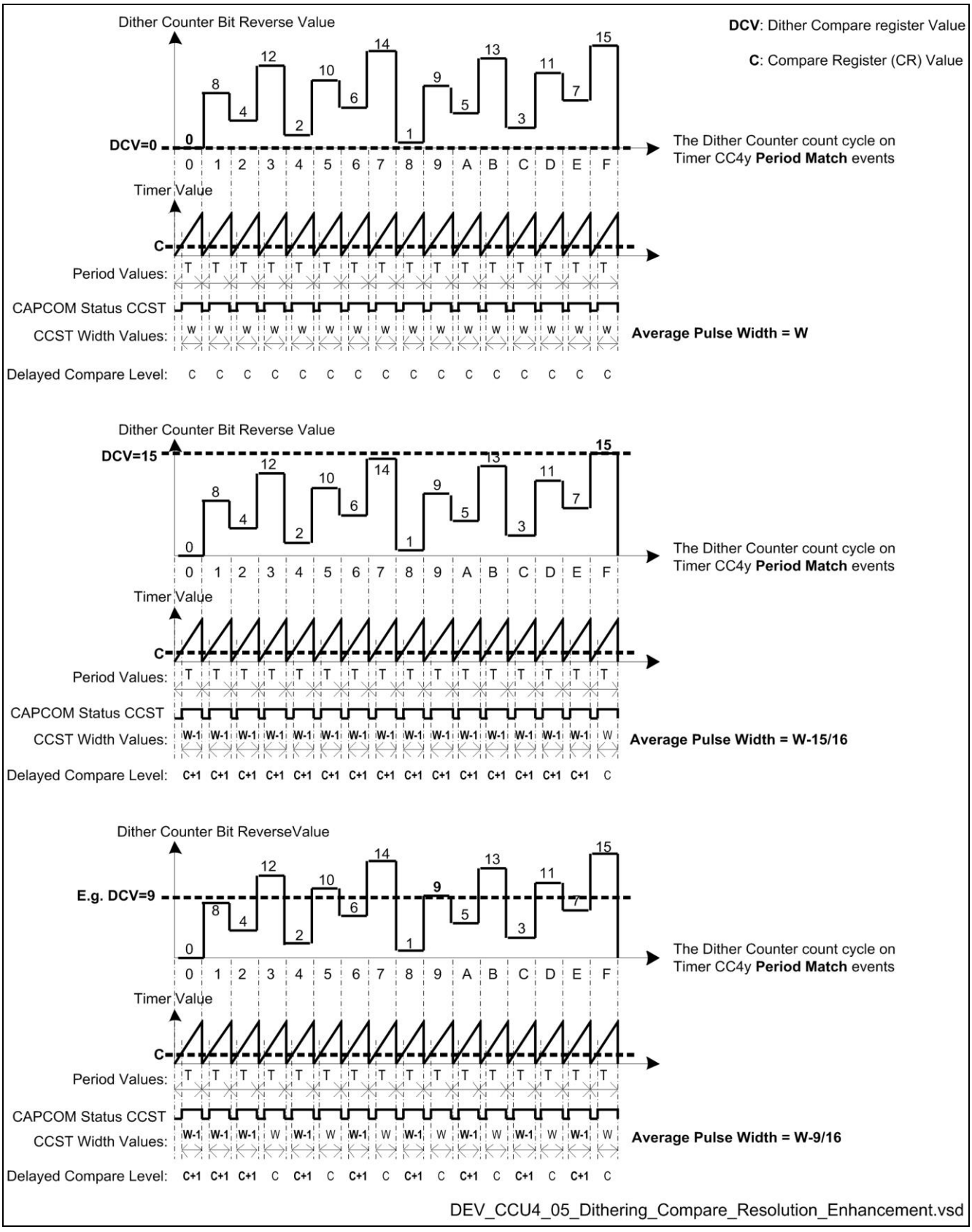


Figure 39 Dithering of Period Width by Compare in Edge Aligned Mode

6.4 Getting Started with CCU4 and Dithering

Dithering Setup

Dither mode and initialization should be setup separately or together with other parameters represented in a Slice y (y=3.0) Timer Control register CC4yTC.

Setting an Average Compare Level by Dithering

Assume that an Average Compare Level of 28.9H by Dithering must be set:

After setting the Dither Compare Match Enable (DITHE) the compare values that are targeted for reload operations should be written into the CC4yCRS respective CC4yDIT shadow registers. However, no transfer from those shadow registers will be done until the Transfer Set Enable bits, SySE, SyDSE in GCSS are preset.

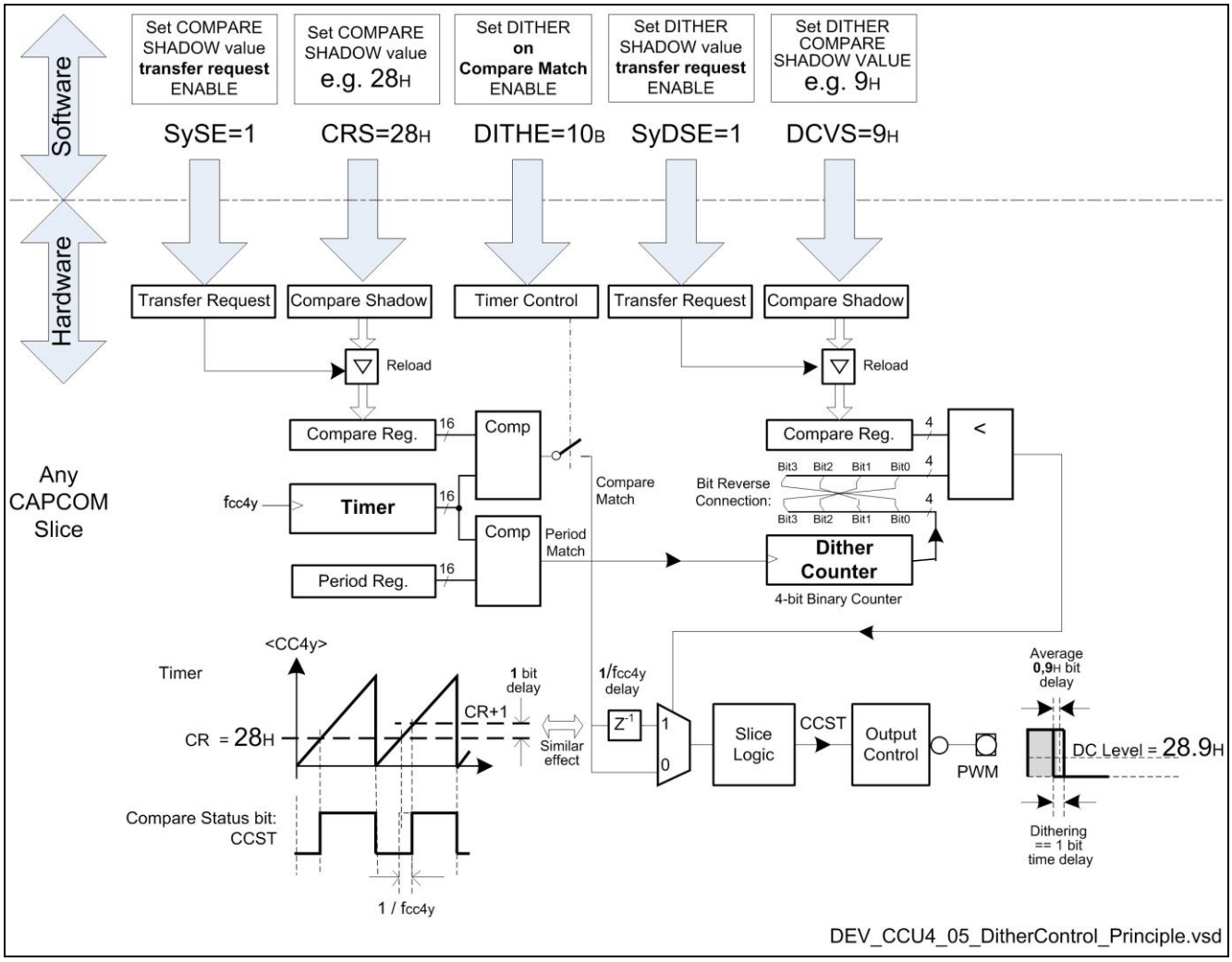


Figure 40 Getting Started Demo – Setting an Average Compare Level of 28.9H by Dithering

6.5 Runtime Handling of CCU4 and Dithering

Reload of Dithering PWM parameters

All CAPCOM4 timers, in any slice configuration, no matter how complex or for what signal pattern, are assured coherent update by hardware of all relevant timer function parameters from update values in

shadow registers that, on a global preset request, are transferred simultaneously to the respective function registers at Period-Match or One-Match.

Shadow Transfer of Compare and Dither Register values

The compare values that are targeted for re-load operations must be written in the CC4yCRS resp. CC4yDIT shadow registers AND the respective Transfer Set Enable bits (SySE, SyDSE in GCSS), must be preset at the latest within the clock cycle of the Period Match (in Edge Aligned Mode) or Period/One Match (in Center Aligned Mode).

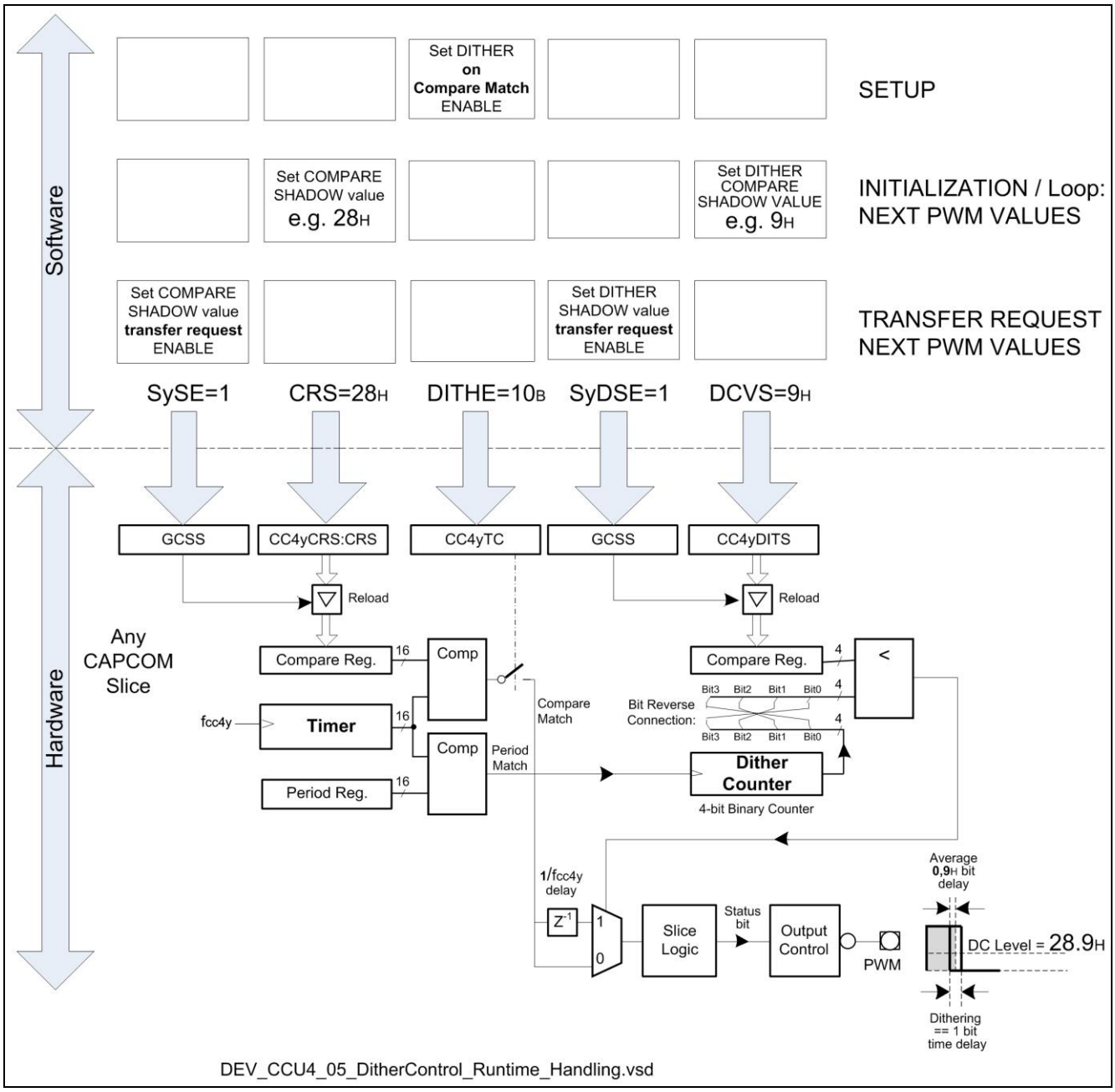


Figure 41 Setup and Reload of Dithering PWM

Top-Level Control of Event Requests to/from a Timer in Dithering Mode

Top-Level control means also conditional control of event requests between a slice and other action providers. The Event Request Unit (ERU1) and the Top-Level Interconnect matrix may combine,

control and link event signals according to user defined request-to-action event patterns. They can be used to supervise Dithering Reload.

6.6 Example Application for CCU4 and Dithering

Demo – Using Dithering in Micro Tick Generation

Micro Ticks are clock pulses that are created and distributed within the time-slots of a pulse train for resolution enhancement purposes, as in the Position Monitoring of a rotating cog wheel. The time-slots are given by a position sensor and detected by a Capture Timer. The Micro Tick Timer is controlled by specific algorithms.

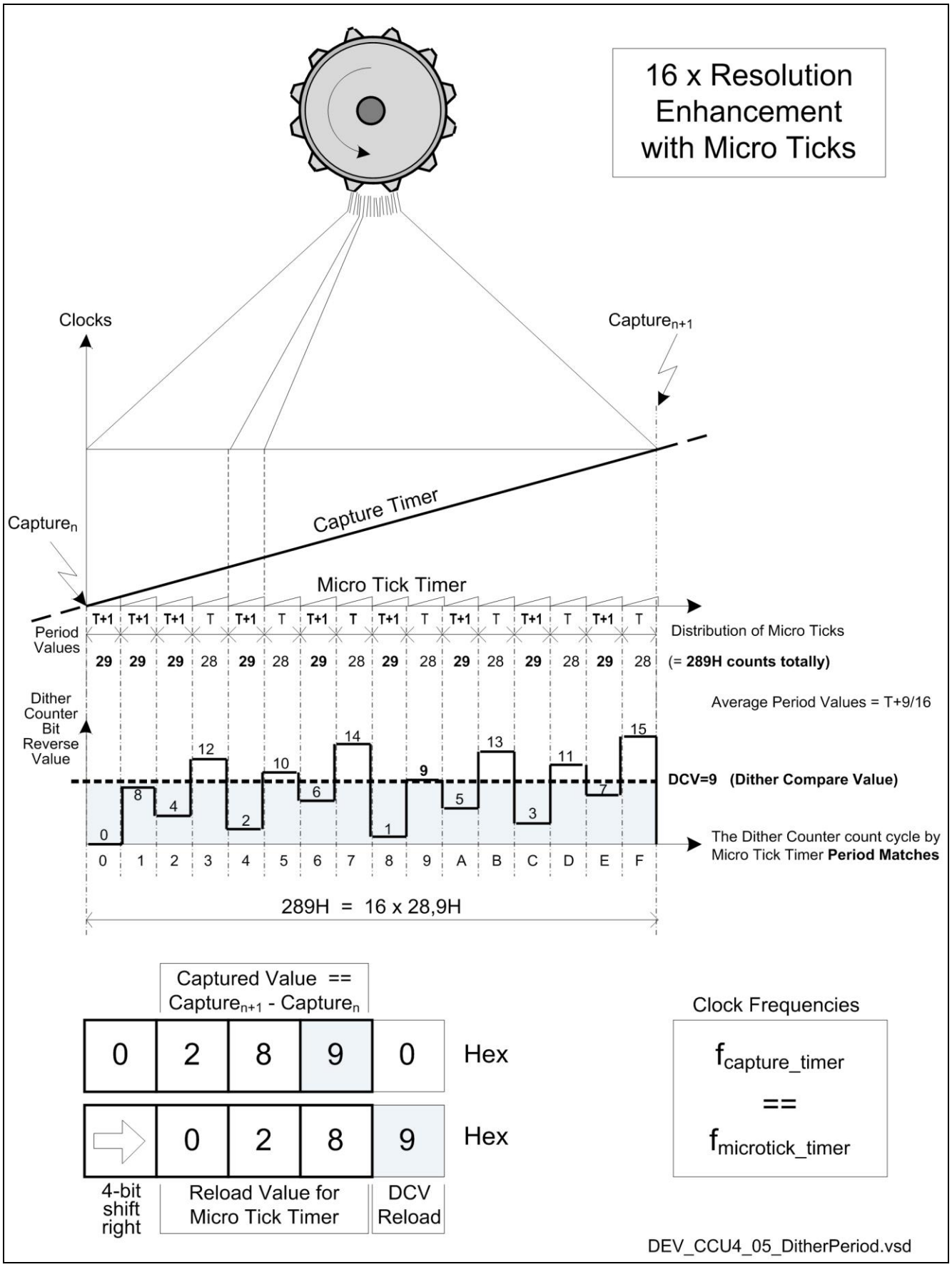


Figure 42 The calculation process for Micro Ticks distribution over period time by Dithering

The controlling algorithms may utilize dithering to optimally distribute the micro ticks. The clock rate quotient between the Micro-Tick and Capture timers could be simplified by a 4-bit shift. However, the algorithms have to take account of time-slot variations too, since the micro ticks are by nature calculated retrospectively.

Demo – Enhancing Micro Tick Resolution

By magnifying the clock frequency quotient difference between the capture timer clock and the micro ticks timer clock by factors of 2, the micro ticks resolution will be enhanced accordingly with sustained optimization in distribution without changing the algorithm. The position measurement precision can be in this way magnified.

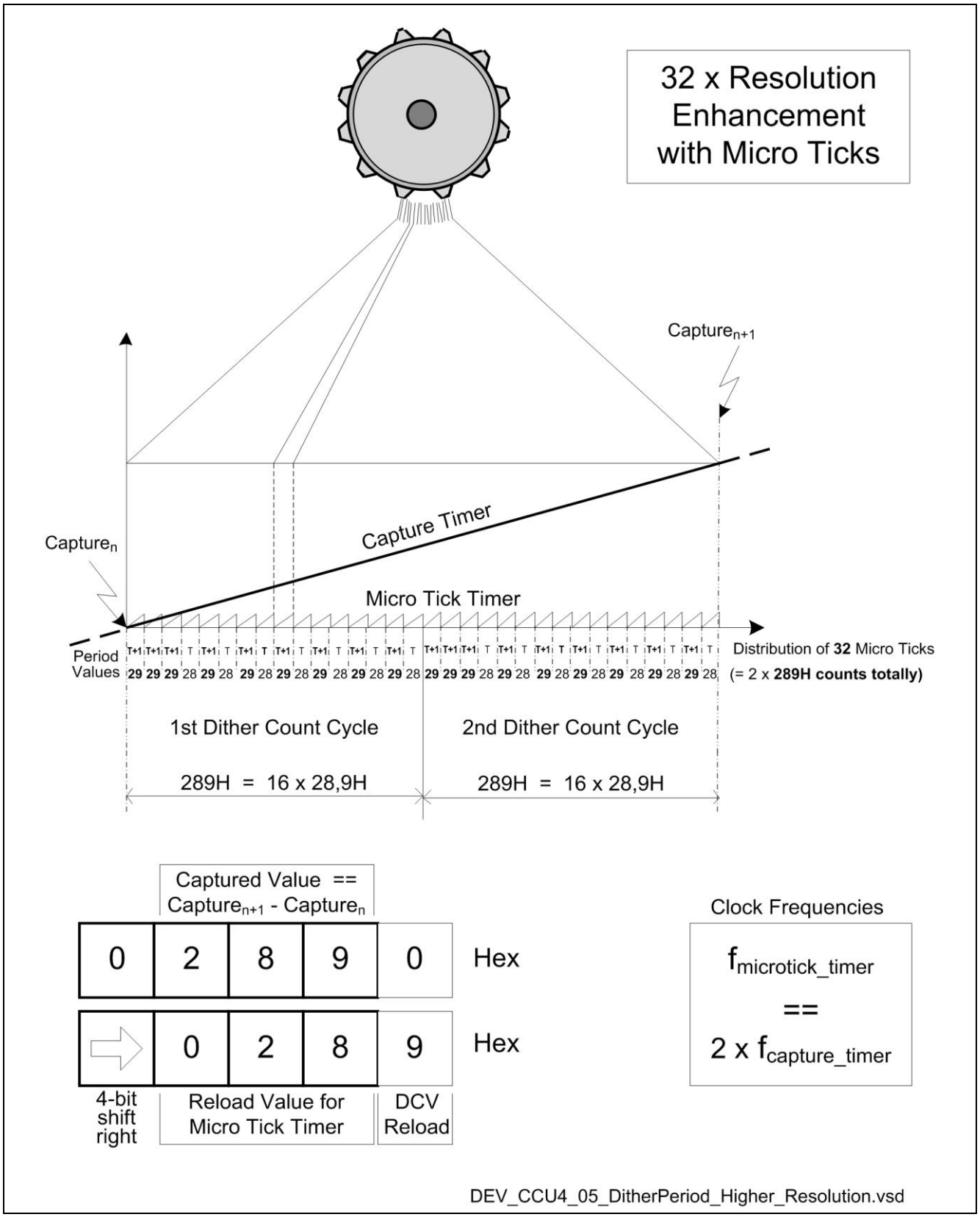


Figure 43 Further enhancement of resolution over period time with Micro Ticks by Dithering

Autonomic Signal Measurement Range Adaption using the Floating Prescaler

7 Autonomic Signal Measurement Range Adaption

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

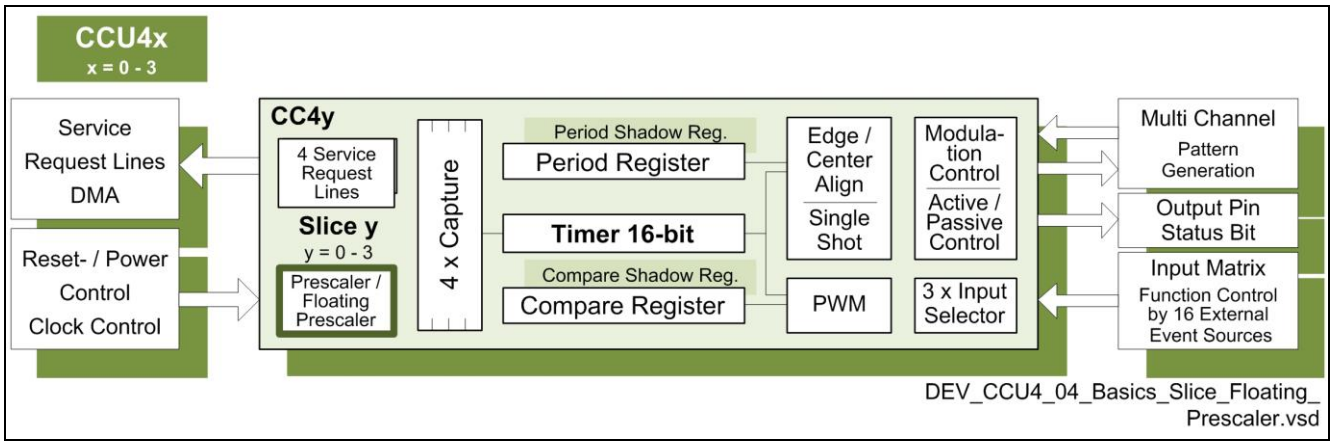


Figure 44 Timer Slice Prescaler with Normal or Floating Prescaler Capability Selectable

Floating Prescaler Use Cases

Floating Prescaler is in the first instance designated for Capture signal monitoring but it also fits for Compare operations.

With Periodic, successive halving of the timer clock frequency, when no capture (or compare) event occurs, the dynamic range can be autonomously adapted to any time length, limited only by the timer configurations.

The Floating Prescaler Principle Function

The following figures show the timing diagrams for the usage of floating prescaler in the different counting modes. There is a Prescaler Initial Value (PSIV) table, which sets the start level of the prescaler value (PVAL) that will be doubled periodically as long as no timer event occurs. This resumes again for the next monitoring cycle.

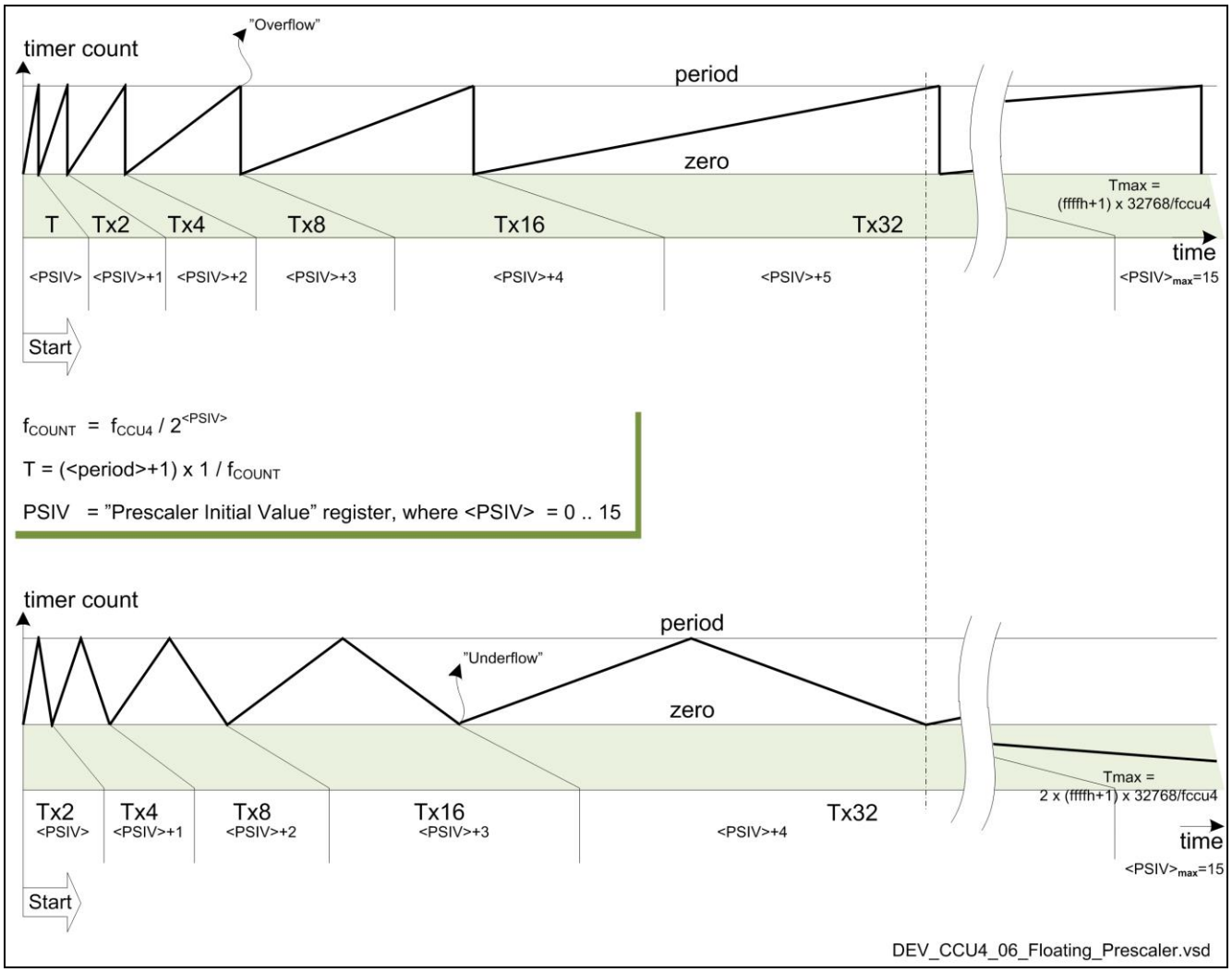


Figure 45 Floating Prescaler Principle

The Floating Prescaler in Signal Measurements by Capture

The cyclic capture procedure with floating prescaler is useful for signals with unpredictable events.

By using floating prescaler in order to automatically tune into an appropriate magnitude of timer clock frequency, you are avoiding CPU load.

Time calculation is based on all successively doubled period lengths and the capture value.

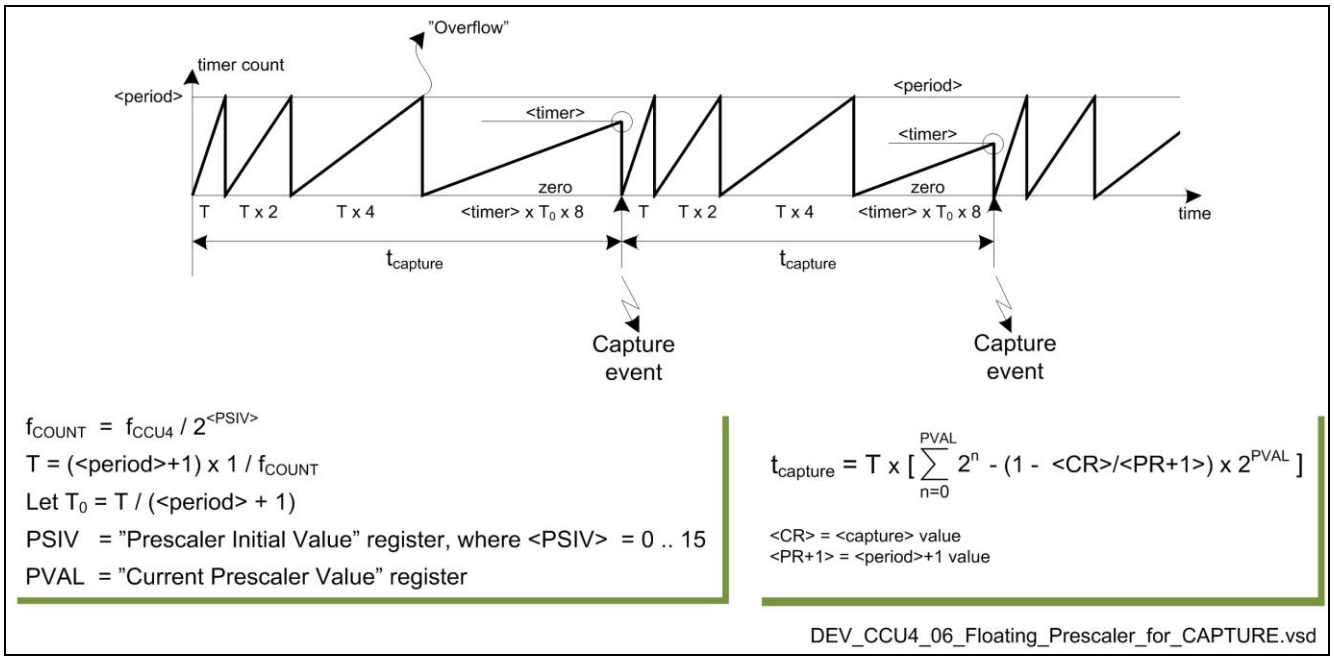


Figure 46 Floating Prescaler For Automatic Dynamic Range Adaption in Time Measurement by Capture

The Floating Prescaler usage with Compare operations

A signal time interval, a count range or PWM periods of arbitrary lengths may be achieved by using the Floating Prescaler.

Period or pulse lengths can be changed coherently via the shadow transfers.

A timer compare event is due if it occurs during a period when the prescaler current value PVAL equals the preset PCMP compare value.

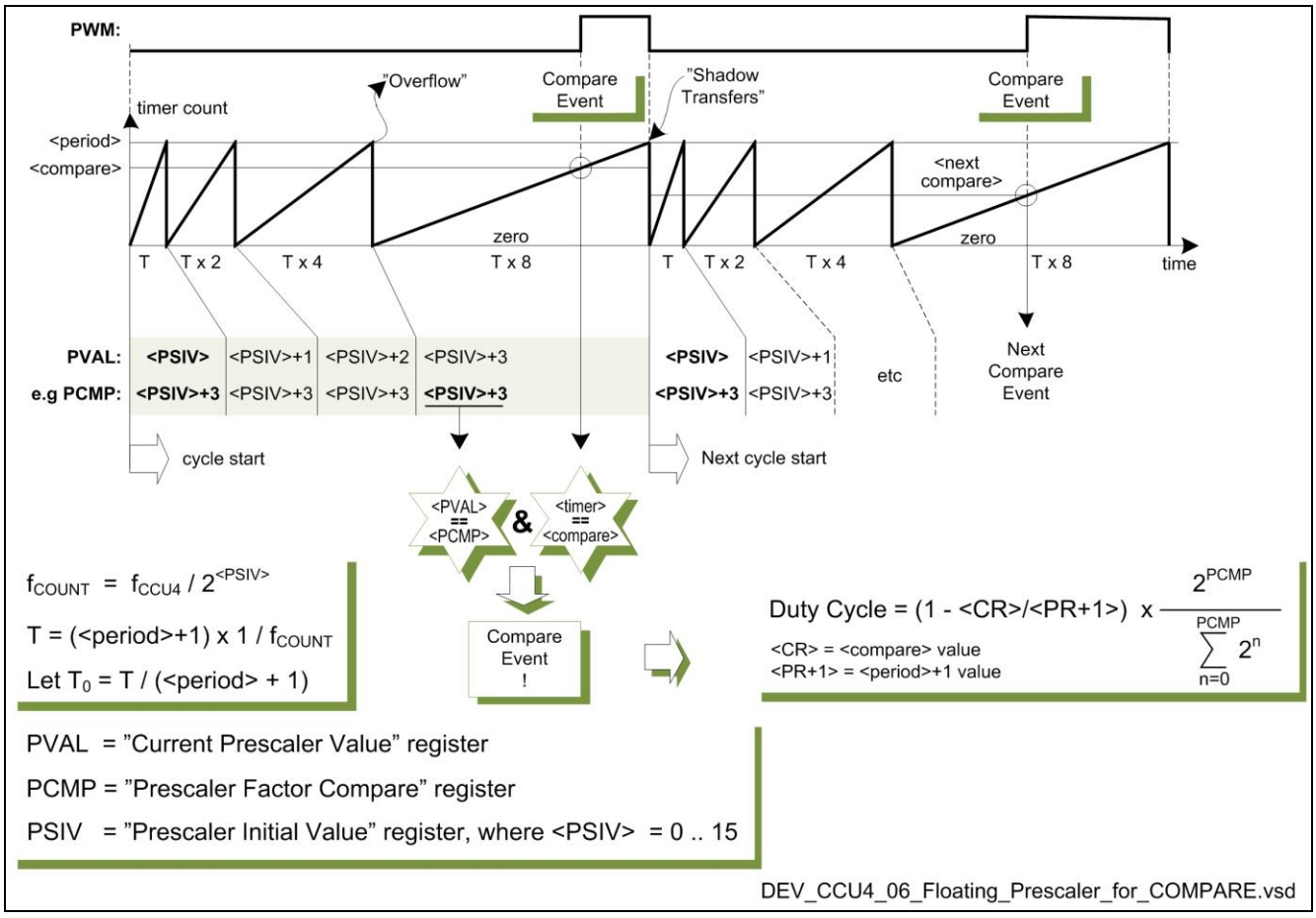


Figure 47 Using Floating Prescaler For Pulse Width Control by Compare (e.g. PWM)

Reload of Floating Prescaler parameters

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

Shadow Transfer of Compare and Floating Prescaler values

The compare values that are targeted for reload operations must be written in the CC4yCRS and CC4yFPCS shadow registers. The respective Transfer Set Enable bits (SySE and SyPSE in GCSS) must be preset at the latest within the clock cycle of Period Match (in Edge Aligned Mode) or Period/One Match (in Center Aligned Mode)

Top-Level Control of Event Requests to/from a Timer with Floating Prescaler

This scenario invokes conditional distribution of event request signals to/from external actors. The Event Request Unit (ERUx) may combine events via the Top-Level Interconnection Matrix, so that a conditional signal flow can be top-level controlled, according to a request-to-action event pattern set by the user.

7.1 Getting Started with CCU4 and Floating Prescaler

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 Prescaler Start-Up

The run bit of the prescaler can be set or cleared by software by writing into the register bits GIDLC.SPRB and GIDLS.CPRB for set and clear respectively. You may also use the run bit of a specific slice by making it responsible for the set/clear. To select the slice to perform this, you must program the GCTRL.PRBC bitfield.

Floating Prescaler Setup Control Registers

Each CCU4xCC4y slice has an individual register bitfield CC4yPSC. PSIV as index to select a clock frequency for each timer in the Timer Clock Division Table. Each register bit CC4yTC. FPE selects normal or floating prescaler mode - and adds a compare register CC4yFPC.PCMP plus its shadow register, if floating mode is selected.

Floating Prescaler Shadow Transfer Control Registers

The reload value in the Shadow Register bitfield CC4yFPCS.PCMPS AND the corresponding Slice Transfer Set Enable bit SyPSE in the GCSS register must be written at the latest within the clock cycle of a Period Match or Period/One Match (regarding edge or center aligned mode) to get a prescaler compare register PCMP update.

Using the Floating Prescaler Associated Control Register Bitfields

- CC4yPSC.PSIV Look-up index for a Prescaler Select Initial Value, used as start value in floating mode.
- CC4yTC.FPE Select FPE =1 for Floating Prescaler Enable (else default mode: Normal, by FPE= 0).
- CC4yPSC.PVAL Current Prescaler Value registers bitfield, read/writable by the user. Updated by HW.
- CC4yPSC.PCMP Floating Prescaler Compare value. The comparison is triggered by the compare match.
- CC4yFPCS.PCMPS Floating Prescaler Compare Shadow registers bitfield.
- GCSS.SyPSE Slice y (y=0-3) Prescaler Shadow transfer Set Enable registers bit.

External Modulation

8 Output Level Modulation on External Events

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

Output Level Modulation Basics

A slice may have its output pin modulated by external events. This function (and the active level of the event source) should be mapped to one of the 3 inputs of the slice in the CC4yCMC and CC4yINS registers. Modulation alternatives can be selected in CC4yTC for either Status Bit Clear or just Output Gating on events.

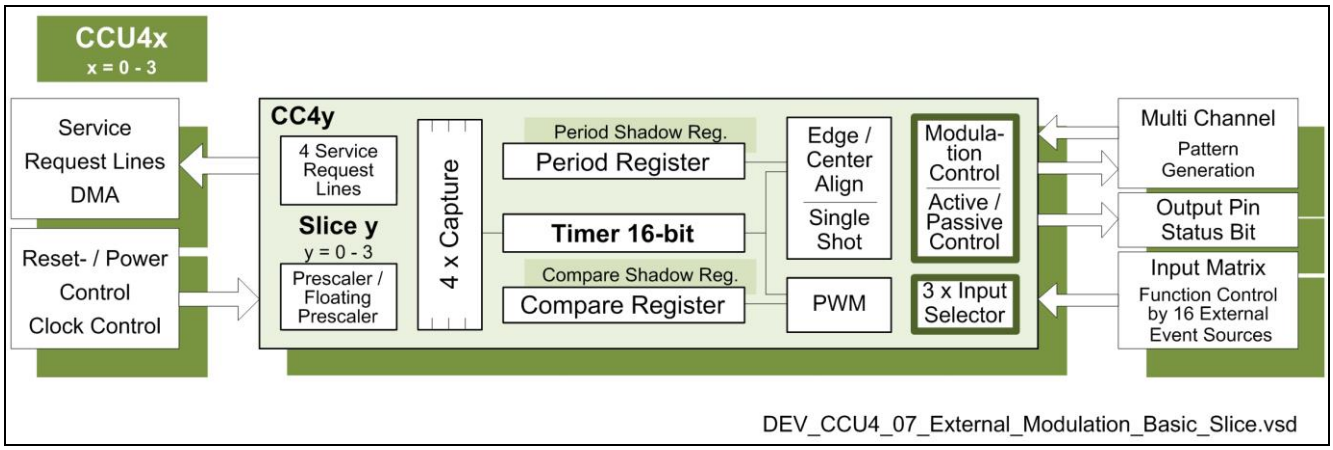


Figure 48 Timer Slice in Status Bit or Output Pin Modulation by External Events Control

Use Cases

Use cases for external modulation can be found in e.g. motor drive and control applications, such as BLDC. The PWM signals that control motor currents by duty cycle are activated and passivized alternately according to the commutation event sequence via an external modulation, which is controlled by commutation logic and sensors.

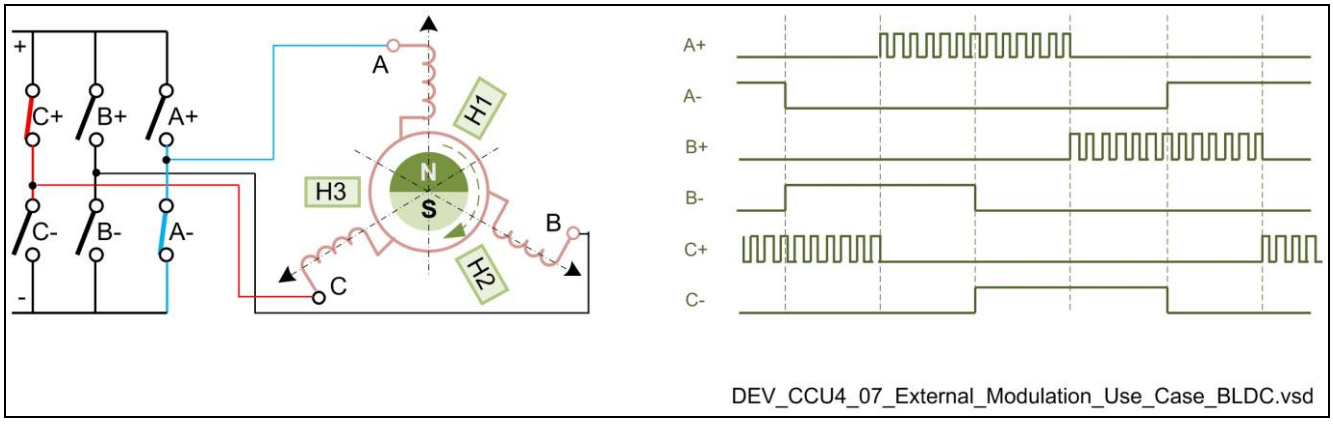


Figure 49 PWM signals in External modulation by BLDC commutation Event Sequences Control

8.1 External Events Control of a Timer Status Bit or Output Pin State

An External Event Control request is a considered edge or level event signal from a peripheral unit or a GPIO. It can be linked to CCU4xCC4y slices input selection stages via a comprehensive matrix. A slice that by any of its 3 events setups detects a considered source-event profile can be Status Bit or Output Pin modulated this way.

External Events Control of Timer Input Functions in general

There are 11 Timer Input Functions (as e.g. Modulate Output), controllable by external events via 3 selectable input lines with configurable source-event profile conditions to the Timer Slices CC4y (y=0-3) of a CCU4x unit (x=0-3) for Start, Stop, Capture0-3, Gate, Up/Down, Load, Count, Bit Override, Trap and Modulate Output control.

Selection of External Events Control Sources and Input Functions

The Modulate Output function can be linked to external trigger requests from sources such as: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU. Pin Connections are given by the Top-Level Interconnect matrix and the CC4yINS[P:A] Input Select vector and Function Select by the CC4yCMC register.

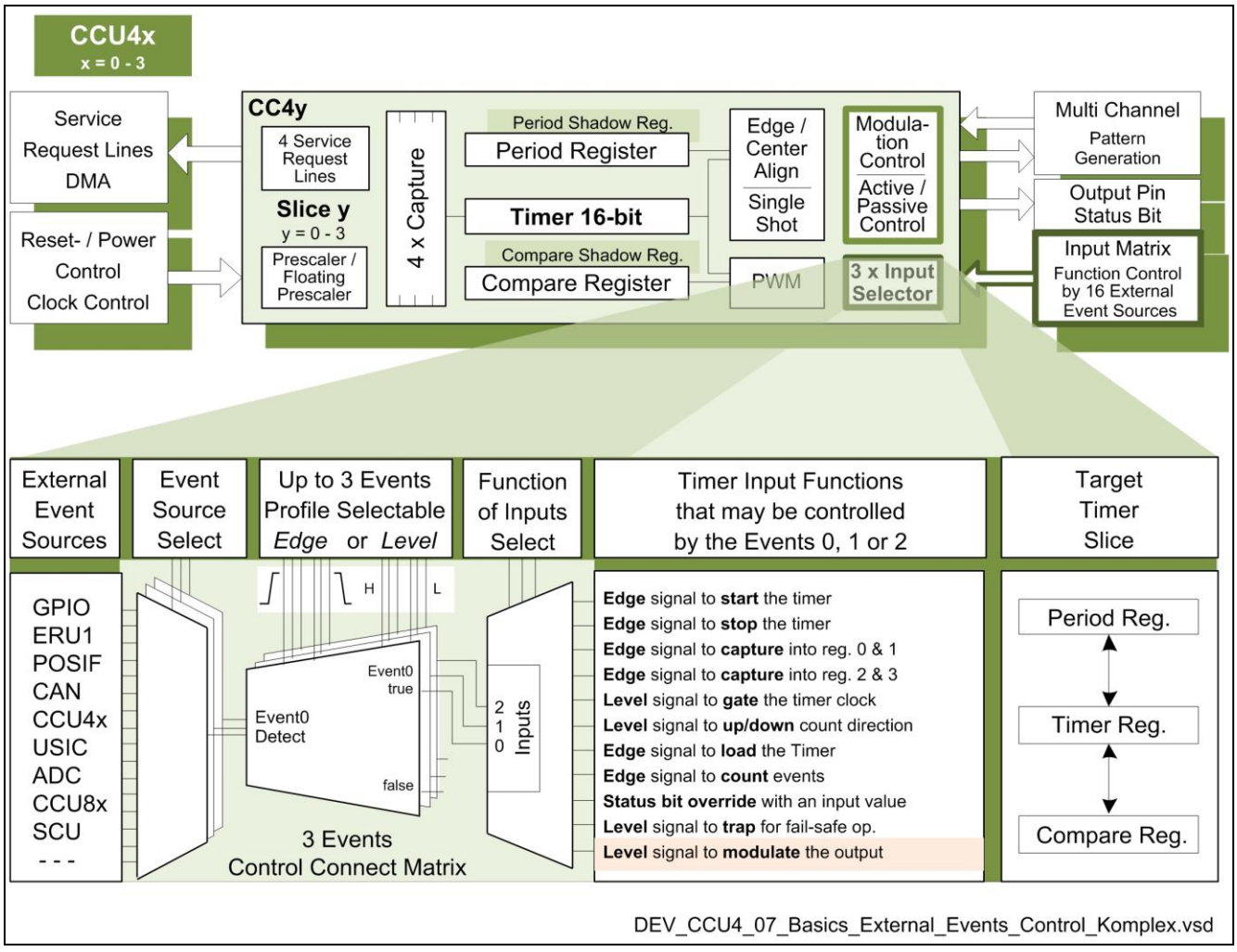


Figure 50 External Events Control of a Timer Slice Input Functions

8.2 Output Level Modulation Principle

A timer slice may have its status bit or output pin modulated by an external event signal. The modulation alternatives can be setup for either Status Bit Clear (including influence on the Output Pin) or Output Gating (i.e. without influence on the Status Bit). The Output PASSIVE/ACTIVE LEVEL rules are still due.

Output Pin PASSIVE / ACTIVE Level Control

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

Synchronized or Unsynchronized Output Level Modulation Schemes

By “synchronized” should be understood that the output pin signal keeps its pulse width until it returns into passive state (i.e. there is no shortening of any active pulse length due to the External Modulation signal) So the entering into inactive state can be synchronized with the PWM period instead of the External Modulation period.

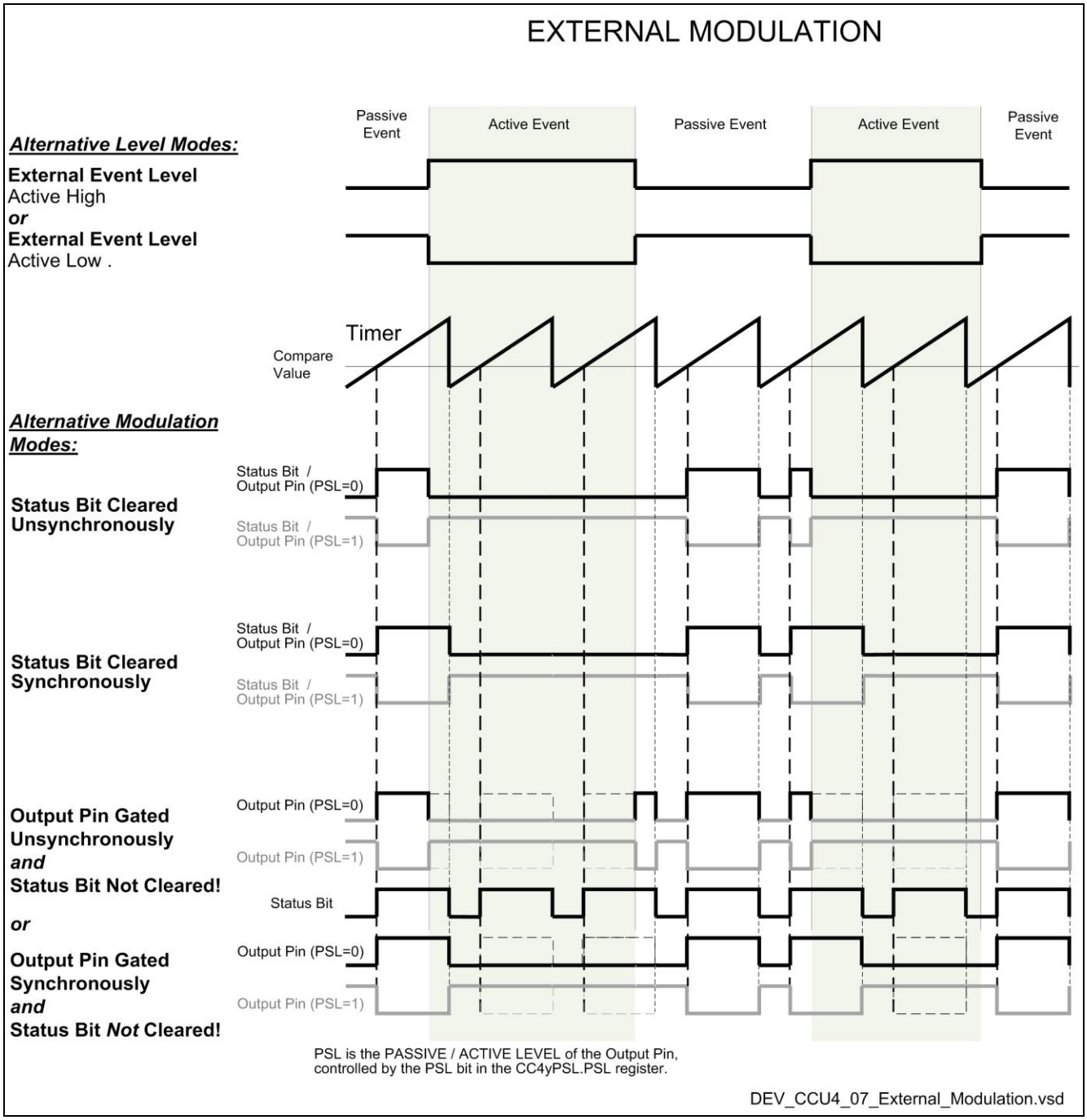


Figure 51 External Modulation by Events Control

8.3 Getting Started with CCU4 and External Modulation

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block

- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 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.

Using the External Modulation Control Register Bitfields

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

Pseudo Code for the External Modulation 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 |= MOS_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Initialize External Modulation
// step 2
CC4yINS &= MOS_SOURCE_AND_EVENT_MODE_BITFIELDS_MASK; //Focus on Source1 events
CC4yINS |= MOS_SOURCE_AND_EVENT_MODE_BITFIELDS_PATTERN;  //Paste setup pattern
// step 3
EMT = TC_STATUS_BIT_CLEAR_OR_OUTPUT_GATING_ONLY; // Set or Clear this bit
// step 4
EMS = TC_UNSYNCHRONOUS_OR_SYNCHRONOUS; // Set or Clear this bit
// Manipulation of the Output Pin PASSIVE/ACTIVE LEVEL control bit (PSL) if needed
// CC4yPSL = ACTUAL_PASSIVE_LEVEL // Use for LOW: 0x00000000 or HIGH: 0x00000001
// (There are no other bits but the PSL bit (CC4yPSL.PSL) in this register!)
```

Setting Up External Modulation

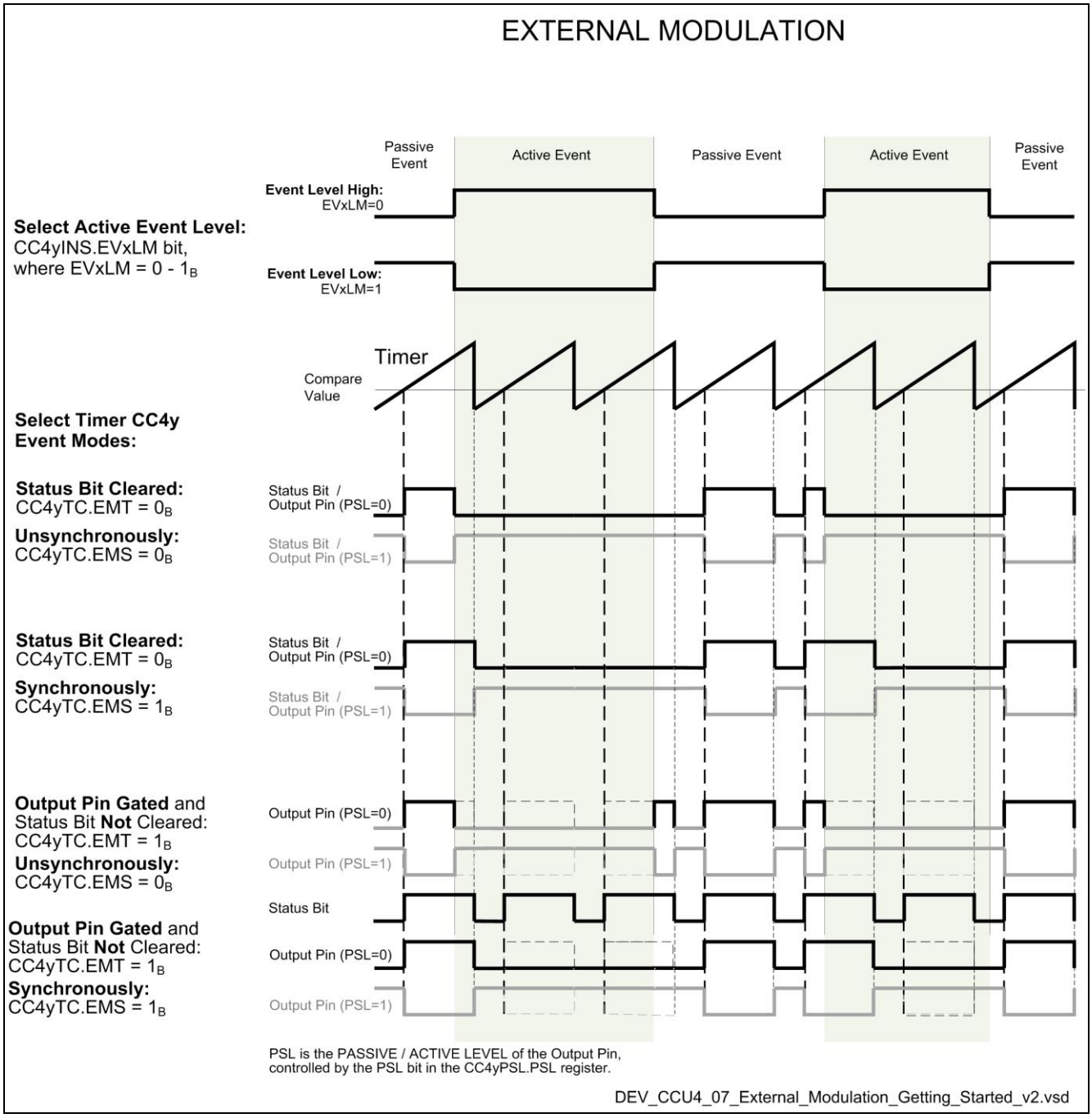


Figure 52 Setting Up External Modulation Events Control Sources and Operation Modes

8.4 Application External Modulation with Current Limitation by Output State Override

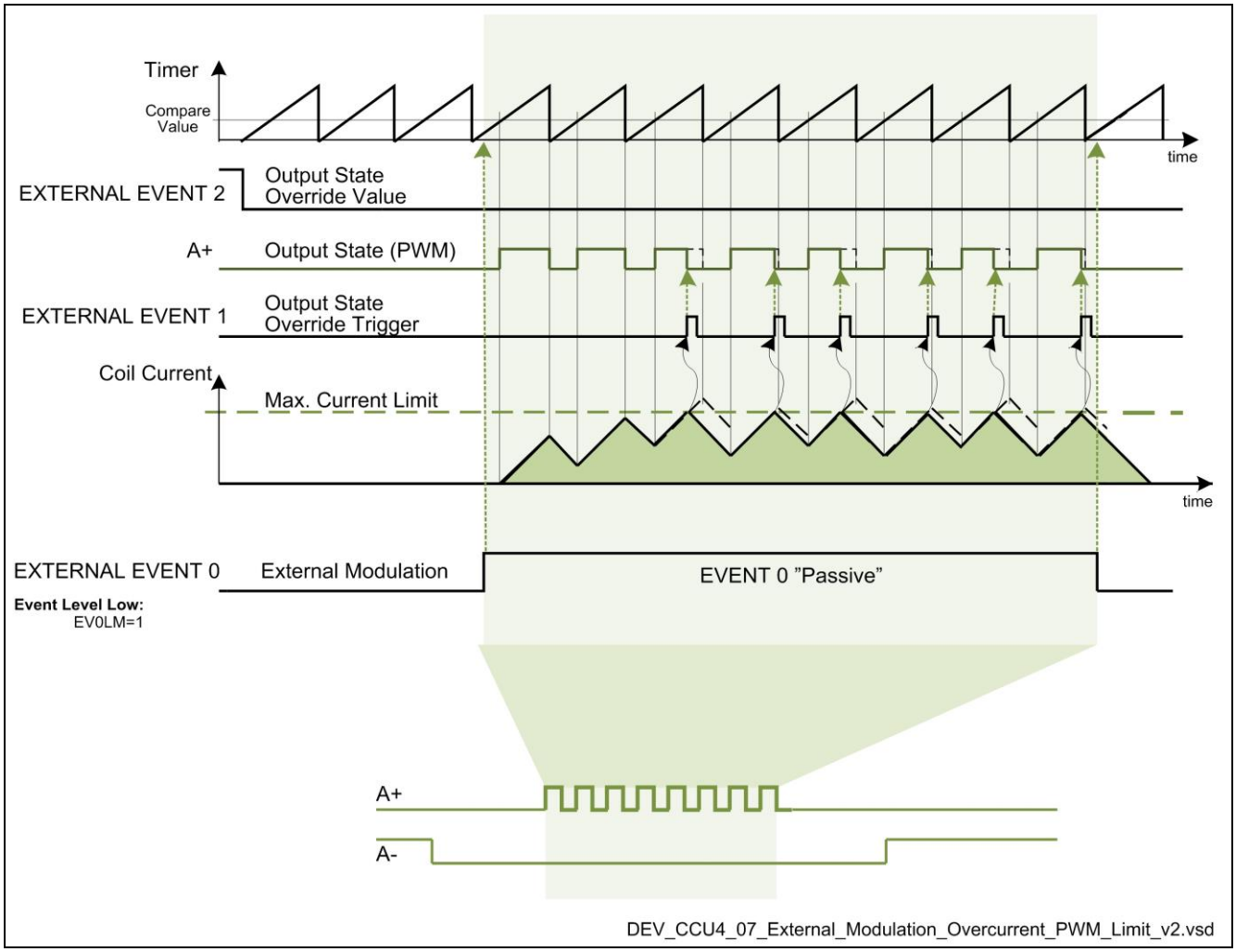


Figure 53 Application External Modulation with Current Limitation by Output State Override

Output State Override

9 Output State Override on External Events

It is possible to override a slice's output state with the value of an external source (or its complement value) on the edge event of another external source. The source properties should be mapped to the CC4yINS register and the override input function by the CC4yCMC register. A use case is for example, output timing control by monitoring.

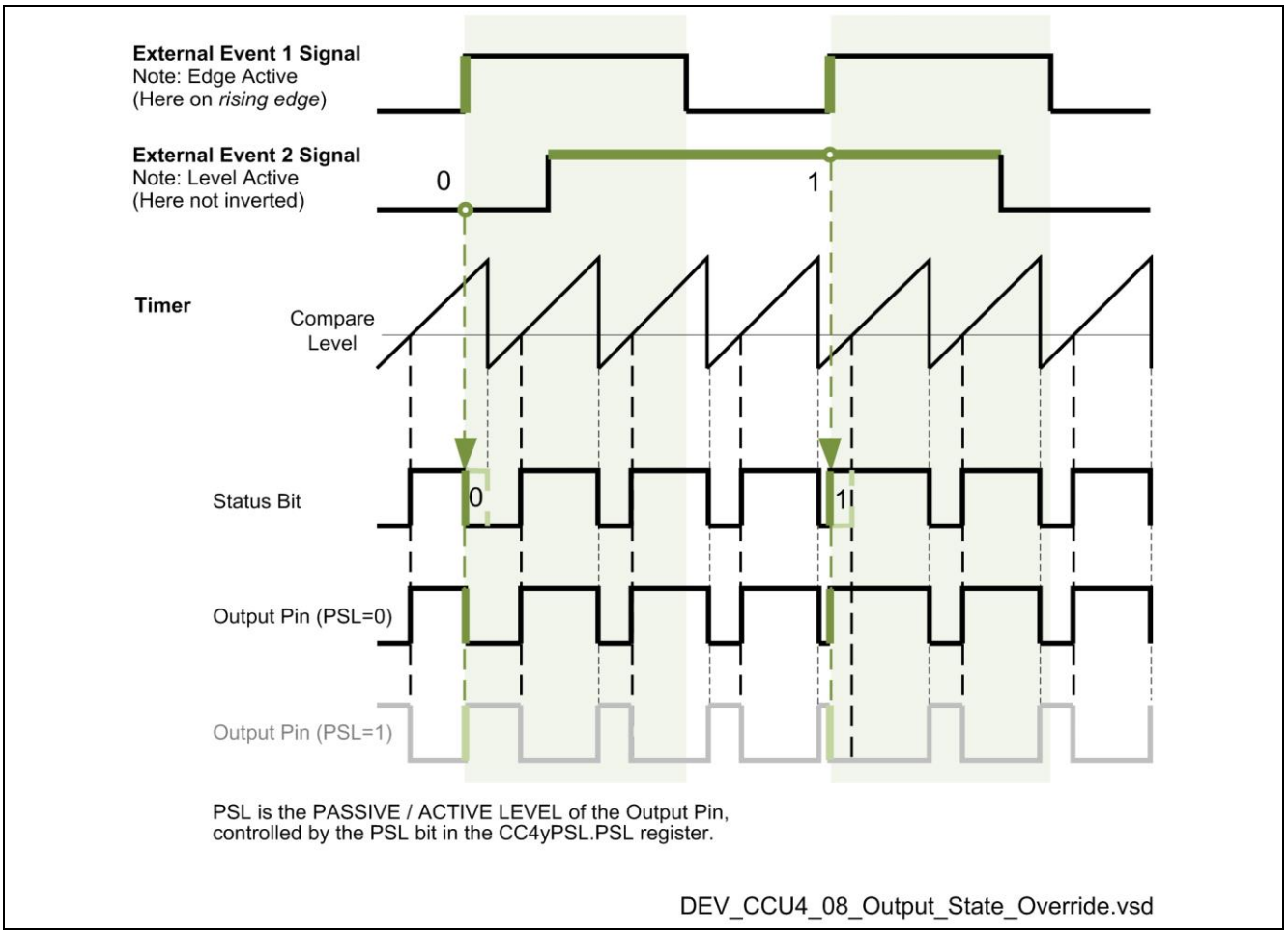


Figure 54 Output State Override

Use Cases

Output State Override is useful for emergency protection (Hardware TRAP) or hysteretic control (simple on/off via external overcurrent comparator for primitive control loop). The ADC boundary flag can be one of the sources for the comparator. For example, the PWM timing may be adjusted by Output State Override with regard to different current boundaries.

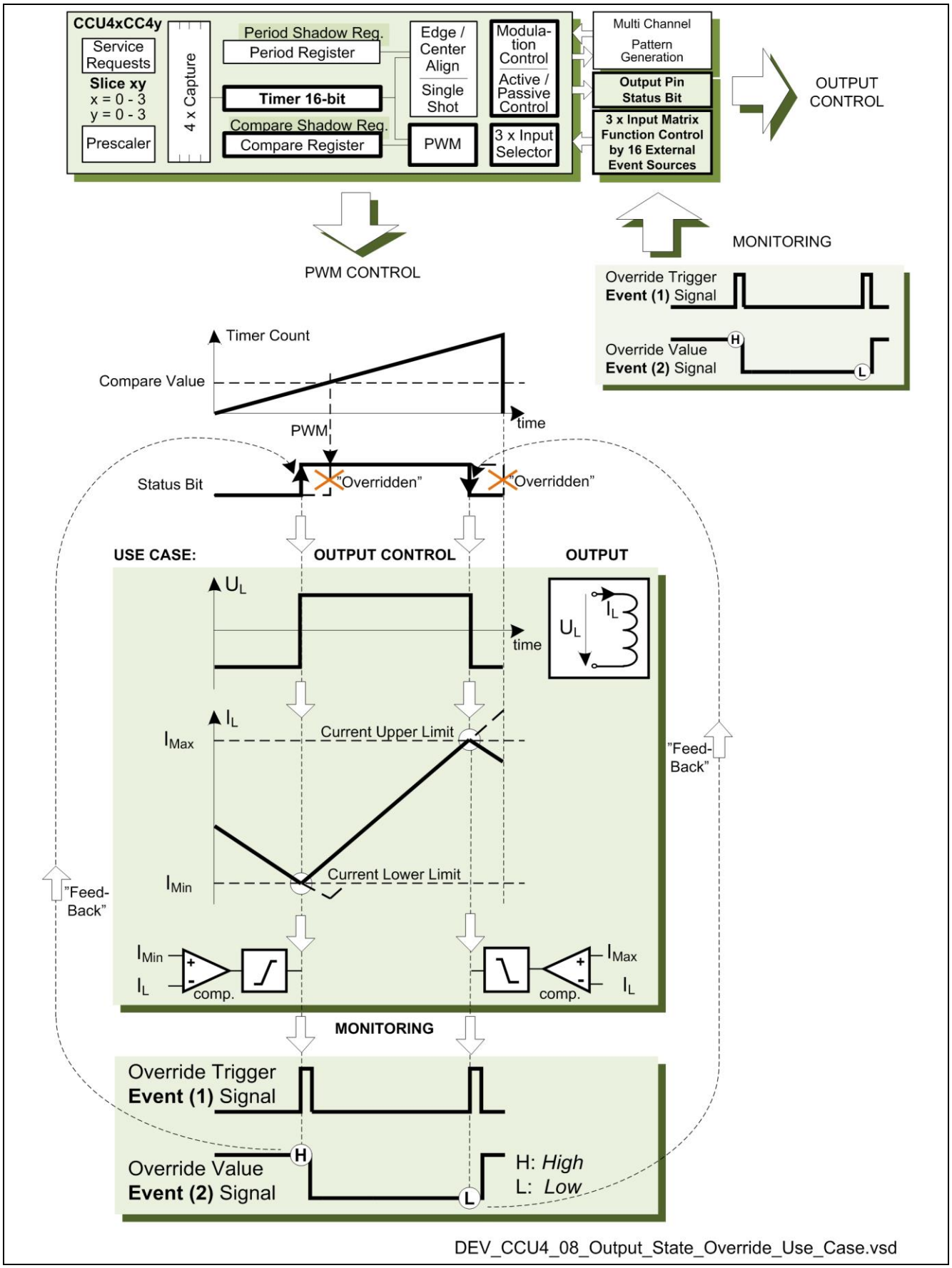

CCU4xCC4y
Service Requests
Slice xy
x = 0 - 3
y = 0 - 3
Prescaler
4 x Capture
Period Shadow Reg.
Period Register
Timer 16-bit
Compare Shadow Reg.
Compare Register
Edge / Center Align
Single Shot
PWM
Modula-tion Control
Active / Passive Control
3 x Input Selector
Multi Channel Pattern Generation
Output Pin Status Bit
3 x Input Matrix Function Control by 16 External Event Sources
OUTPUT CONTROL
PWM CONTROL
MONITORING
Override Trigger Event (1) Signal
Override Value Event (2) Signal
Timer Count
Compare Value
time
PWM
Status Bit
"Overridden"
"Overridden"
USE CASE:
OUTPUT CONTROL
OUTPUT
UL
IL
Current Upper Limit
IMax
IMin
Current Lower Limit
"Feed-Back"
"Feed-Back"
comp.
comp.
MONITORING
Override Trigger Event (1) Signal
Override Value Event (2) Signal
H: High
L: Low
DEV_CCU4_08_Output_State_Override_Use_Case.vsd

Figure 55 Use case – Current control and Protection by Monitoring using Output State Override

9.1 External Events Control of a Timer Output State

An External Event Control request is a considered edge or level event signal from a peripheral unit or a GPIO. It can be linked to CCU4xCC4y slices input selection stages via a comprehensive matrix. A slice that by any of its 3 events setups detects a considered source-event profile can be Output State Overridden remotely-like this way!

External Events Control of Timer Input Functions in general

There are 11 Timer Input Functions (as e.g. Bit Override), controllable by external events via 3 selectable input lines with configurable source-event profile conditions to the Timer Slices CC4y (y=0-3) of a CCU4x unit (x=0-3) for Start, Stop, Capture0-3, Gate, Up/Down, Load, Count, Status Bit Override, Trap and Modulate Output control.

Selection of External Events Control Sources and Input Functions

The Output State Override function can be linked to external trigger requests from sources such as: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU. Pin Connections are given by the Top-Level Interconnect matrix and CC4yINS[P:A] Input Select vector and Function Select by the CC4yCMC register.

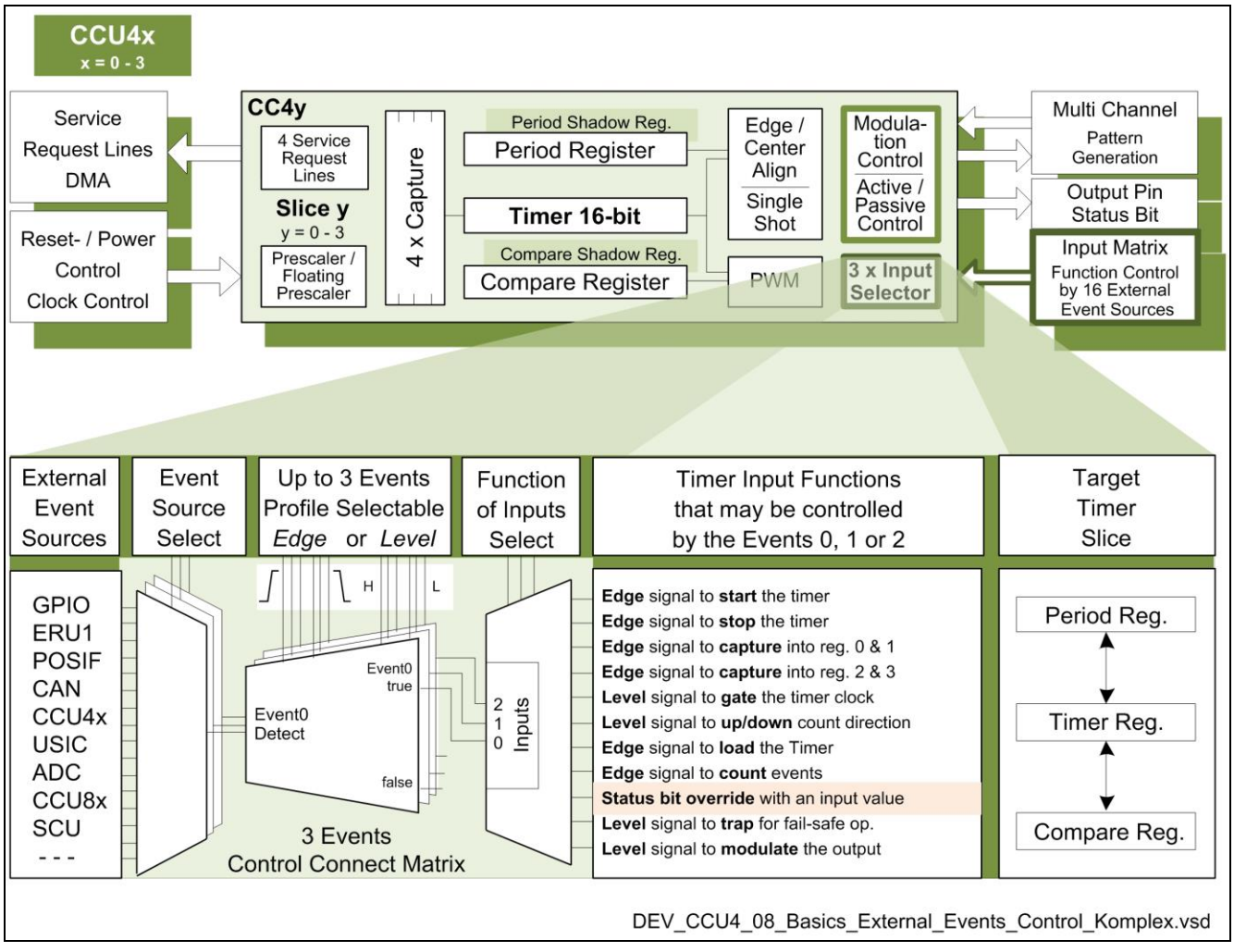


Figure 56 External Events Control of a Timer Slice Input Functions

9.2 Getting Started with CCU4 and Output State Override

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 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.

Using the Output State Override Control Register Bitfields

- CC4yCMC.OFS Input Function Selection bit to enable the Output State Override Function
- CC4yINS.<Event1 bitfields> Event1 Source EV1IS (CC4yINS[P:A]) and Edge Mode EV1EM bitfields
- CC4yINS.<Event2 bitfields> Event2 Source EV2IS (CC4yINS[P:A]) and Level Mode EV2LM bitfields
- CC4yTC.<Mode bitfields> Timer Mode selections e.g. Modes/Extended modes, Counting Schemes etc.

Note: The Output State Override Function is Edge Sensitive.

Note: The Carries of the Override State Value is Level Sensitive.

Note: Selection of Event Numbers are Default!!! I.e. Event 1 is Trigger Edge signal, so the Output State Override time point is controlled by Source 1. Event 2 is Carrier of Level signal, so the Output State Override value is controlled by the level of Source 2.

Pseudo Code for the Output State Override 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 |= OFS_FUNCTION_AND_EVENTS_BITFIELDS // Enable the Output State Override
// step 2
CC4yINS &= OFS_SOURCE1_AND_EVENT1_MODES_BITFIELDS_MASK; //Focus on Source1 events
CC4yINS |= OFS_SOURCE1_AND_EVENT1_MODES_BITFIELDS_PATTERN;  //Paste setup pattern
// step 3
CC4yINS &= OFS_SOURCE2_AND_EVENT2_MODES_BITFIELDS_MASK; //Focus on Source1 events
CC4yINS |= OFS_SOURCE2_AND_EVENT2_MODES_BITFIELDS_PATTERN;  //Paste setup pattern
```

```
// step 2 & 3 could naturally be merged into one operation sequence, as follows:
CC4yINS &= OFS_SOURCES_AND_EVENTS_MODES_BITFIELDS_MASK; //Focus on all bitfields
CC4yINS |= OFS_SOURCES_AND_EVENTS_MODES_BITFIELDS_PATTERN; //Paste both setups!

// Manipulation of the Output Pin PASSIVE/ACTIVE LEVEL control bit (PSL) if needed
// CC4yPSL = ACTUAL_PASSIVE_LEVEL // Use for LOW: 0x00000000 or HIGH: 0x00000001
// (There are no other bits but the PSL bit (CC4yPSL.PSL) in this register!)
```

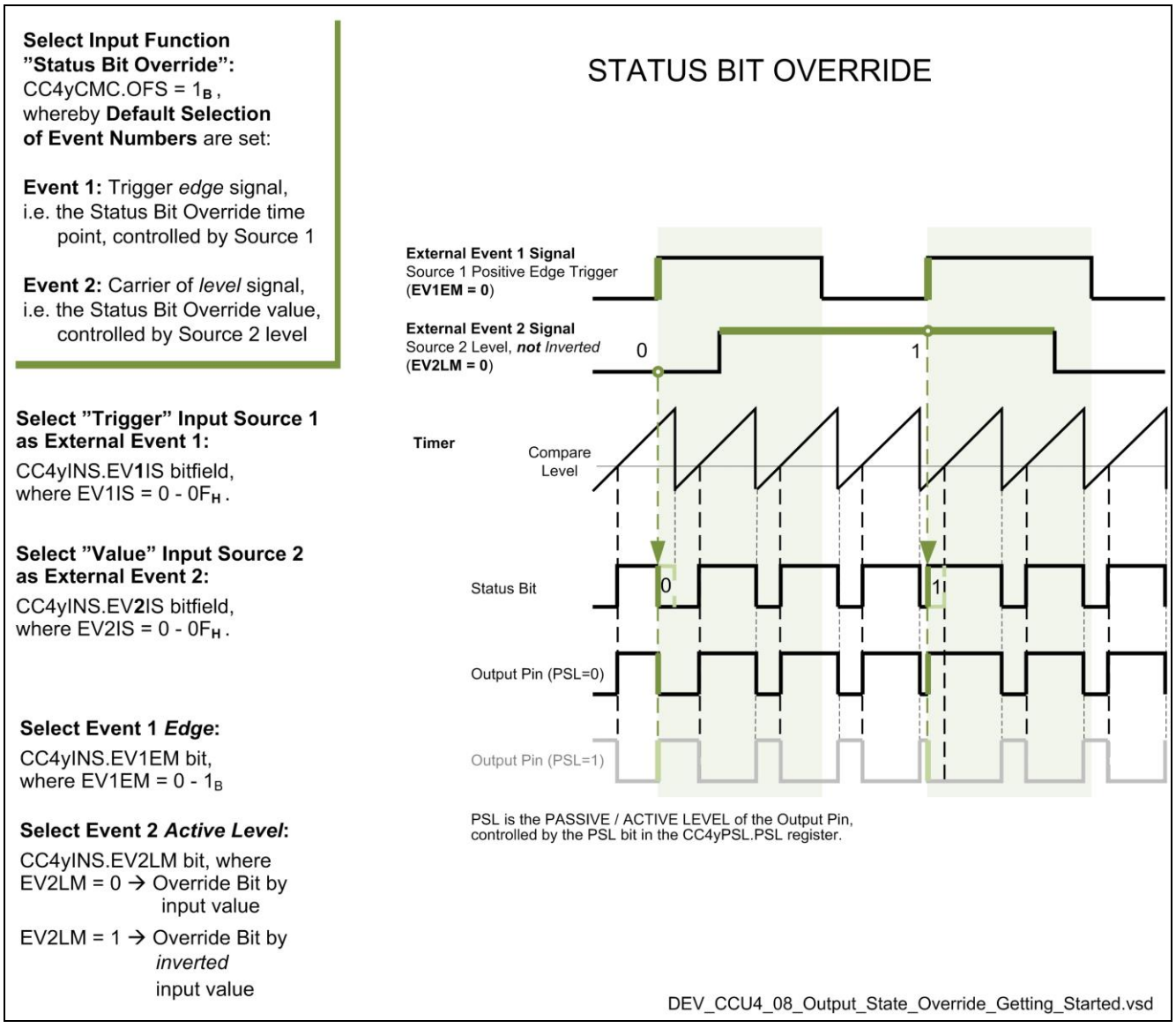


Figure 57 Setting Output State Override Parameter (Control Registers)

9.3 Example Application for CCU4 and Output State Override

Principle of Peak and Hold with Current Limitation by Output State Override

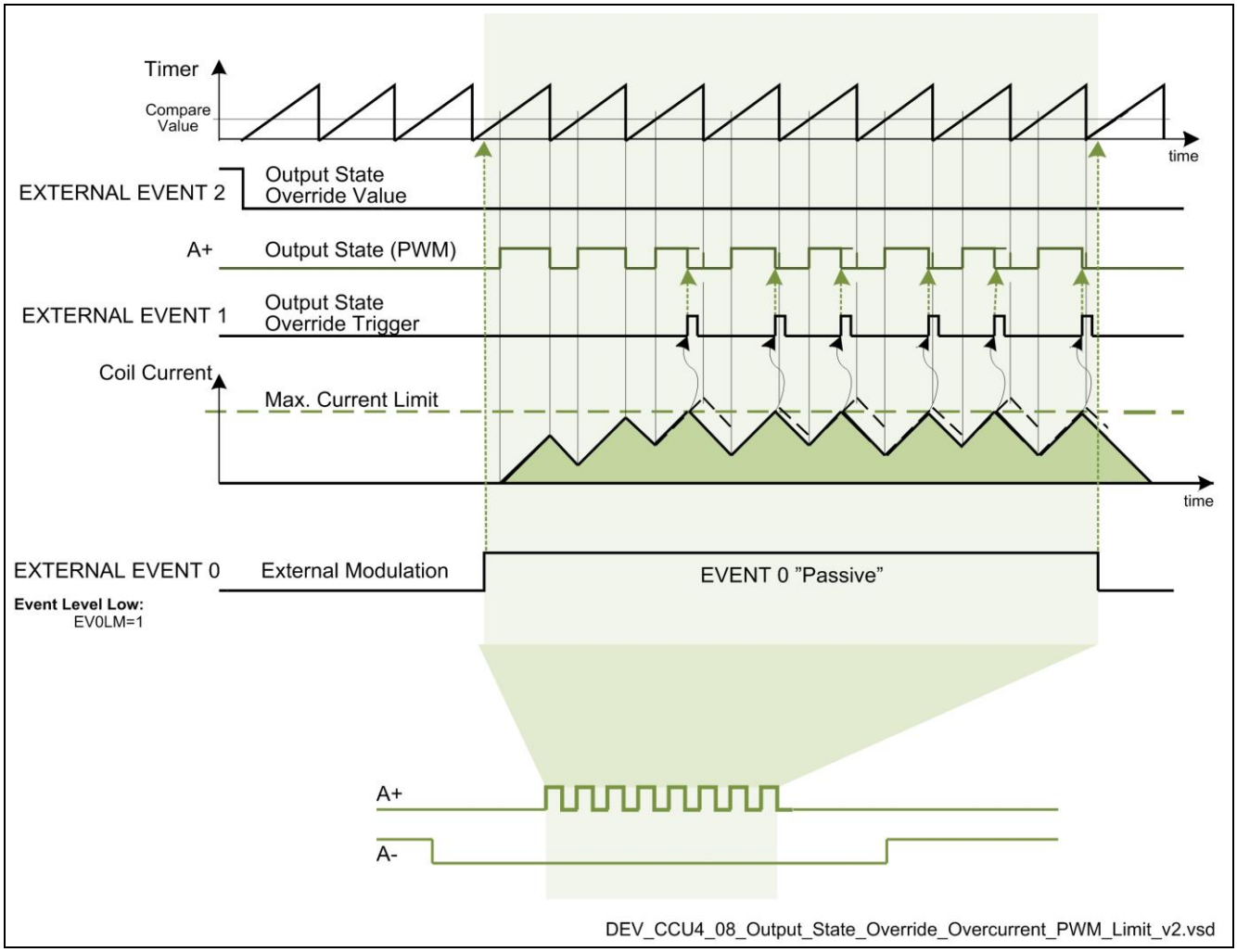


Figure 58 Principle of Peak & Hold with Current Limitation by Output State Override

External Load

10 Timer Reload on External Events

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

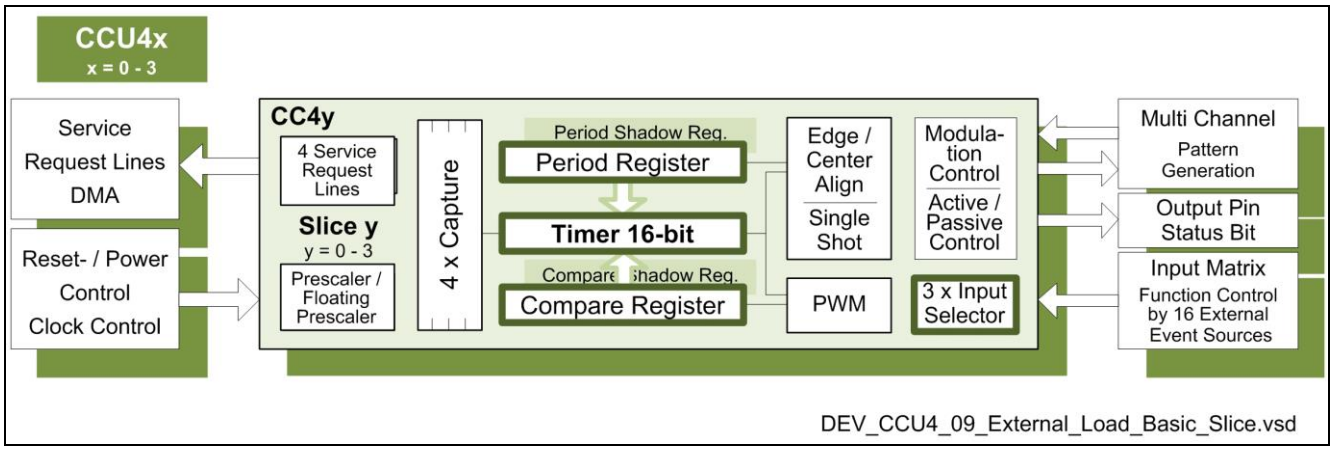


Figure 59 CCU4 Basic Slice and Timer Load on External Events

Timer Load Basics

An external source may trigger a Timer Reload transfer from either the compare register CRy (if counting up) or from the period register PRy (if counting down). The count direction bit CC4yTCST.CDIR can be set by hardware by another source. The input source(s) and function(s) should be mapped to the CC4yINS & CC4yCMC registers.

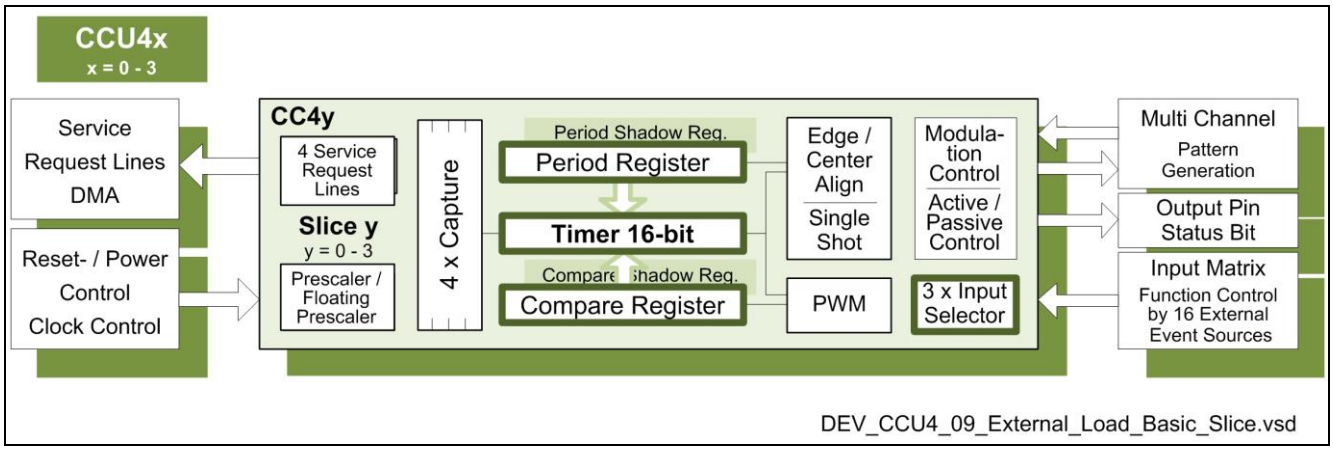


Figure 60 Timer Load Selection Control

Timer Load by External Events Control

This scenario regards how to link the Timer Load input function to external trigger requests from sources such as: GPIO, ERU, POSIF, CAN, CCU4x, USIC, ADC, CCU8x or SCU. Pin Connections are given by the Top-Level Interconnect and the CC4yINS[P:A] Input Select and Load function by the CC4yCMC.LDS.

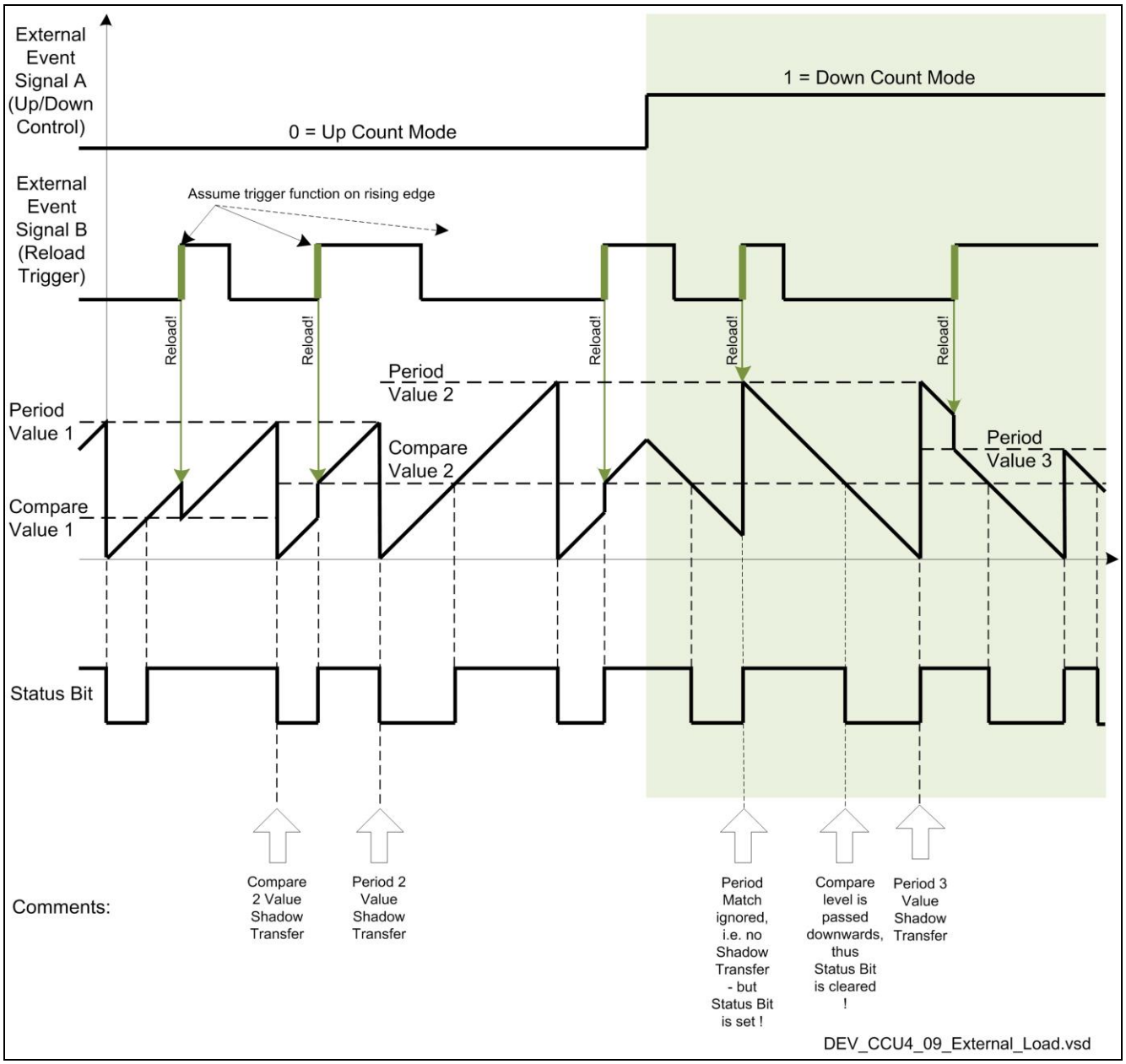


Figure 61 Timer Reload from Compare Register or Period Register on External Events

10.1 Getting Started with CCU4 and External Load

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 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 Status bit CC4yTCST.CDIR controls the counting direction.

Using the External Load Control Registers Bitfields

- CC4yCMC.LDS Map the Load Function Selector to an Event Input Line Number x=0-2
- CC4yINS.<Event-x bitfields> Event-x Source EVxIS (CC4yINS[P:A]) and Edge Mode EVxEM bitfields

Using the External Up/Down Control Registers Bitfields

- CC4yCMC.UDS Map the Up/Down Function Selector to an Event Input Line Number x=0-2
- CC4yINS.<Event-x bitfields> Event-x Source EVxIS (CC4yINS[P:A]) and Level Mode EVxLM bitfields

Note: The Up/Down Counting Direction control works indirectly via the unwritable CC4yTCST.CDIR status bit

Note: The External Load function is Edge Sensitive.

Note: The External Up/Down Direction Control function is Level Sensitive.

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 Event Number 'x' for the LDS Input Function is always x == <LDS>-1 (when <LDS> != 0).

Pseudo Code for the External Load 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 |= LDS_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Initialize Load and Event Nr
// step 2
CC4yINS &= LDS_SOURCE_AND_EVENT_MODES_BITFIELDS_MASK; //Focus on bitfields
CC4yINS |= LDS_SOURCE_AND_EVENT_MODES_BITFIELDS_PATTERN;  //Paste setup pattern
// step3
CC4yCMC |= UDS_FUNCTION_AND_EVENT_NUMBER_BITFIELD //Initialize UDS and Event Nr
// step 4
CC4yINS &= UDS_SOURCE_AND_EVENT_MODES_BITFIELDS_MASK; // Focus on bitfields
CC4yINS |= UDS_SOURCE_AND_EVENT_MODES_BITFIELDS_PATTERN;  //Paste setup pattern

// step 1 & 3 could naturally be merged into one operation sequence, as follows:
// Make a MASK OPERATIONS first for bitfield focusing - if not first time - else:
CC4yCMC |= LDS_AND_UDS_FUNCTIONS_AND_EVENTS_NRS_BITFIELDS //Init Ext. Load and UDS
```

```
// step 2 & 4 could naturally be merged into one operation sequence, as follows:
CC4yINS &= LDS_UDS_SOURCES_AND_EVENTS_MODES_BITFIELDS_MASK; //Focus on bitfields
CC4yINS |= LDS_UDS_SOURCES_AND_EVENTS_MODES_BITFIELDS_PATTERN; //Paste all setups!
```

Counting Scheme

11 Counting Modes and Control

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

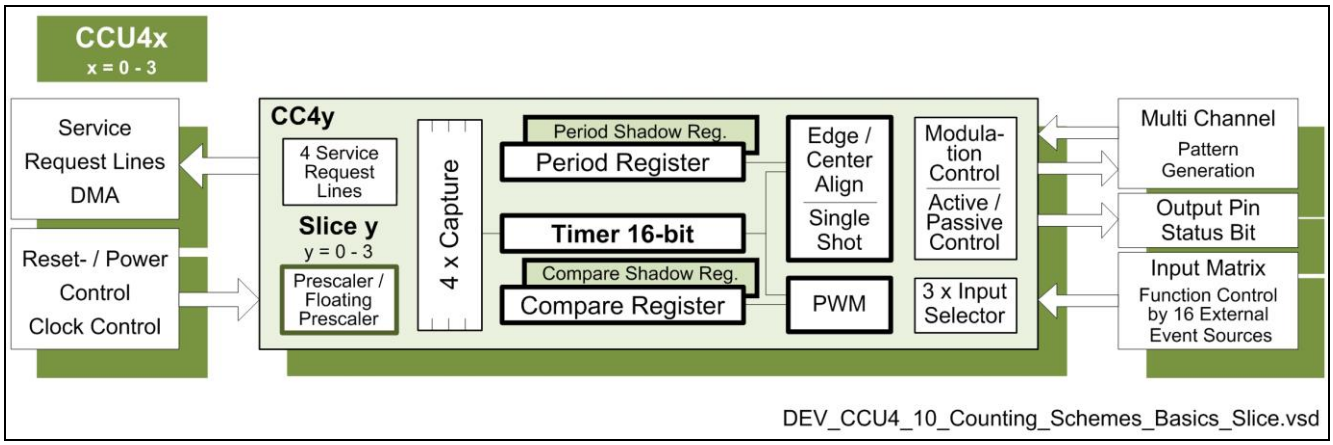


Figure 62 CAPCOM4 Basic Slices

Basic Timer Functions

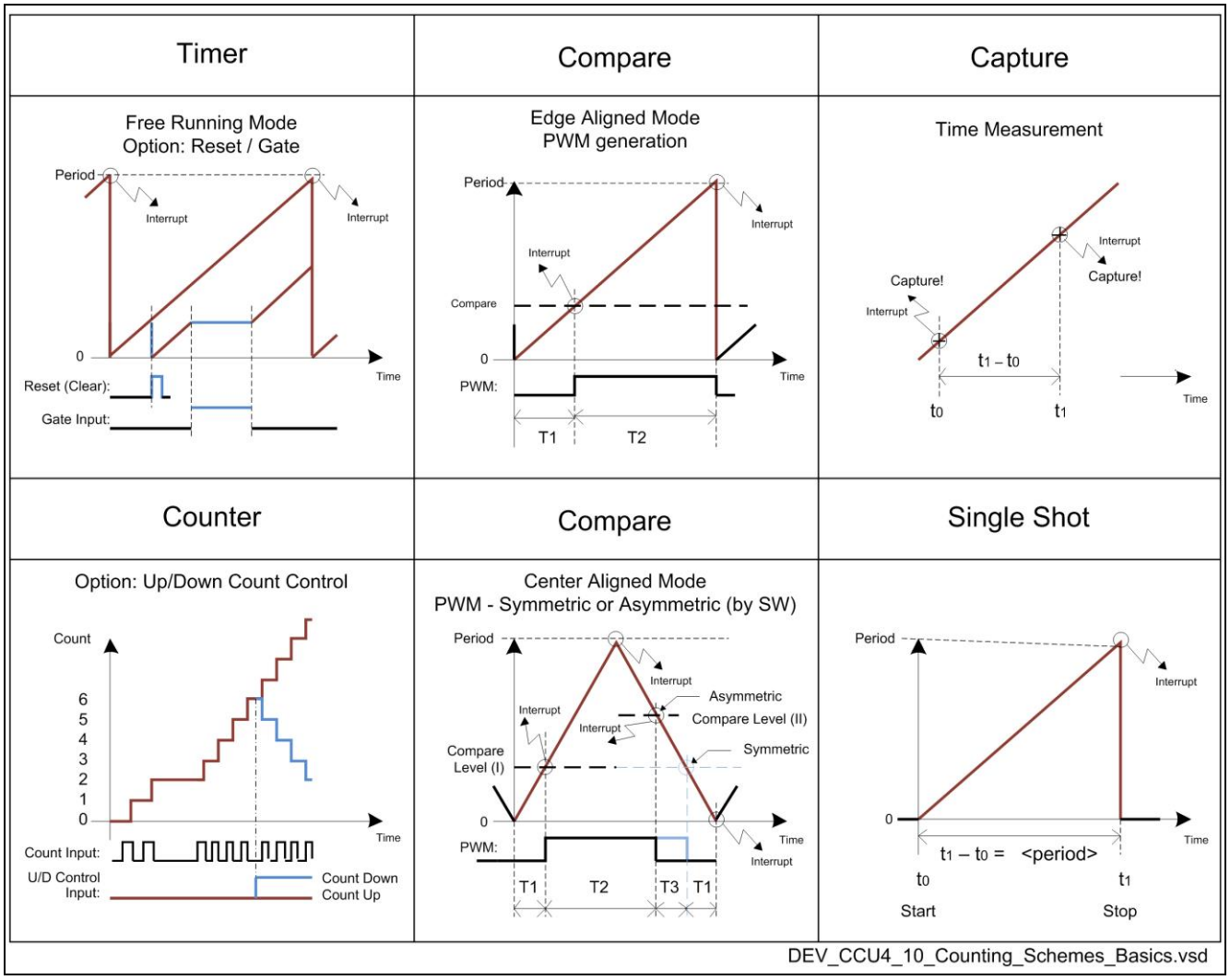


Figure 63 CAPCOM4 Basic Counting Schemes

PWM Range 0 to 100%

When the timer register value is equal or greater than the compare register value, the Status Bit (CCST or even named CCU4xSTy) will be set to one (=1). Otherwise it is set to zero (=0).

The dynamic PWM range can be set to any value from 0 to 100%.

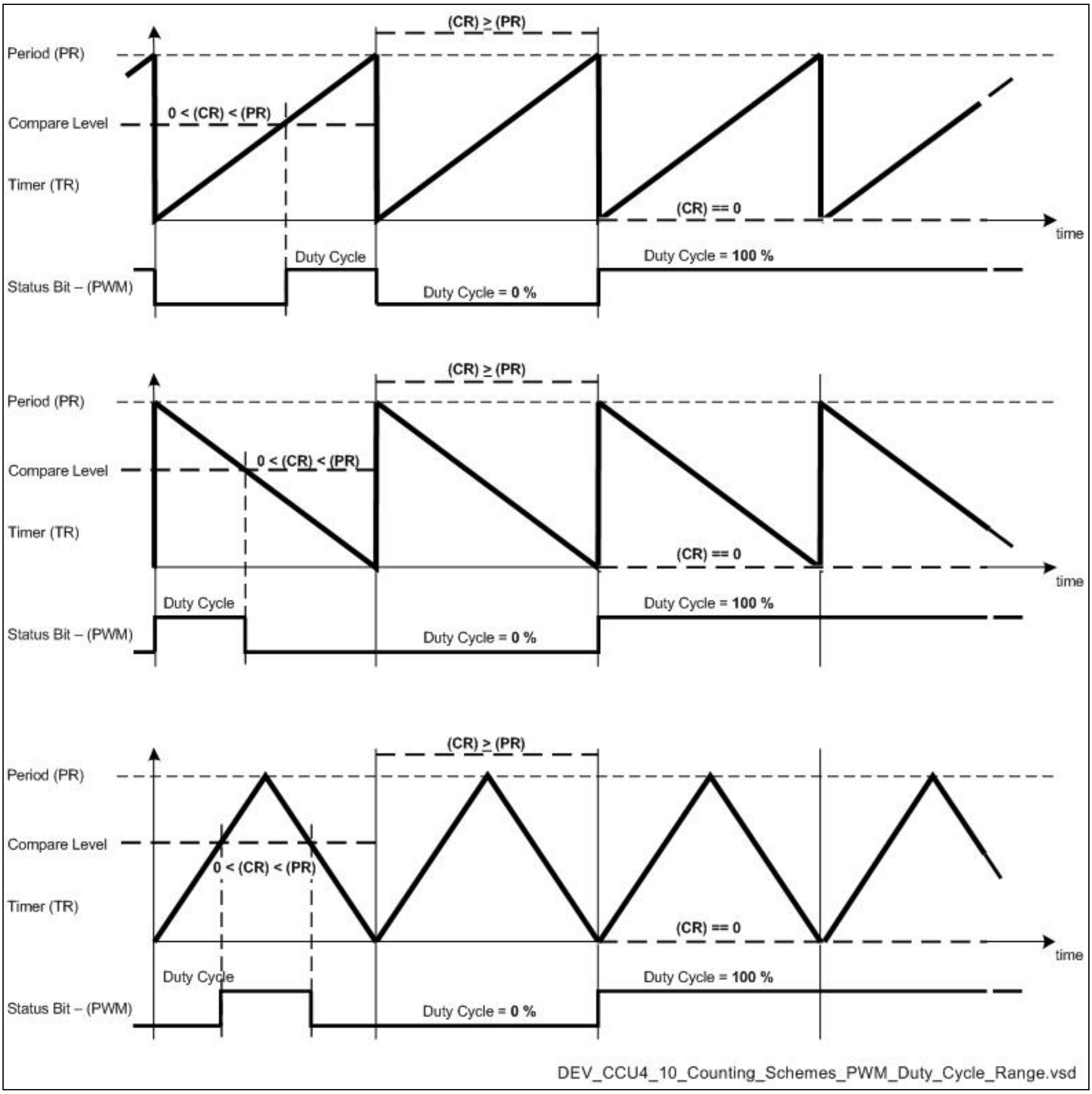


Figure 64 PWM Range in Edge Aligned or Center Aligned Mode

11.1 Counting Schemes in Detail

Counting Rules

One count cycle after the Counting Direction status bit CC4yTCST.CDIR has been changed, the timer counting direction will have been changed at the next count. In Center Aligned Mode this bit will be changed from 0 to 1, one cycle after a Period-Match state is entered, and from 1 to 0, one count cycle after a One-Match is entered.

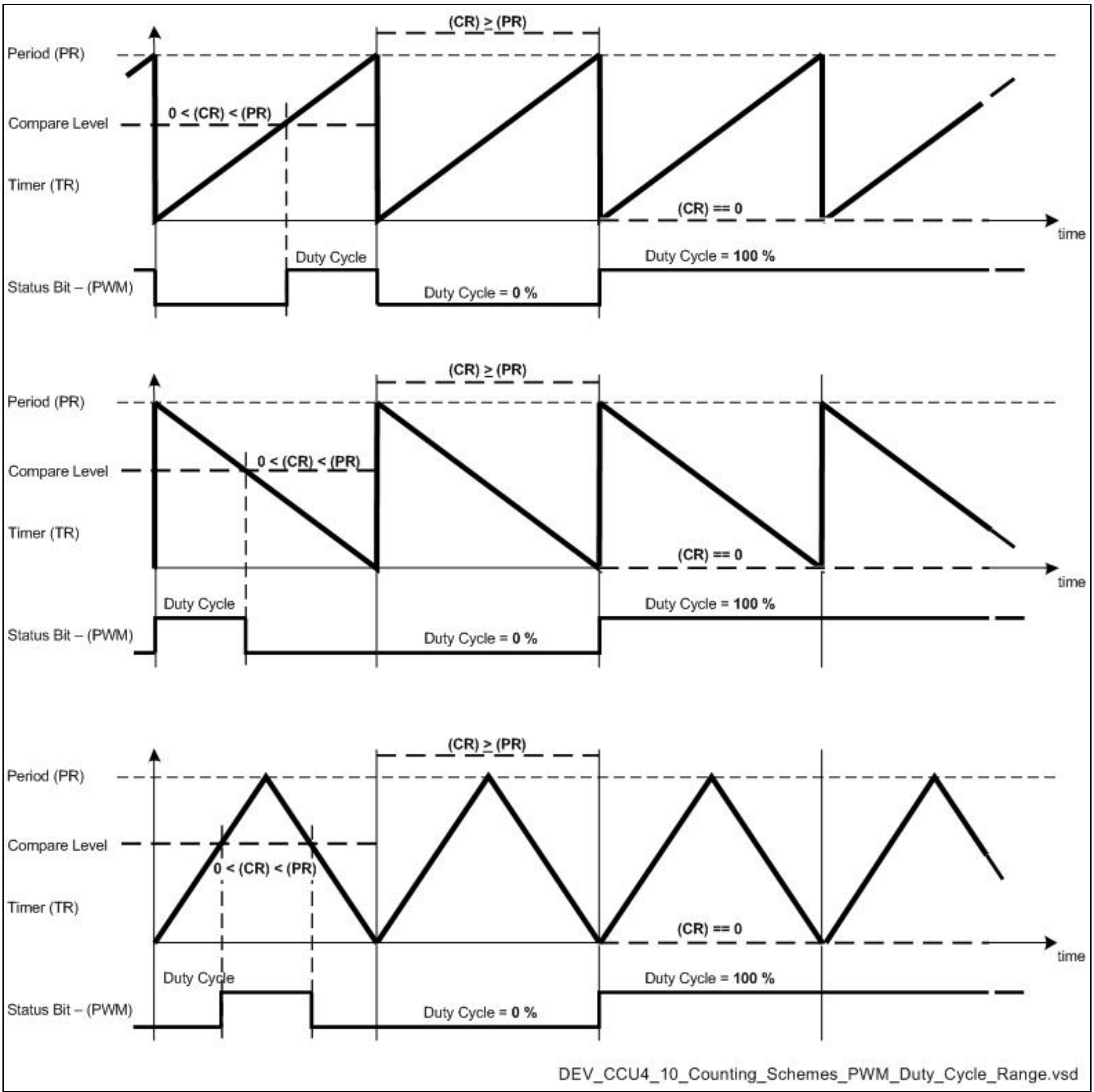


Figure 65 PWM Range in Edge Aligned or Center Aligned Mode

11.2 Reload Schemes in Detail

Reload by Shadow Transfer Rules

Reload of registers by Shadow Transfers from associated shadow registers will occur according to the following:

- In the next clock cycle after a Period Match while counting up
- In the next clock cycle after an One Match while counting down
- Immediately if the timer is stopped and a transfer request was triggered

Note: This implies that it is still due time to Set Transfer Enable requests as late as on a Period Match event, by using a Period Match Interrupt Service with a shorter latency time than the last clock cycle before the Shadow Transfer occurs. Normally the Compare Event Interrupts are used for this purpose. However, it shows that even a compare register value equal to the period register value is accurate for Transfer Enable request settings.

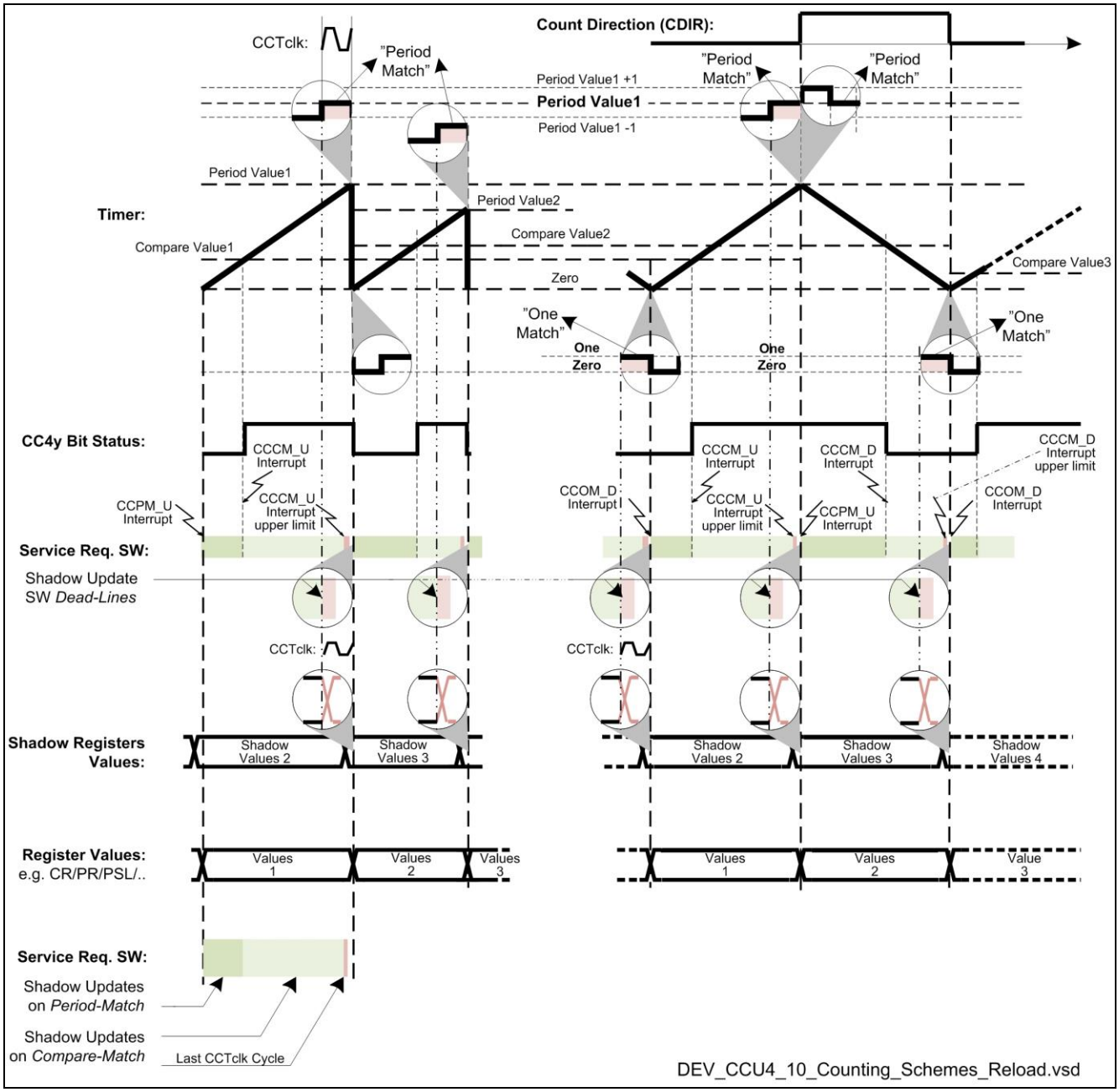


Figure 66 Reload Schemes via Shadow Register (Shadow Transfer Timing)

Shadow Transfer for Coherent Signal Pattern Update

All CAPCOM4 timers, in any slice configuration, are assured coherent update by hardware 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-Match or One-Match.

The Global Shadow Transfer Set Enable Register

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

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, for example SySE in GCSS, must be preset at the latest within the clock cycle of Period Match (in Edge Aligned Mode) or Period/One Match (in Center Aligned Mode)

Compound Shadow Transfers

Beside the Compare (CR) values, there are also the timer Period register (PR) and the PWM Active/Passive control bit (PSL) that are updated simultaneously on the SySE flag. Dithering or Floating Prescaler values may get simultaneous update by the SyDSE and SyPSE request flags.

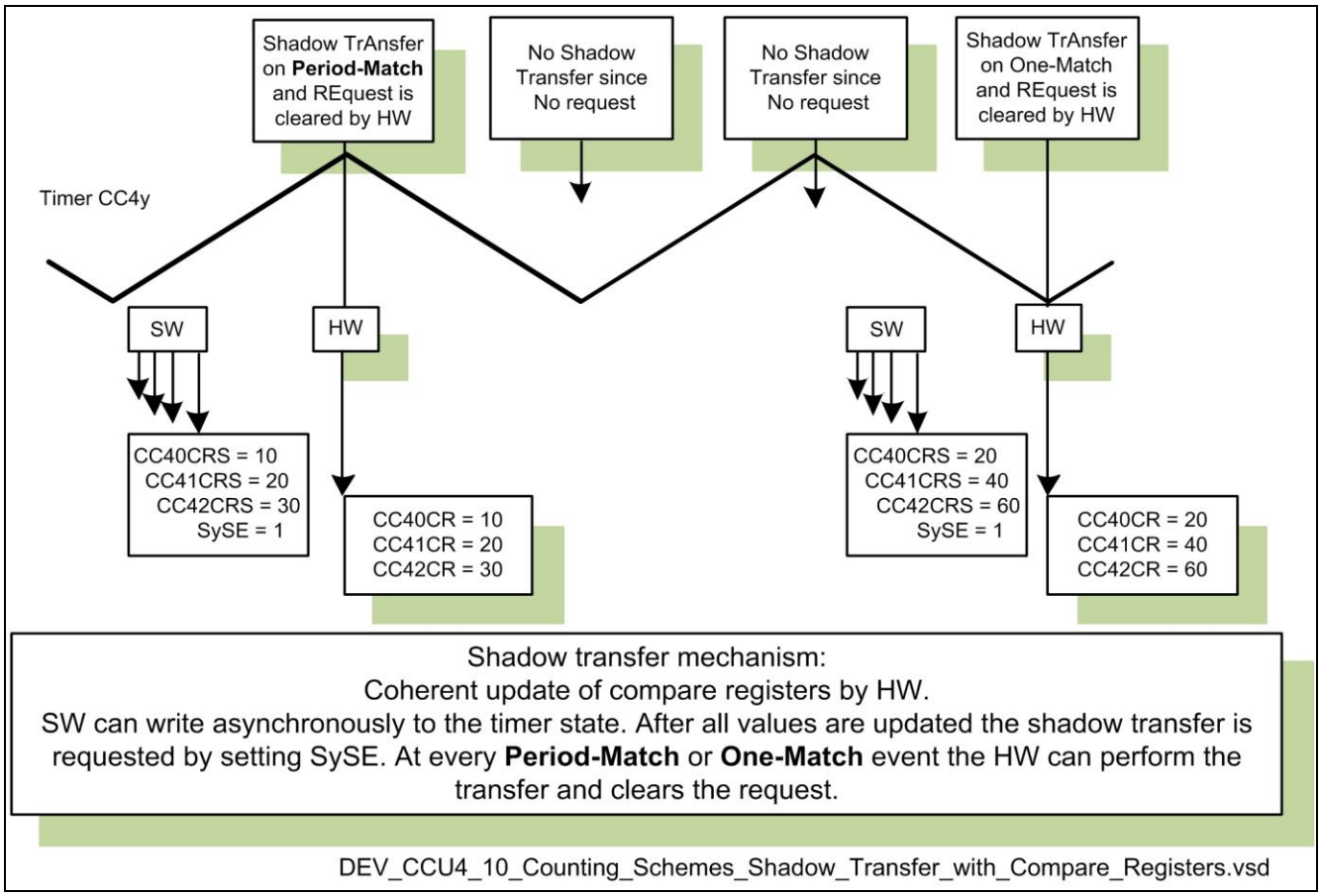


Figure 67 The Shadow Transfer Mechanism (Center Aligned Mode)

11.3 Start Counting Scheme

There are two ways to start a timer, either directly by software setting the Timer Run Bit Set (TRBS) or indirectly by hardware 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.

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

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 as output pattern generation of many different kinds.

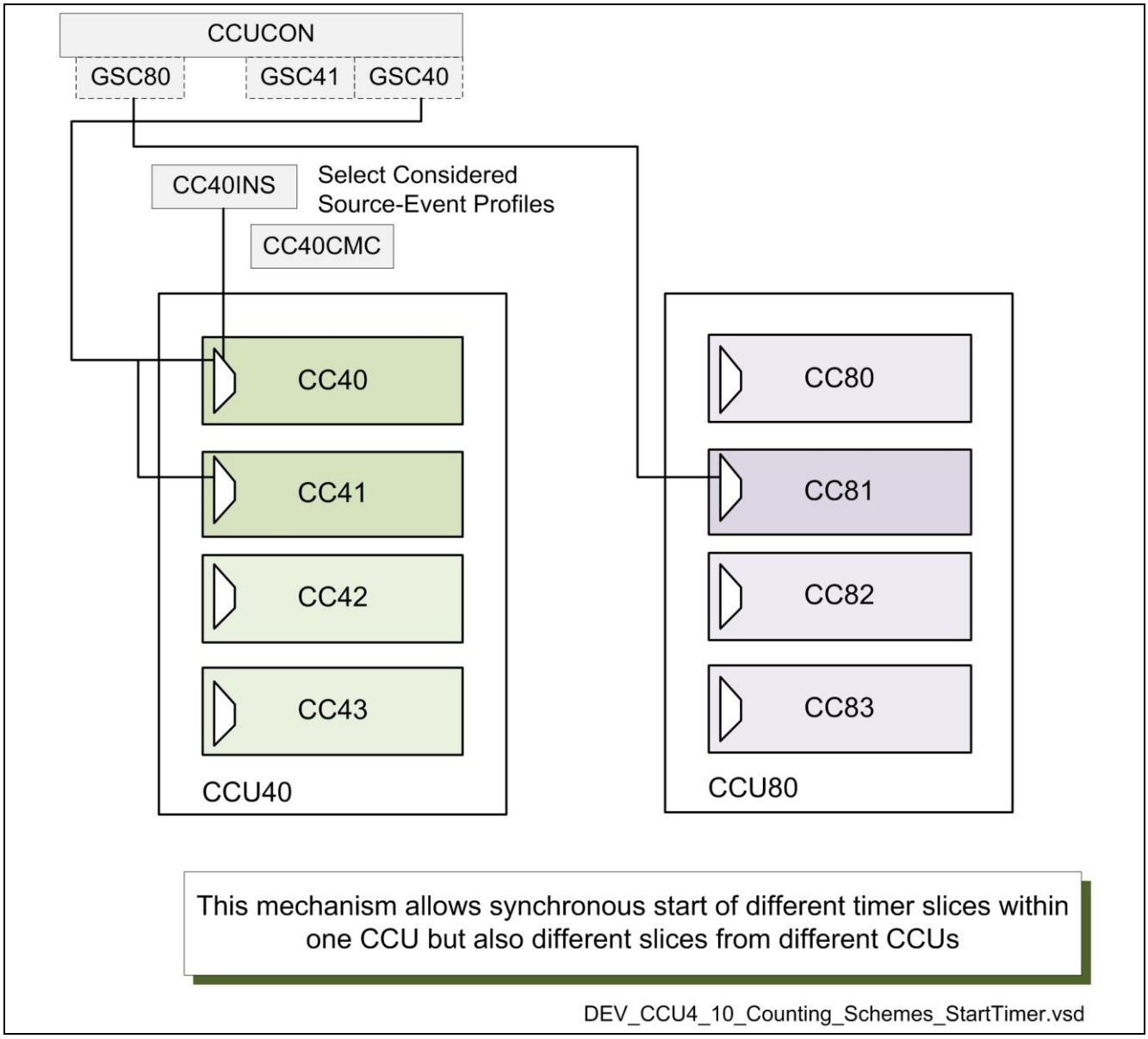


Figure 68 Synchronous Global Start of Timer Slices on different CAPCOM units (CCU4 or CCU8)

11.4 Getting Started with CCU4 and Counting Schemes

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 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:

For example, Start, Extended Start respective Stop, Extended Stop.

Initialize and Start the Timer

After the Start procedures have been executed, the software may:

- Initialize the slice timer CC4yTIMER or clear it with a CC4yTCCLR.TCC bit=1 command
- Start the timer with a Set Timer Run command in the CC4yTCSET.TRBS bit

The timer may be stopped at any time by software with a CC4yTCCLR.TRBC command.

Using the Timer Counting Modes Control Register

There is only one register that is concerned, CC4yTC register bitfields:

- CC4yTC.CMOD — CAPCOM Mode selection bit may be used: 0 → Compare / 1 → Capture mode
- CC4yTC.TCM — Timer Counting Mode Selection bit: 0 → Edge-Aligned / 1 → Center-Aligned
- CC4yTC.TSSM — Set (if intended!) Single Shot Mode by this bit
- CC4yPR.PR — Write the timer count cycle upper limit into the period register

Options:

- CC4yCRS.CRS — Write the required Compare Level value into the compare shadow register
- CC4yPSL.PSL (CC4yPSL.PSL) — PASSIVE/ACTIVE control bit: (CCU4xOUTy**)** = (CCUxSTy) xor
- CC4yTC.STRM Control — Extended Start function on External Start when using External Events
- CC4yTC.ENDM Control — Extended Stop function on External Stop when using External Events
- CC4yTIMER command) — Initialize a timer start value (or flush by the CC4yTCCLR.TCC bit=1

- CC4yTCSET.TRBS Start Timer by software

Top Level Interconnect

12 Top-Level Interconnect Control

The comprehensive matrix of interconnect lines enables signals to get directly from one system unit to another. For example, an ADC conversion can be requested to start via an internal signal from the timer.

The Top-Level interconnect matrix supports immediate interconnections between modules as shown in the following diagram.

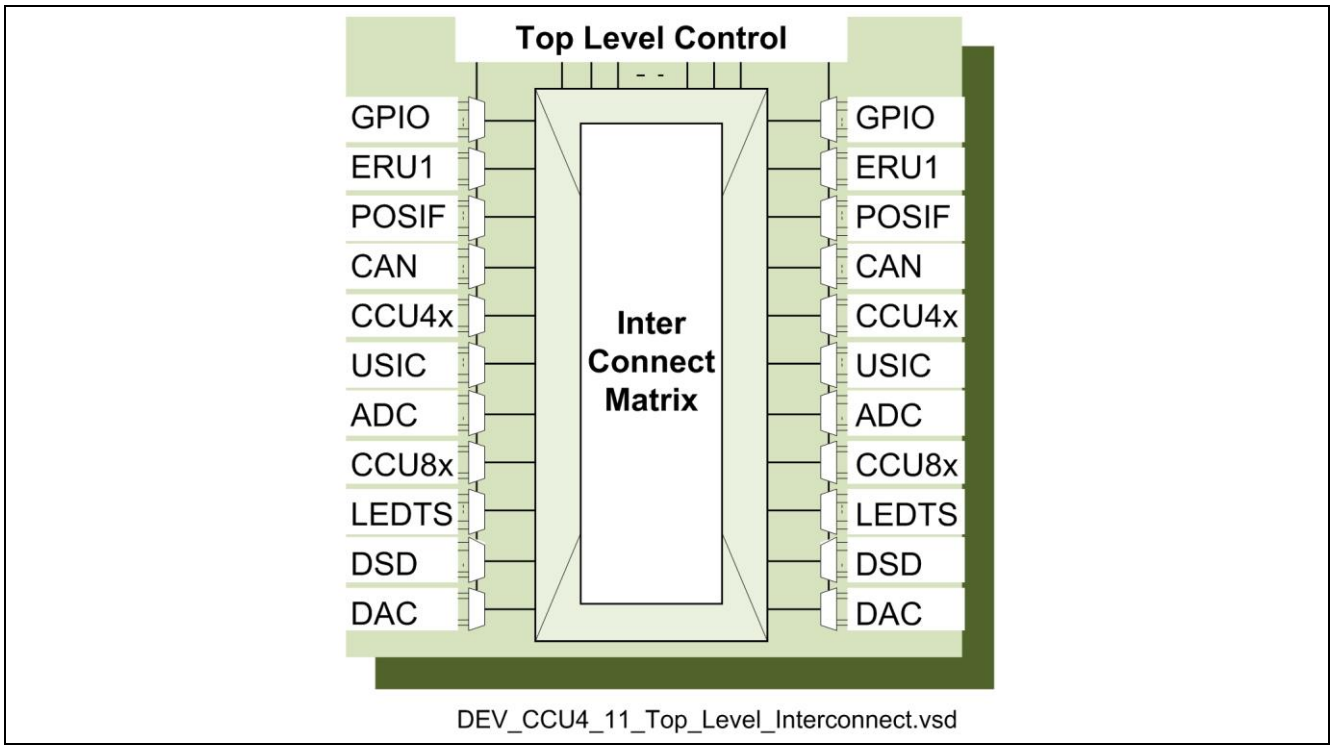


Figure 69 The Top–Level Interconnect Matrix

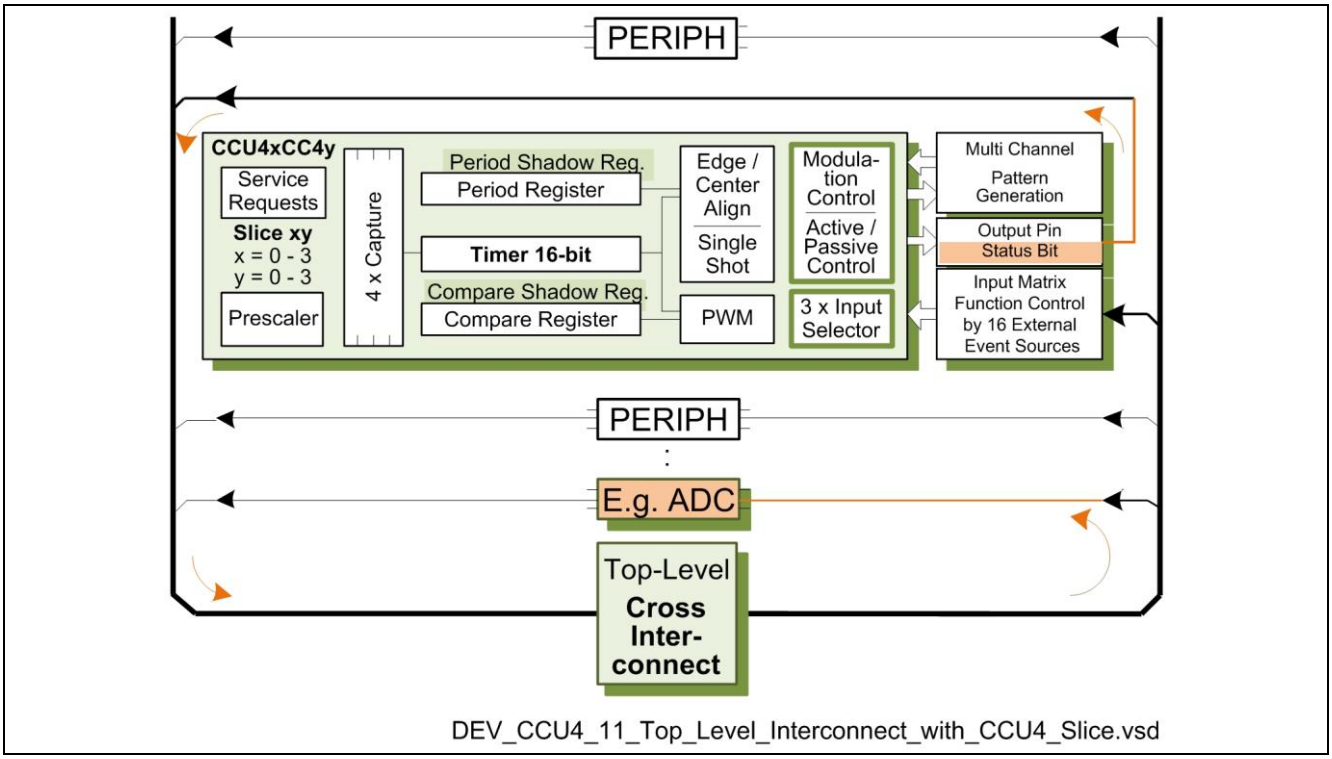


Figure 70 Example with an ADC cross connected to a Timer via the Top–Level Interconnect Matrix

12.1 Top-Level Control of Event Requests to/from a Timer Slice

The Top-Level Control of Event Requests is a flexible control and cross unit distribution of event signals to/from a slice. The Event Request Unit (ERU1) may combine events via the Top-Level Interconnect matrix. The cross unit event flow can be top-level controlled without any CPU load according to the request-to-action event.

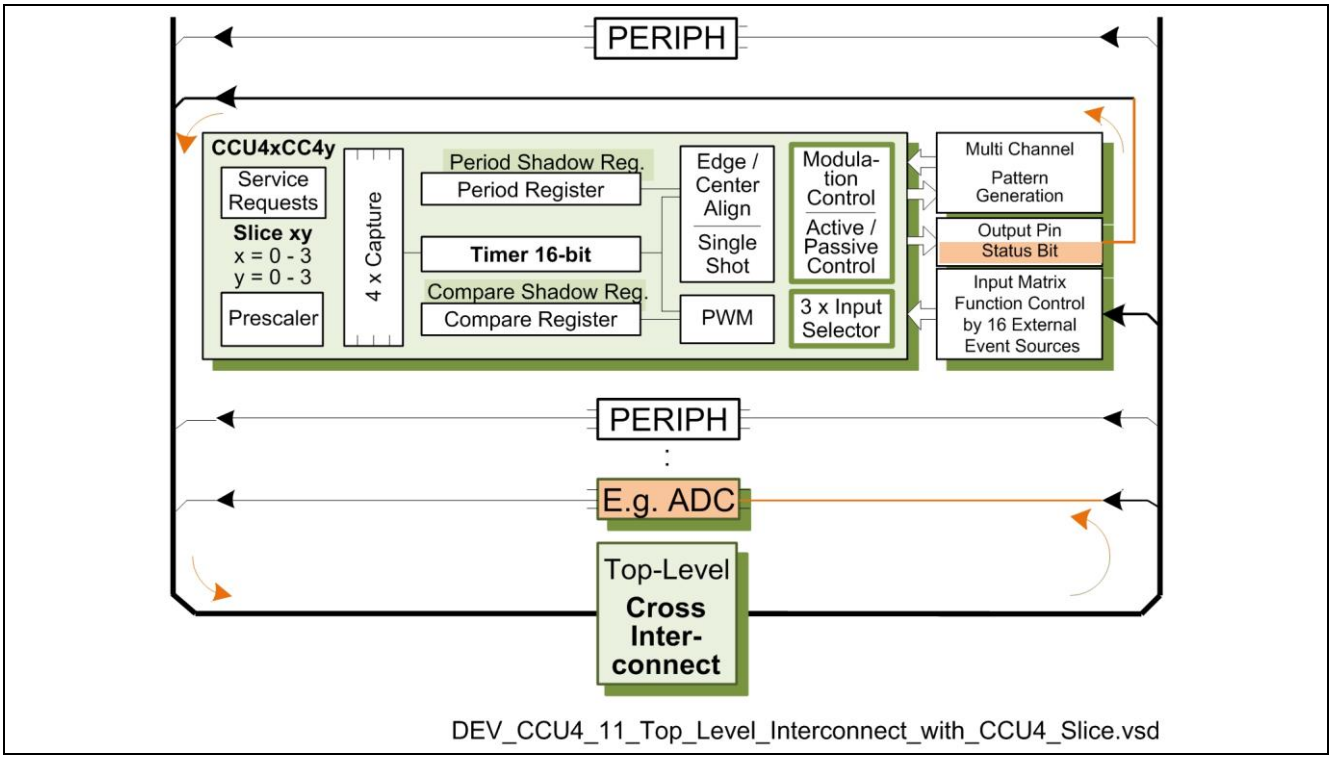


Figure 71 Top-Level Interconnect with conditional event request control by the Event Request Unit

CCU4 Using Top-Level Interconnect Control with Event Request Unit (ERU)

A slice interface to the ERU1 and to the Top-Level Interconnect matrix can be shown by a simplified scheme. To complete the picture of capable interaction, this scheme also shows how operations can be extended to involve DMA transfers (by the GPDMA), triggered by a handler (DLR) on the Service Request Lines (SRn).

CCU4 and ERU Use Cases

If an application requires ADC conversions to start on timer events under specific conditions, for example when not directly via a Top-Level Interconnect matrix path, then the ERU1 is able to offer an alternative signal path. It may involve independence on a port pin, time window due to a second timer, or a certain sequence of event patterns.

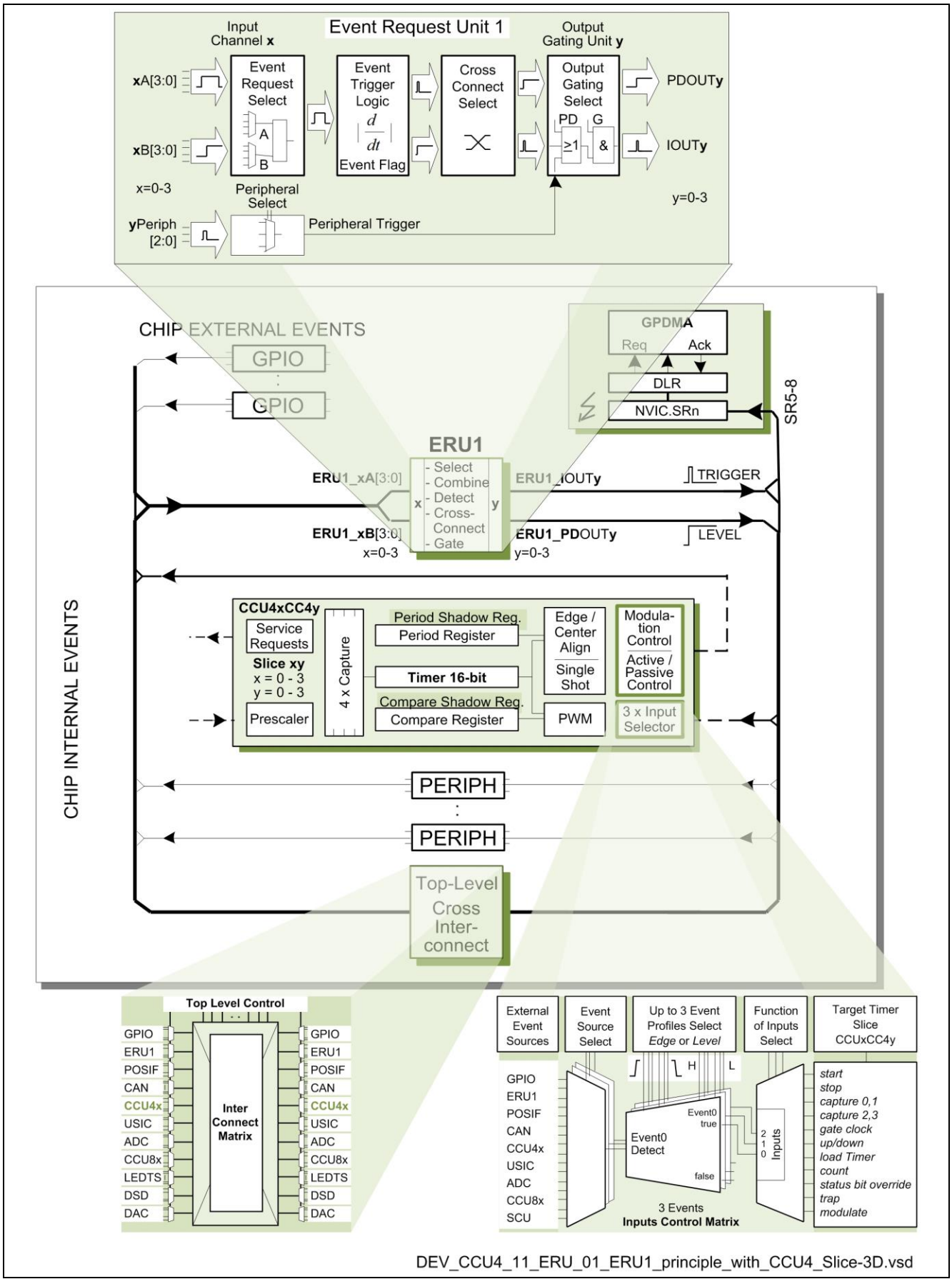

Event Request Unit 1
Input Channel x
Output Gating Unit y
xA[3:0]
xB[3:0]
x=0-3
Event Request Select
A
B
Event Trigger Logic
Event Flag
Cross Connect Select
Output Gating Select
PD
G
≥1
&
PDOUTy
IOUTy
y=0-3
Peripheral Select
yPeriph [2:0]
Peripheral Trigger
CHIP EXTERNAL EVENTS
GPIO
GPIO
GPDMA
Req
Ack
DLR
NVIC.SRn
SR5-8
ERU1
ERU1_xA[3:0]
ERU1_xB[3:0]
x=0-3
- Select
- Combine
- Detect
- Cross-Connect
- Gate
ERU1_IOUTy
ERU1_PDOUTy
y=0-3
TRIGGER
LEVEL
CHIP INTERNAL EVENTS
CCU4xCC4y
Service Requests
Slice xy
x = 0 - 3
y = 0 - 3
Prescaler
4 x Capture
Period Shadow Reg.
Period Register
Timer 16-bit
Compare Shadow Reg.
Compare Register
Edge / Center Align
Single Shot
PWM
Modula-tion Control
Active / Passive Control
3 x Input Selector
PERIPH
PERIPH
Top-Level Cross Inter-connect
Top Level Control
GPIO
ERU1
POSIF
CAN
CCU4x
USIC
ADC
CCU8x
LEDTS
DSD
DAC
Inter Connect Matrix
External Event Sources
Event Source Select
Up to 3 Event Profiles Select
Edge or Level
Function of Inputs Select
Target Timer Slice CCUxCC4y
H
L
Event0 true
Event0 Detect
false
2
1
0
Inputs
start
stop
capture 0,1
capture 2,3
gate clock
up/down
load Timer
count
status bit override
trap
modulate
3 Events
Inputs Control Matrix
DEV_CCU4_11_ERU_01_ERU1_principle_with_CCU4_Slice-3D.vsd

Figure 72 Details of a CCU4 Timer Slice system interactions by Top-Level Interconnect Control

12.2 Getting Started with CCU4 and Cross Interconnections

Getting Started with the CAPCOM4 Unit CCU4x

In the following examples the ERU1 and CCU4 Initialization Sequence is assume before start.

Using the CAPCOM4 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.

CAPCOM4 External Event Control Register Bitfields

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

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 Event Number 'x' for the Input Function is x == <function>-1 (when <function> != 0). <function>== 0 means "disabled".

Getting Started with Event Request Unit – ERU1

There are one Input Selection Register and two Control Registers per input channel x (x=0-3) to setup the entire functionality of an ERU on a Top Level and channel wise target the Output Channels y (y=0-3):

- EXISEL Input Selection Register
- EXICONx Input and Trigger Control Register
- EXOCONx Output Control Register

Initializations and functions are determined by writing to the respective register bitfields.

Cross Interconnection Example – Using CCU4, ERU1, ADC and GPIO

ADC Request on Port Pin P2.1 State AND the Timer CCU40CC40 Status Bit

Setup ERU1 Input Channel x=0, targeting the Output Channel y=2 (IOUT2), from which a link to the ADC Trigger Request N Input, Group 2 or 3 can be routed.

```
// Event Request for input A & B in the ERU1_EXISEL register bitfields
EXS0A = 2; // select ERU1_0A2 – i.e. A connected to CCU40.ST0 status bit 0
EXS0B = 0; // select ERU1_0B0 – i.e. B connected to a GPIO port pin P2.1

// Select logical combinations that should be taken into account as event
request
```

```
NA = 0; // input A is used directly, i.e. not inverted
NB = 0; // input B is used directly, i.e. not inverted
SS = 3; // source combination, i.e. logical condition: input A AND input B

// Event Trigger Logic conditions for trigger, level detect and event edge
PE = 1; // Trigger pulse enable, i.e. an output trigger pulse will be
       // created
LD = 0; // define event flag sticky, i.e. it will not be rebuild by HW, but
       // by SW
RE = 1; // detection on Rising Edge, i.e. event edge on positive signal
       // transition
FE = 0; // no detection on Falling Edge

// Perform Cross Connect Selection, i.e. target output channel y (here y=2)
// Event Trigger Logic Output Pulse TR0 from input channel x (here x=0):
// Register for the following mapping is EXICONx, i.e. here ERU1_EXICON0:
       // CCU4x.IN2(K)     - CAPCOM4 Unit CCU4x, Slice CC42 Timer Input
       // CCU8x.IN2(G)     - CAPCOM8 Unit CCU8x, Slice CC42 Timer Input
       // VADC.G2REQTRN    - ADC Trigger Request N Input, Group 2
       // VADC.G3REQTRN    - ADC Trigger Request N Input, Group 3
       // ERU1.1B3         - A trigger feed-back to ERU1 Channel Input 1B3
       // NVIC.SR7         - A trigger to Service Request Line nr 7
       // POSIF0.MSET(F)   - POSIF 0 Multi-Channel Next Pattern Update Set
       // POSIF1.MSET(F)   - POSIF 1 Multi-Channel Next Pattern Update Set

// Event Output Trigger Control 2 Register ERU1_EXOCON2 for output gating
// Output Gating functions – Here: just a straight forward alternative:
// Target register for the following mapping is EX0CONy, i.e. here
// ERU1_EXOCON2:
GEEN = 0; // disable “Gating Event Enable on Pattern Detection Changes”
GP   = 1; // “Output Gating Select on Pattern Detection” =1, to activate
       // IOUT2
IPEN0, IPEN1, IPEN2, IPEN3 = 0; // disable Pattern Detection Enable flags
```

Event Request Unit Block Diagrams and Control Registers

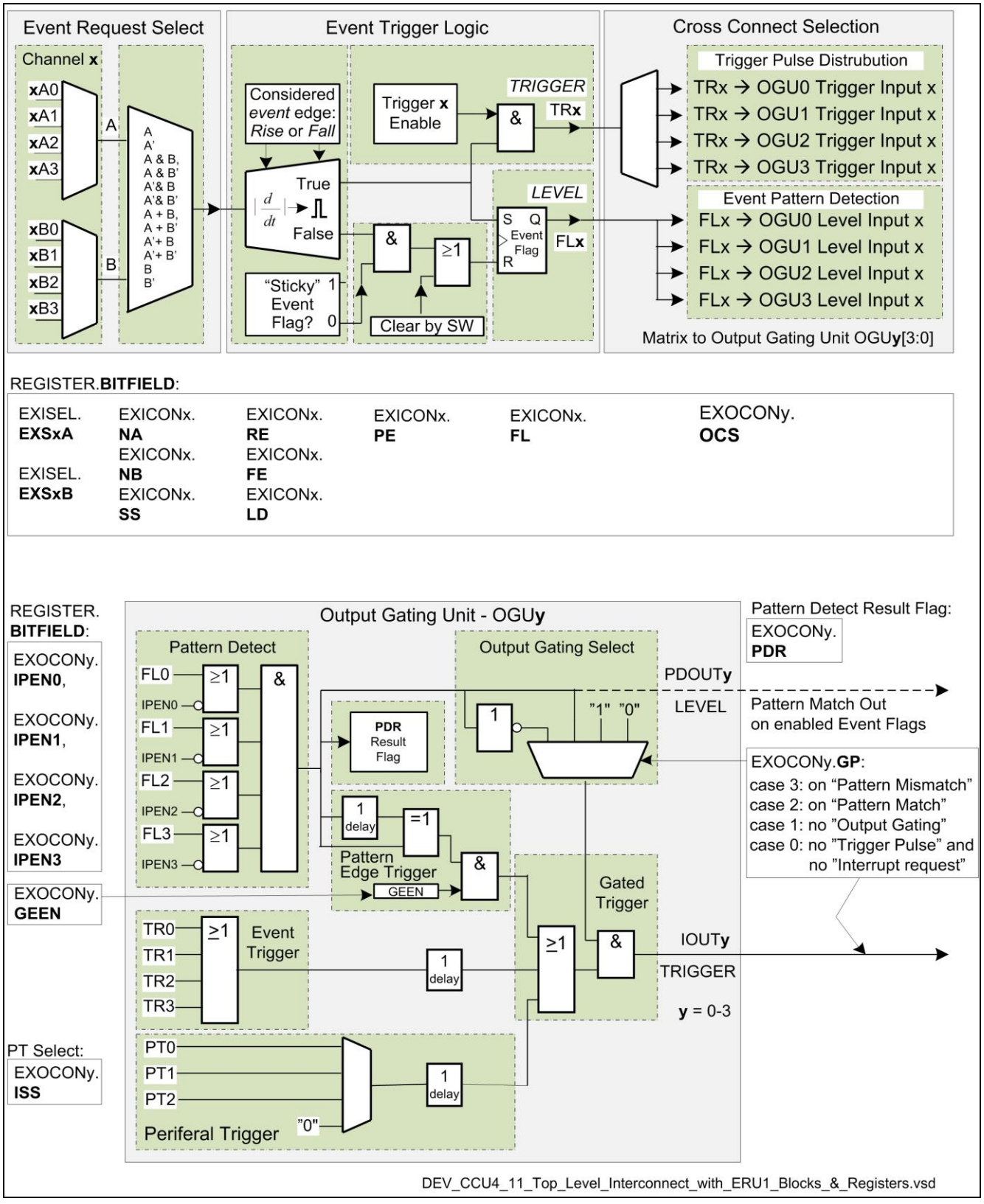


Figure 73 Event Request Unit Block Diagrams and Control Registers

12.3 Runtime Handling of CCU4 and Cross Interconnections

Handling Event Concatenation using ERU1 and Pattern Match Detection

In the following example the intention is to show a state-machine-like configuration setup of the ERU1. It will be used to explain how a certain sequence of events can be concatenated on a Top-Level control and detected by an Event Pattern Match on the ERU output gating stage. The scenario is for example a De-bounce Rejection Filter.

Note: No runtime software involved.

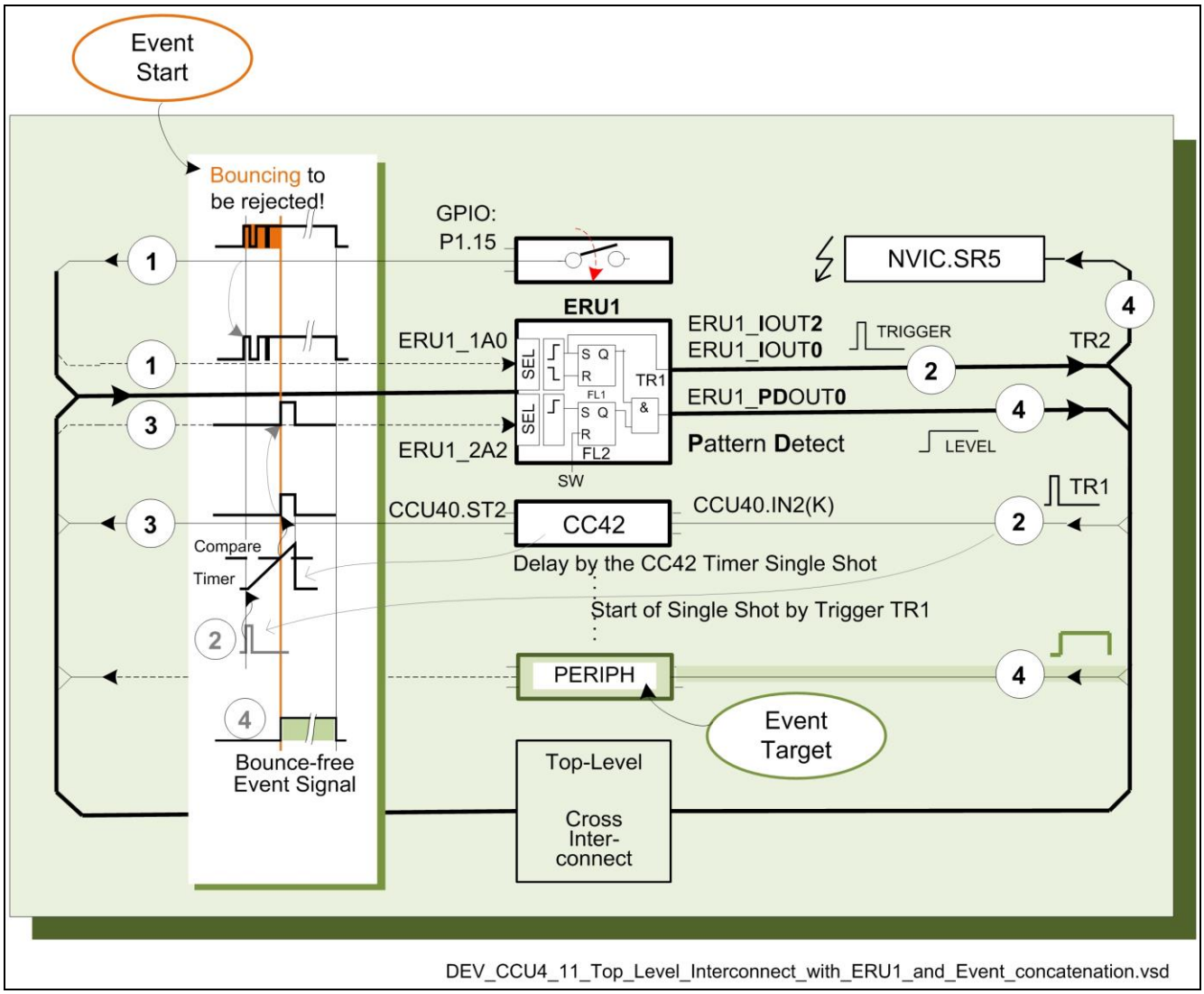


Figure 74 Concatenation of events with Time Window for a press-button De-bounce Rejection Filter

- Bouncing signal from SWITCH-ON. This signal is connected to the ERU1 Channel 1 Input Selection Pin A0 (ERU1_1A0). The event will be detected on Rising Edge, memorized in event flag FL1 and generate a trigger pulse TR1 on the Output Gating Unit 2 (IOUT2) after the ERU1 internal Cross Connect Stage.
- Trigger TR1 on Output Gating Unit 2 (IOUT2) starts the Timer CCU40T42 in Single Shot mode by an External Event Control command (including the Extended Mode condition for Flush/Start).
- On the single shot timer Compare Event that starts the Single Shot time frame, the CCU40ST2 status bit ST2 will be set. This signal is connected to the ERU1 Channel 2 Input Selection Pin A2

(ERU1_2A2) and will set the sticky flag FL2 and send a trigger pulse TR2 on IOUT0 after the Cross Connect Stage.

- On the Output Gating Unit 0 (IOUT0) pin the trigger pulse TR2 can be distributed to the Service Request Line nr 5. The PDOUT0 output pin of the Output Gating Unit 0 there will be a Pattern Match Detection by the AND-Gate and the flags FL1 and FL2. This is the de-bounced press-button switch signal, extracted by event concatenation, which is sent to the Peripheral Unit as the “Event Target”.

Multi-Phase Output control using the Multi-Channel Mode

13 Multi-Phase Output Pattern Control and Synchronization

The CAPCOM4 (CCU40/../43) is a multi-purpose timer unit for signal monitoring/conditioning and Pulse Width Modulation (PWM) signal generation. It is designed with repetitive structures with multiple timer slices that have the same base functionality. The internal modularity of CCU4, translates into a software friendly system for fast code development and portability between applications.

The following image shows the main function blocks of one of the four CC4y slices on a CCU4x.

Multi-Channel Mode Basics

The output pattern CC4yOUT of any slice that has been set in multi-channel mode (CC4yTC.MCME = 1_B), can be controlled in parallel by an external pattern that is present on the Multi-Channel Input MCIy. In order to achieve synchronization, a Set Shadow Transfer Enable MCSS and a Pattern Set PSy request must be set.

The Multi-Channel Slice Function

In Multi-Channel Mode the Status Bit (CCST) is delayed by 3 Module Clocks. So that the AND operation between the Status Bit and the requested Multi-Channel Input value MCIy is still valid during the Set Pattern request PSy. An External Modulation or Status Bit Override on external events control can be added if required.

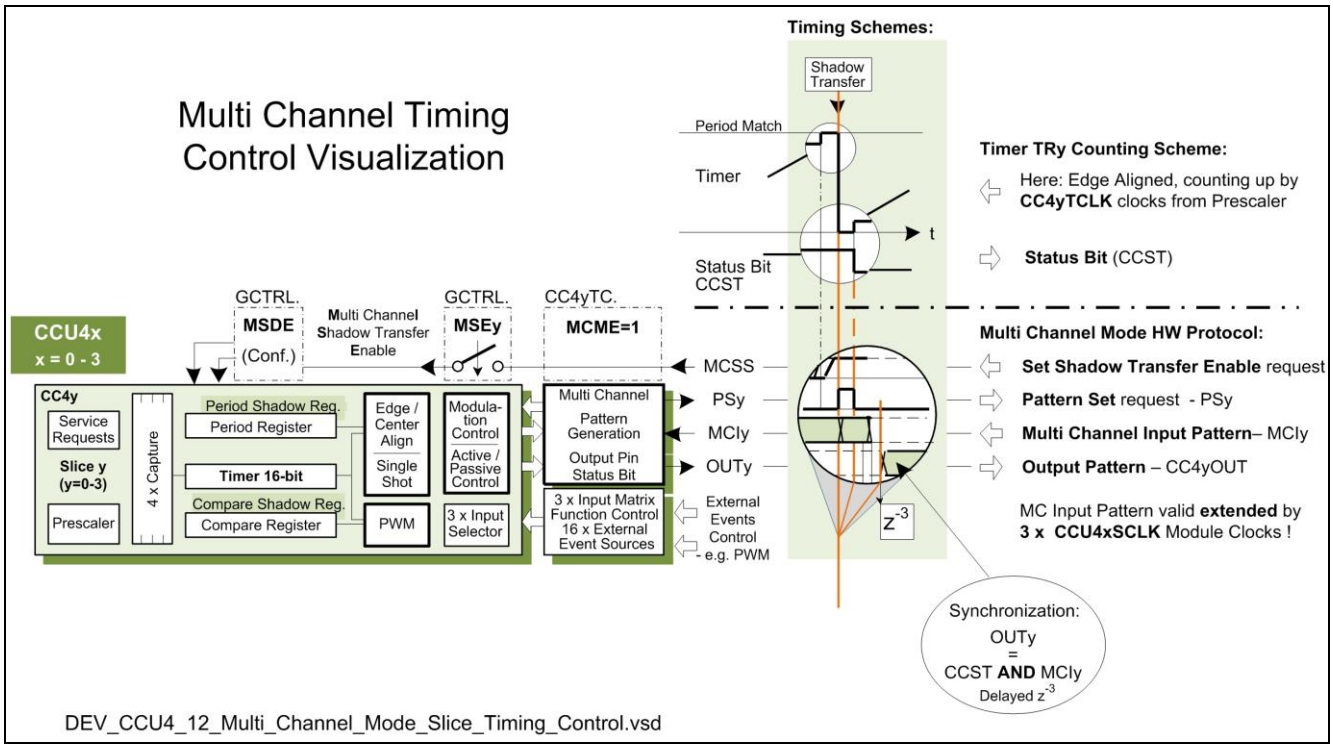


Figure 75 Multi-Channel Mode HW Protocol Visualization

13.1 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, the CCU4xMCSS trigger can be used for the synchronization. The synchronization will be performed via the GCTRL.MSEy bit field base on the shadow transfers configured on the GCTRL.MSDE field.

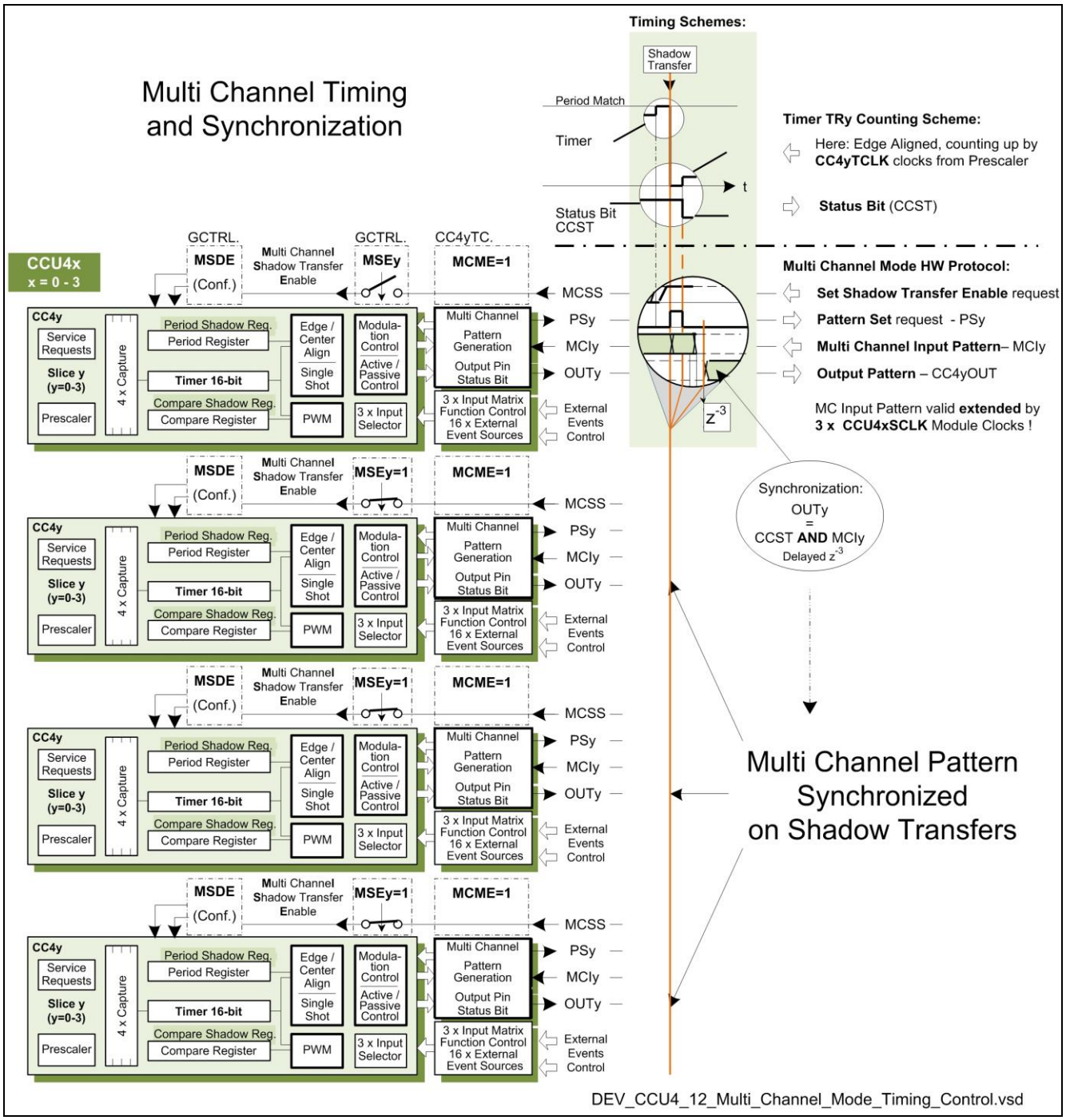


Figure 76 Multi-Channel Timing and Synchronization

CCU4 Shadow Transfer for Coherent Signal Pattern Update

All CAPCOM4 timers, in any slice configuration, are assured coherent update by hardware of all relevant timer function parameters. The values in shadow registers are updated on a global preset request simultaneously to all function registers at a Period-Match or One-Match.

The Global Shadow Transfer Set Enable Register

The Global register, GCSS, carrying all enable-flags that have to be preset by software to selectively activate the targeted Shadow Transfer Requests. It can be cleared by hardware after the transfer. The real-time correctness can achieved from these logic operations which are essential for safe power switching.

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, for example SySE in GCSS, must be preset at the latest within the clock cycle of Period Match (in Edge Aligned Mode) or Period/One Match (in Center Aligned Mode)

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 can also be simultaneous updated via the SyDSE and SyPSE request flags.

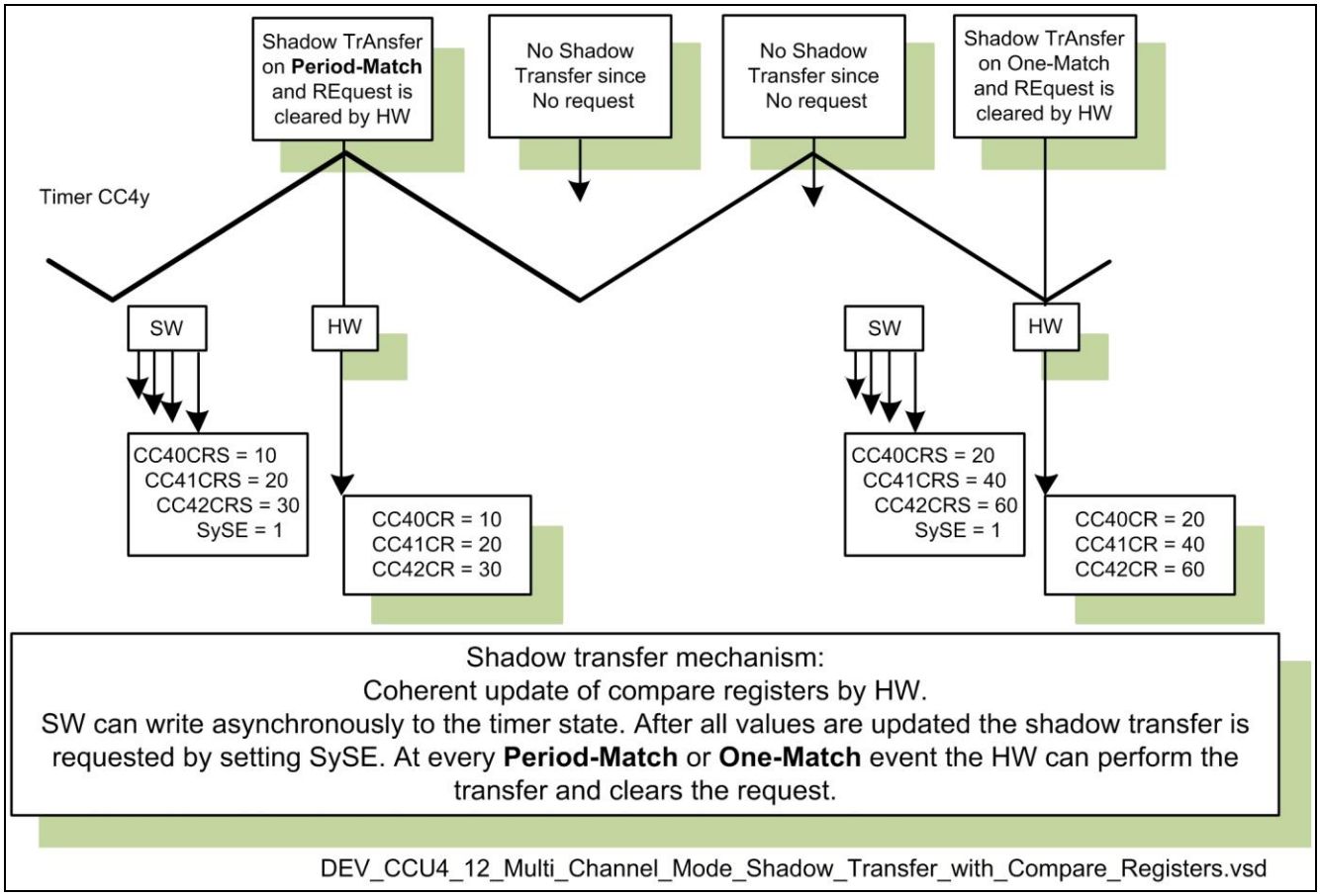


Figure 77 The Shadow Transfer Mechanism (Center Aligned Mode)

13.2 Start Timers

There are two ways to start a timer, either directly by software setting the Timer Run Bit Set (TRBS) or indirectly by hardware 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 with CAN, ADC, USIC, IO, CCU4/8, ERU1, POSIF etc.

Global Start of CAPCOM Units

There is a way to get a synchronized start of CAPCOM Units, both CCU4x and CCU8x:

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

Global Start of the CAPCOM Units CCU4 and CCU8

The synchronization of the starting of different timers can be achieved and all timings can be controlled in parallel, regardless of which CAPCOM unit the timers belong.

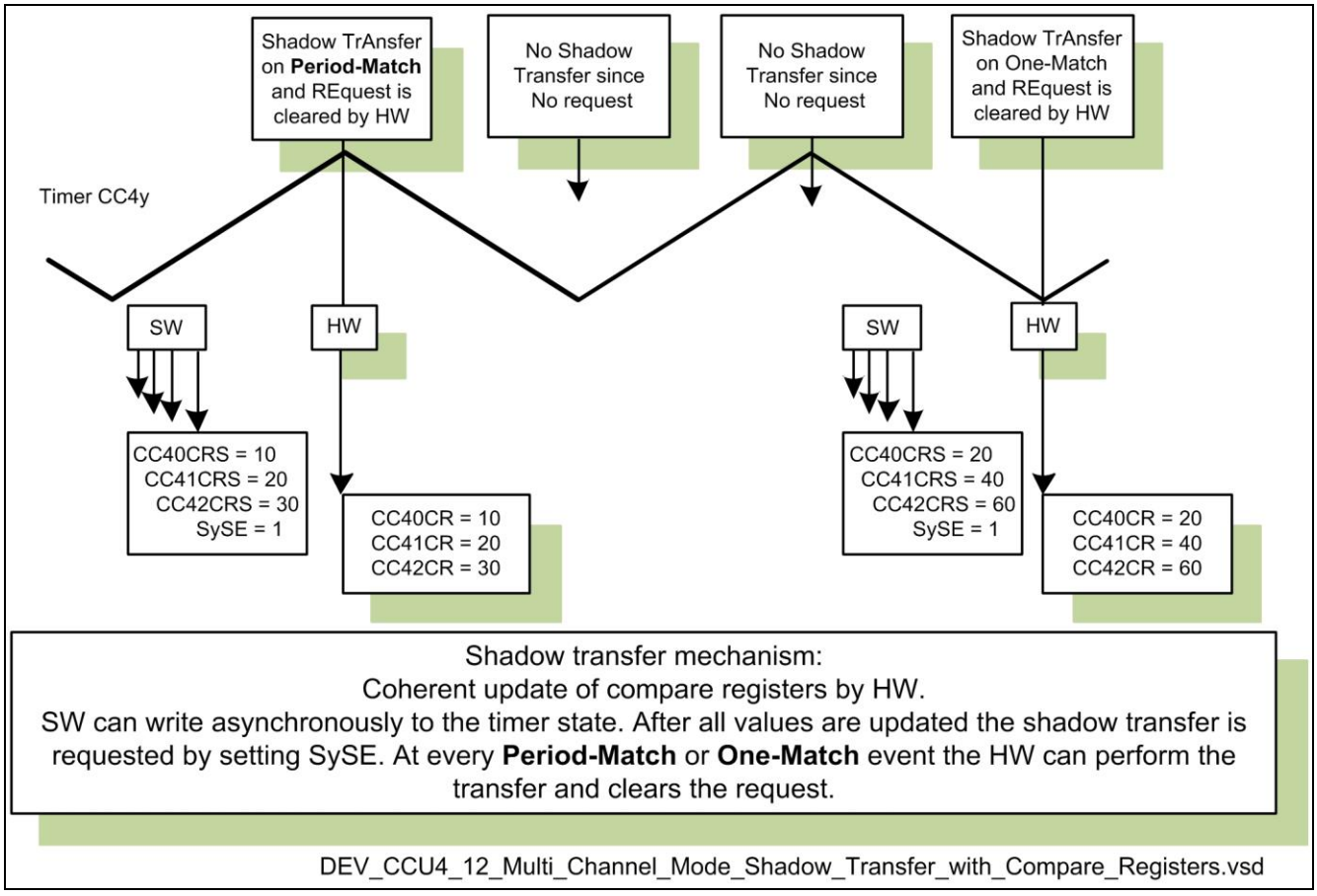


Figure 78 Synchronous Global Start of Timer Slices on different CAPCOM units (CCU4 or CCU8)

13.3 Getting Started with CCU4 and Multi-Channel Mode

Initialization Sequence

Before the start and execution of timer slice software for the first time, the CCU4 must have been initialized appropriately using the following sequence:

- Apply Reset
- Release Reset
- Enable Clock
- Enable Prescaler Block
- Configure Global Control
- Configure Slice(s) Functions, Interrupts and Start-up

Start-Up Enable

In the last part of the CCU4 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 Setting Compare Mode Control

The default state for the CAPCOM timer is the Compare Mode.

The compare level will be set in the compare register CC4yCR with 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.

Setting the Output Pins PASSIVE / ACTIVE Level Control

The PASSIVE/ACTIVE state of a slice internal output (for example the status bits CC4yST), is controlled by the compare level and the External Modulation Mode. The external output pin CCU4xOUTy can be controlled to Passive Low / Active High or vice-versa via the CC4yPSL.PSL bit.

Initialize and Start the Timer

After the Start procedures have been executed, the software may:

- Initialize the slice timer CC4yTIMER or clear it with a CC4yTCCLR.TCC bit=1 command
- Start the timer with a Set Timer Run command in the CC4yTCSET.TRBS bit

The timer may be stopped at any time by software with a CC4yTCCLR.TRBC command.

Multi-Channel Mode Enable

A slice has to be set in multi-channel 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 configure the shadow transfers by the GCTRL.MSDE bitfield.

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.

Using the Respective Multi Channel Slices Control Registers

- GCTRL.MSDE Configure transfer for Compare / Compare+Period+Prescaler / and even Dither
- CC4yTC.MCME Enable Multi Channel Mode
- GCTRL.MSEy Enable Shadow Transfer Request
- CC4yTC.CMOD CAPCOM Mode selection bit should be =0 for Compare Mode (default)
- CC4yTC.TCM Set Timer Counting Mode selection bit: 0 → Edge-Aligned / 1 → Center-Aligned

- CC4yCRS.CRS Write the required Compare Value into the CRS compare shadow register
- CC4yPR. PR Write the timer count cycle upper limit, Period Match, into the period register
- CC4yPSL.PSL Set PASSIVE/ACTIVE by 0: CCU4xOUTy Passive Low 1: CCU4xOUTy Active Low

Optionally:

- CC4yTIMER Initialize a timer start value (or flush by the CC4yTCCLR.TCC bit=1 command)
- CC4yTC.CLST Enable a shadow transfer each time a clearing action is done by SW or External Event
- GCSS.SySE Set request at least once for a Shadow Transfer to Initialize the Compare Registers
- Global Start and CC4yCMC Map start control to CCUCON and External Events Control to CC4yINS

External Event Control Setup Register

External Events Control would be useful for example, 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, such as Start.

Using the External Start Control Register Bitfields

CC4yCMC**.**STRTS Input Function Selection and Event Number x=0-2 Input Line Selection bitfield

CC4yINS**.**<Event-x bitfields> Event-x Source EVxIS (CC4yINS[P:A]) and Event Edge Mode EVxEM bitfields

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

Using the External Stop Control Register Bitfields

Timers can be stopped by External Events Control. The following Control Register Bitfields should be used then:

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

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

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
```

13.4 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, for example a POSIF unit. The PWM duty-cycle can be modulated linearly from 0 to 100% by varying the compare level.

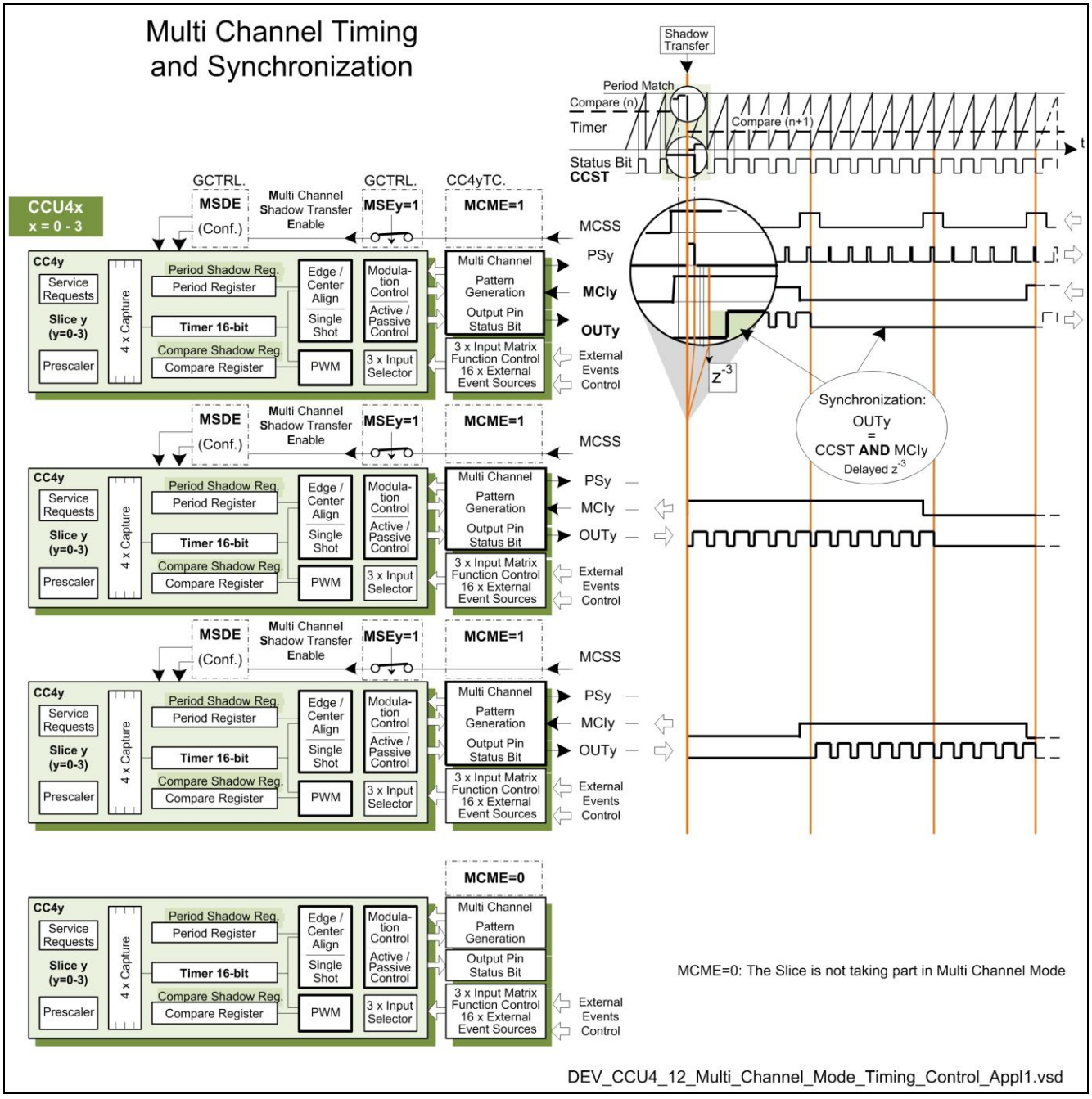


Figure 79 Multi-Channel Timing and Synchronization Application in Runtime

13.5 Example Application for CCU4 and Multi-Channel Mode

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 required motor control pattern.

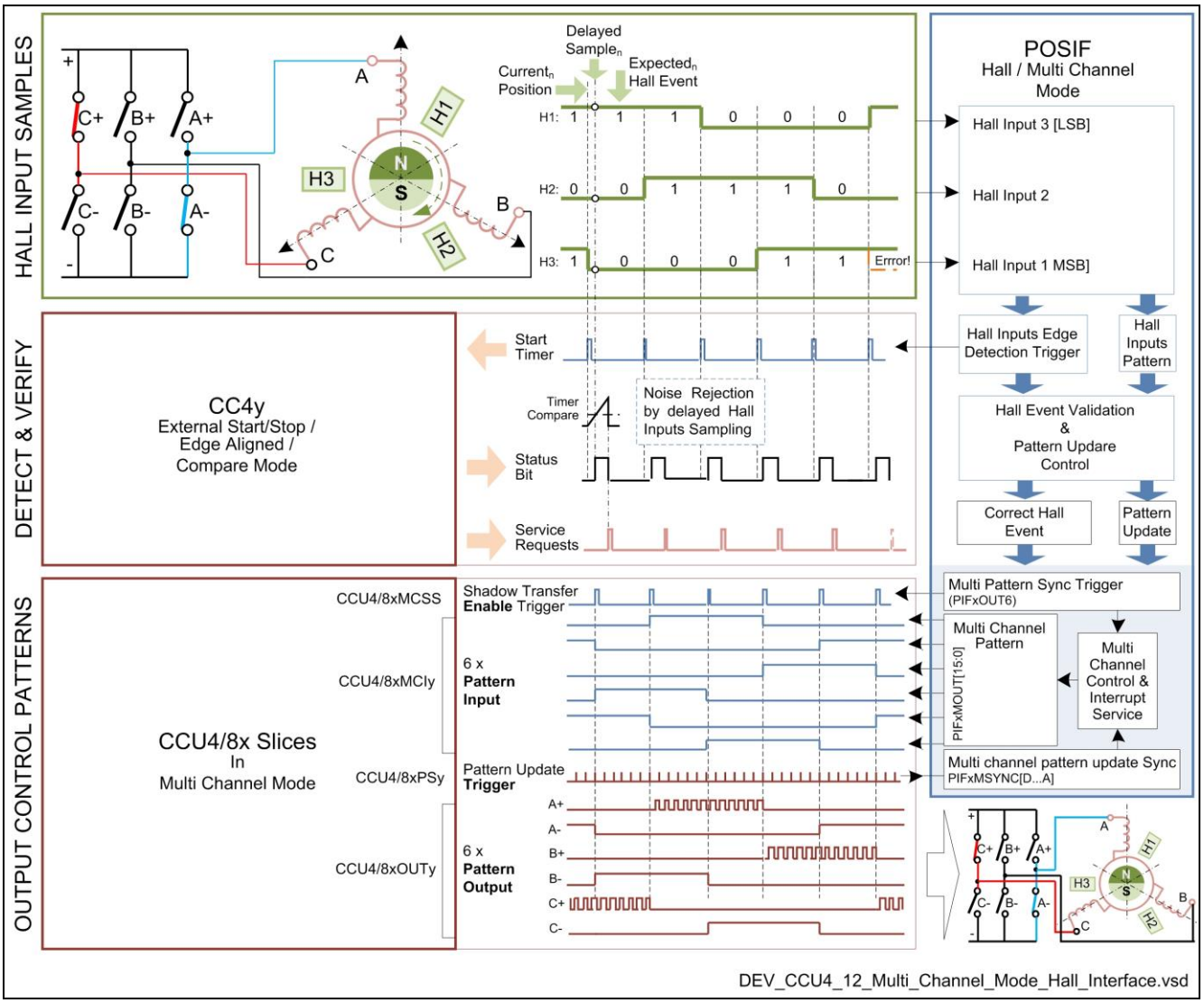


Figure 80 Using POSIF in Hall Sensor Mode and Control by Multi Channel Interface

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

Published by Infineon Technologies AG