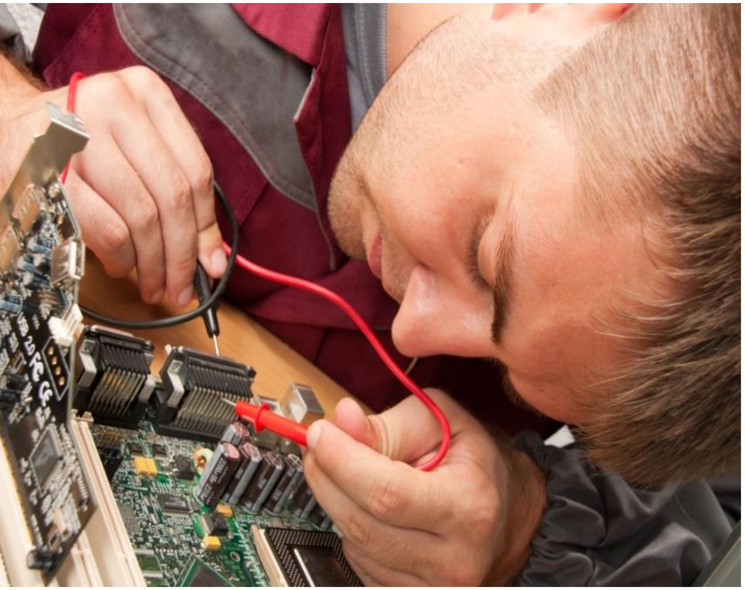
TECHNISCHE
HOCHSCHULE
DEGGENDORF



Microcontroller Programming (7)

Gerald Kupris, 26.11.2013

innovativ & lebendig – Bildungsregion DonauWald

Lectures Microcontroller Programming WS2013/14

08.10.2013	Microcontroller, Programming and Debuging Interfaces
15.10.2013	Reading and Writing of Registers
22.10.2013	I/O-Pins, Reading and Writing of Single Bits
29.10.2013	Clock Generation, CPU und Computing Power
05.11.2013	Interrupts
12.11.2013	**No lecture !**
19.11.2013	Memory
26.11.2013	Timer and PWM, Watchdog Timer
03.12.2013	Analog to Digital Converter
10.12.2013	Serial Interfaces: SPI, IIC and UART
17.12.2013	Additional Explanation of the Freescale Cup Cars
14.01.2014	Project Work on the Freescale Cup Cars
21.01.2014	Project Work on the Freescale Cup Cars

Hands-On Workshops Microcontroller Programming

08.10.2013	Workshop 1: Preparation of the Work Place
15.10.2013	Workshop 2: Loading and Debugging of Programs
22.10.2013	Workshop 3: Using the GPIO Pins
29.10.2013	Workshop 4: Clock Generation and Calculations
05.11.2013	Workshop 5: Interrupts
12.11.2013	**No Workshop !**
19.11.2013	Workshop 6: Using the Flash Memory
26.11.2013	Workshop 7: Timer and Pulse Width Modulation (PWM)
03.12.2013	Workshop 8: Analog to Digital Conversion
10.12.2013	Workshop 9: Serial Communication
17.12.2013	Project work on the Freescale Cup Cars
14.01.2014	Project work on the Freescale Cup Cars
21.01.2014	Project work on the Freescale Cup Cars

Participation on all workshops is required for admittance to the final project!

Start Time Tuesday:	**15:45 p.m.**
New Start Time Thursday:	**14:45 p.m.**

Freescale Cup Rules of Participation

All Teams will receive a pre-assembled Freescale Cup Car.
This car can be / should be modified (HW and SW) as wanted, as long as it complies to the **2014 EMEA Challenge Rules** (published on V drive).

All teams have to submit a technical report of their vehicle.

The Semester Final Race will take place 21.-23. January 2014 in ITC1.

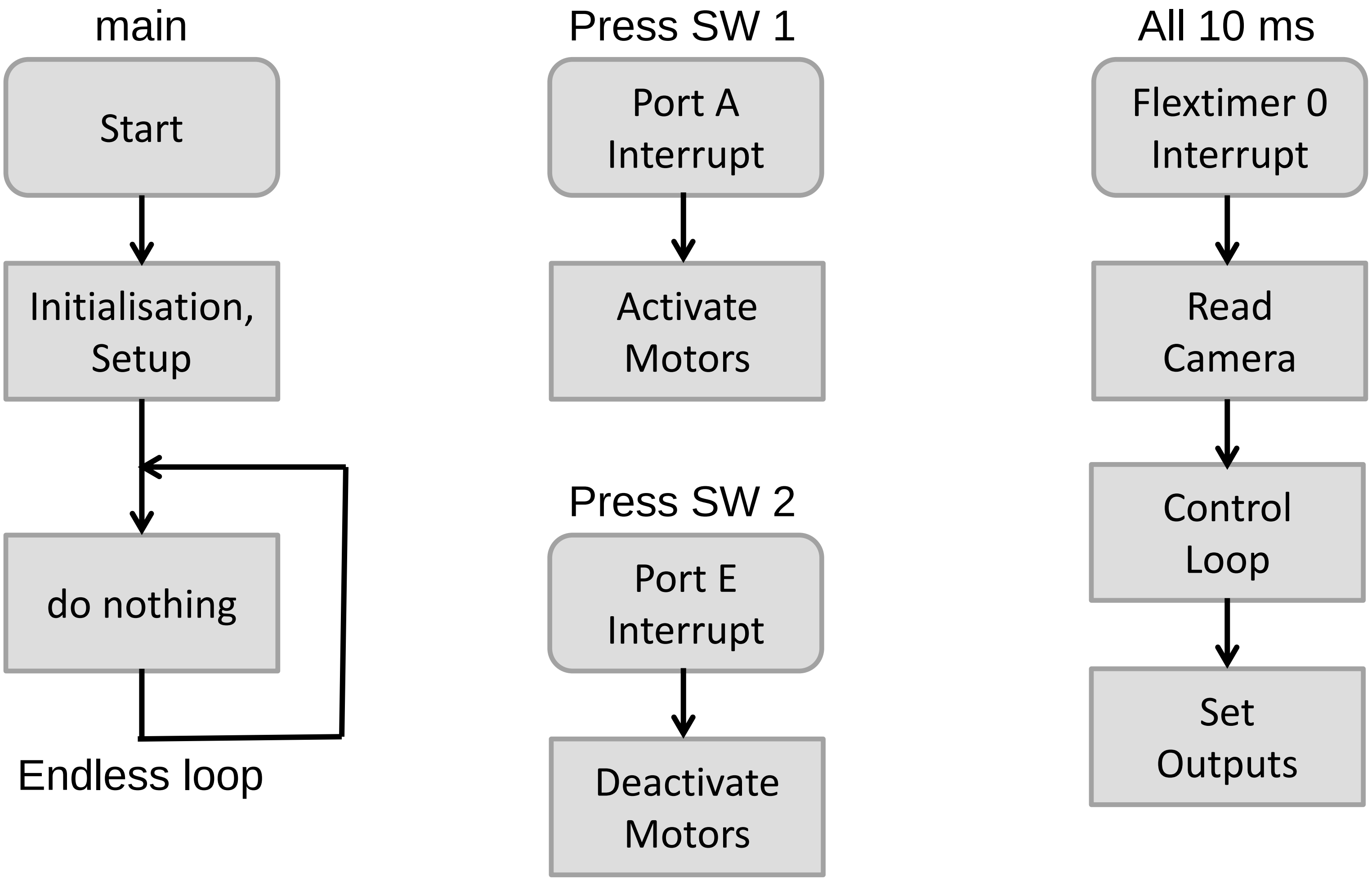
The grade of the course will depend on the results of the semester finals and on the quality of the technical report.

The best teams will participate in the German Qualification Race:
Tuesday, 18 Mar 2014: Munich, Germany at the University of Applied Sciences Munich - attendees of all teams in Germany and UK.

The winners will qualify for the 2014 EMEA Final Race 29. and 30. April 2014 in Erlangen.

Freescale Cup Basic Software Structure

Example project for K60N512, but can be changed to any other MCU



Timer

A timer is a specialized type of clock for measuring time intervals.

Computer systems usually have at least one hardware timer. These are typically **digital counters** that either increment or decrement at a fixed frequency, which is often configurable, and which interrupt the processor when reaching zero, or alternatively a counter with a sufficiently large word size that it will not reach its counter limit before the end of life of the system.

More sophisticated timers may have **comparison logic** to compare the timer value against a specific value, set by software, that triggers some action when the timer value matches the preset value. This might be used, for example, to measure events or generate **pulse width modulated (PWM)** waveforms to control the speed of motors.

One specialist use of hardware timers in computer systems is as **watchdog timers**, that are designed to perform a hardware reset of the system if its software fails.

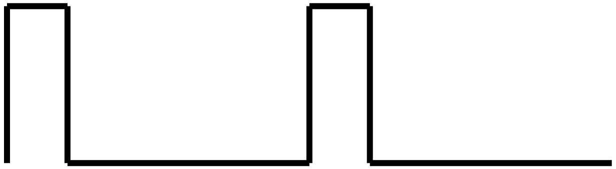
Pulse Width Modulation (PWM)

Pulse-width modulation (PWM), or pulse-duration modulation (PDM), is a modulation technique that conforms the width of the pulse, formally the pulse duration, based on modulator signal information. Although this modulation technique can be used to encode information for transmission, its main use is to allow the control of the power supplied to electrical devices, especially to inertial loads such as motors.

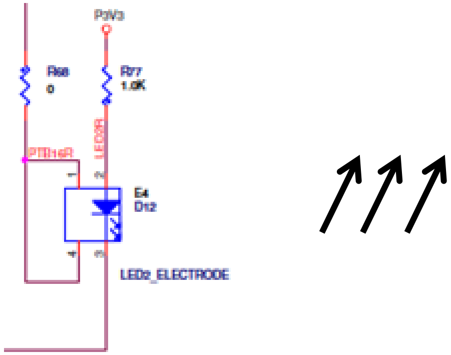
The **PWM switching frequency** has to be much faster than what would affect the load, which is to say the device that uses the power. Typically switchings have to be done several times a minute in an electric stove, 120 Hz in a lamp dimmer, from few kilohertz (kHz) to tens of kHz for a motor drive and well into the tens or hundreds of kHz in audio amplifiers and computer power supplies.

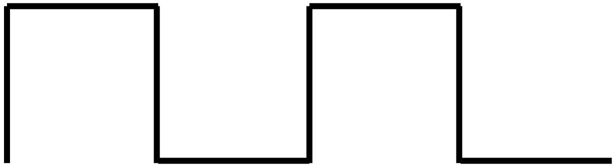
The term **duty cycle** describes the proportion of 'on' time to the regular interval or 'period' of time; a low duty cycle corresponds to low power, because the power is off for most of the time. Duty cycle is expressed in percent, 100% being fully on.

Modulating the Brightness of a LED

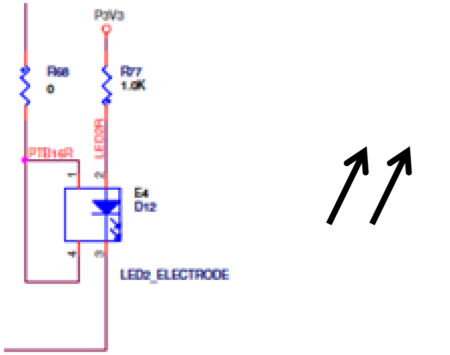


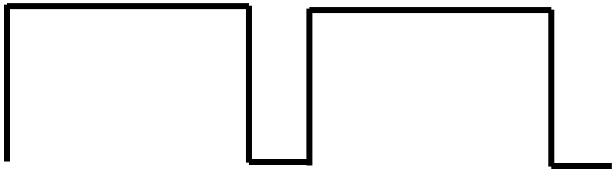
20 %



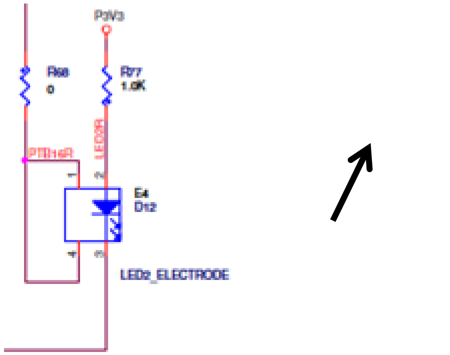


50 %

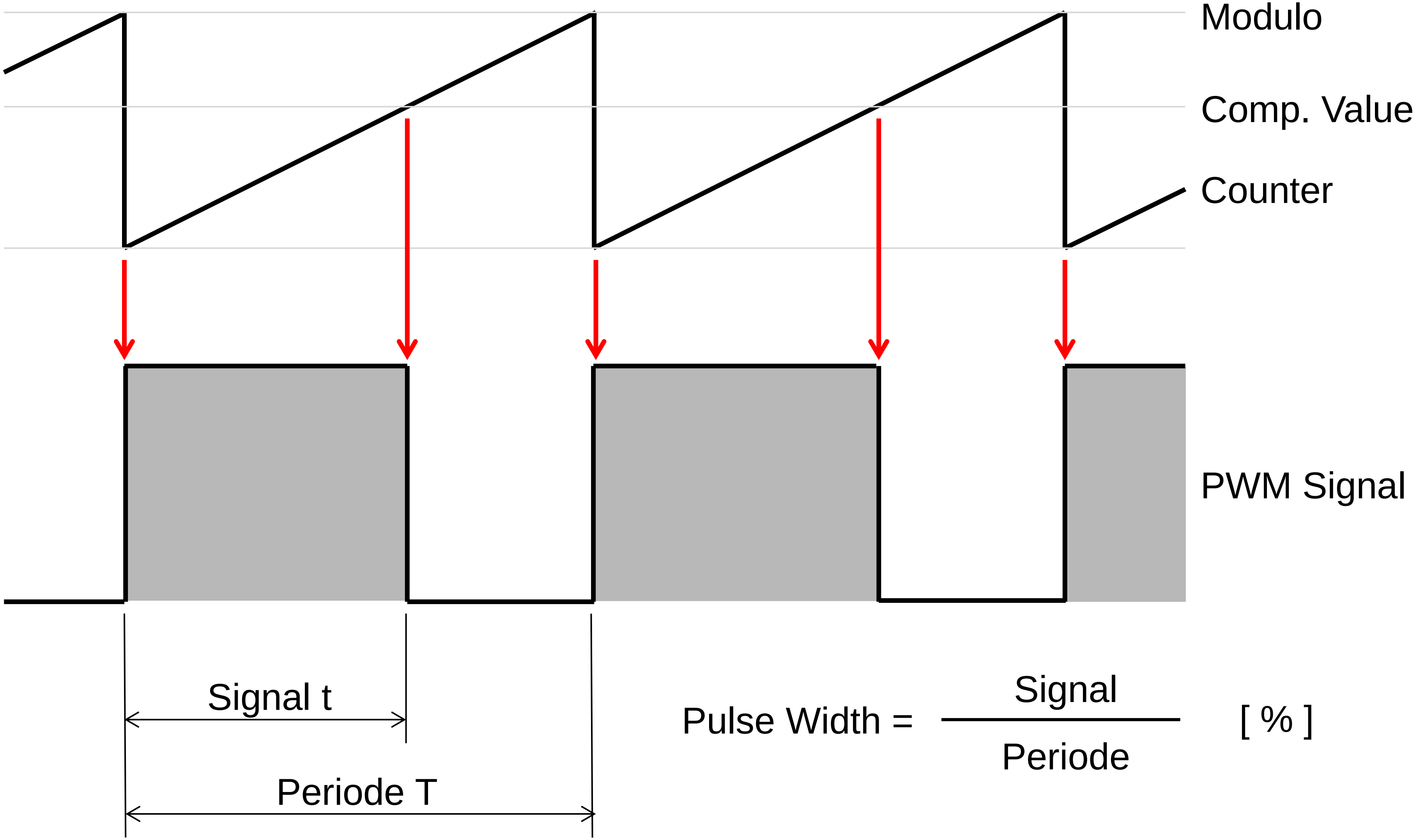




80 %

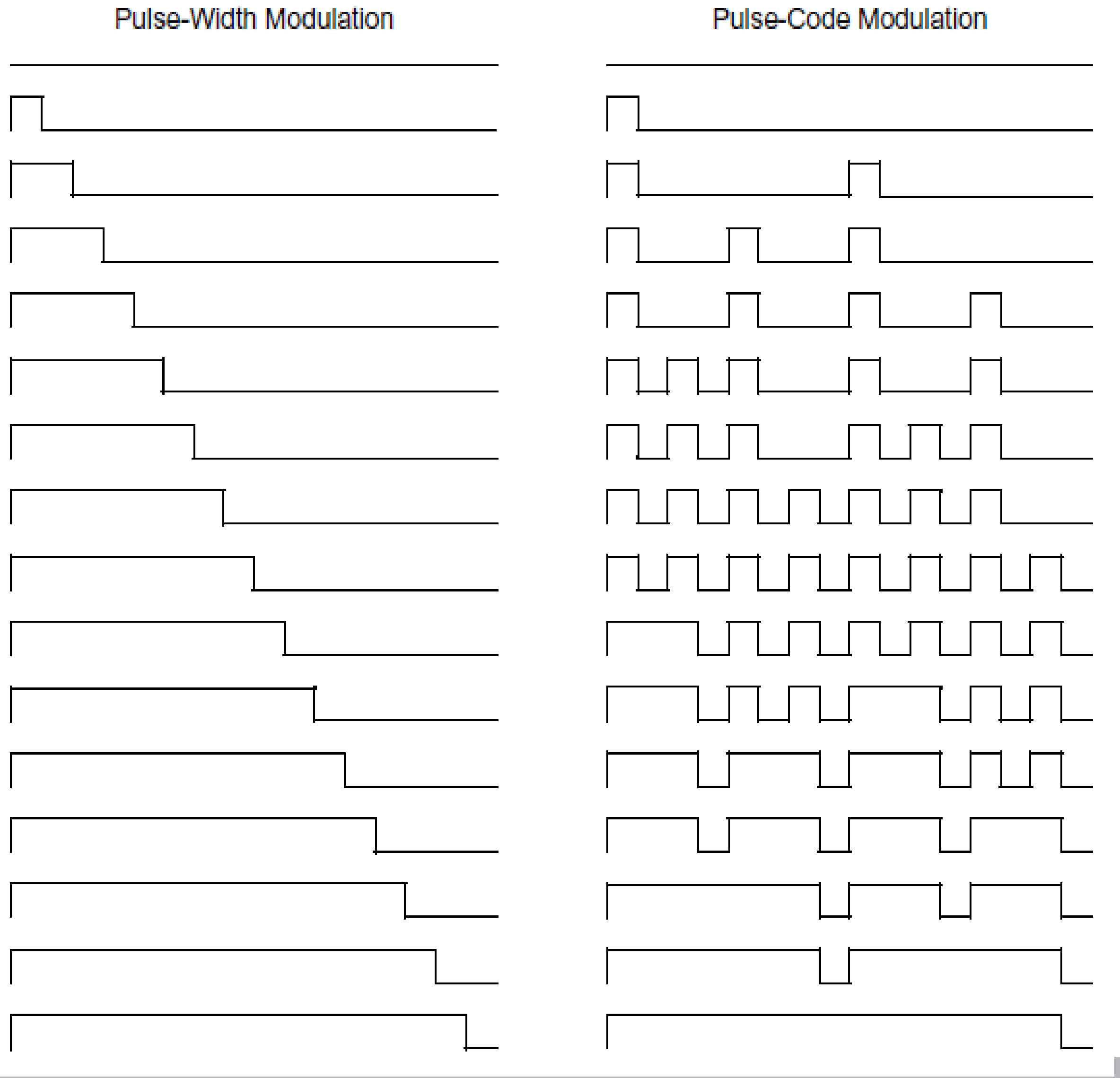


PWM Signal



$$\text{Pulse Width} = \frac{\text{Signal}}{\text{Periode}} \quad [\ \%\ ]$$

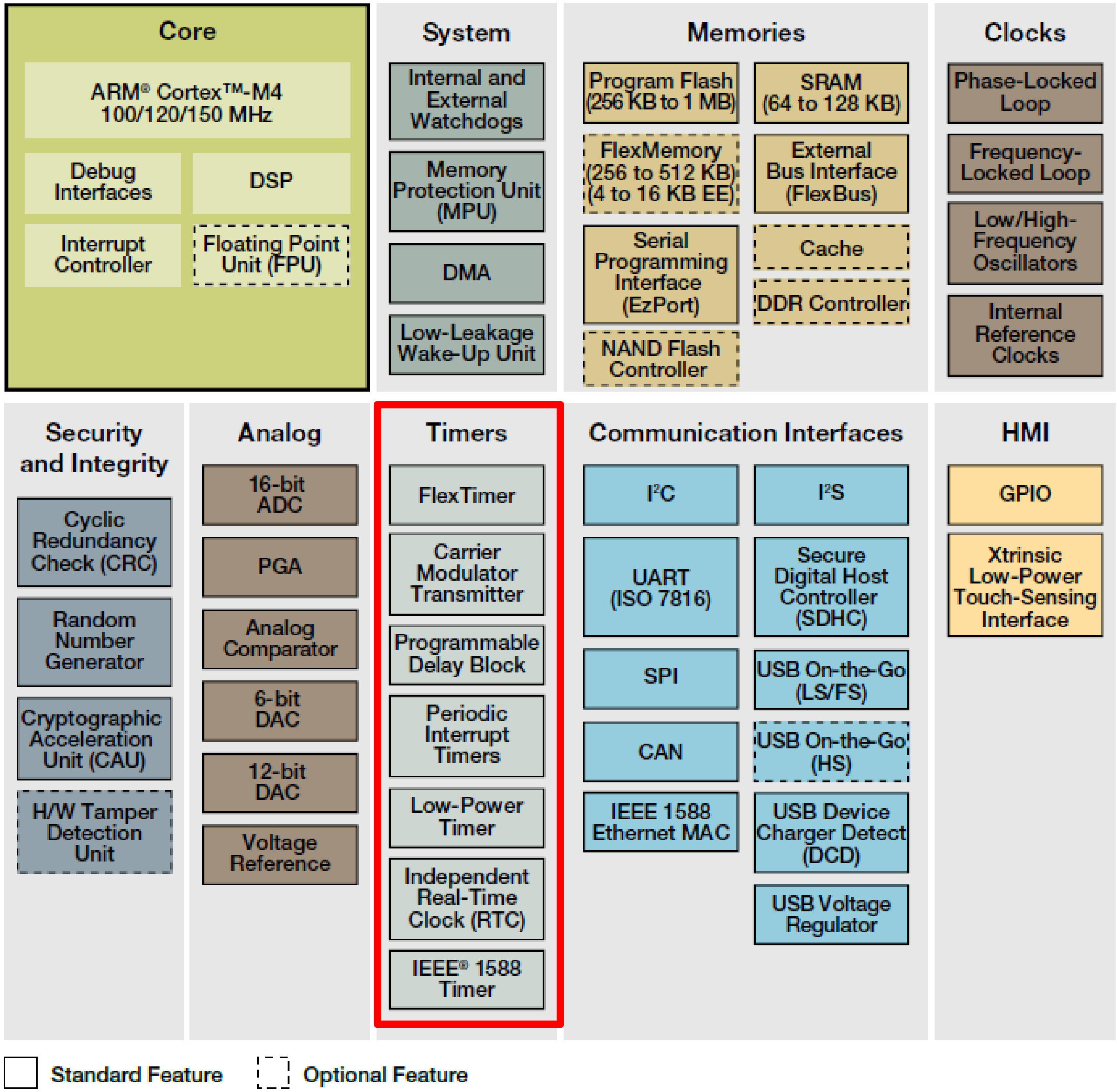
PWM and PCM



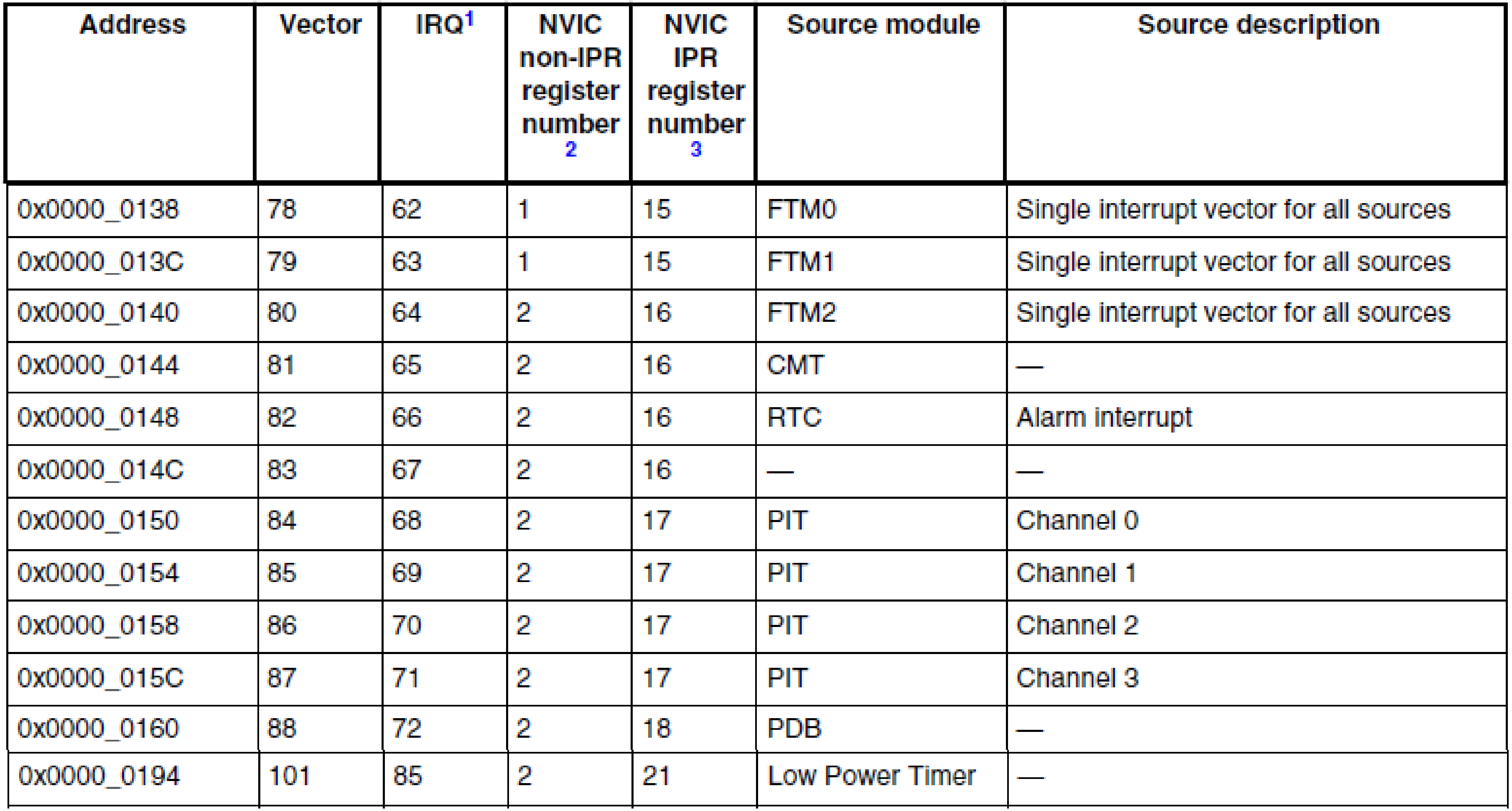
Timer on the Kinetis K60

- Programmable delay timer block (PDB)
- Flexible timer modules (FTM)
- Periodic interrupt timers (PIT)
- Low-power timer (LPTimer)
- Carrier modulator timer (CMT)
- Real-time clock (RTC)
- IEEE 1588 timers
- Watchdog timer (WDOG)

Timer on the Kinetis K60

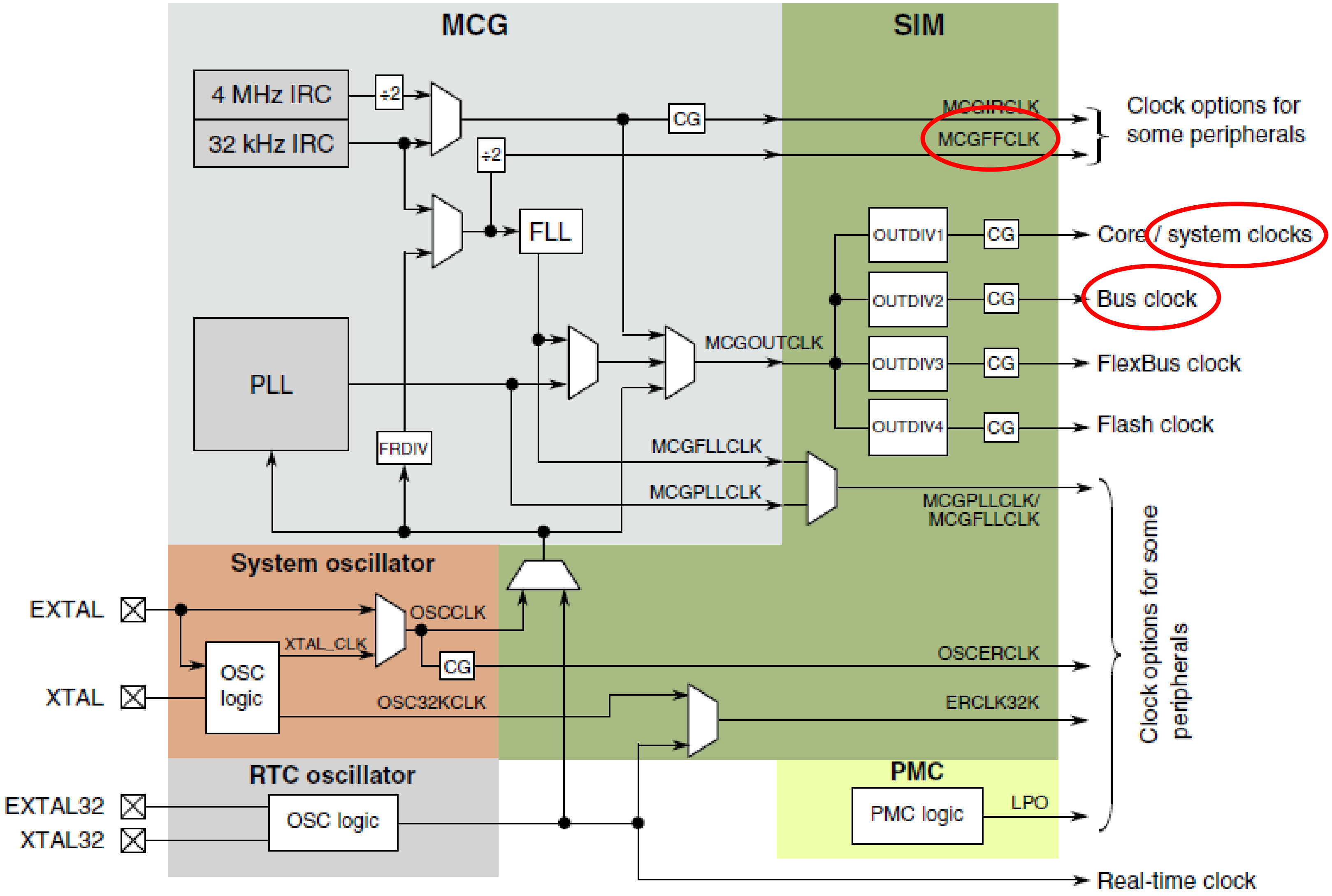

Core
ARM® Cortex™-M4 100/120/150 MHz
Debug Interfaces
DSP
Interrupt Controller
Floating Point Unit (FPU)
System
Internal and External Watchdogs
Memory Protection Unit (MPU)
DMA
Low-Leakage Wake-Up Unit
Memories
Program Flash (256 KB to 1 MB)
SRAM (64 to 128 KB)
FlexMemory (256 to 512 KB) (4 to 16 KB EE)
External Bus Interface (FlexBus)
Serial Programming Interface (EzPort)
Cache
DDR Controller
NAND Flash Controller
Clocks
Phase-Locked Loop
Frequency-Locked Loop
Low/High-Frequency Oscillators
Internal Reference Clocks
Security and Integrity
Cyclic Redundancy Check (CRC)
Random Number Generator
Cryptographic Acceleration Unit (CAU)
H/W Tamper Detection Unit
Analog
16-bit ADC
PGA
Analog Comparator
6-bit DAC
12-bit DAC
Voltage Reference
Timers
FlexTimer
Carrier Modulator Transmitter
Programmable Delay Block
Periodic Interrupt Timers
Low-Power Timer
Independent Real-Time Clock (RTC)
IEEE® 1588 Timer
Communication Interfaces
I²C
I²S
UART (ISO 7816)
Secure Digital Host Controller (SDHC)
SPI
USB On-the-Go (LS/FS)
CAN
USB On-the-Go (HS)
IEEE 1588 Ethernet MAC
USB Device Charger Detect (DCD)
USB Voltage Regulator
HMI
GPIO
Xtrinsic Low-Power Touch-Sensing Interface
Standard Feature
Optional Feature

Timer Interrupts on Kinetis K60



Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0138	78	62	1	15	FTM0	Single interrupt vector for all sources
0x0000_013C	79	63	1	15	FTM1	Single interrupt vector for all sources
0x0000_0140	80	64	2	16	FTM2	Single interrupt vector for all sources
0x0000_0144	81	65	2	16	CMT	—
0x0000_0148	82	66	2	16	RTC	Alarm interrupt
0x0000_014C	83	67	2	16	—	—
0x0000_0150	84	68	2	17	PIT	Channel 0
0x0000_0154	85	69	2	17	PIT	Channel 1
0x0000_0158	86	70	2	17	PIT	Channel 2
0x0000_015C	87	71	2	17	PIT	Channel 3
0x0000_0160	88	72	2	18	PDB	—
0x0000_0194	101	85	2	21	Low Power Timer	—

Clock Distribution of Kinetis K60

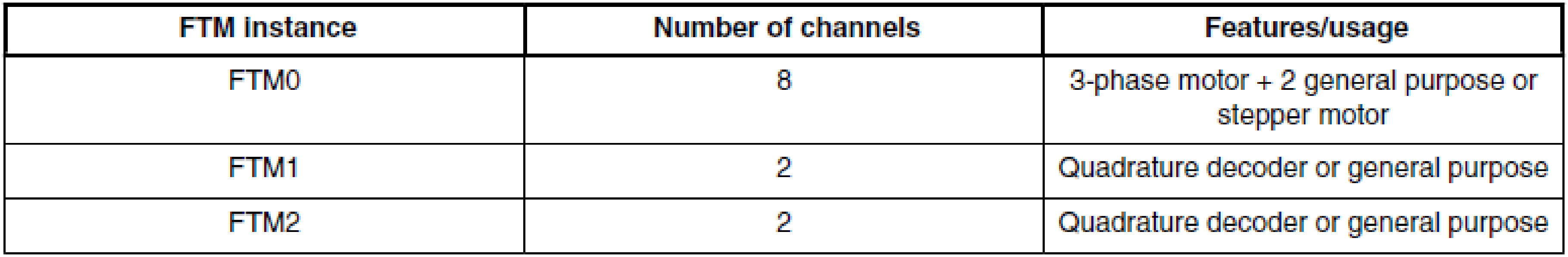


CG — Clock gate

FlexTimer (FTM)

The FlexTimer Module (FTM) is a two to eight channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications.
The FTM **time reference is a 16-bit counter** that can be used as an unsigned or signed counter.

The Kinetis K60 device contains three FlexTimer modules.



FTM instance	Number of channels	Features/usage
FTM0	8	3-phase motor + 2 general purpose or stepper motor
FTM1	2	Quadrature decoder or general purpose
FTM2	2	Quadrature decoder or general purpose

FTM features

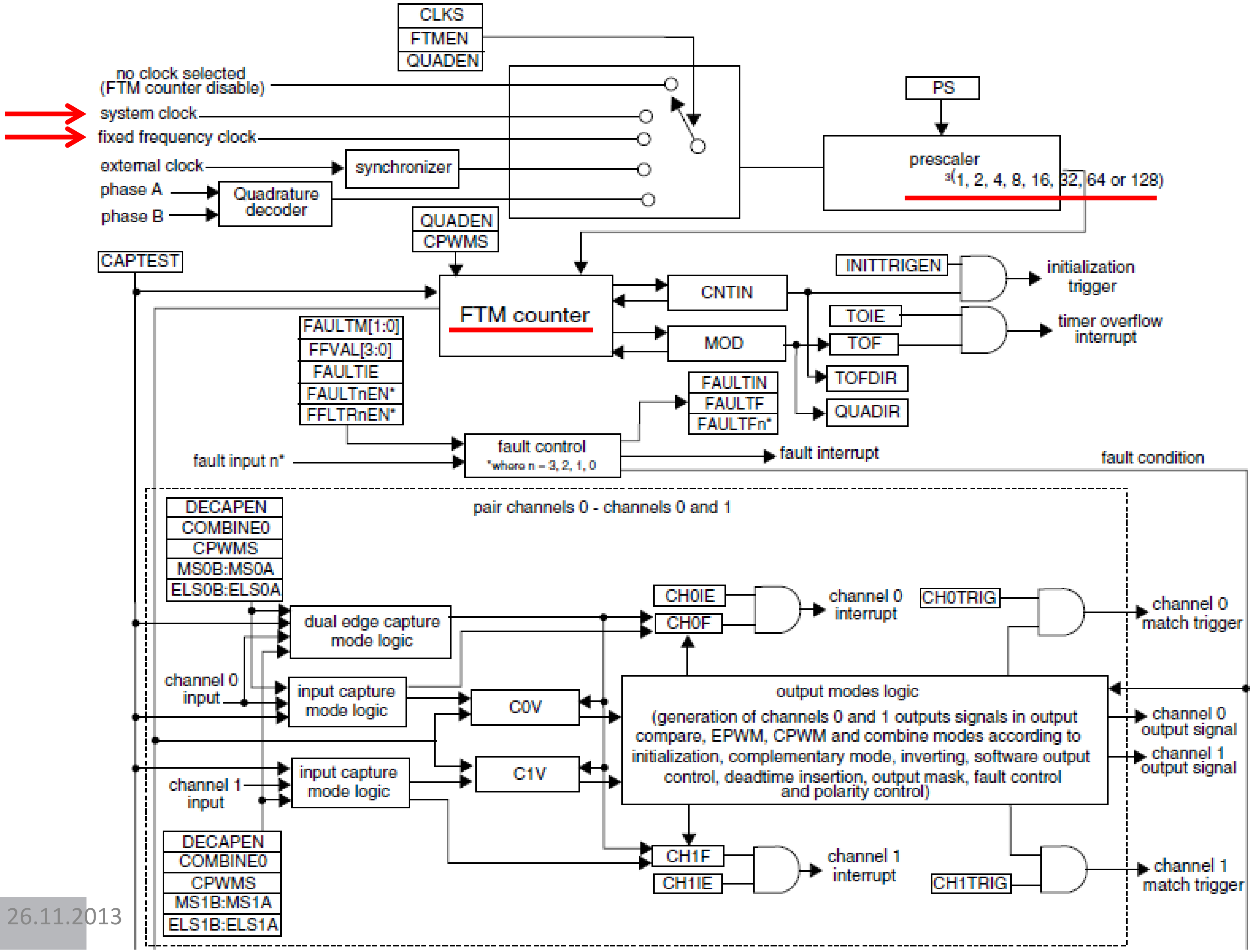
The FTM features include:

- FTM source clock is selectable
  - Source clock can be the system clock, the fixed frequency clock, or an External clock
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the system clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source

- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- FTM has a **16-bit counter**
  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down

- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode

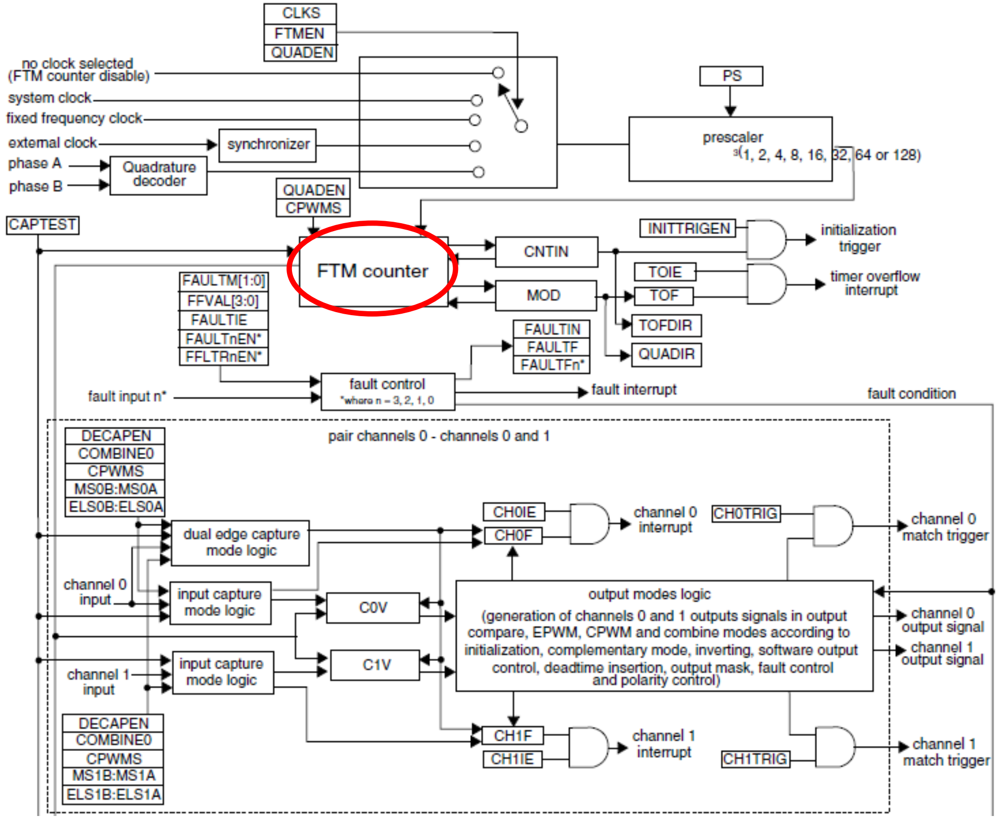
Flex Timer Block Diagram



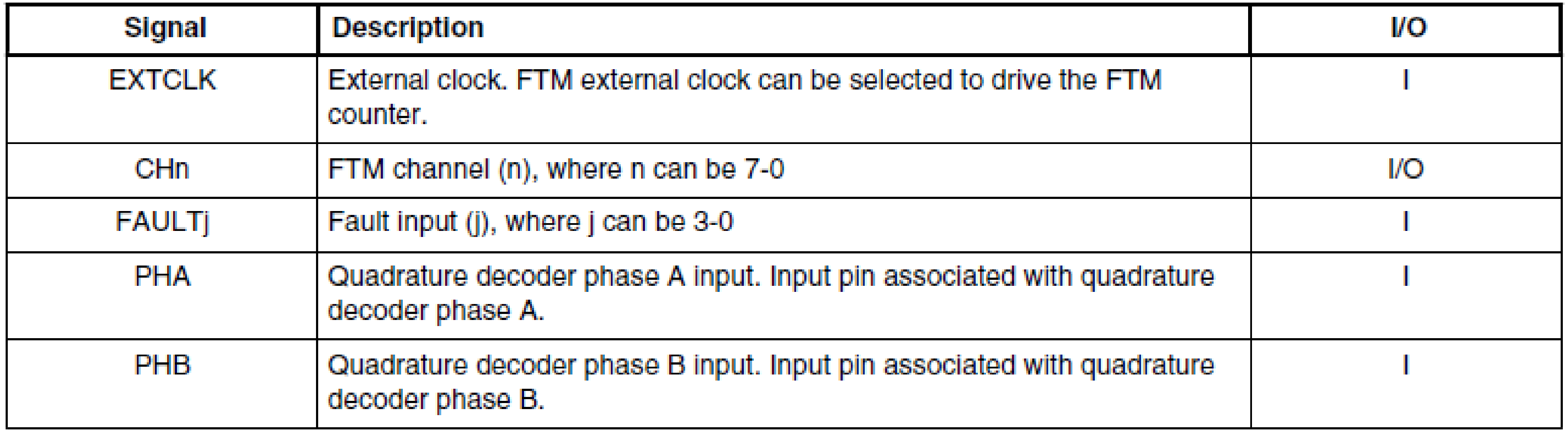
FTM Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler. The FTM counter has these modes of operation:

- Up Counting
- Up-Down Counting
- Quadrature Decoder Mode



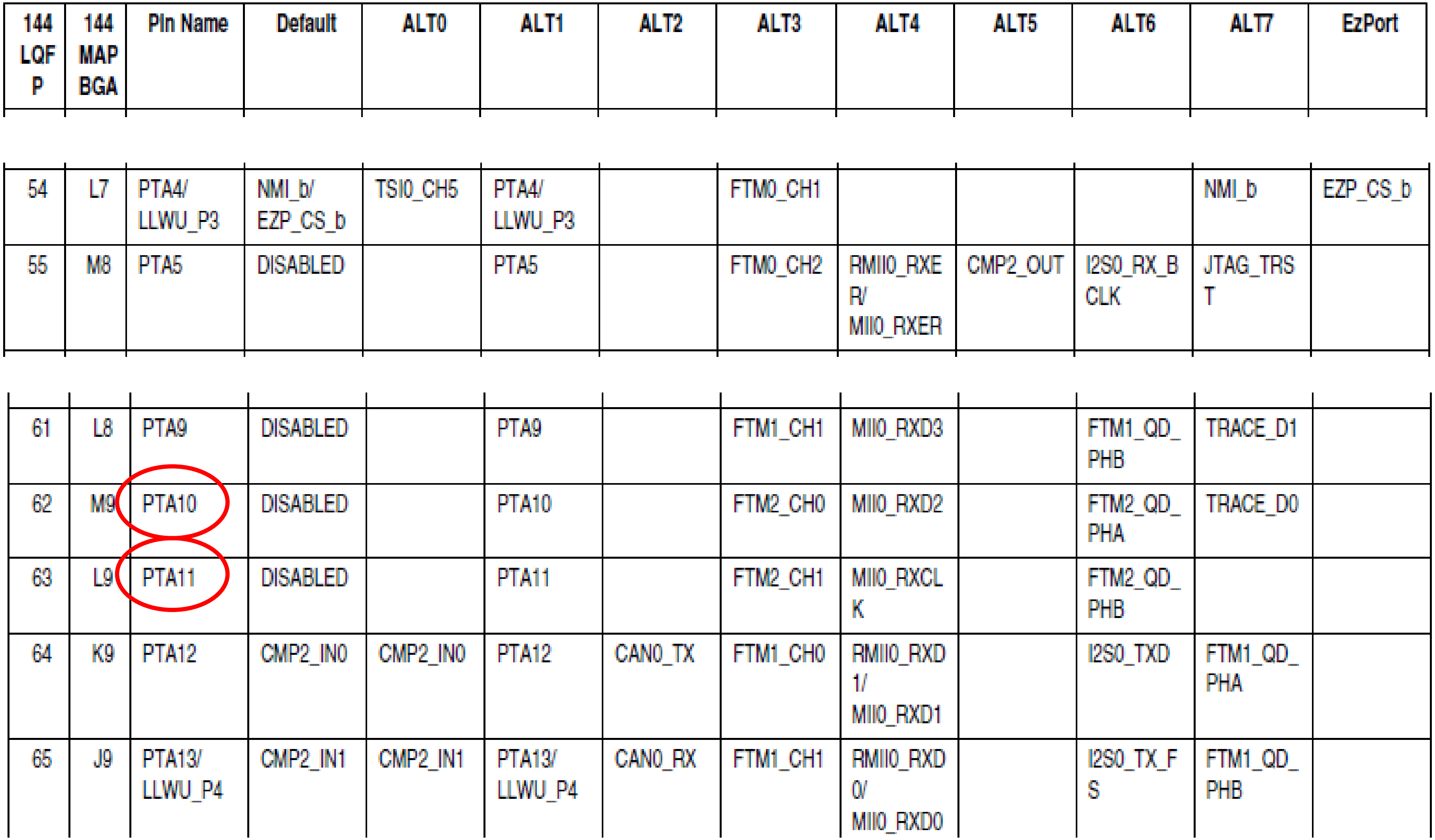
FTM external Signals



Signal	Description	I/O
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I
CHn	FTM channel (n), where n can be 7-0	I/O
FAULTj	Fault input (j), where j can be 3-0	I
PHA	Quadrature decoder phase A input. Input pin associated with quadrature decoder phase A.	I
PHB	Quadrature decoder phase B input. Input pin associated with quadrature decoder phase B.	I

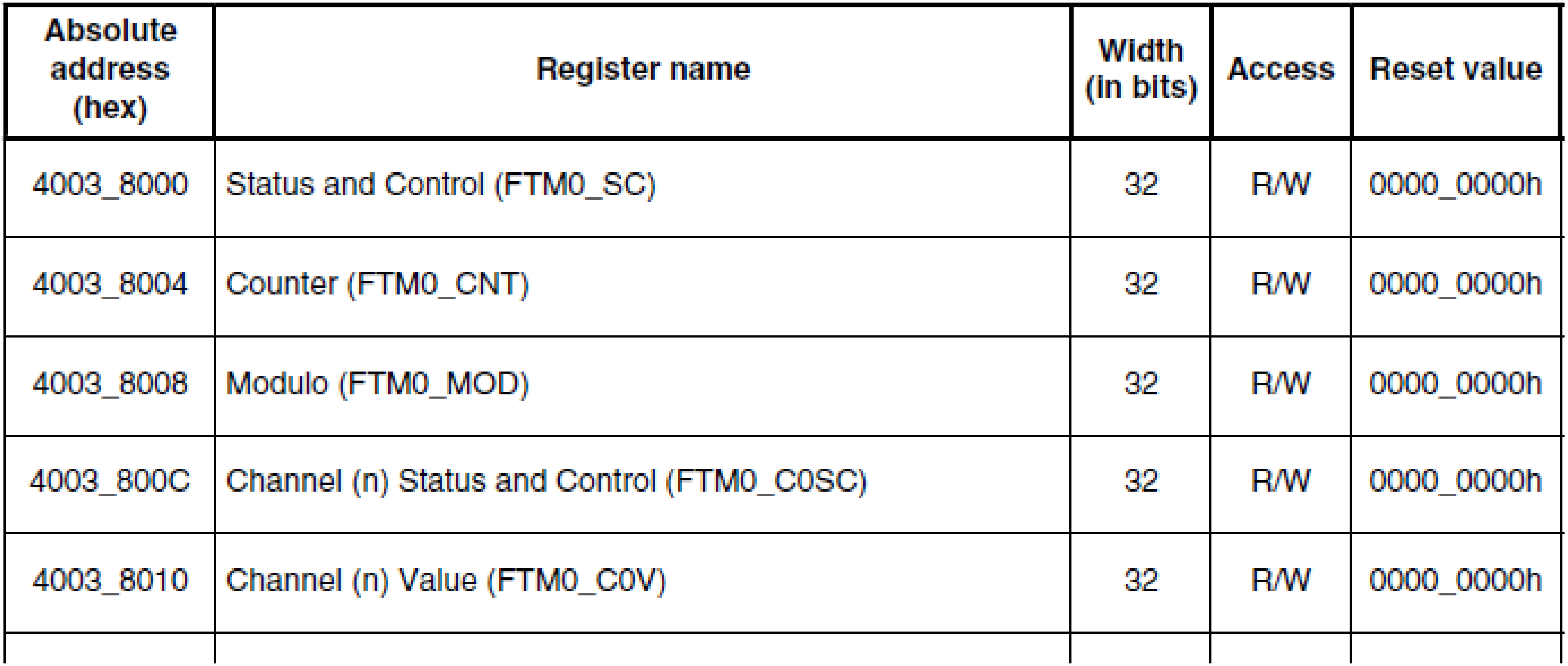
Each FTM channel can be configured to operate either as input or output. The Direction associated with each channel, input or output, is selected according to the mode assigned for that channel.

Pins of the FlexTimer Moduls



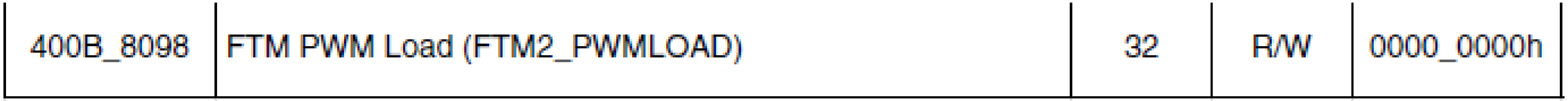
144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
54	L7	PTA4/LLWU_P3	NMI_b/EZP_CS_b	TSI0_CH5	PTA4/LLWU_P3		FTM0_CH1				NMI_b	EZP_CS_b
55	M8	PTA5	DISABLED		PTA5		FTM0_CH2	RMII0_RXER/MII0_RXER	CMP2_OUT	I2S0_RX_BCLK	JTAG_TRST	
61	L8	PTA9	DISABLED		PTA9		FTM1_CH1	MII0_RXD3		FTM1_QD_PHB	TRACE_D1	
62	M9	PTA10	DISABLED		PTA10		FTM2_CH0	MII0_RXD2		FTM2_QD_PHA	TRACE_D0	
63	L9	PTA11	DISABLED		PTA11		FTM2_CH1	MII0_RXCLK		FTM2_QD_PHB		
64	K9	PTA12	CMP2_IN0	CMP2_IN0	PTA12	CAN0_TX	FTM1_CH0	RMII0_RXD1/MII0_RXD1		I2S0_TXD	FTM1_QD_PHA	
65	J9	PTA13/LLWU_P4	CMP2_IN1	CMP2_IN1	PTA13/LLWU_P4	CAN0_RX	FTM1_CH1	RMII0_RXD0/MII0_RXD0		I2S0_TX_FS	FTM1_QD_PHB	

FlexTimer Register Memory Map



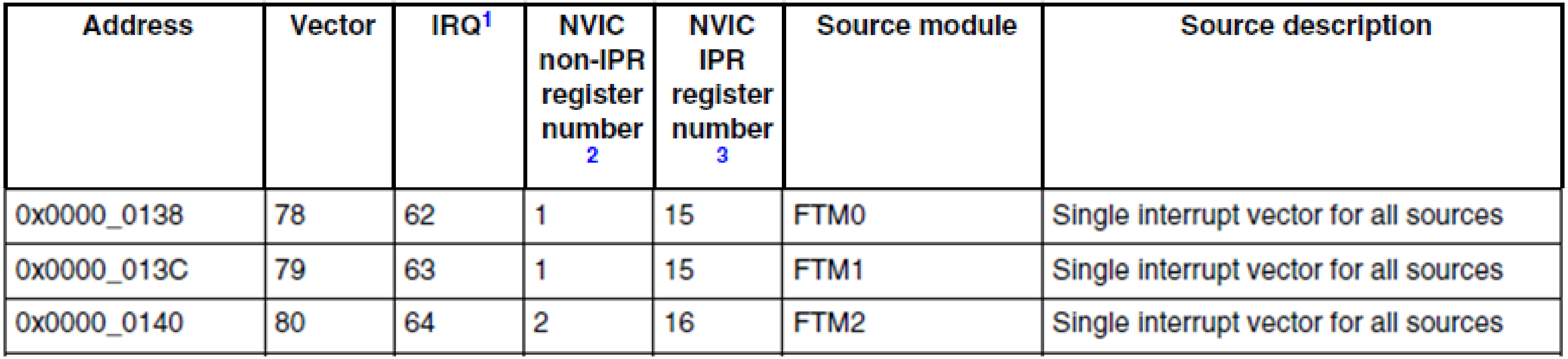
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4003_8000	Status and Control (FTM0_SC)	32	R/W	0000_0000h
4003_8004	Counter (FTM0_CNT)	32	R/W	0000_0000h
4003_8008	Modulo (FTM0_MOD)	32	R/W	0000_0000h
4003_800C	Channel (n) Status and Control (FTM0_C0SC)	32	R/W	0000_0000h
4003_8010	Channel (n) Value (FTM0_C0V)	32	R/W	0000_0000h

... alltogether 117 registers ...



400B_8098	FTM PWM Load (FTM2_PWMLOAD)	32	R/W	0000_0000h

FlexTimer Interrupts



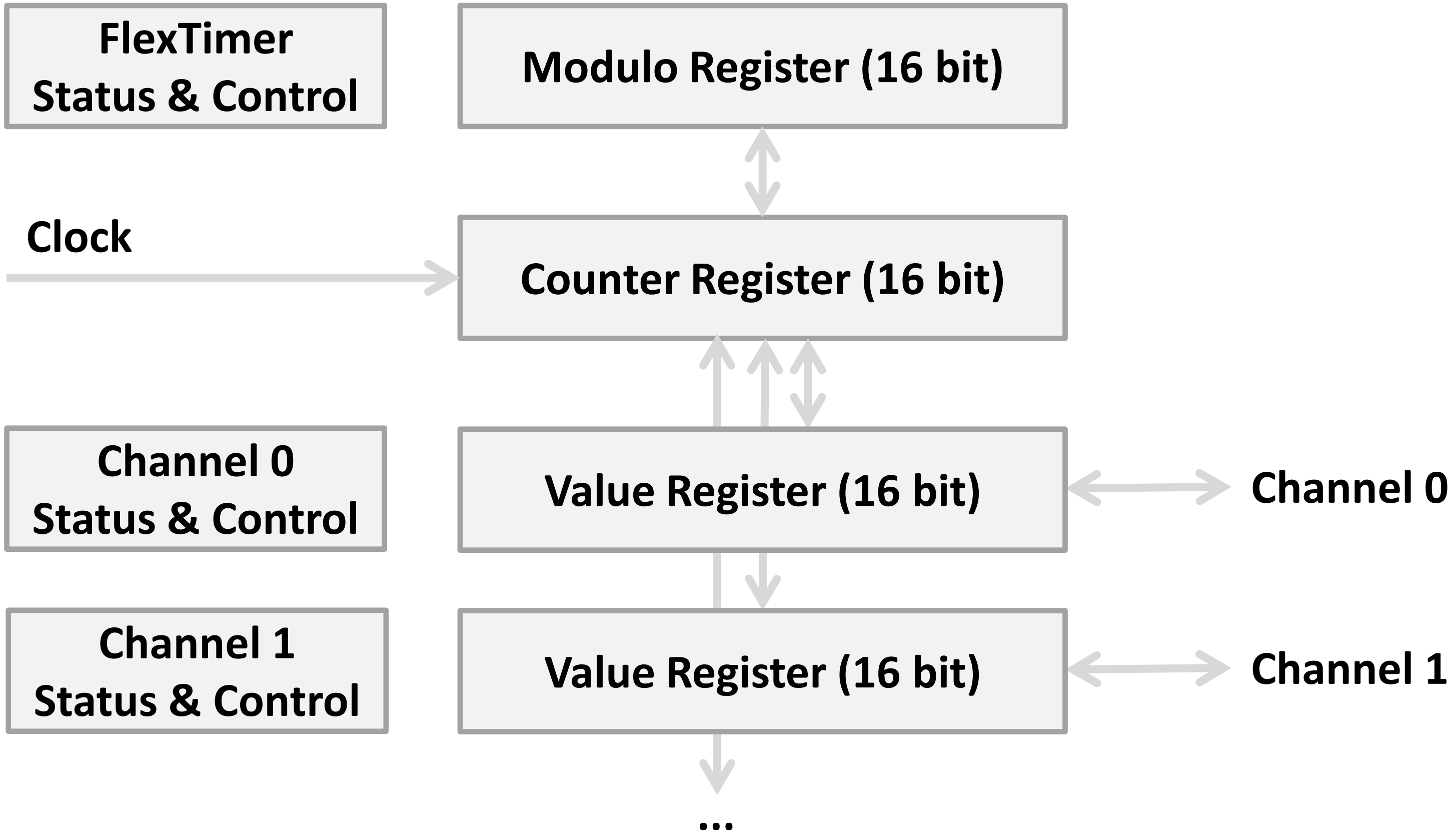
Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0138	78	62	1	15	FTM0	Single interrupt vector for all sources
0x0000_013C	79	63	1	15	FTM1	Single interrupt vector for all sources
0x0000_0140	80	64	2	16	FTM2	Single interrupt vector for all sources

Timer Overflow Interrupt
The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

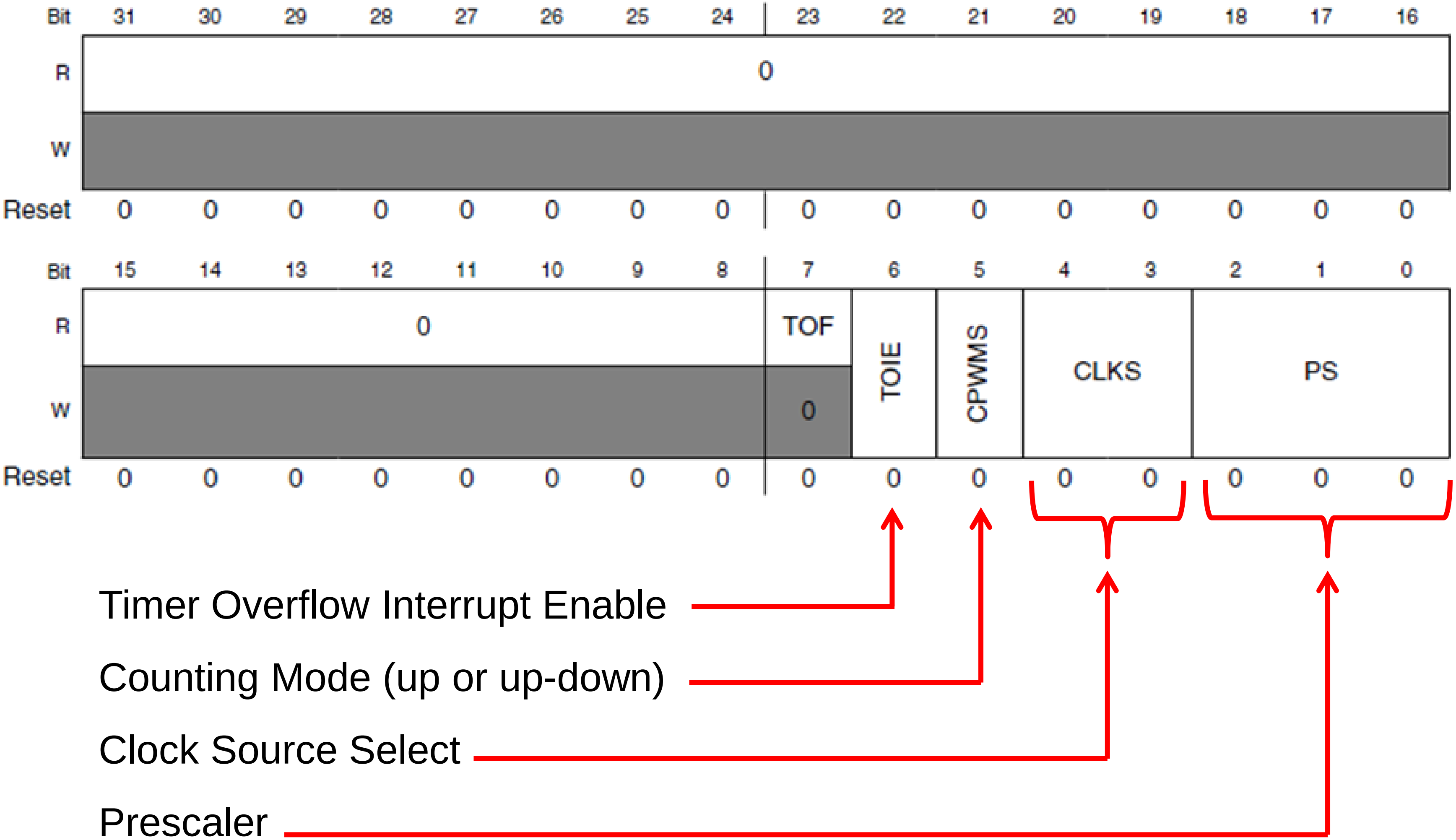
Channel (n) Interrupt
The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

Fault Interrupt
The fault interrupt is generated when (FAULTIE = 1) and (FAULTF = 1).

Important FlexTimer Registers



FlexTimer Status and Control (FTMx_SC)



Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TOF	TOIE	CPWMS	CLKS		PS		
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

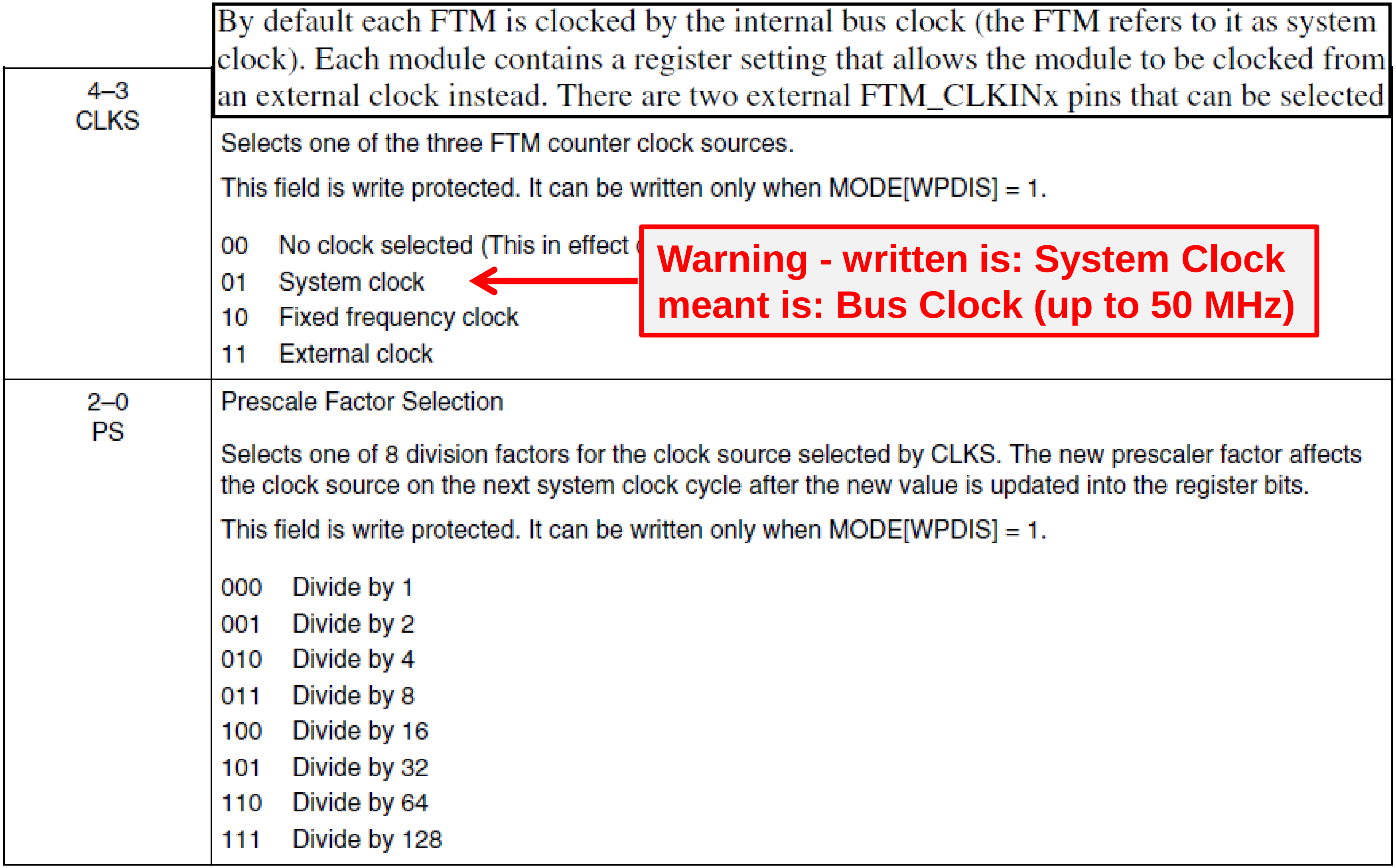
Timer Overflow Interrupt Enable

Counting Mode (up or up-down)

Clock Source Select

Prescaler

FlexTimer Status and Control (FTMx_SC)

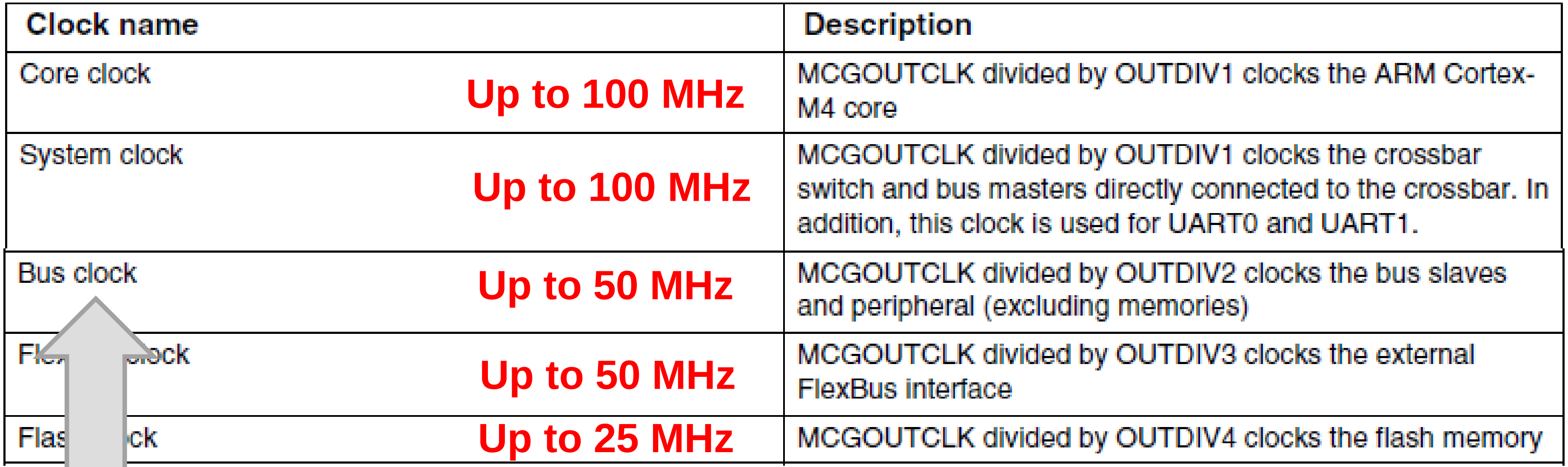


By default each FTM is clocked by the internal bus clock (the FTM refers to it as system clock). Each module contains a register setting that allows the module to be clocked from an external clock instead. There are two external FTM_CLKINx pins that can be selected

4–3 CLKS	Selects one of the three FTM counter clock sources. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 No clock selected (This in effect 01 System clock 10 Fixed frequency clock 11 External clock
2–0 PS	Prescale Factor Selection Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next system clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128

Warning - written is: System Clock meant is: Bus Clock (up to 50 MHz)

Recapitulation:
The most important frequencies of the MK60N512VMD10



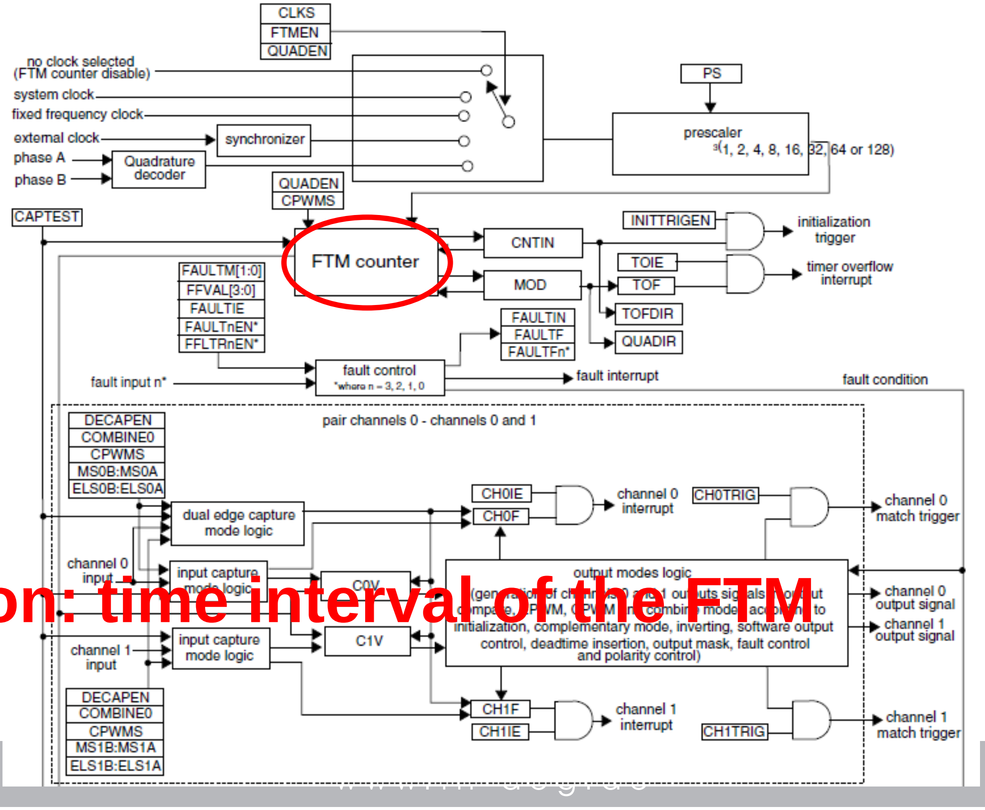
Clock name	Description
Core clock Up to 100 MHz	MCGOUTCLK divided by OUTDIV1 clocks the ARM Cortex-M4 core
System clock Up to 100 MHz	MCGOUTCLK divided by OUTDIV1 clocks the crossbar switch and bus masters directly connected to the crossbar. In addition, this clock is used for UART0 and UART1.
Bus clock Up to 50 MHz	MCGOUTCLK divided by OUTDIV2 clocks the bus slaves and peripheral (excluding memories)
FlexBus clock Up to 50 MHz	MCGOUTCLK divided by OUTDIV3 clocks the external FlexBus interface
Flash clock Up to 25 MHz	MCGOUTCLK divided by OUTDIV4 clocks the flash memory

Bus Clock up to 50 MHz

FTM Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler. The FTM counter has these modes of operation:

- Up Counting
- Up-Down Counting
- Quadrature Decoder Mode



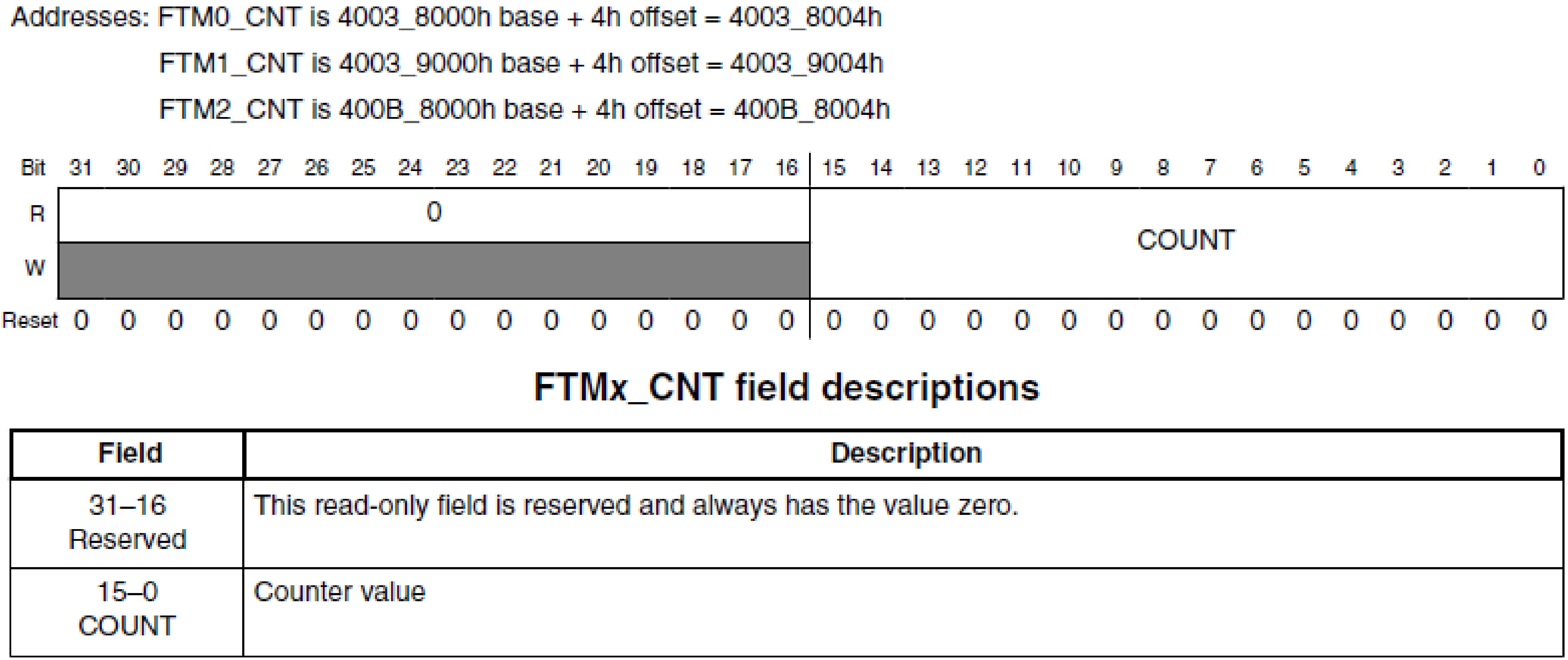
Calculation: time interval of the FTM

Set the Timer Interrupt to 10 ms

```
// Set MCGOUTCLK to 100 MHz
    MCG_C1 |= MCG_C1_CLKS(2);      // Use external clock source
    MCG_C5 = MCG_C5_PRDIV(24);     // Divide clock source by 25
    MCG_C6 = MCG_C6_PLLS_MASK | 0x1A; // PLL factor 50
    MCG_C1 &= ~MCG_C1_CLKS_MASK;   // Switch to PLL output
    SIM_CLKDIV1 = 0x01130000;      // Bus Clock = 50 MHz

// FlexTimer0 Clock Source
    FTM0_SC |= FTM_SC_CLKS(1);     // Bus Clock = 50 MHz

// FlexTimer0 Clock Prescaler
    FTM0_SC |= FTM_SC_PS(5);       // Prescaler = 32

// FlexTimer0 Modulo Register
    FTM0_MOD = 15625;              // Total Period 10 ms

// FlexTimers Enable Flextimer 0 Overflow Interrupt 10 ms
    FTM0_SC |= FTM_SC_TOIE_MASK;   // Overflow Interrupt 10 ms
```

FlexTimer Counter Register (FTMx_CNT)

The CNT register contains the FTM counter value.
Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value (CNTIN).



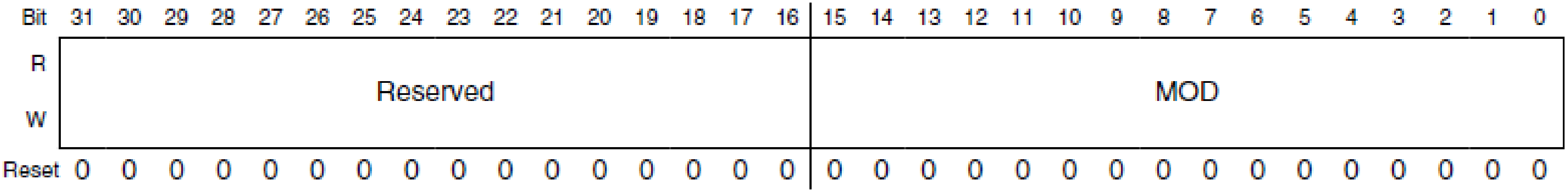
Addresses: FTM0_CNT is 4003_8000h base + 4h offset = 4003_8004h
FTM1_CNT is 4003_9000h base + 4h offset = 4003_9004h
FTM2_CNT is 400B_8000h base + 4h offset = 400B_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W	▨																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_CNT field descriptions

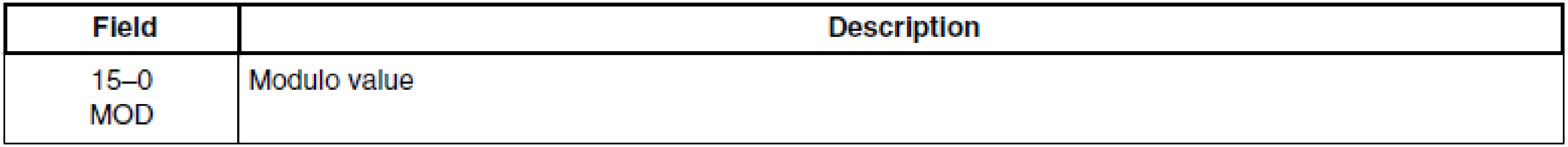
Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 COUNT	Counter value

FlexTimer Modulo Register (FTMx_MOD)



Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																MOD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_MOD field descriptions



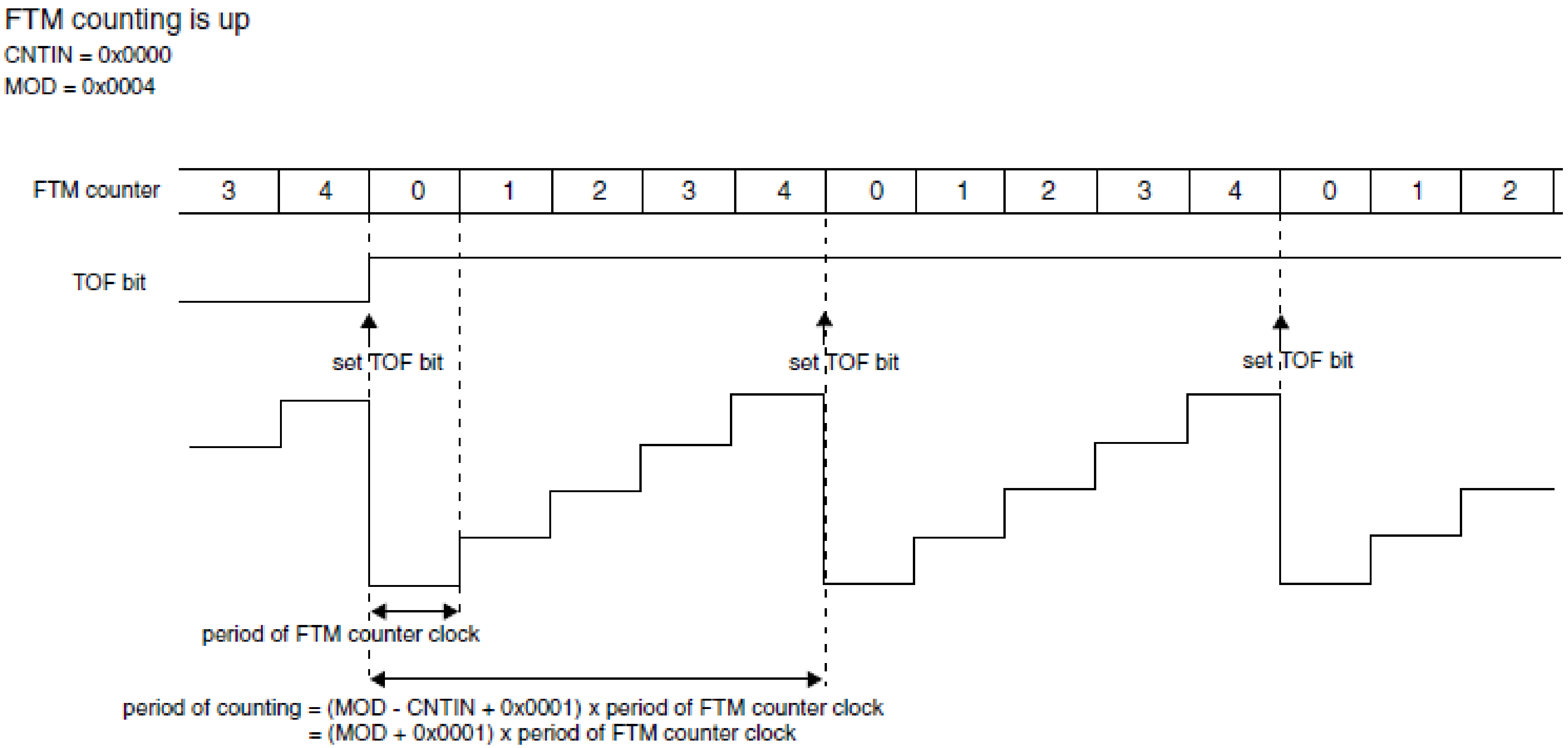
Field	Description
15–0 MOD	Modulo value

The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock, and the next value of FTM counter depends on the selected counting method:

- up counting
- up-down counting
- quadrature mode

Up Counting

Up counting is selected when (QUADEN = 0) and (CPWMS = 0).
CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.



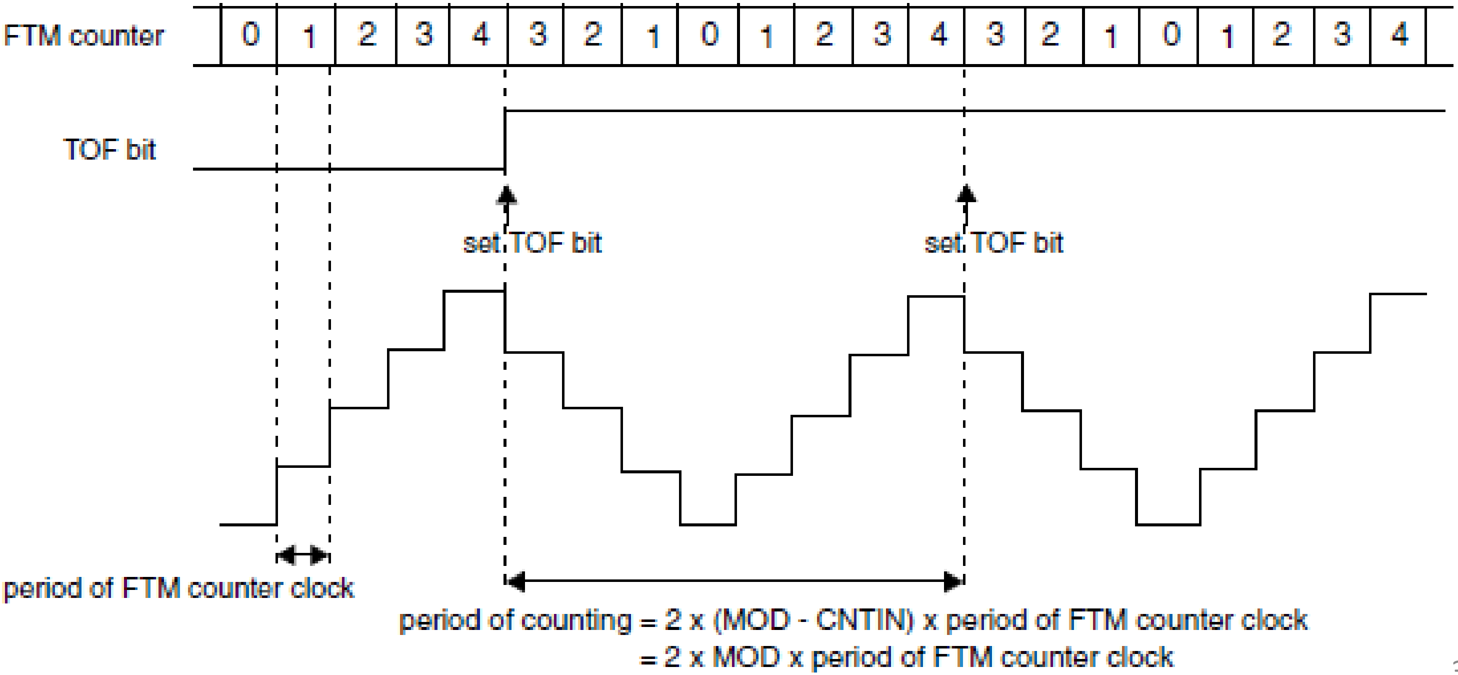
FTM counting is up
CNTIN = 0x0000
MOD = 0x0004

period of counting = (MOD - CNTIN + 0x0001) x period of FTM counter clock
= (MOD + 0x0001) x period of FTM counter clock

Up-Down Counting

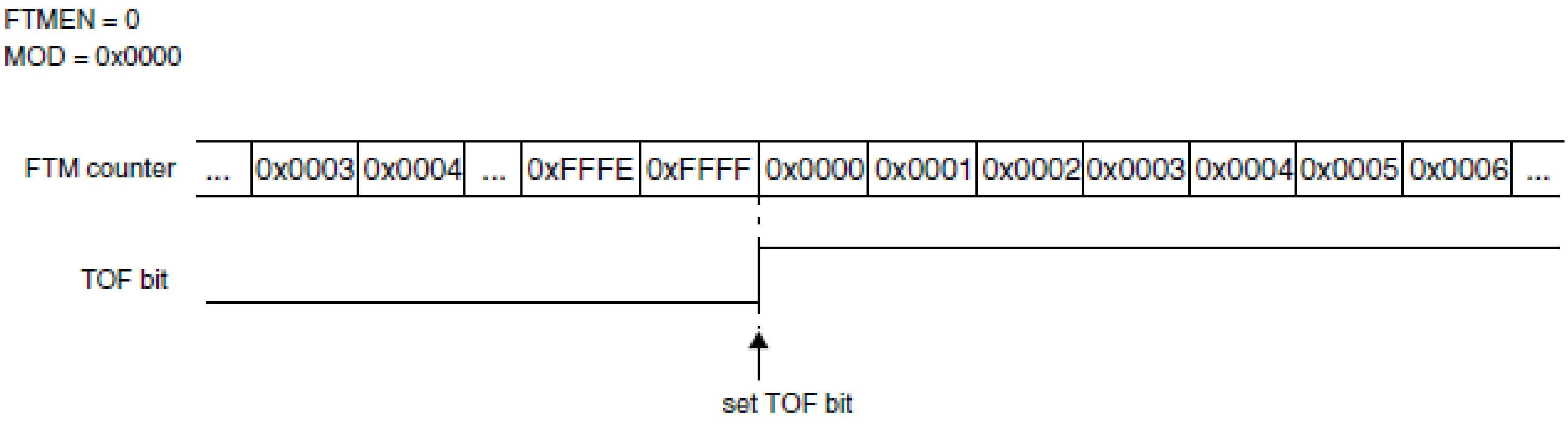
Up-down counting is selected when (QUADEN= 0) and (CPWMS = 1).
CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

FTM counting is up-down
CNTIN = 0x0000
MOD = 0x0004


FTM counter
0 1 2 3 4 3 2 1 0 1 2 3 4 3 2 1 0 1 2 3 4
TOF bit
set TOF bit
set TOF bit
period of FTM counter clock
period of counting = 2 x (MOD - CNTIN) x period of FTM counter clock
= 2 x MOD x period of FTM counter clock

Free Running Counter

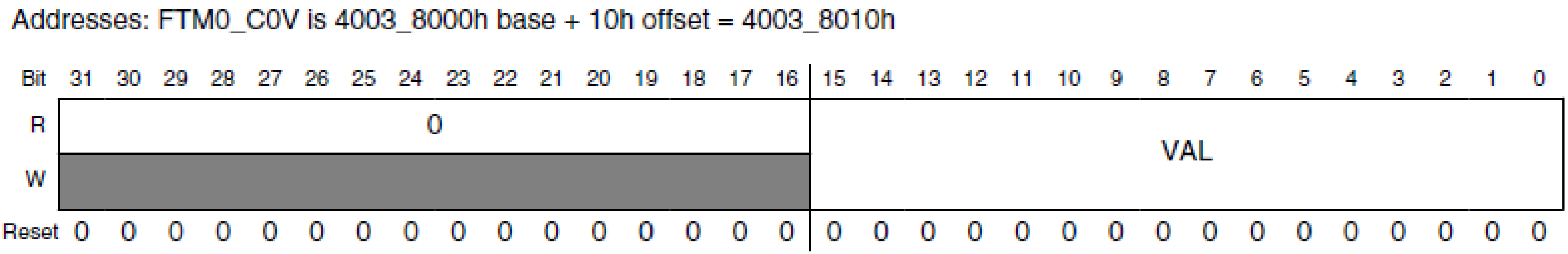
If (FTMEN = 0) and (MOD = 0x0000 or MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000.



Channel (n) Value Register (FTMx_CV)

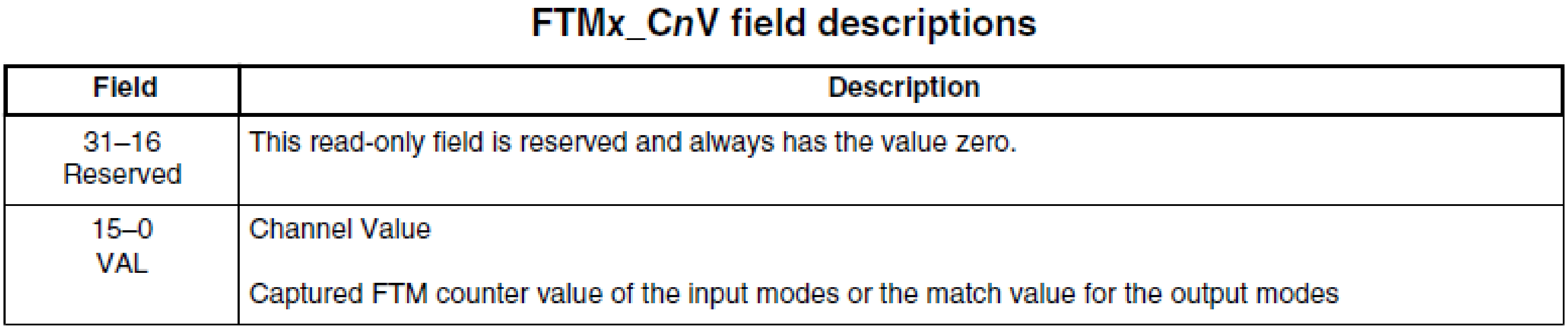
Sometimes also called Value Compare Register!

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.



Addresses: FTM0_C0V is 4003_8000h base + 10h offset = 4003_8010h

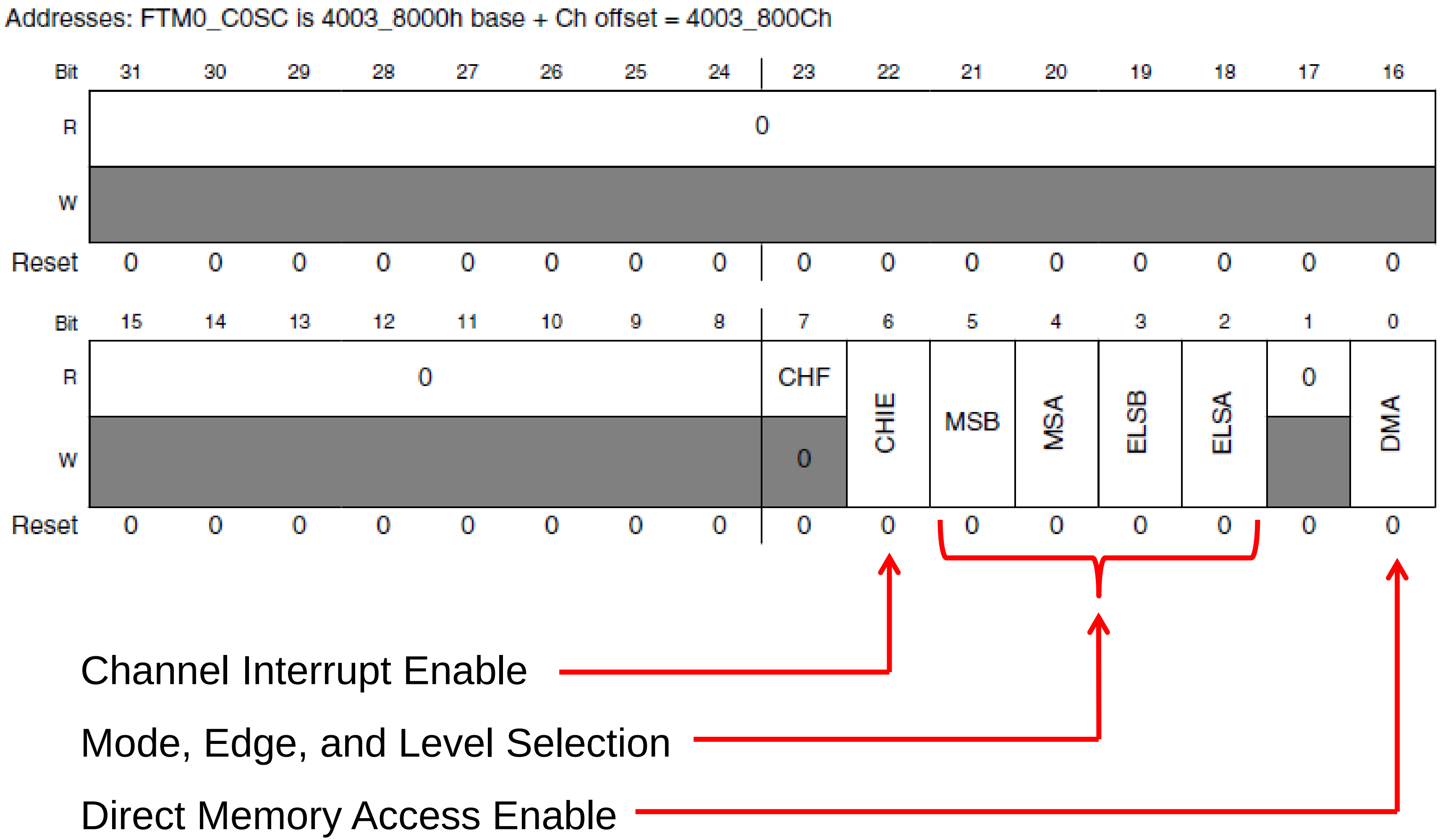
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																VAL															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



FTMx_CnV field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 VAL	Channel Value Captured FTM counter value of the input modes or the match value for the output modes

Channel (n) Status and Control (FTMx_CSC)



Addresses: FTM0_C0SC is 4003_8000h base + Ch offset = 4003_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

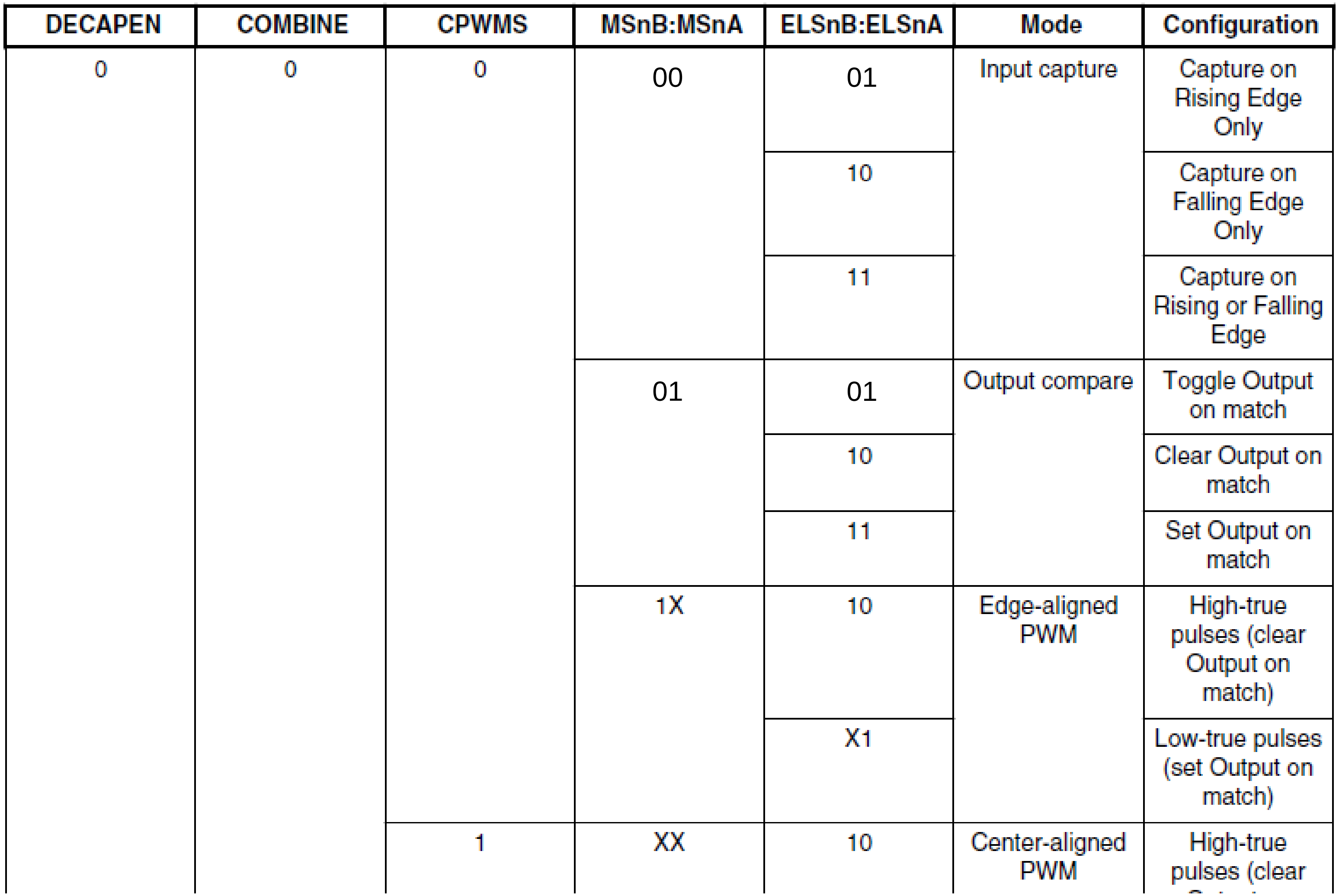
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CHF	CHIE	MSB	MSA	ELSB	ELSA	0	DMA
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Channel Interrupt Enable

Mode, Edge, and Level Selection

Direct Memory Access Enable

Channel (n) Mode, Edge, and Level Selection

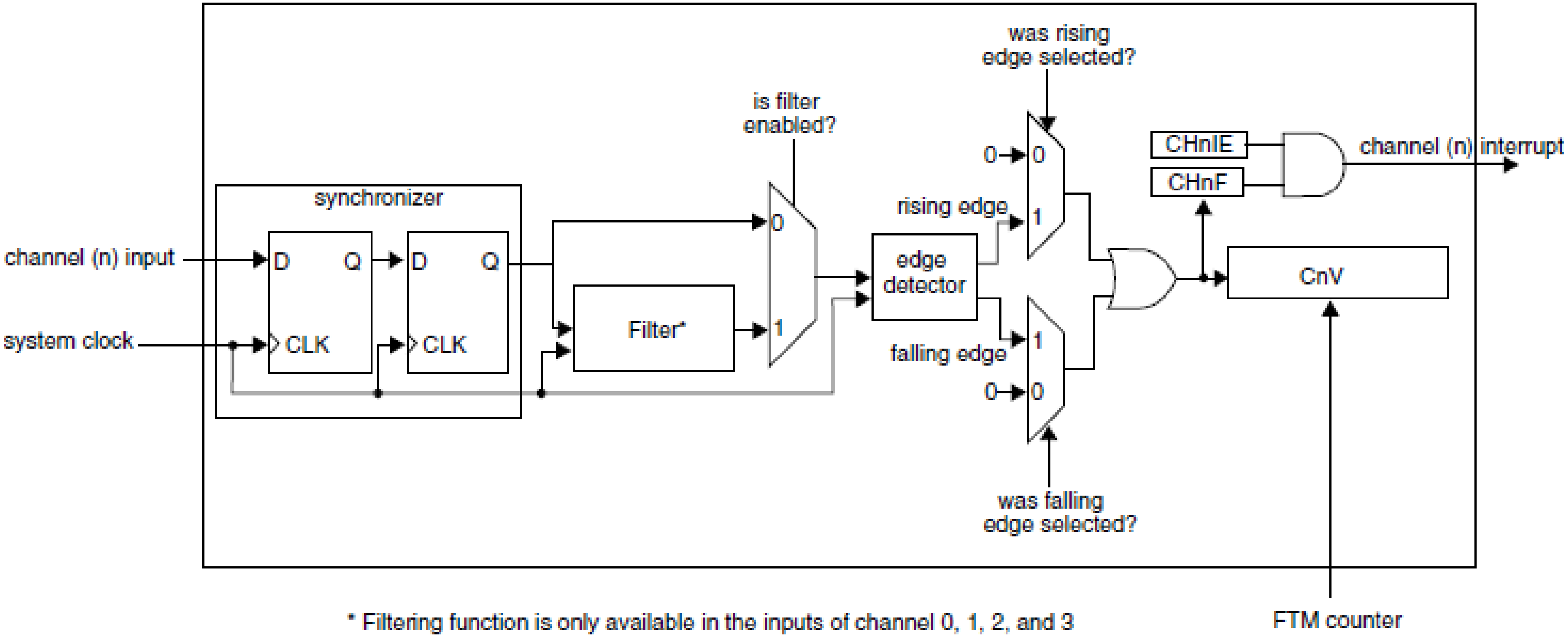


DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
0	0	0	00	01	Input capture	Capture on Rising Edge Only
				10		Capture on Falling Edge Only
				11		Capture on Rising or Falling Edge
			01	01	Output compare	Toggle Output on match
				10		Clear Output on match
				11		Set Output on match
			1X	10	Edge-aligned PWM	High-true pulses (clear Output on match)
				X1		Low-true pulses (set Output on match)
		1	XX	10	Center-aligned PWM	High-true pulses (clear

Input Capture Mode

The input capture mode is selected when (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

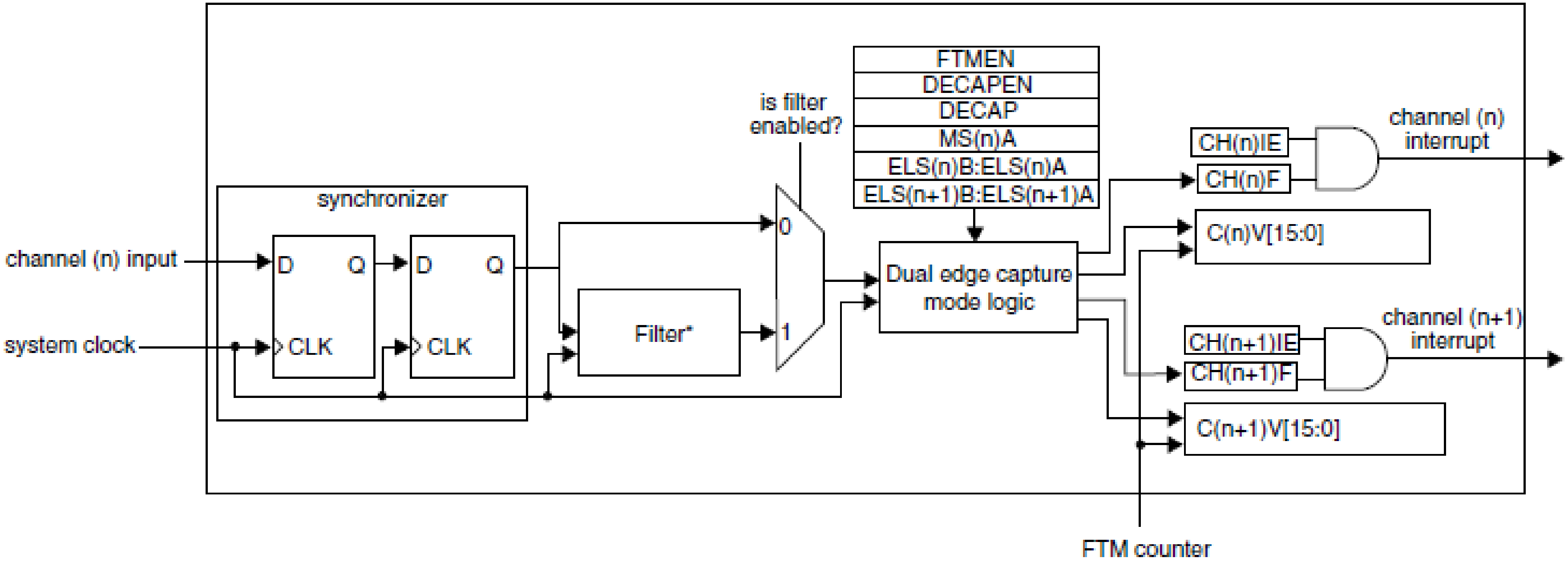
When a selected edge occurs on the channel input, the current value of the FTM Counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1



* Filtering function is only available in the inputs of channel 0, 1, 2, and 3

Dual Edge Capture Mode

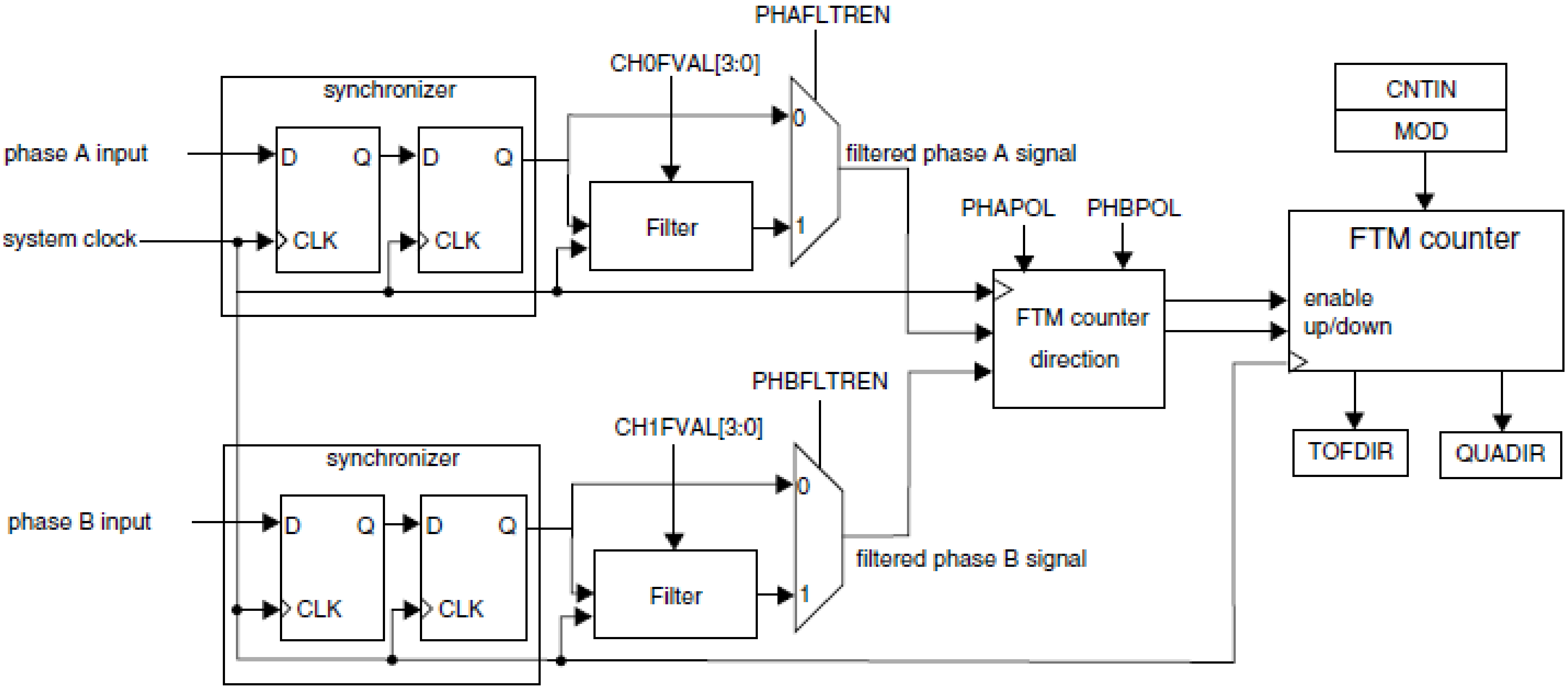
The dual edge capture mode is selected if FTMEN = 1 and DECAPEN = 1. This mode allows to measure a pulse width or period of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode when n is 0 or 2.



* Filtering function for dual edge capture mode is only available in the channels 0 and 2

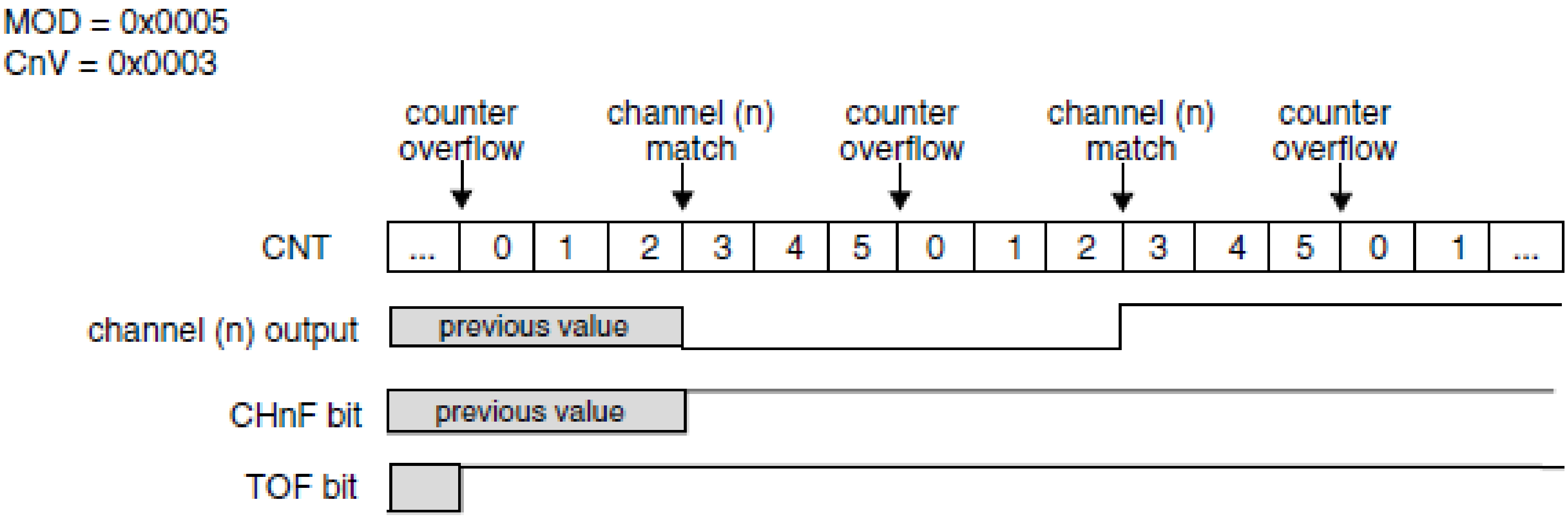
Quadrature Decoder Mode

The quadrature decoder mode is selected if (FTMEN = 1) and (QUADEN = 1). The quadrature decoder mode uses the input signals phase A and B to control the FTM counter increment and decrement. The following figure shows the quadrature decoder block diagram.



Output Compare Mode

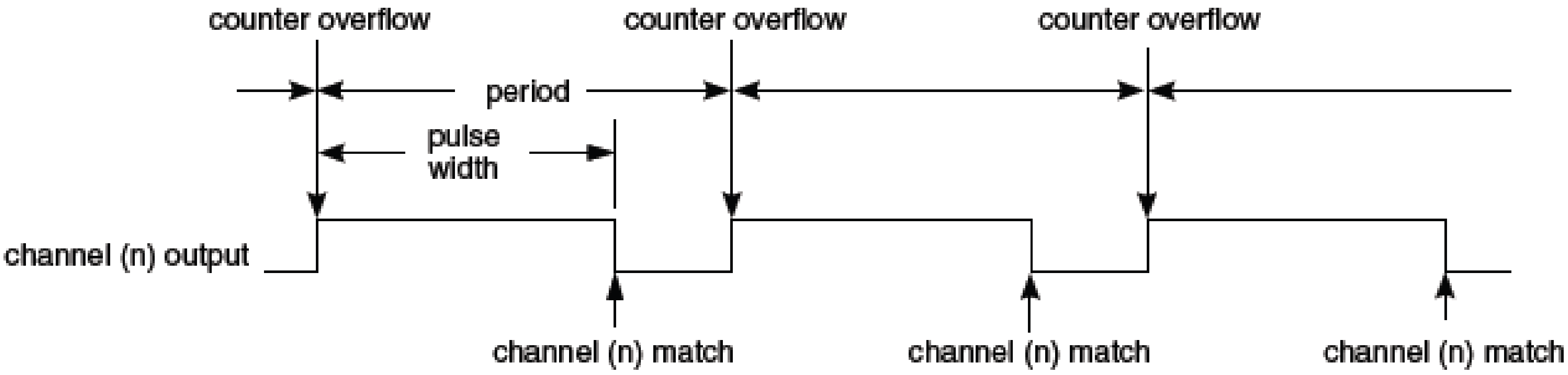
The output compare mode is selected when (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0), and (MSnB:MSnA = 0:1).
In output compare mode, the FTM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in The CnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled.



Example of the Output Compare Mode when the Match Toggles the Output

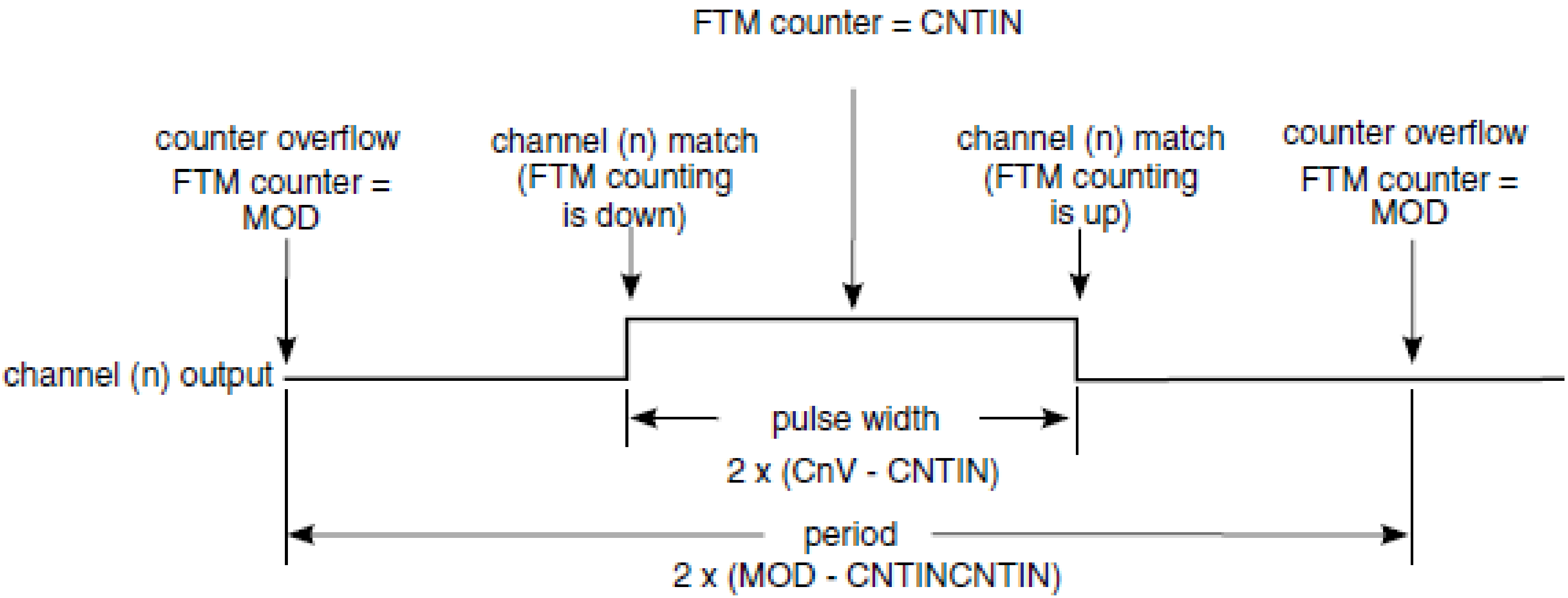
Edge-Aligned PWM (EPWM) Mode

The edge-aligned mode is selected when (QUADEN = 0), (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0), and (MSnB = 1).
The EPWM period is determined by (MOD − CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV − CNTIN).
The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnV), that is, at the end of the pulse width.
This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.



Center-Aligned PWM (CPWM) Mode

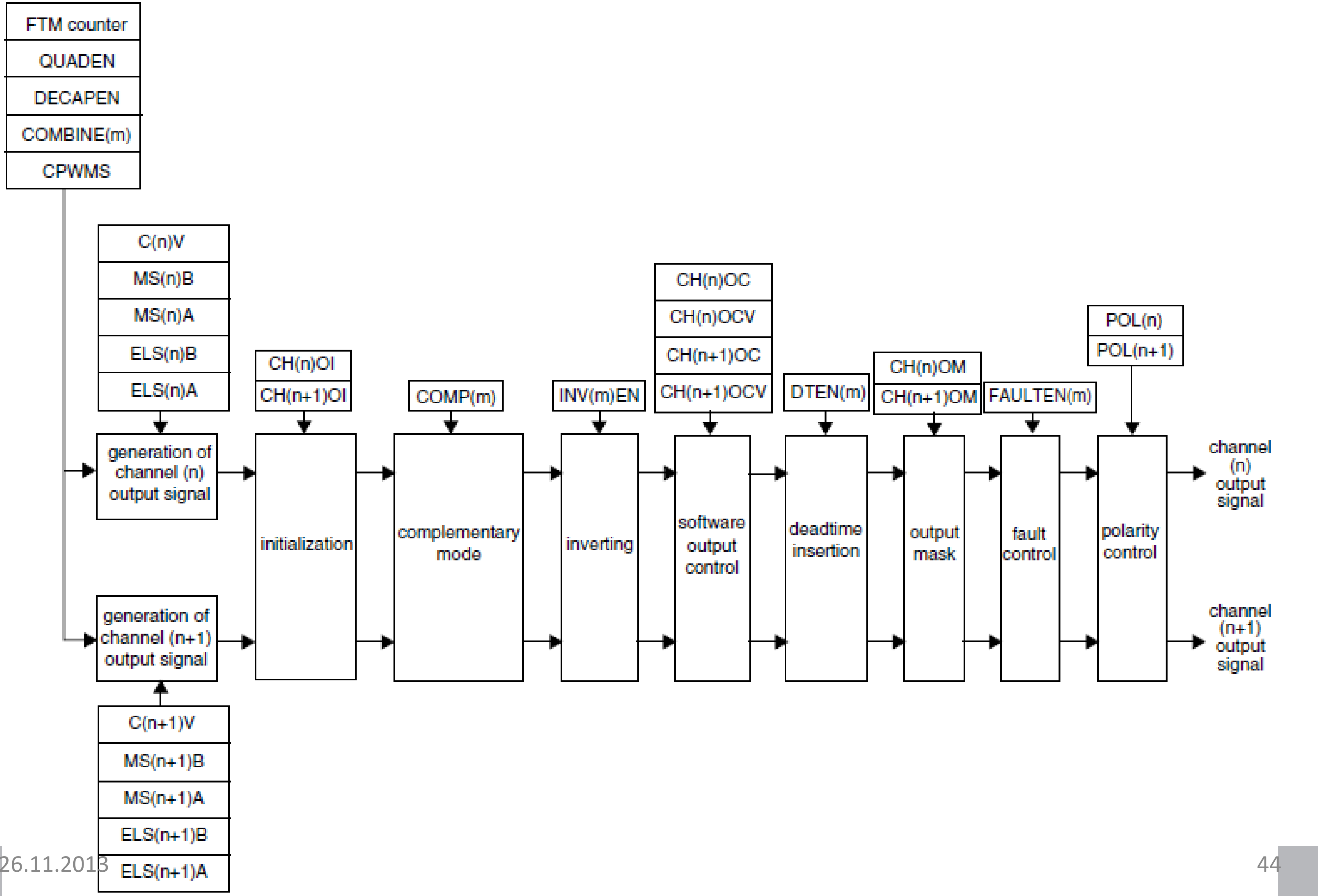
The center-aligned mode is selected when (QUADEN = 0), (DECAPEN = 0), (COMBINE = 0), and (CPWMS = 1).
The CPWM pulse width (duty cycle) is determined by 2 × (CnV − CNTIN) and the period is determined by 2 × (MOD − CNTIN).
In the CPWM mode, the FTM counter counts up until it reaches MOD and then Counts down until it reaches CNTIN.



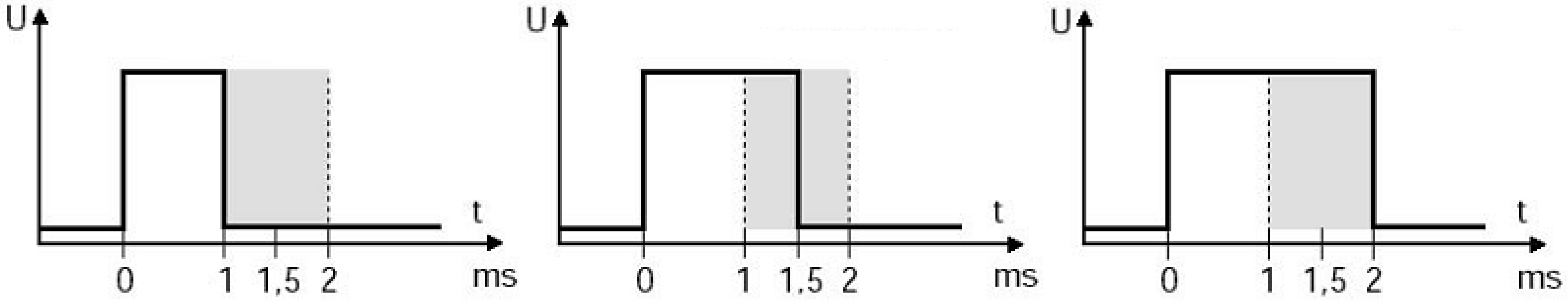
FTM Additional Features

- Combined Mode
- PWM Synchronization
- Counter Synchronization
- Channel Inverting
- Software Output Control
- Deadtime Insertion
- Output Mask
- Fault Control
- Polarity Control
- DMA transfer request

Features Priority

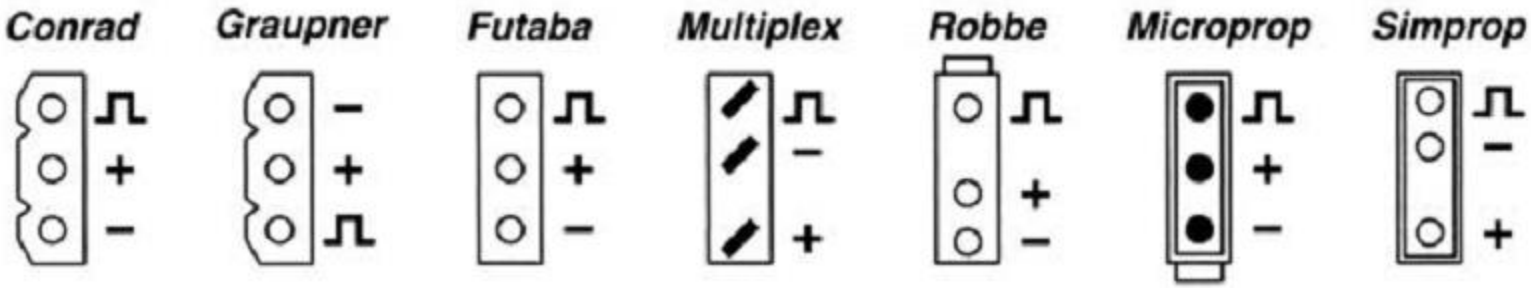


Control of Servo Motors

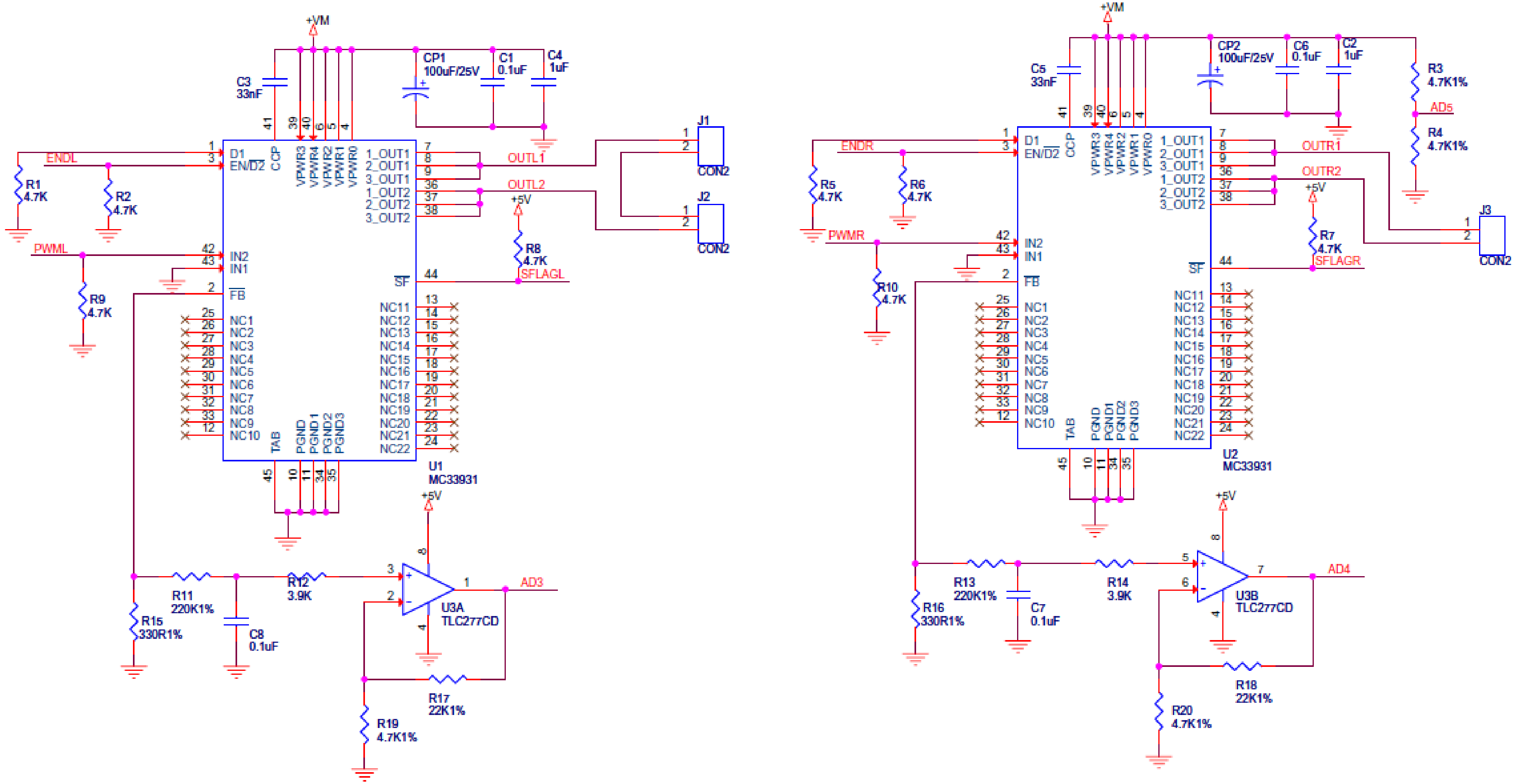


Minimum Neutral Maximum

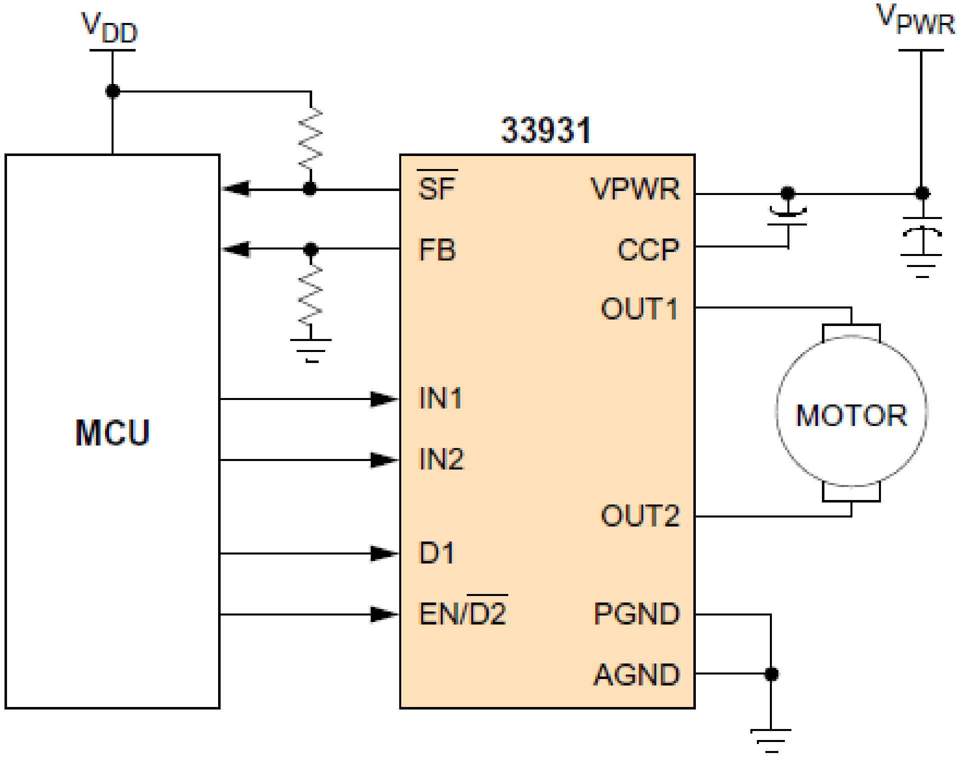
The servo motor should receive a new impulse all 10 - 20 ms. The duration of the impulse should be betrween 1 ms and 2 ms and defines the position of the servo.



DC Motor Control with two MC33931



MC33931 5.0 A Throttle Control H-bridge

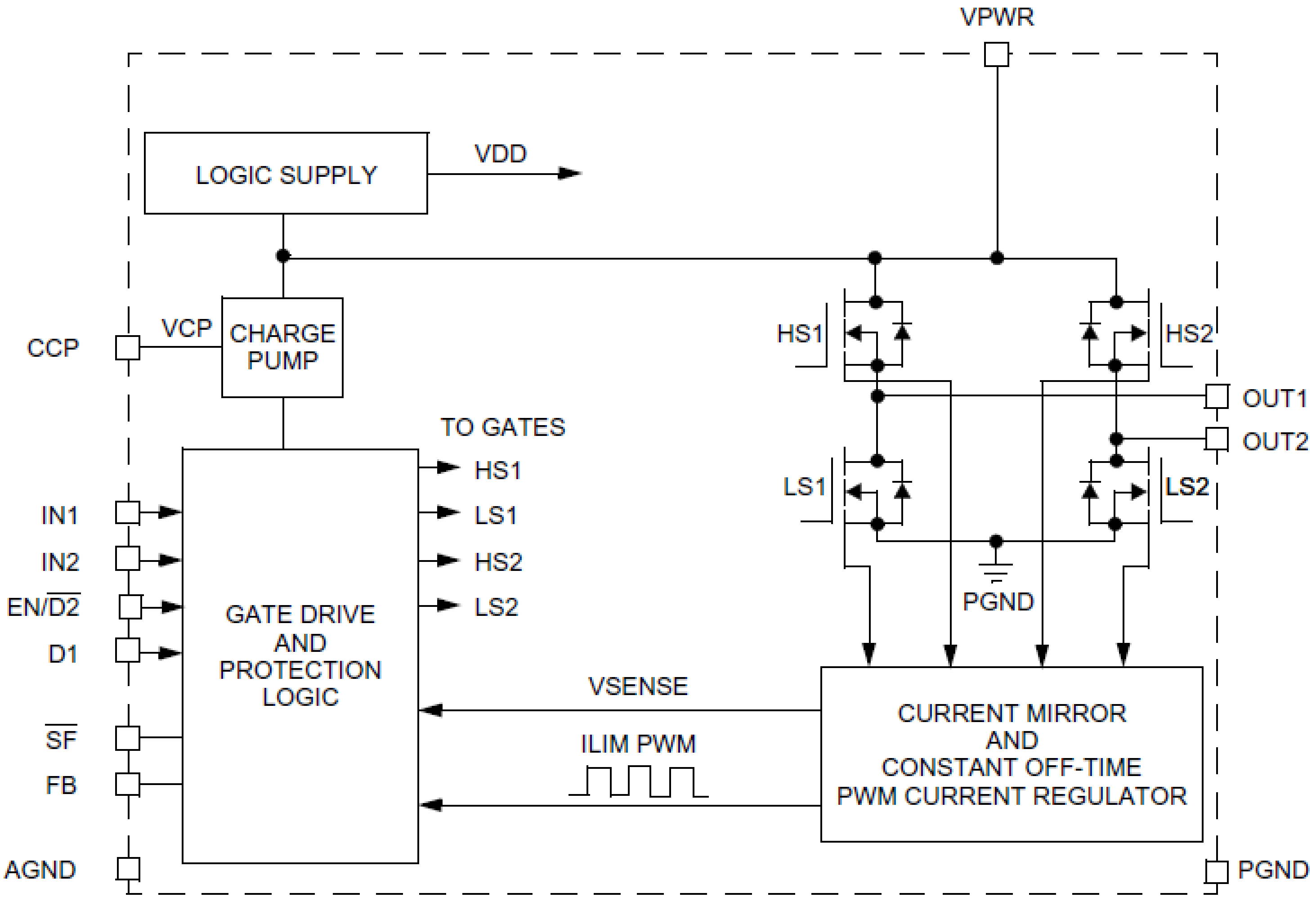


DC Motor

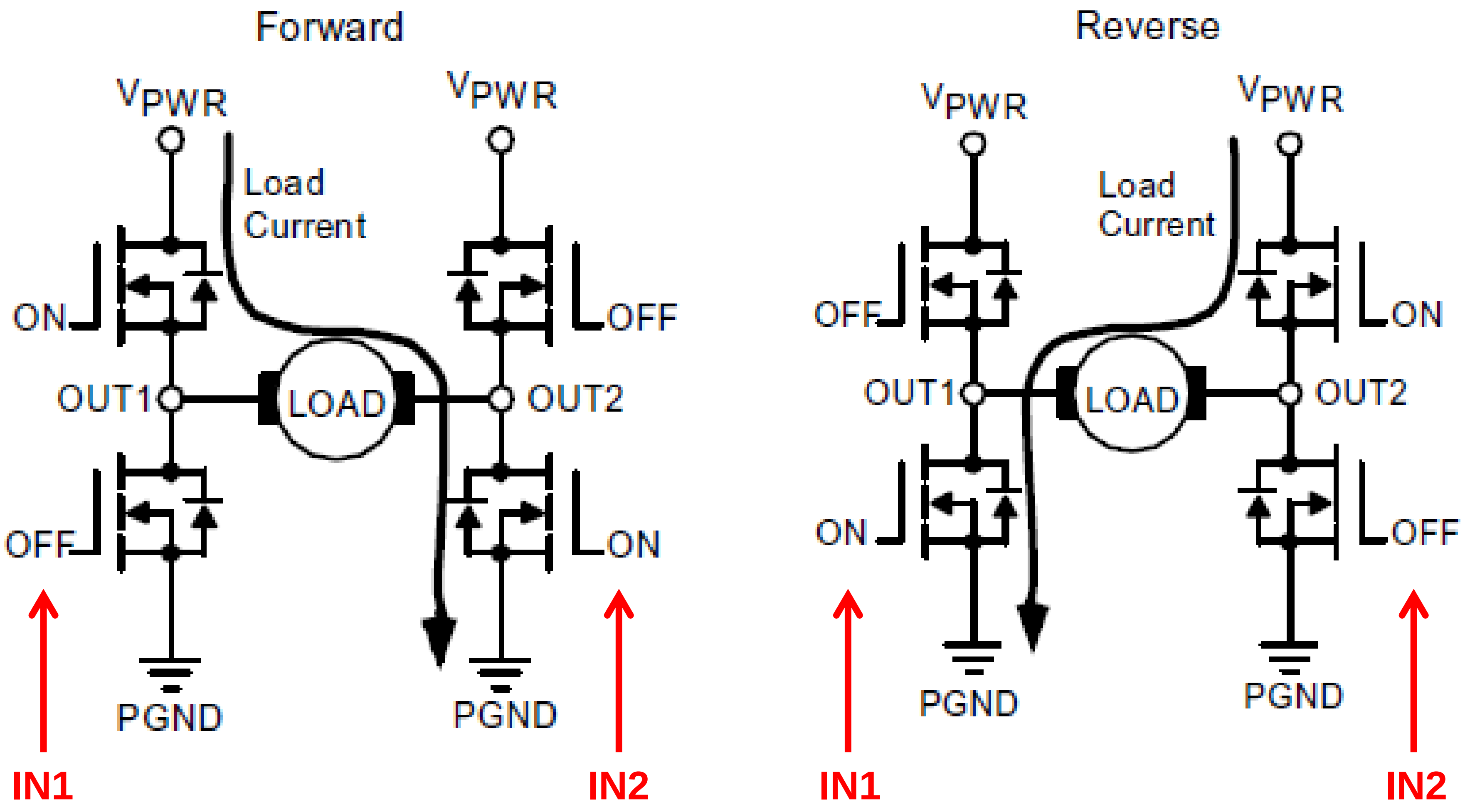
IN1	PWM Input 1
IN2	PWM Input 2
D1	Disable Input 1
EN//D2	Enable / Disable Input 2
/SF	Status Flag Output
FB	Analog Feedback Output

OUT1	Power Output
OUT2	Power Output

MC33931 Block Diagram

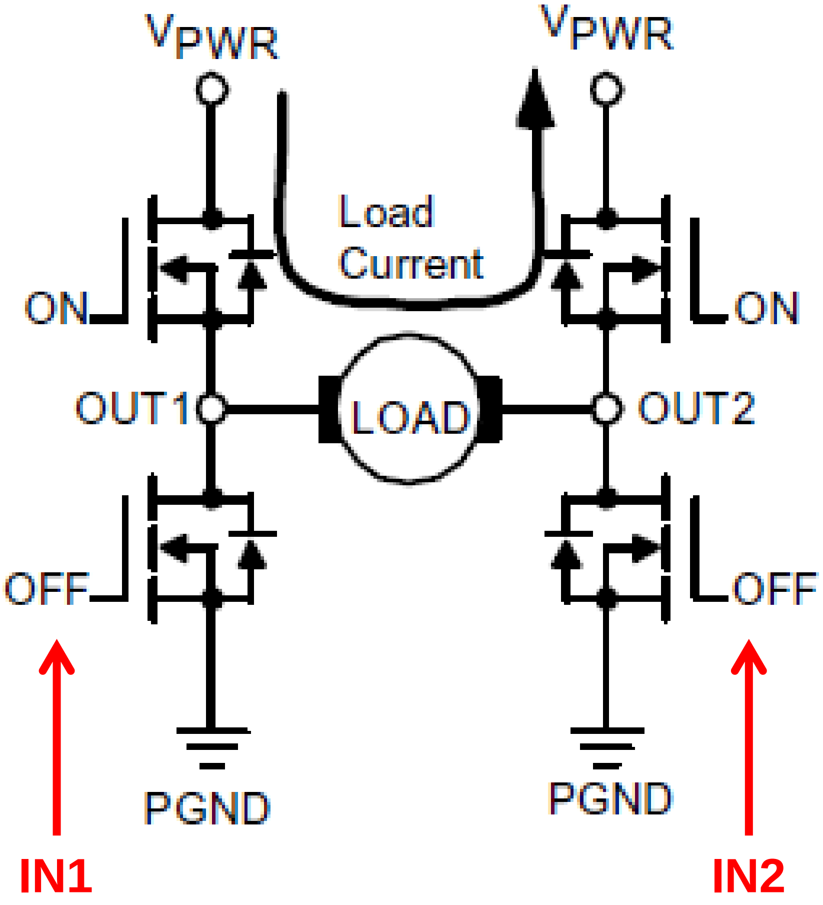


DC Motor Forward / Reverse

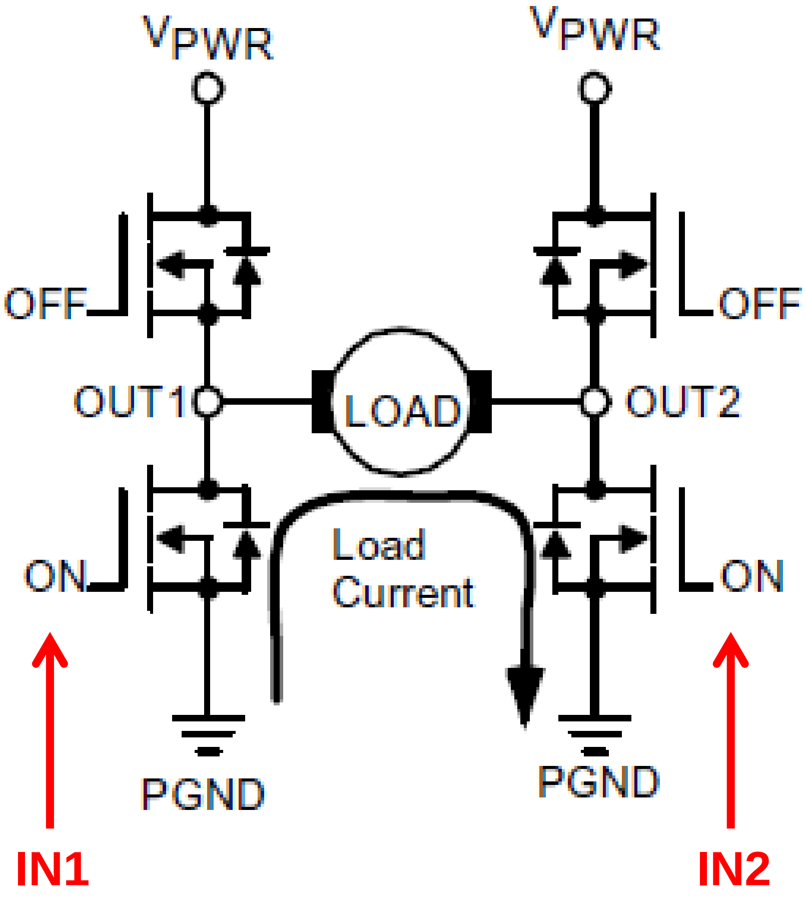


DC Motor Stopp

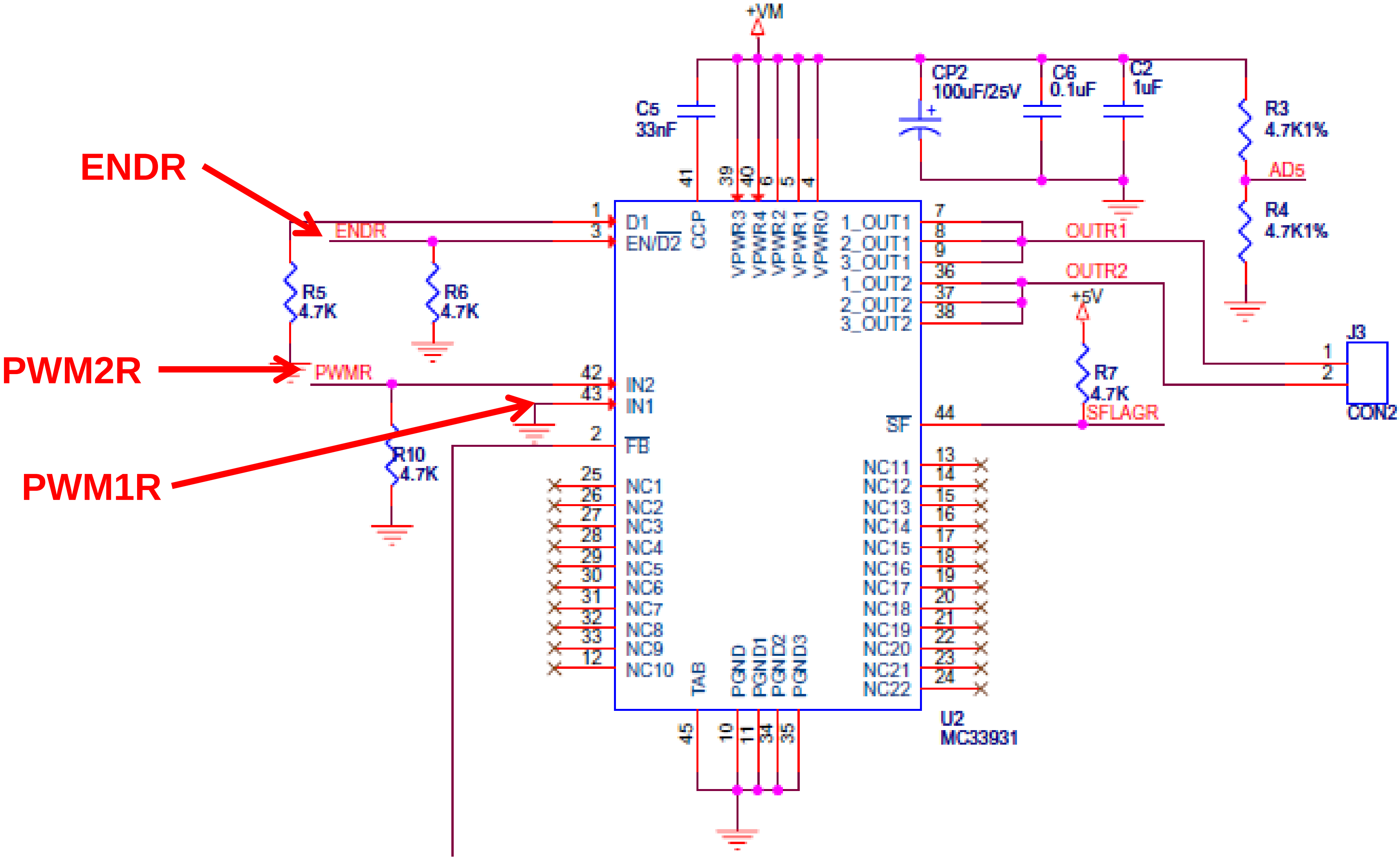
High Side Recirculation
(Forward)



Low Side Recirculation
(Forward)

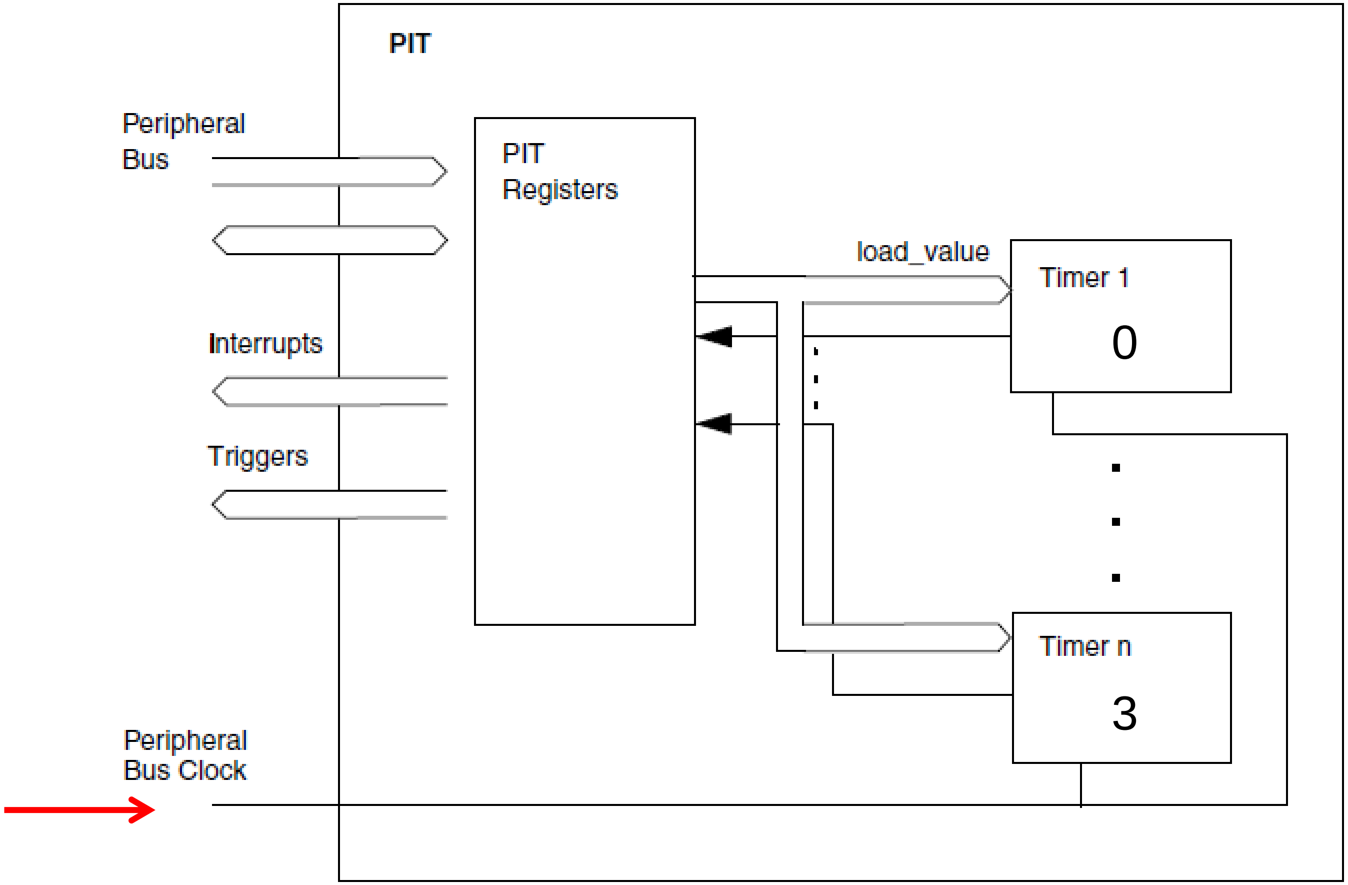


Modification for „active Breaking“



Periodic Interrupt Timer (PIT)

The PIT timer module is an array of timers that can be used to raise interrupts and Trigger DMA channels.



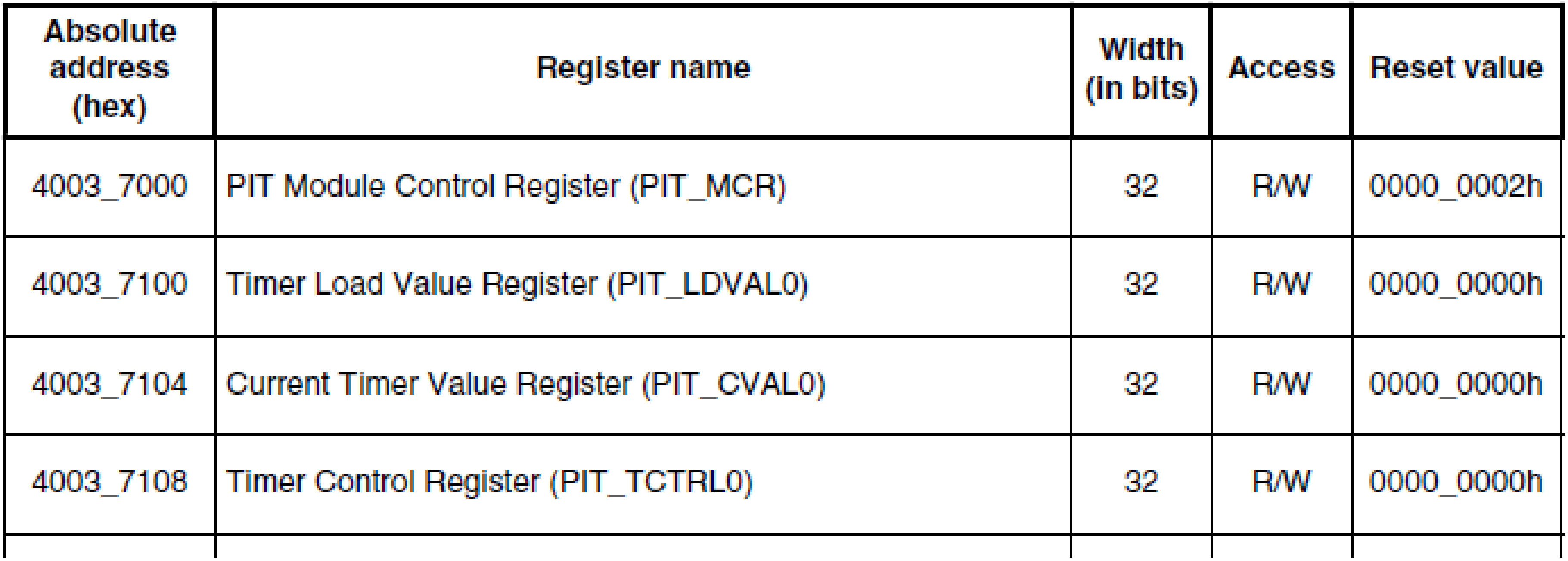
Periodic Interrupt Timer Features

The main features of this block are:

- Timers can generate DMA trigger pulses
- Timers can generate interrupts
- All interrupts are maskable
- Independent timeout periods for each timer

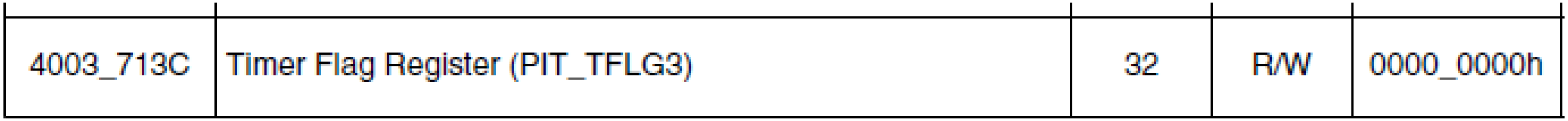
Signal Description: The PIT module has no external pins.

PIT Memory Map



Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4003_7000	PIT Module Control Register (PIT_MCR)	32	R/W	0000_0002h
4003_7100	Timer Load Value Register (PIT_LDVAL0)	32	R/W	0000_0000h
4003_7104	Current Timer Value Register (PIT_CVAL0)	32	R/W	0000_0000h
4003_7108	Timer Control Register (PIT_TCTRL0)	32	R/W	0000_0000h

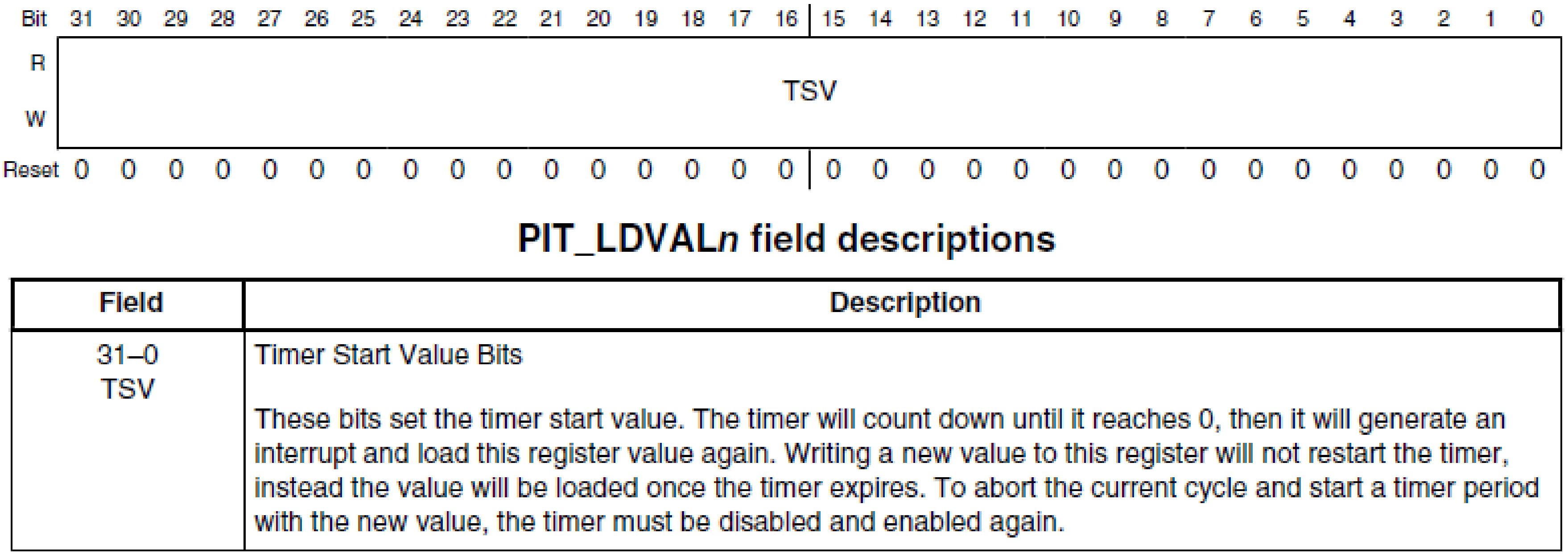
... alltogether 17 registers ...



4003_713C	Timer Flag Register (PIT_TFLG3)	32	R/W	0000_0000h

PIT Function

The timers generate triggers at periodic intervals, when enabled. They load their Start values, as specified in their **LDVAL** registers, then count down until they reach 0. Then they load their respective start value again. Each time a timer reaches 0, it will generate a trigger pulse and set the interrupt flag.



Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSV																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PIT_LDVAL*n* field descriptions

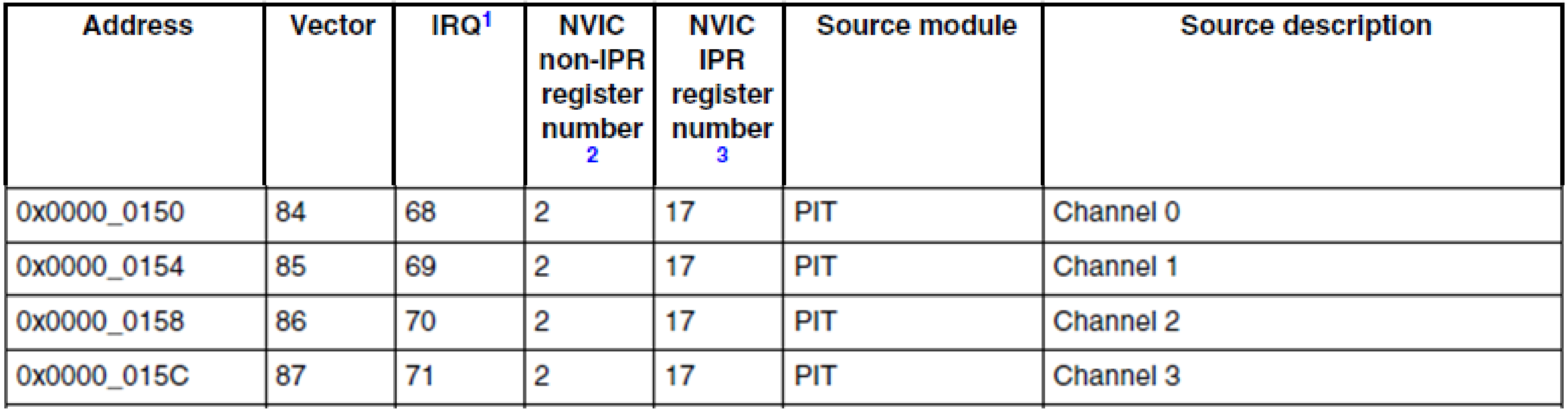
Field	Description
31–0 TSV	Timer Start Value Bits These bits set the timer start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer, instead the value will be loaded once the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.

Calculation: time intervals of the PIT

PIT Interrupts

All of the timers support interrupt generation.

All interrupts can be enabled or masked (by setting the TIE bits in the TCTRL registers). A new interrupt can be generated only after the previous one is cleared.



Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0150	84	68	2	17	PIT	Channel 0
0x0000_0154	85	69	2	17	PIT	Channel 1
0x0000_0158	86	70	2	17	PIT	Channel 2
0x0000_015C	87	71	2	17	PIT	Channel 3

Low Power Timer (LPTMR)

The low power timer (LPTMR) can be configured to operate as a time counter (with optional prescaler) or as a pulse counter (with optional glitch filter) across all power modes, including the low leakage modes. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

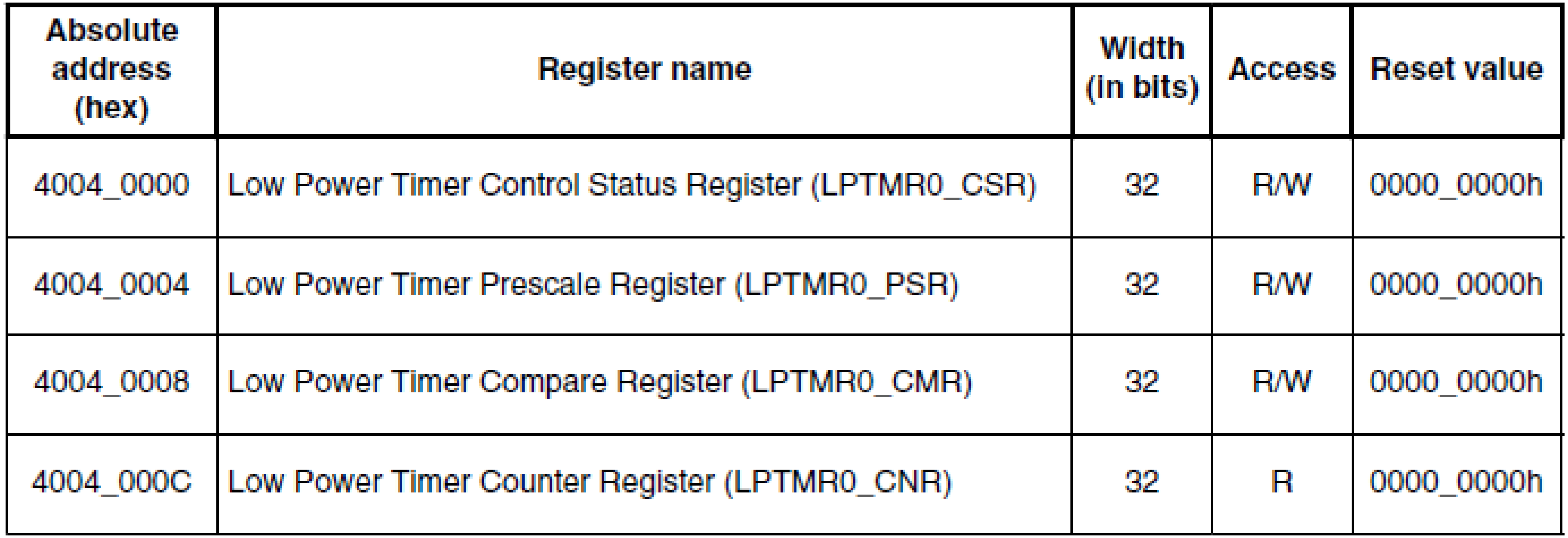
The LPTMR module's features include:

- 16-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wakeup from any low power mode
  - Hardware trigger output
  - Counter supports free-running mode or reset on compare

- Configurable clock source for prescaler/glitch filter

- Configurable input source for pulse counter
  - Rising edge or falling edge

LPTMR Memory Map

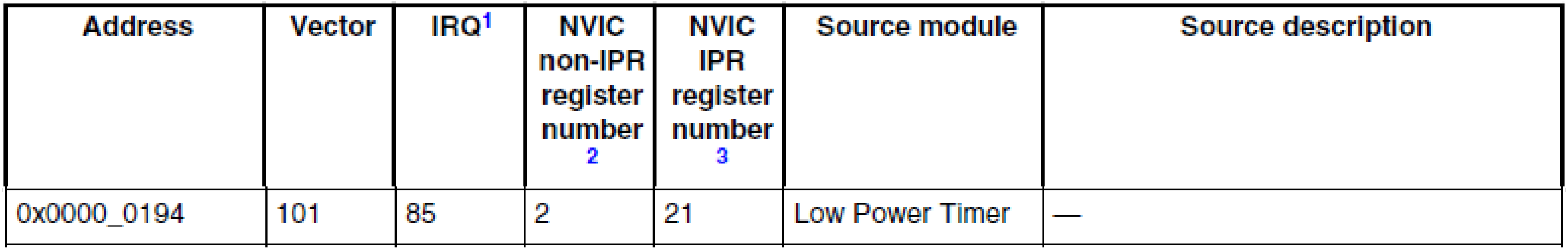


Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4004_0000	Low Power Timer Control Status Register (LPTMR0_CSR)	32	R/W	0000_0000h
4004_0004	Low Power Timer Prescale Register (LPTMR0_PSR)	32	R/W	0000_0000h
4004_0008	Low Power Timer Compare Register (LPTMR0_CMR)	32	R/W	0000_0000h
4004_000C	Low Power Timer Counter Register (LPTMR0_CNR)	32	R	0000_0000h

LPTMR Interrupt

The LPTMR interrupt is generated whenever the CSR[TIE] and CSR[TCF] are set. The CSR[TCF] is cleared by disabling the LPTMR or by writing a logic one to it. The CSR[TIE] can be altered and the CSR[TCF] can be cleared while the LPTMR Is enabled.

The LPTMR interrupt is generated asynchronously to the system clock and can be used to generate a wakeup from any low power mode, including the low leakage Modes.



Address	Vector	IRQ[1]	NVIC non-IPR register number[2]	NVIC IPR register number[3]	Source module	Source description
0x0000_0194	101	85	2	21	Low Power Timer	—

Watchdog Timer (WDOG)

The Watchdog Timer (WDOG) keeps a watch on the system functioning and resets it in case of its failure. Some reasons for such failures are: run-away software code and the stoppage of the system clock that in a safety critical system can lead to serious consequences.
In such cases, the watchdog brings the system into a safe state of operation.
The watchdog monitors the operation of the system by expecting periodic communication from the software, generally known as servicing or refreshing The watchdog. If this periodic refreshing does not occur, the watchdog resets the system.

In its simplest form, the watchdog timer runs continuously off a clock source and expects to be serviced periodically, failing which it resets the system. This ensures that the software is executing correctly and has not run away in an unintended direction. Software can adjust the period of servicing or the time-out value for the watchdog timer to meet the needs of the application.

To enhance the independence of watchdog from the system, it runs off an inde-Pendent LPO oscillator clock. You can also switch over to an alternate clock source if required, through a control register bit.

WDOG Features

- Independent clock source input (independent from CPU/bus clock). Choice between two clock sources:
  - LPO Oscillator
  - External system clock

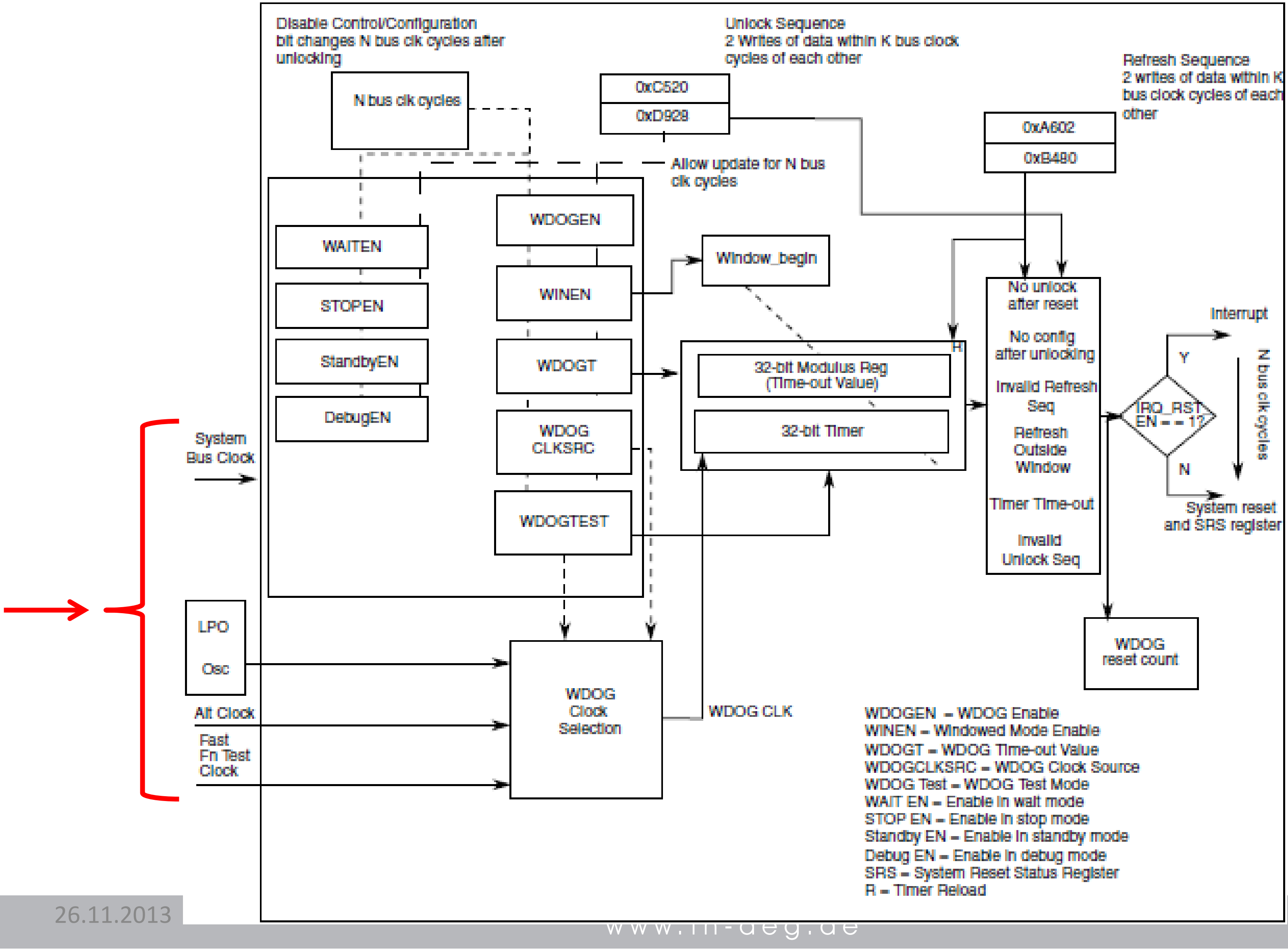
- Unlock sequence for allowing updates to write-once WDOG control/configuration bits.

- All WDOG control/configuration bits are writable once only within 256 bus clock cycles of being unlocked.

- Programmable time-out period specified in terms of number of WDOG clock cycles.

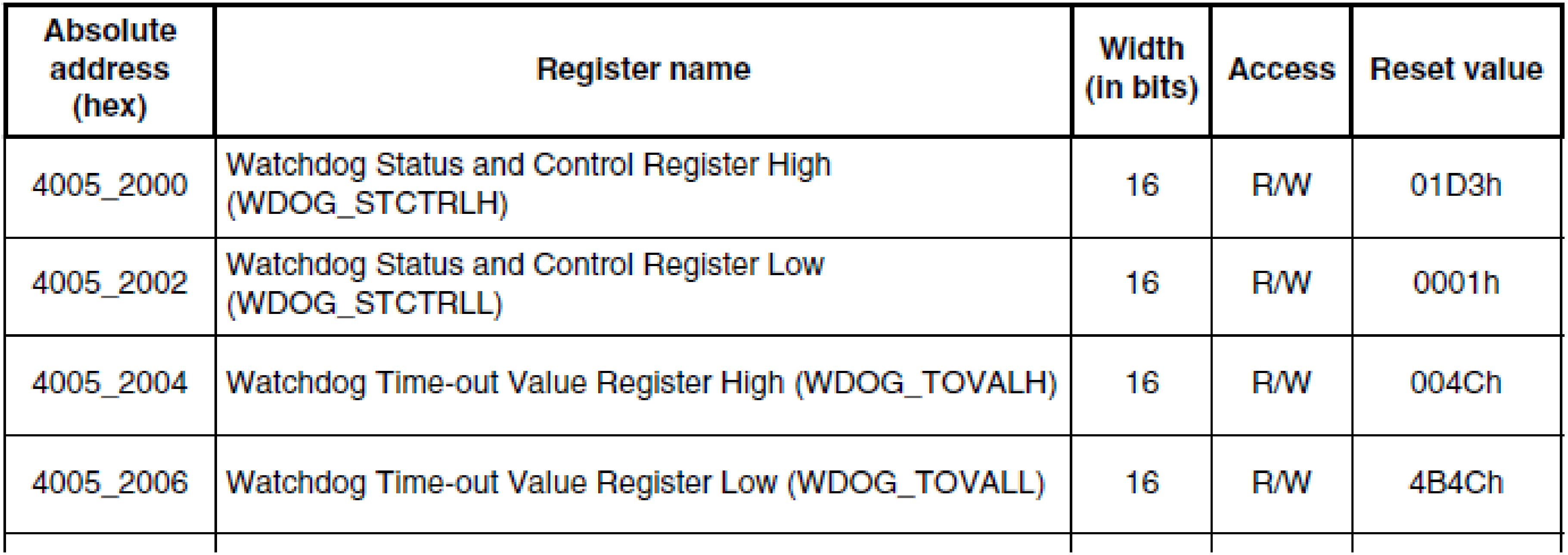
- Ability to test WDOG timer and reset with a flag indicating watchdog test.
  - Quick test—Small time-out value programmed for quick test.
  - Byte test—Individual bytes of timer tested one at a time.
  - Read-only access to the WDOG timer—Allows dynamic check that WDOG timer is operational.

WDOG Block Diagram



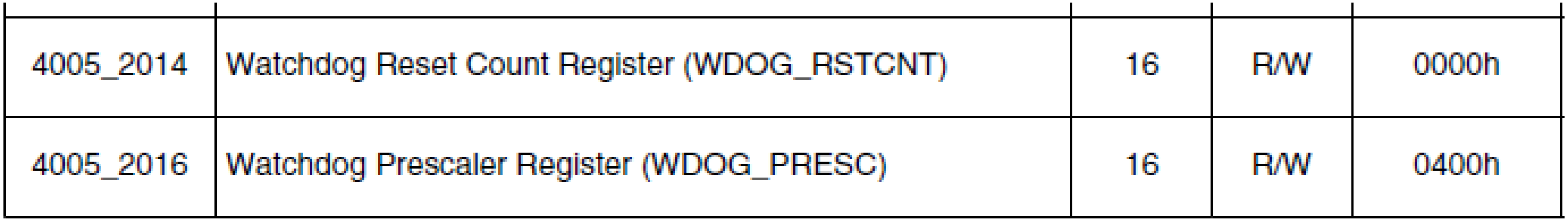
WDOGEN – WDOG Enable
WINEN – Windowed Mode Enable
WDOGT – WDOG Time-out Value
WDOGCLKSRC – WDOG Clock Source
WDOG Test – WDOG Test Mode
WAIT EN – Enable in wait mode
STOP EN – Enable in stop mode
Standby EN – Enable in standby mode
Debug EN – Enable in debug mode
SRS – System Reset Status Register
R – Timer Reload

WDOG Memory Map



Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4005_2000	Watchdog Status and Control Register High (WDOG_STCTRLH)	16	R/W	01D3h
4005_2002	Watchdog Status and Control Register Low (WDOG_STCTRLL)	16	R/W	0001h
4005_2004	Watchdog Time-out Value Register High (WDOG_TOVALH)	16	R/W	004Ch
4005_2006	Watchdog Time-out Value Register Low (WDOG_TOVALL)	16	R/W	4B4Ch

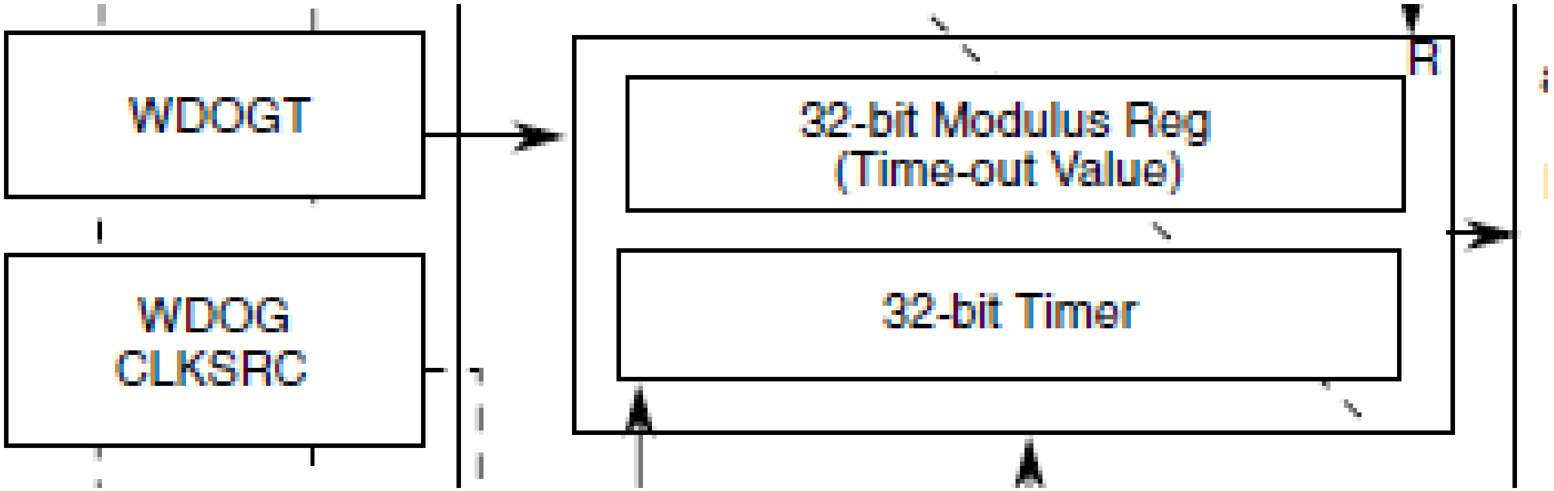
... insgesamt 12 Register ...



4005_2014	Watchdog Reset Count Register (WDOG_RSTCNT)	16	R/W	0000h
4005_2016	Watchdog Prescaler Register (WDOG_PRESC)	16	R/W	0400h

WDOG Time Out

Module clock	Chip clock
LPO Oscillator	1 kHz LPO Clock
Alt Clock	Bus Clock
Fast Test Clock	Bus Clock
System Bus Clock	Bus Clock



Refreshing the Watchdog

A robust refreshing mechanism has been chosen for the watchdog.

A valid refresh is a write of 0xA602 followed by 0xB480 within 20 bus clock cycles to the watchdog refresh register.

```
0xA602 = 1010011000000010

0xB480 = 1011010010000000
```

If these two values are written more than 20 bus cycles apart or if something other than these two values is written to the register, a watchdog reset (or interrupt - then Reset if enabled) is issued to the system.
A valid refresh makes the watchdog timer restart on the next bus clock.

To prevent unintended modification of the watchdog's control and configuration register bits, you are allowed to update them only within a period of 256 bus clock cycles after unlocking.

Disable Watchdog

For code development and debugging, it is best to disable the watchdog. This requires unlocking the watchdog first. Keep in mind that there are timing requirements for the execution of the unlock steps. The two step unlock sequences must execute within 20 clock cycles of each other. Therefore interrupts must be disabled and single-step debugging cannot be done during this section.

```
/* disable all interrupts */
asm(" CPSID i");

/* Write 0xC520 to the unlock register */
WDOG_UNLOCK = 0xC520;

/* Followed by 0xD928 to complete the unlock */
WDOG_UNLOCK = 0xD928;

/* enable all interrupts */
asm(" CPSIE i");

/* Clear the WDOGEN bit to disable the watchdog */
WDOG_STCTRLH &= ~WDOG_STCTRLH_WDOGEN_MASK;
```

kinetis_sysinit.c

Literature and Links

http://www.arm.com/products/processors/cortex-m/cortex-m3.php

http://www.arm.com/products/processors/technologies/instruction-set-architectures.php

http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=K60

K60 Sub-Family Reference Manual

Kinetis Peripheral Module Quick Reference

Cortex-M4 Technical Reference Manual

http://www.arm.com/products/processors/cortex-m/index.php

TECHNISCHE HOCHSCHULE DEGGENDORF THD

Technische Hochschule Deggendorf – Edlmairstr. 6 und 8 – 94469 Deggendorf