NC STATE UNIVERSITY

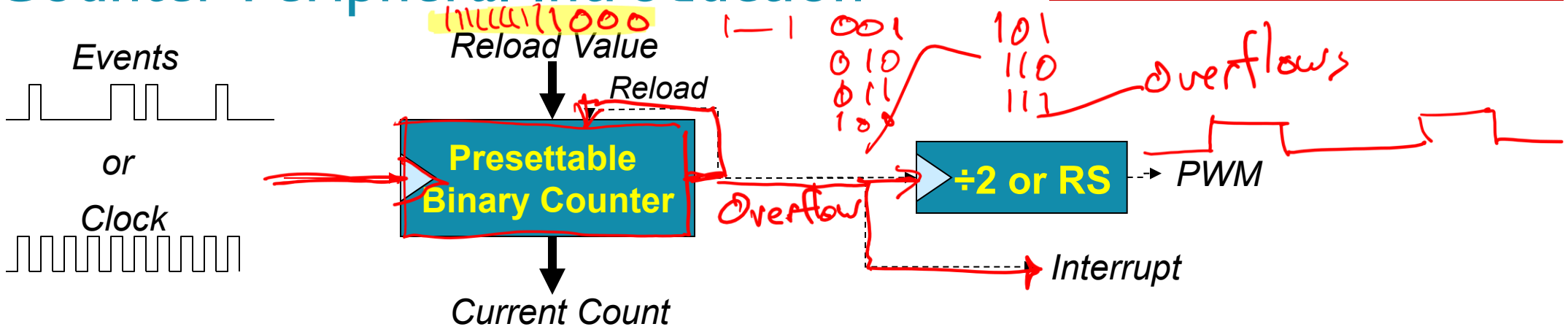
Timer Peripherals

Timer Peripherals Available

NC STATE UNIVERSITY

- In KL25 MCU
  - PIT - Periodic Interrupt Timer
    - Can generate periodically generate interrupts or trigger DMA (direct memory access) transfers
  - TPM - Timer/PWM Module
    - Connected to I/O pins, has input capture and output compare support
    - Can generate PWM signals
    - Can generate interrupts and DMA requests
  - LPTMR - Low-Power Timer
    - Can operate as timer or counter in all power modes (including low-leakage modes)
    - Can wake up system with interrupt
    - Can trigger hardware
  - Real-Time Clock
    - Driven by external 32.768 kHz crystal
    - Tracks elapsed time (seconds) in 32-bit register
    - Can set alarm
    - Can generate 1Hz output signal and/or interrupt
    - Can wake up system with interrupt
  - Watchdog Timer (WDT)
    - Resets CPU if not refreshed before expiration
    - Covered in separate slide set
- In Cortex-M0+
  - SysTick – System Tick Timer
    - Part of CPU core's peripherals
    - Can generate periodic interrupt

Timer/Counter Peripheral Introduction



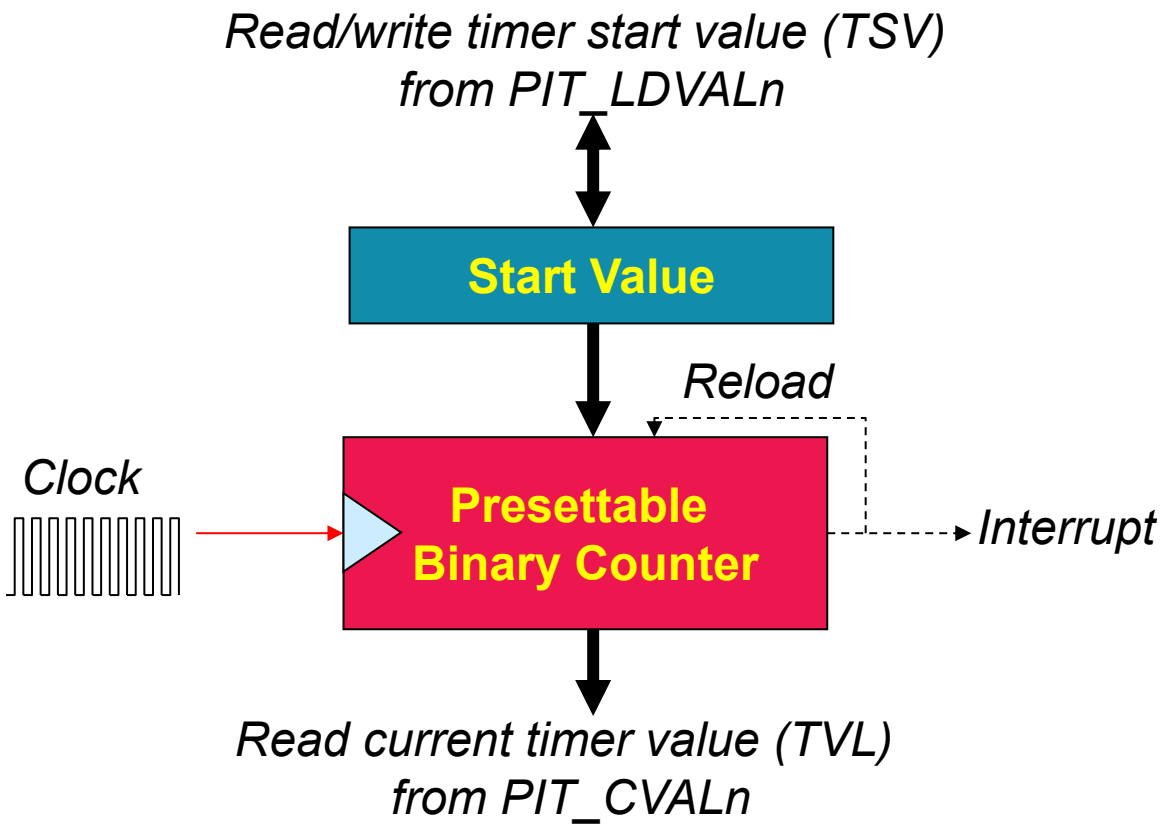
- Common peripheral for microcontrollers
- Based on presettable binary counter, enhanced with configurability

- Two basic modes
  - **Counter mode:** count **pulses** which indicate **events** (e.g. odometer pulses)
  - **Timer mode**: clock source is periodic, so counter value is proportional to **elapsed time** (e.g. stopwatch)

- Main configurable options
  - Current count **value**
  - Count **reload value**
  - Count **direction:** up or down
  - Counter's **clock source**
  - Counter's **overflow/underflow action**
    - Generate interrupt
    - Reload counter, resume counting
    - Toggle hardware output signal
    - Stop!

PERIODIC INTERRUPT TIMER

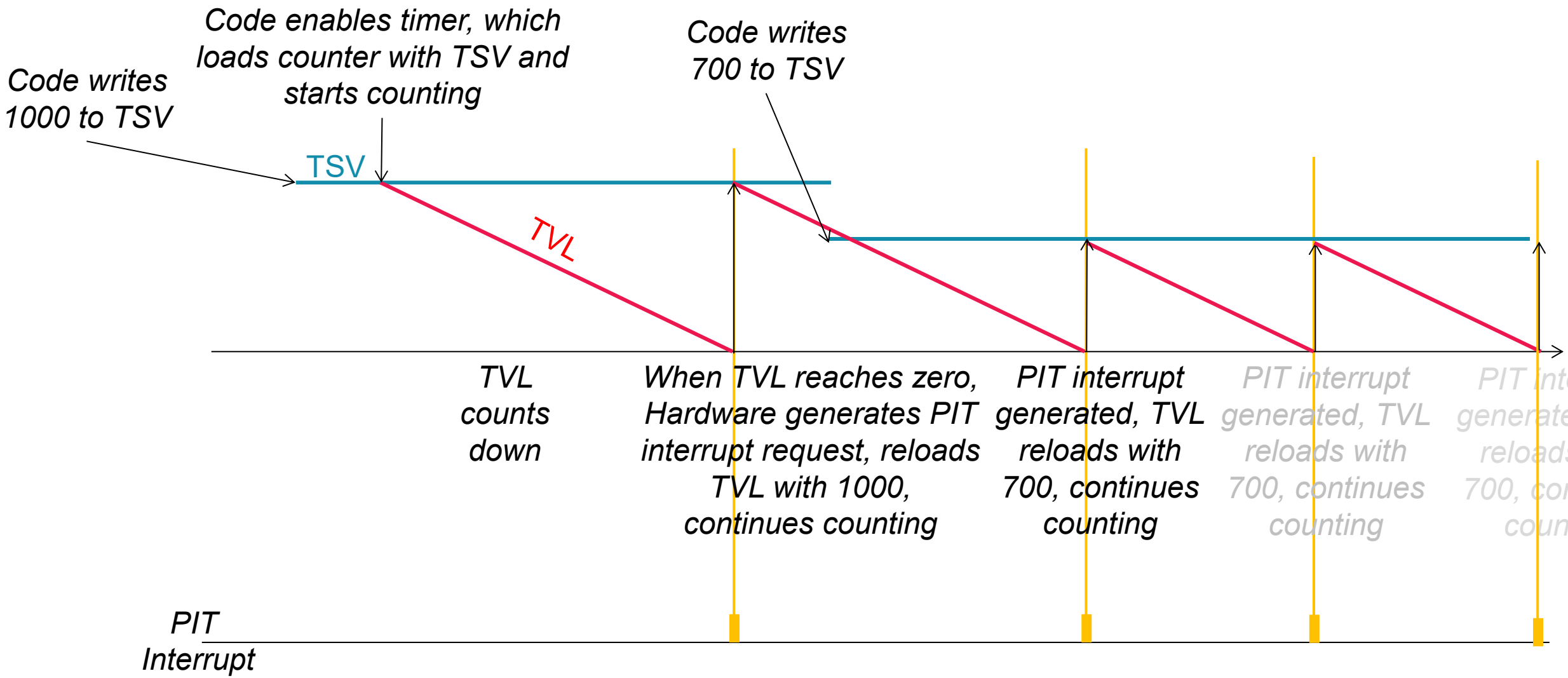
Periodic Interrupt Timer



- Generates periodic interrupts using a 32-bit counter
- Load start value (32-bit) from LDVAL
- Counter decrements with each clock pulse
  - Fixed clock source for PIT - Bus Clock from Multipurpose Clock Generator - e.g. 24 MHz
- When timer value (CVAL) reaches zero
  - Generates interrupt
  - Reloads timer with start value

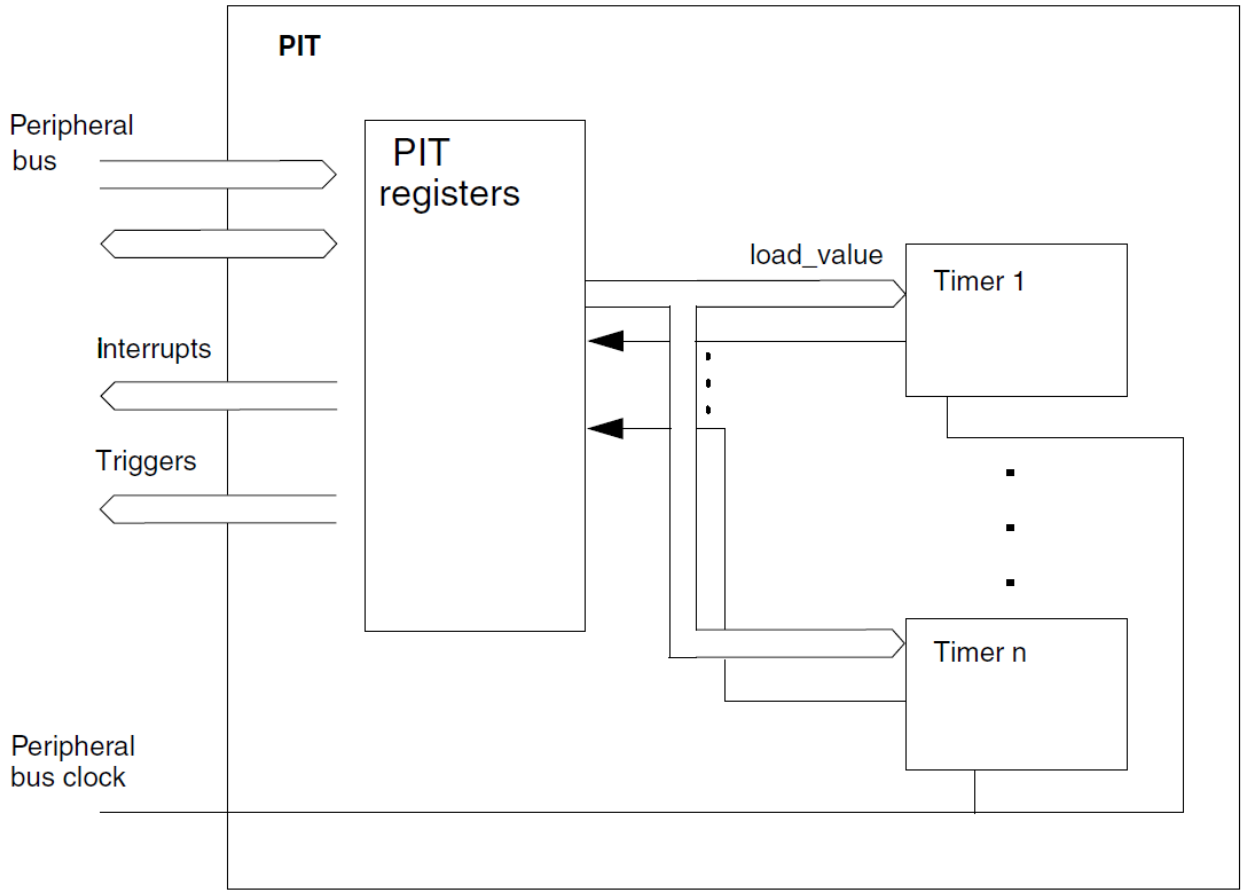
Periodic Interrupt Timer

NC STATE UNIVERSITY



PIT Configuration

- 1: Enable clock gating!
  - SIMCGC6 PIT
- 2: Module Control Register (PIT->MCR)
  - MDIS - Module disable
    - 0: module enabled
    - 1: module disabled (clock disabled)
  - FRZ - Freeze - stops timers in debug mode
    - 0: timers run in debug mode
    - 1: timers are frozen (don't run) in debug mode
- **Multiple channels** within a PIT
  - KL25Z has two channels
- Can chain timers together to create 64-bit timer



Control of Each Timer Channel n

NC STATE UNIVERSITY

- CMSIS Interface:
  - General PIT settings accessed as struct: PIT->MCR, etc.
  - Channels are accessed as an array of structs: PIT->CHANNEL[n].LDVAL, etc
- PIT_LDVALn: Load value (PIT->CHANNEL[n].LDVAL)
- PIT_CVALn: Current value (PIT->CHANNEL[n].CVAL)
- PIT_TCTRLn: Timer control (PIT->CHANNEL[n].TCTRL)
  - CHN: Chain
    - 0: independent timer operation, uses own clock source
    - 1: timer n is clocked by underflow of timer n-1
  - TIE: Timer interrupt enable
    - 0: Timer will not generate interrupts
    - 1: Interrupt will be requested on underflow (i.e. when TIF is set)
  - TEN: Timer enable
    - 0: Timer will not count
    - 1: Timer is enabled, will count
- PIT_TFLGn: Timer flags
  - TIF: Timer interrupt flag
    - 1: Timeout has occurred

Configuring the PIT

NC STATE UNIVERSITY

- Enable clock to PIT module

```
SIM->SCGC6 |= SIM_SCGC6_PIT_MASK;
```

- Enable module, freeze timers in debug mode

```
PIT->MCR &= ~PIT_MCR_MDIS_MASK;
PIT->MCR |= PIT_MCR_FRZ_MASK;
```

- Initialize PIT0 to count down from starting_value

```
PIT->CHANNEL[0].LDVAL = PIT_LDVAL_TSV(starting_value);
```

- No chaining of timers

```
PIT->CHANNEL[0].TCTRL &= ~PIT_TCTRL_CHN_MASK;
```

Calculating Load Value

NC STATE UNIVERSITY

- Goal: generate an interrupt every T seconds
- LDV = round($T*f_{count}$ - 1)
  - -1 since the counter counts down to 0
  - Round since LDV register is an integer, not a real number
    - Rounding provides closest integer to desired value, resulting in minimum timing error
- Example: Interrupt every 137.41 ms
  - LDV = 137.41 ms * 24 MHz - 1 = 3297839
- Example: Interrupt with a frequency of 91 Hz
  - LDV = (1/91 Hz)*24 MHz - 1 = round (263735.2637-1) = 263734

Configuring the PIT and NVIC for Interrupts

NC STATE UNIVERSITY

- Configure PIT
  - Let the PIT channel generate interrupt requests

```
PIT->CHANNEL[0].TCTRL |= PIT_TCTRL_TIE_MASK;
```

- Configure NVIC
  - Set PIT IRQ priority

```
NVIC_SetPriority(PIT_IRQn, 128); // 0, 64, 128 or 192
```

  - Clear any pending IRQ from PIT

```
NVIC_ClearPendingIRQ(PIT_IRQn);
```

  - Enable the PIT interrupt in the NVIC

```
NVIC_EnableIRQ(PIT_IRQn);
```

- Make sure interrupts are not masked globally

```
__enable_irq();
```

Interrupt Handler

NC STATE UNIVERSITY

- One interrupt for entire PIT
- CMSIS ISR name: PIT_IRQHandler
- ISR activities
  - Determine which channel triggered interrupt

```
if (PIT->CHANNEL[n].TFLG & PIT_TFLG_TIF_MASK) {
```

  - Clear interrupt request flag for channel by writing one to it

```
PIT->CHANNEL[0].TFLG |= PIT_TFLG_TIF_MASK;
```

  - Do the ISR's work

Starting and Stopping the Timer Channel

NC STATE UNIVERSITY

- Start the timer channel

```
PIT->CHANNEL[0].TCTRL |= PIT_TCTRL_TEN_MASK;
```

- Stop the timer channel

```
PIT->CHANNEL[0].TCTRL &= ~PIT_TCTRL_TEN_MASK;
```

Example: Stopwatch

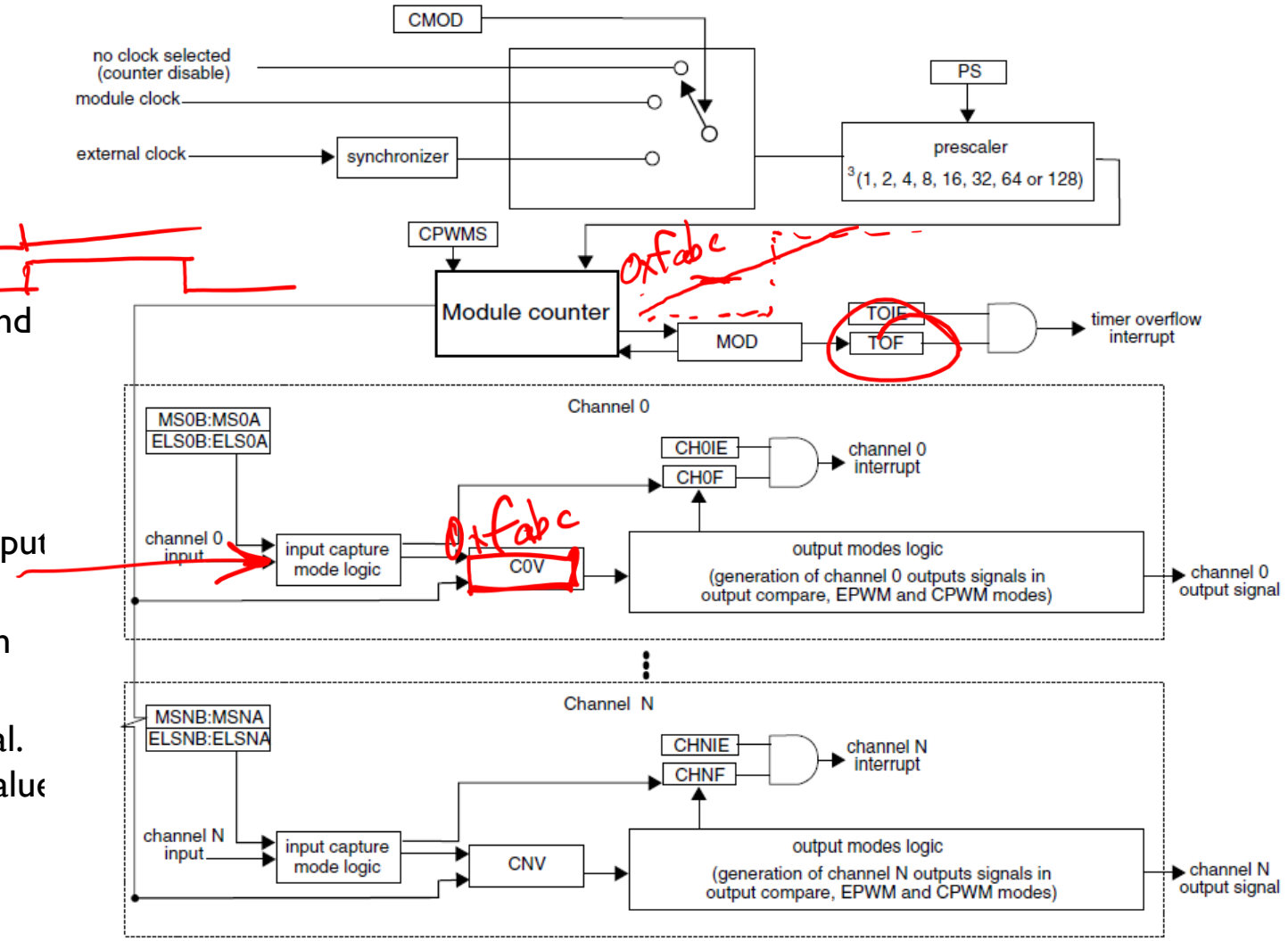
NC STATE UNIVERSITY

- Measure time with 100 us resolution
- Display elapsed time, updating screen every 10 ms
- Controls
  - S1: toggle start/stop
- Use PIT
  - Counter increment every 100 us
    - Set to PIT Channel 0 to expire every 100 us
    - Calculate load value LDVAL = round (100 us * 24 MHz -1) = 2399
  - LCD Update every 10 ms
    - Update LCD every nth PIT ISR
    - n = 10 ms/100us = 100
    - Don't update LCD in ISR! Too slow.
    - Instead set flag LCD_Update in ISR, poll it in main loop

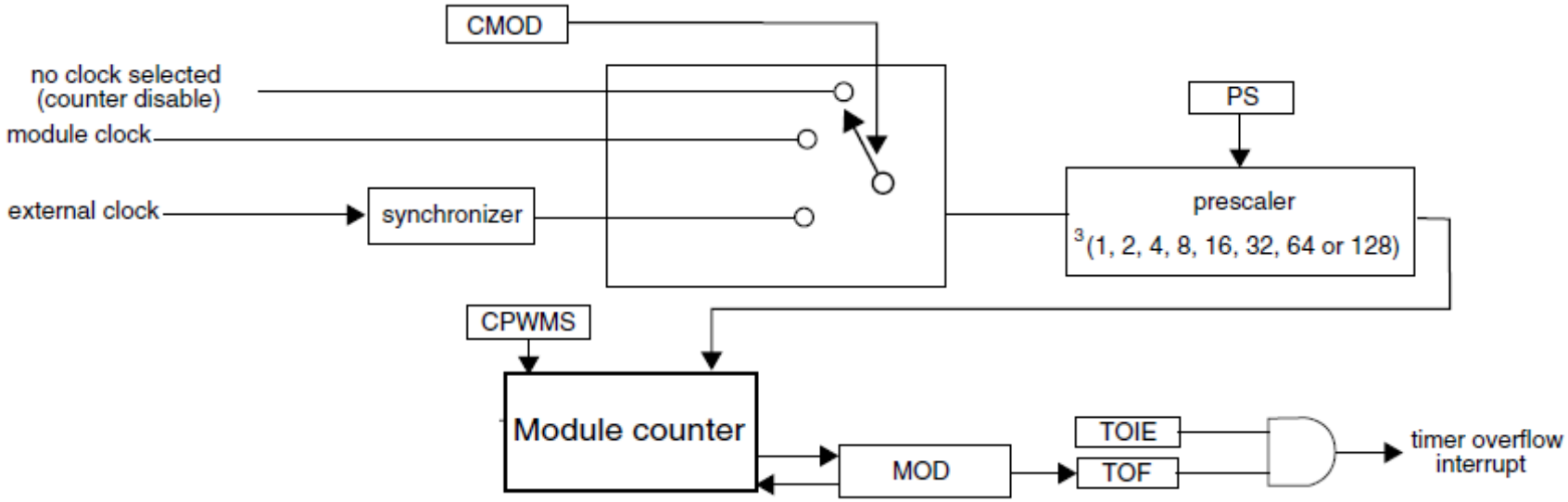
TIMER/PWM MODULE (TPM)

TPM - Timer/PWM Module

- Core: Module counter
  - Two clock options - external or internal
  - Prescaler to divide clock by 1 to 128
  - 16-bit counter
    - Can count up or up/down
    - Can reload with set load value or wrap around (to FFFF or 0000)
- Multiple (6) independent channels
  - 3 modes
    - Capture Mode: capture timer's value when input signal changes
    - Output Compare: Change output signal when timer reaches certain value
    - PWM: Generate pulse-width-modulated signal. Width of pulse is proportional to specified value
  - Each channel can generate interrupt, DMA request, hardware trigger on overflow
  - One I/O pin per channel: TPM_CHn

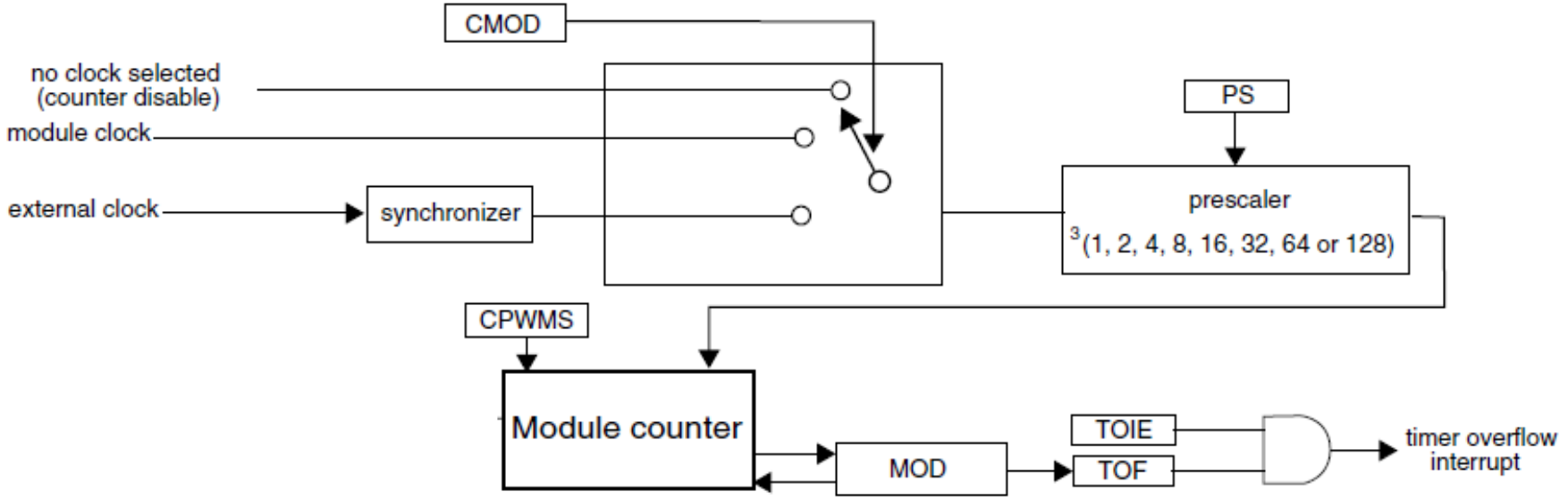


Timer Configuration



- Clock source
  - CMOD: selects internal or external clock
- Prescaler
  - PS: divide selected clock by 1, 2, 4, 8, 16, 32,64, 128
- Count Mode: direction
  - CPWMS: count up (0) or up & down (1)
- Count Modulus: value to counts up to
  - MOD: 16-bit
  - Timer overflows when counter goes past MOD value
  - Up counting: 0, 1, 2, … MOD, 0/Overflow, 1, 2, … MOD
  - Up/down counting: 0, 1, 2, … MOD, MOD-1/Interrupt, MOD-2, … 2, 1, 0, 1, 2, …
- DMA: Enable DMA transfer on overflow
- TOF: Flag indicating timer has overflowed

Basic Counter Mode



- Count external events applied on input pin
  - Set CMOD = 01 to select external clock
  - Set PS = 000 (unless division needed
- Timer overflow flag TOF set to 1 upon receiving MOD * prescaler pulses
- Can generate interrupt if TOIE is set

2-0 PS	Prescaler Factor
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

Count Mode and Modulo - Counting Up

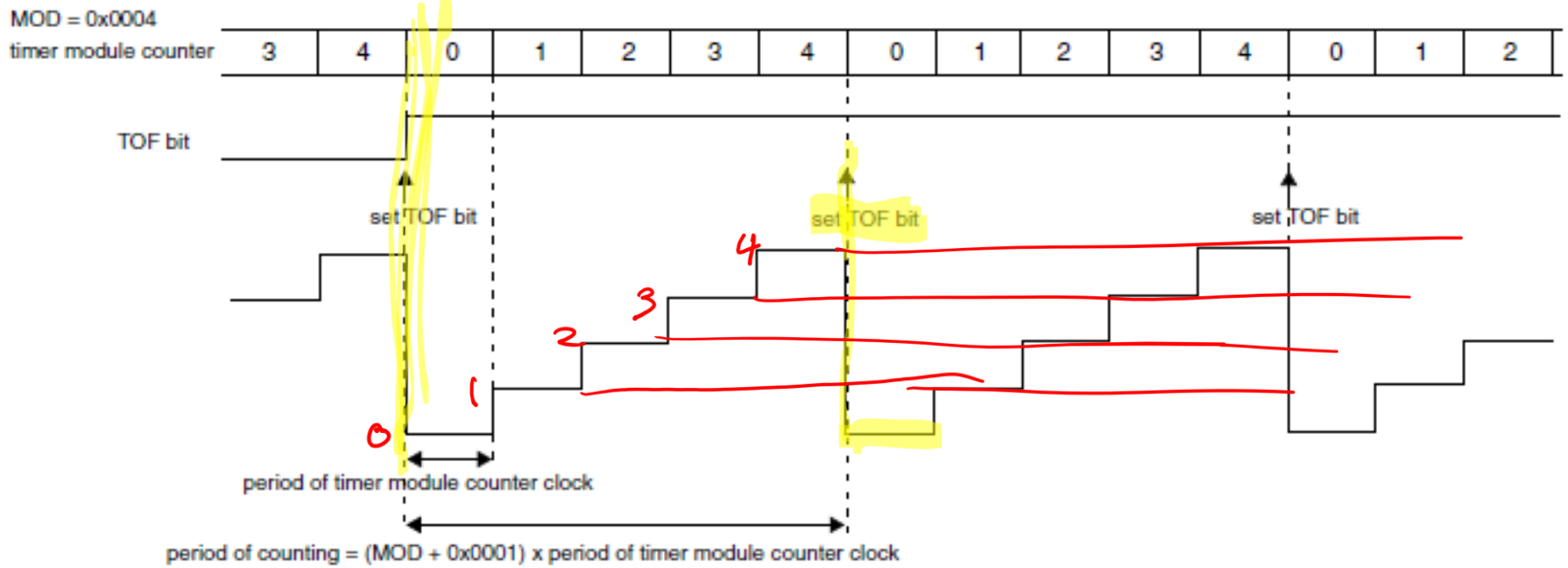


Figure 31-79. Example of TPM Up Counting

- Counter increments with each clock tick
- When counter reaches MOD,
  - set TOF bit (timer overflow)
  - reset counter value to 0
- Frequency of overflows is timer clock frequency / (1 + MOD)

Count Mode and Modulo - Counting Up and Down

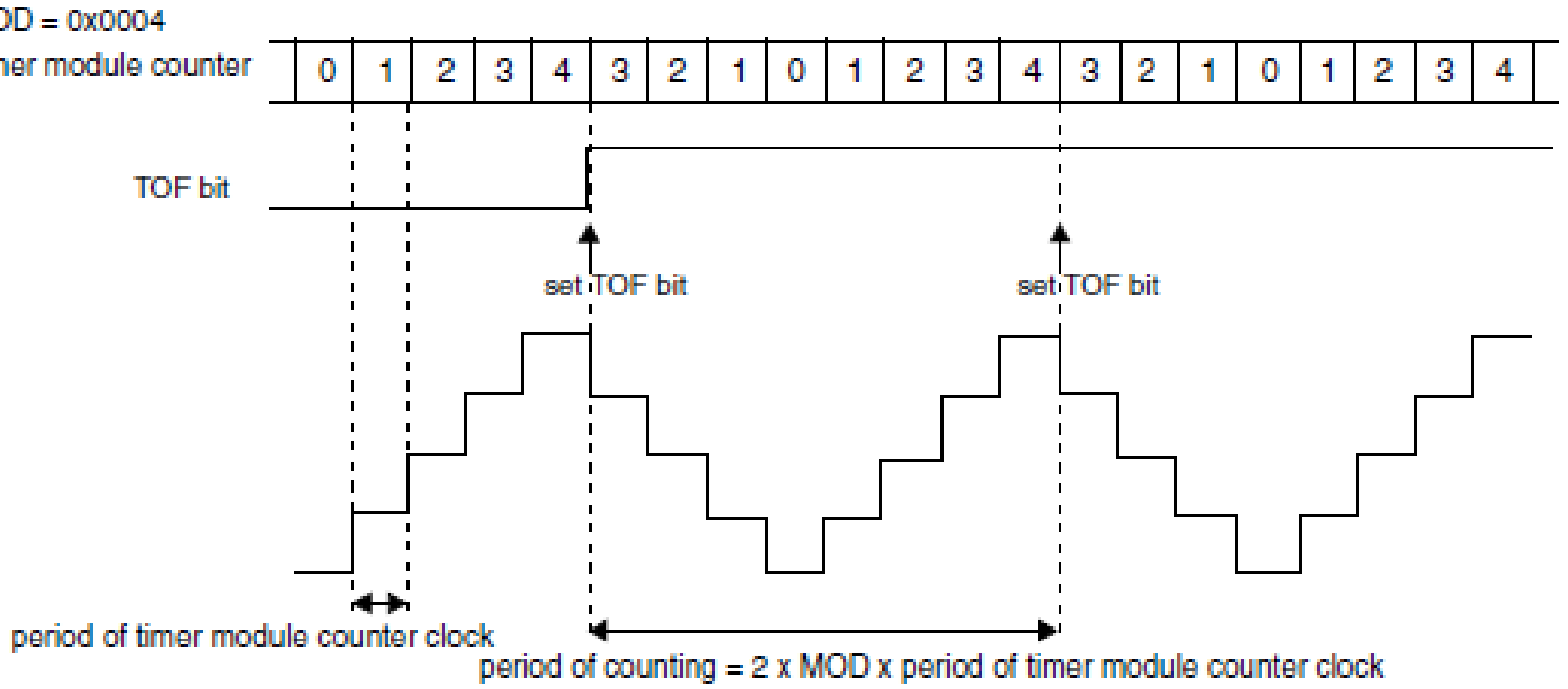
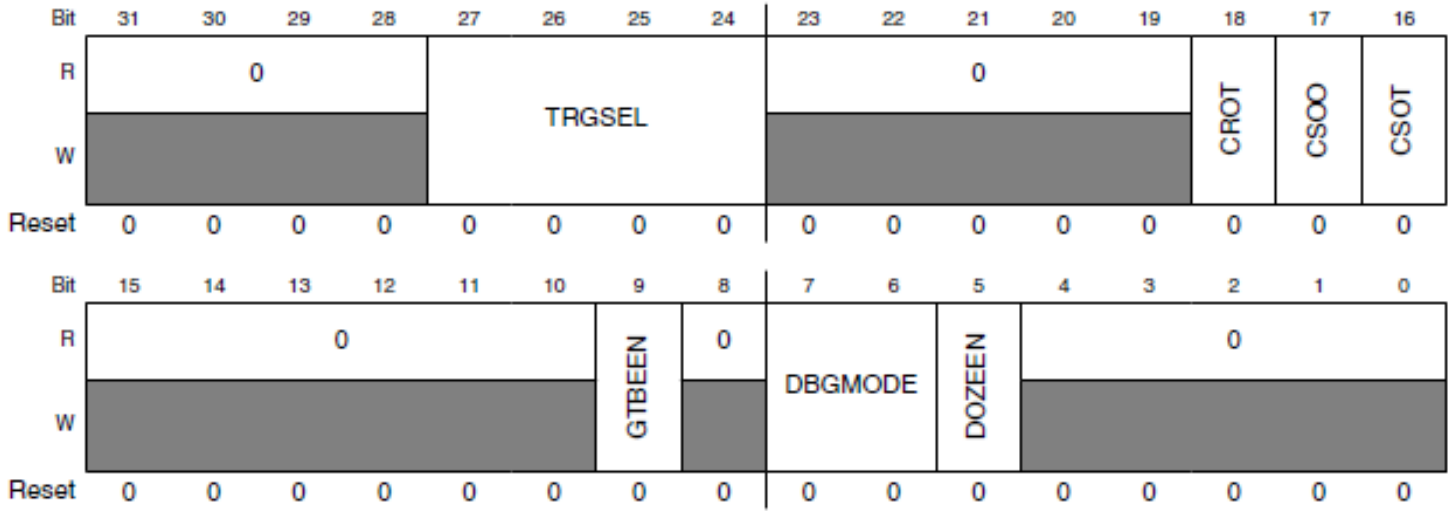


Figure 31-80. Example of Up-Down Counting

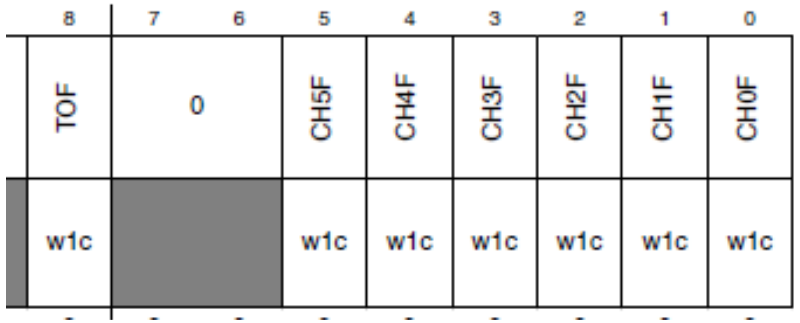
- Up-counting phase
  - Counter increments with each clock tick
  - When counter reaches MOD, set TOF bit (timer overflow), set to down-count mode
- Down-counting phase
  - Counter decrements with each clock tick
  - When counter reaches 0, set to up-count mode
- Frequency of overflows is timer clock frequency / (2 * MOD)

TPM Configuration (TPMx_CONF)



- TRGSEL - input trigger select
- CROT - counter reload on trigger
- CSOO - counter stop on overflow
- CSOT - counter start on trigger
- GTBEEN - external global time base enable (rather than LPTPM counter)
- DBGMODE - let LPTPM counter increment during debug mode
- DOZEEN - pause LPTPM when in doze mode

TPM Status (TPMx_STATUS)



8	7	6	5	4	3	2	1	0
TOF	0		CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
w1c			w1c	w1c	w1c	w1c	w1c	w1c

- TOF - Counter has overflowed
- CHxF - Channel event has occurred (event depends on mode)

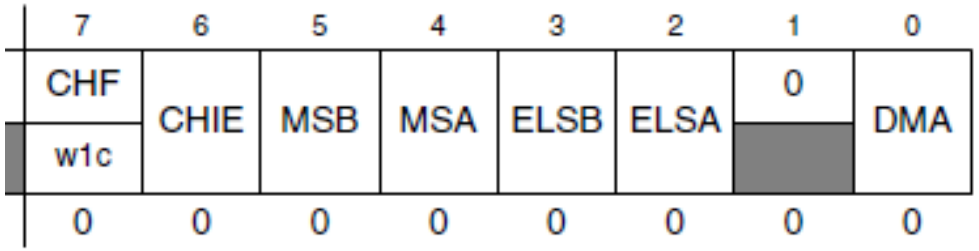
Major Channel Modes

NC STATE UNIVERSITY

- Input Capture Mode
  - Capture timer's value when input signal changes
    - Rising edge, falling edge, both
  - How long after I started the timer did the input change?
    - Measure time delay
- Output Compare Mode
  - Modify output signal when timer reaches specified value
    - Set, clear, pulse, toggle (invert)
  - Make a pulse of specified width
  - Make a pulse after specified delay
- Pulse Width Modulation
  - Make a series of pulses of specified width and frequency

Channel Configuration and Value

- Configuration: TPMx_CnSC



7	6	5	4	3	2	1	0
CHF	CHIE	MSB	MSA	ELSB	ELSA	0	DMA
w1c							
0	0	0	0	0	0	0	0

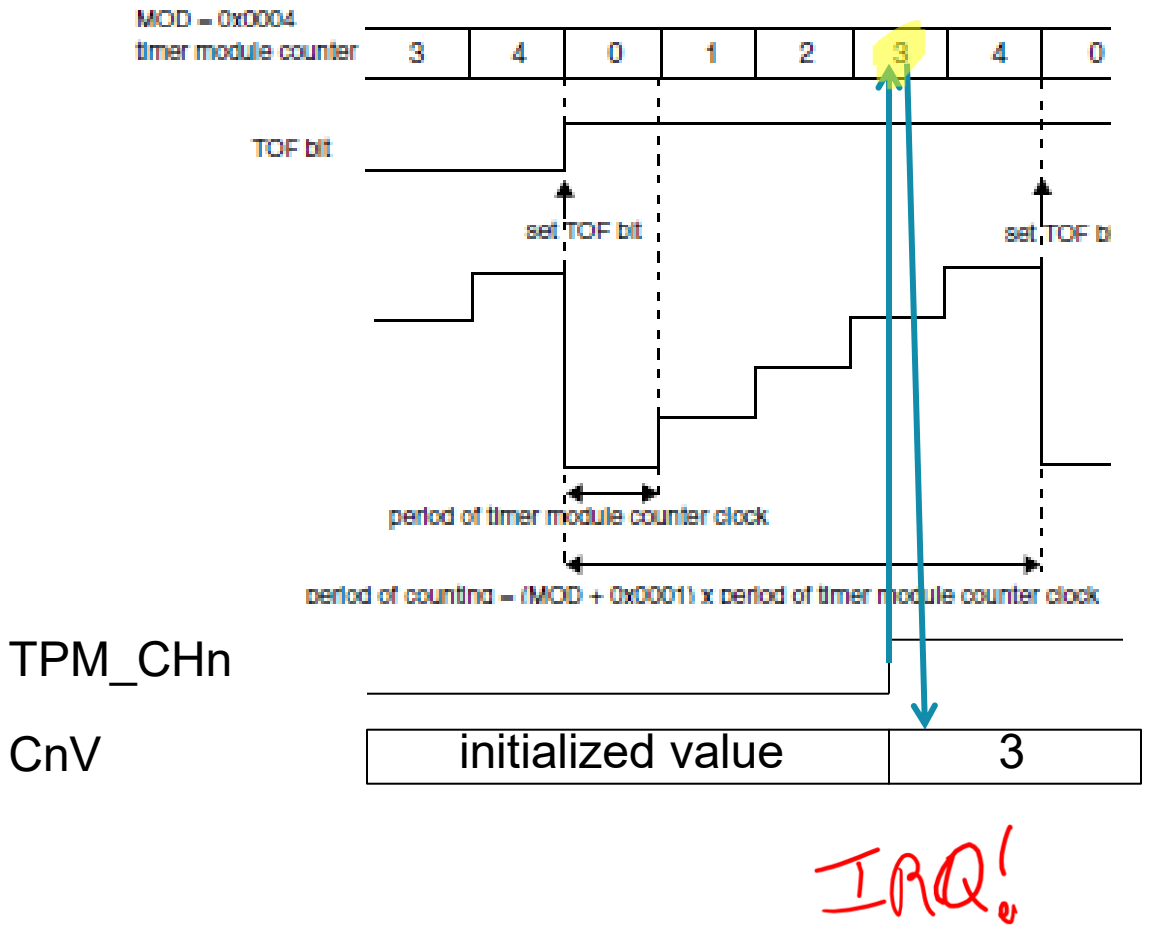
  - CHF - set when event occurs
  - CHIE - enable channel to generate an interrupt
  - MSB:MSA - mode select
  - ELSB:ELSA - edge or level select
  - DMA - enable DMA transfers

- Value: TPMx_CnV
  - 16-bit value for output compare or input capture

Input Capture Mode

- Select mode with CPWMS = 0, MSnB:MSnA = 00
- TPM_CHn I/O pin operates as edge-sensitive input
  - ELSnB:ELSnA select rising (01) or falling edge (10) or both (11)

- When valid edge is detected on TPM_CHn…
  - Current value of counter is stored in CnV
  - Interrupt is enabled (if CHnIE = 1)
  - CHnF flag is set (after 3 clock delay)



Wind Speed Indicator (Anemometer)

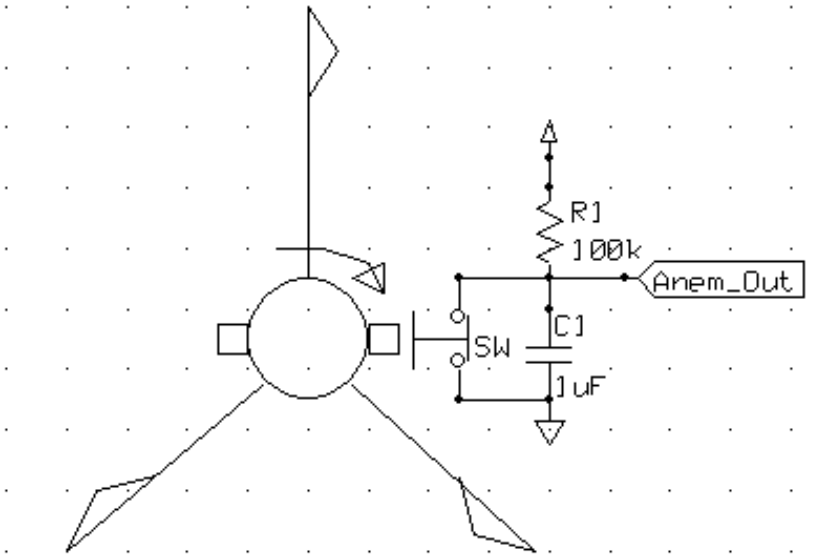
- Rotational speed (and pulse frequency) is proportional to wind velocity

- Two measurement options:
  - Frequency (best for high speeds)
  - Width (best for low speeds)

- Can solve for wind velocity v

$$v_{wind} = \frac{K * f_{clk}}{T_{anemometer}}$$

- How can we use the TPM for this?
  - Use Input Capture Mode to measure period of input signal



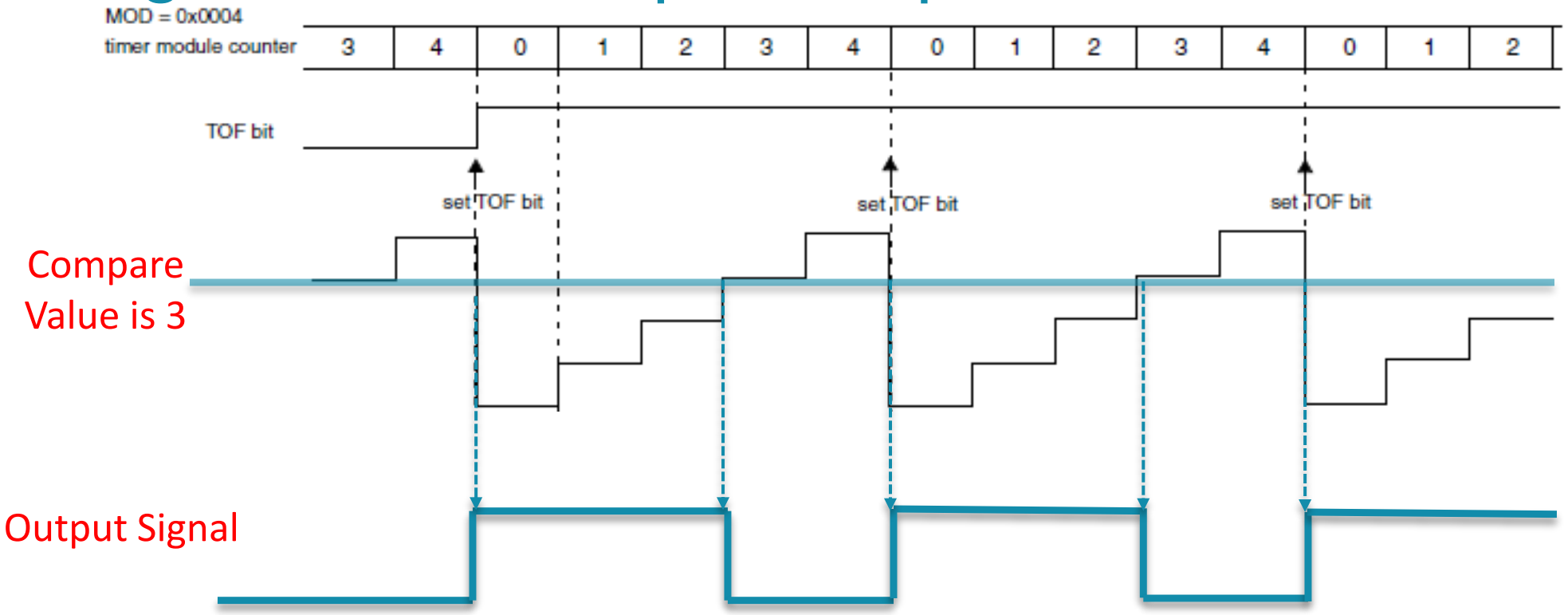
TPM Capture Mode for Anemometer

NC STATE UNIVERSITY

- Configuration
  - Set up TPM to count at given speed from internal clock
  - Set up TPM channel for input capture on rising edge

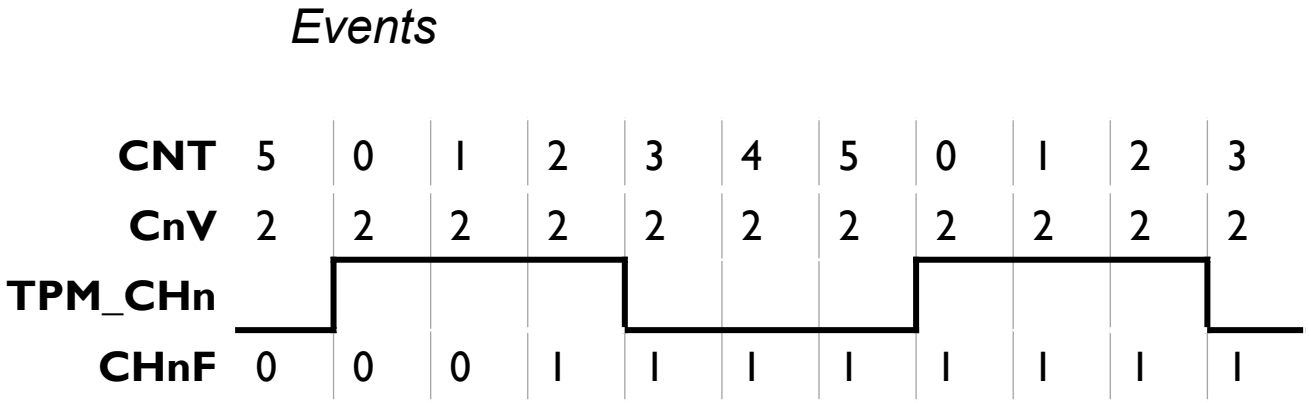
- Operation: Repeat
  - First TPM interrupt - on rising edge
    - Reconfigure channel for input capture on falling edge
    - Clear TPM counter, start it counting
  - Second TPM interrupt - on falling edge
    - Read capture value from CnV, save for later use in wind speed calculation
    - Reconfigure channel for input capture on rising edge
    - Clear TPM counter, start it counting

Creating PWM with Output Compare Mode



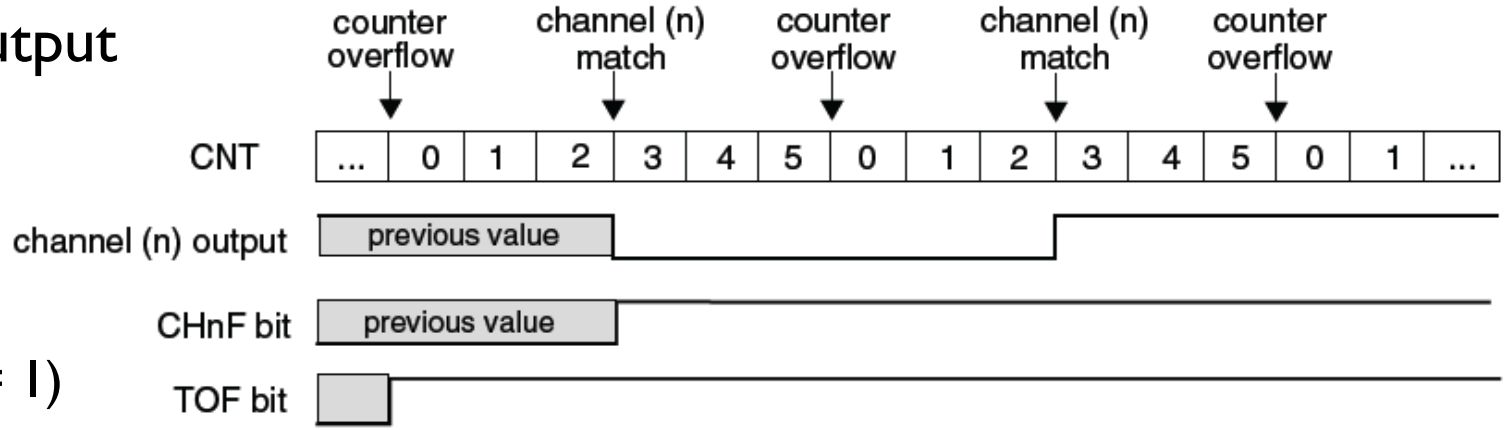
- Basic idea
  - Set PWM output to 1 when counter starts (e.g. at 0)
  - Clear PWM output to 0 when counter reaches channel's specified compare value (e.g. 3)
- Resulting pulse width is proportional to channel's compare value

Output Compare Mode



MSnB	ELSnB	ELSnA	Output Action
0	0	1	Toggle
0	0	0	Clear
0	1	1	Set
1	1	0	Pulse low
1	x	1	Pulse high

- Select mode with CPWMS = 0, MSnA = 1
- TPM_CHn I/O pin operates as output
- When CNT matches CnV …
  - Output signal TPM_CHn is updated
  - CHnF flag is set
  - CHnI Interrupt is enabled (if CHnIE = 1)
- Can select one of multiple output signal actions on match
  - Toggle, clear, set, pulse low, pulse high

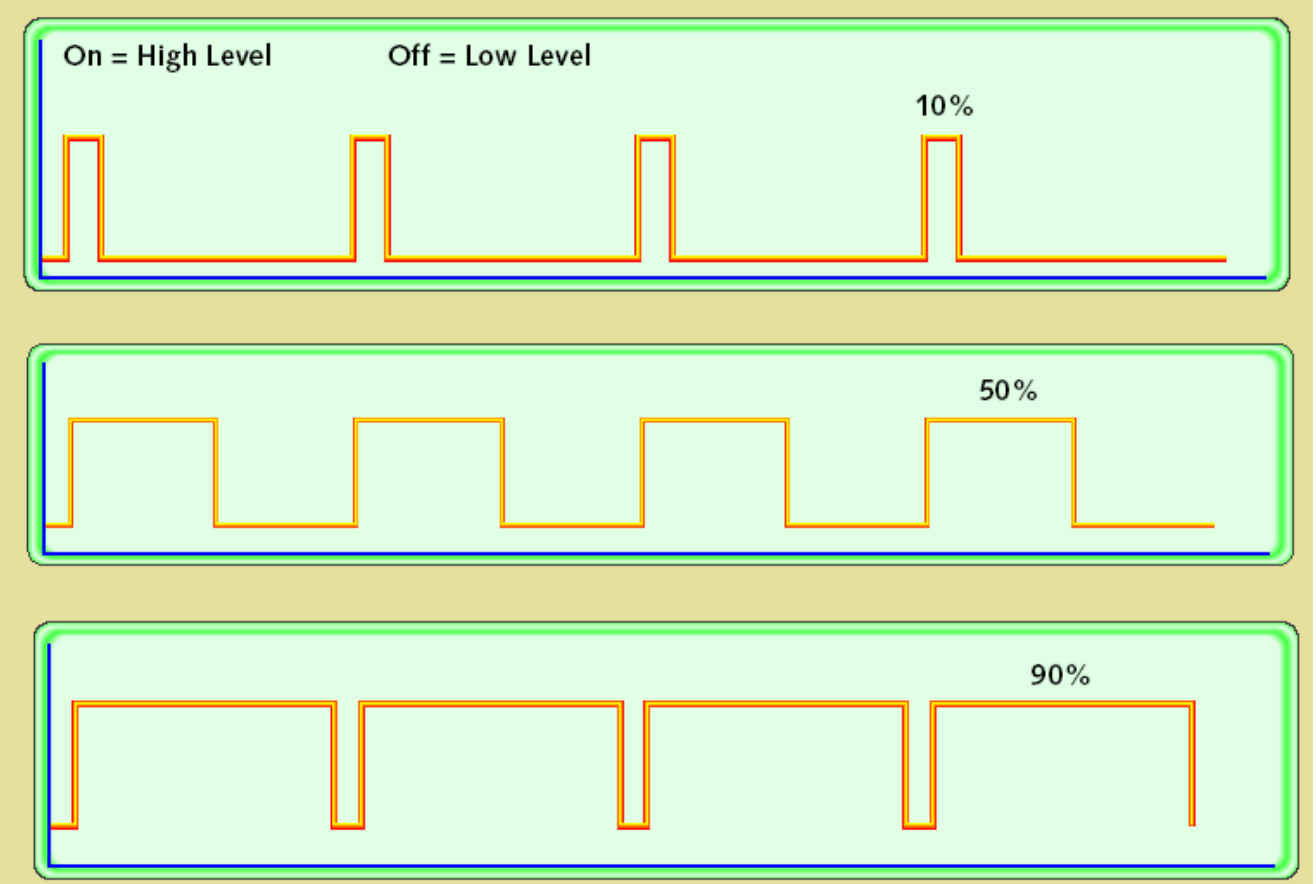


Pulse-Width Modulation

NC STATE UNIVERSITY

- Allows a single digital signal to send more than two values (0, 1)
- Simple encoding: value is the fraction of time signal is a 1
- Signal can easily be averaged to create an analog voltage

- PWM signal characteristics
  - Modulation frequency – how many pulses occur per second (fixed)
  - Period – 1/(modulation frequency)
  - On-time – amount of time that each pulse is on (asserted)
  - Duty-cycle – on-time/period
  - Adjust *on-time* (hence *duty cycle*) to represent the analog value

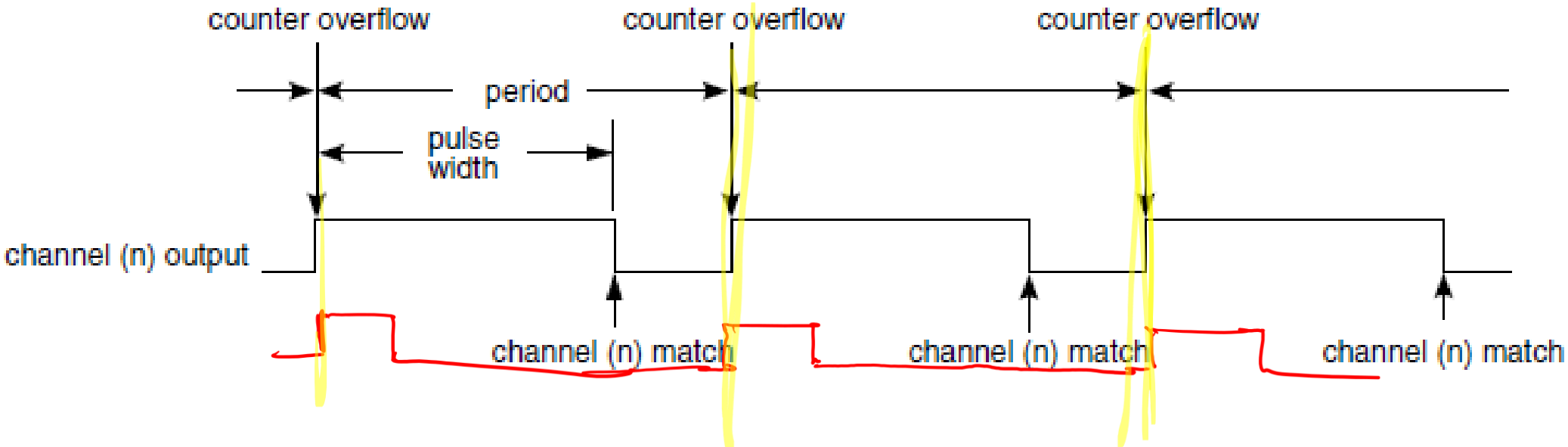


Uses of Pulse-Width Modulation

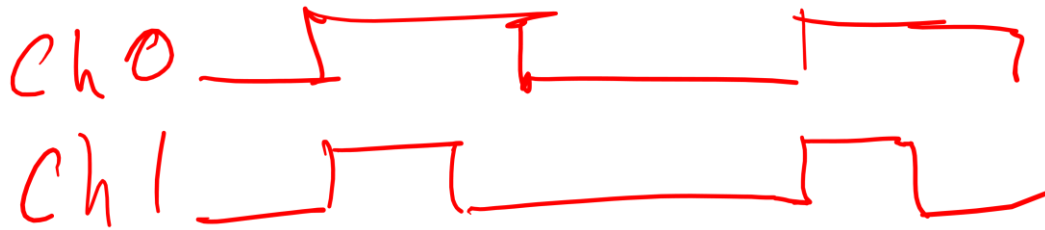
NC STATE UNIVERSITY

- **Digital communication** is less sensitive to noise than analog methods
  - PWM provides a *digital encoding* of an *analog* value
  - Much less vulnerable to noise
- **Digital power amplifiers** are more efficient and less expensive than analog power amplifiers
  - Applications: motor speed control, light dimmer, switch-mode power conversion
  - Load (motor, light, etc.) responds slowly, averages PWM signal

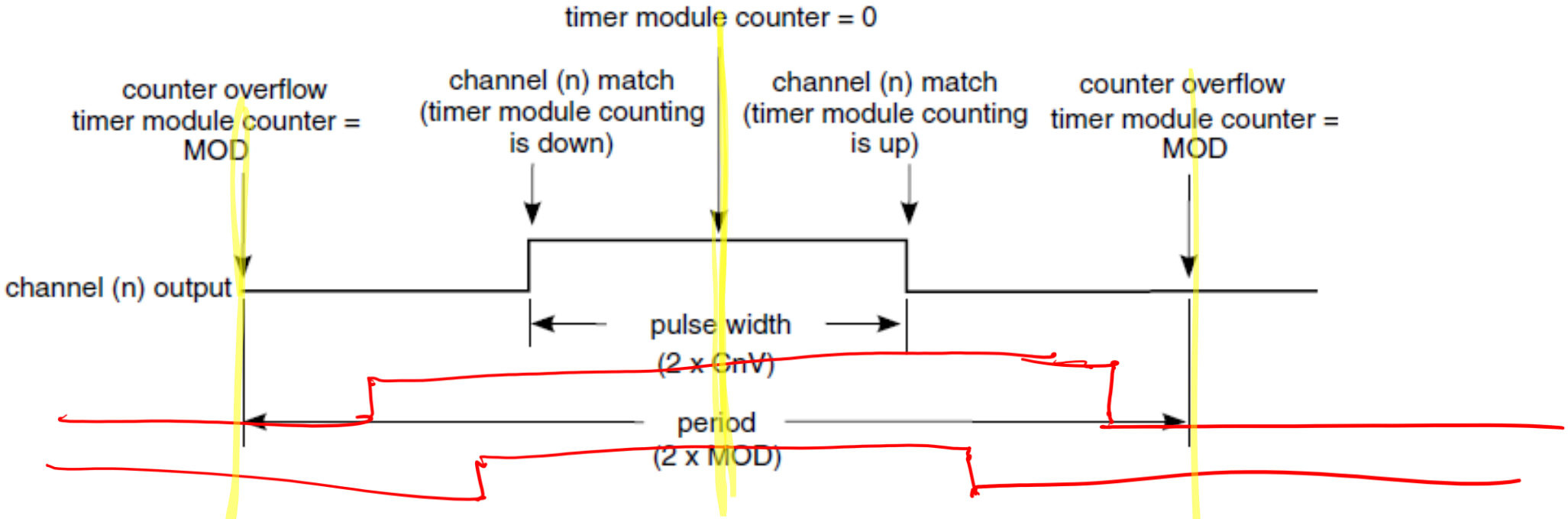
Edge-Aligned PWM Mode



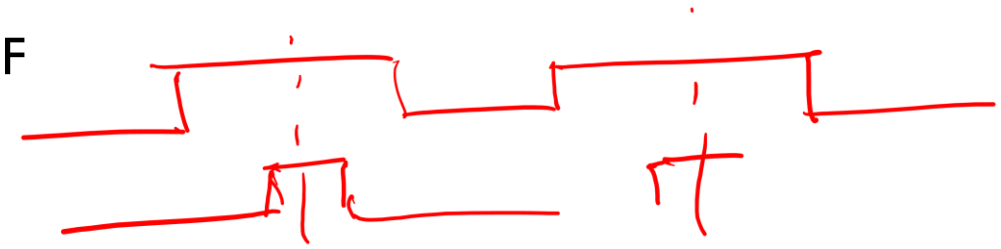
- Edge-aligned - leading edges of signals from all PWM channels are aligned
  - Uses count up mode
  - Period = (MOD + 1) cycles
  - Pulse width = (CnV) cycles
- MSnB:MSnA = 01, CPWMS = 0
  - ELSnB:ELSnA = 10 - high-true pulses
  - ELSnB:ELSnA = x1 - low-true pulses



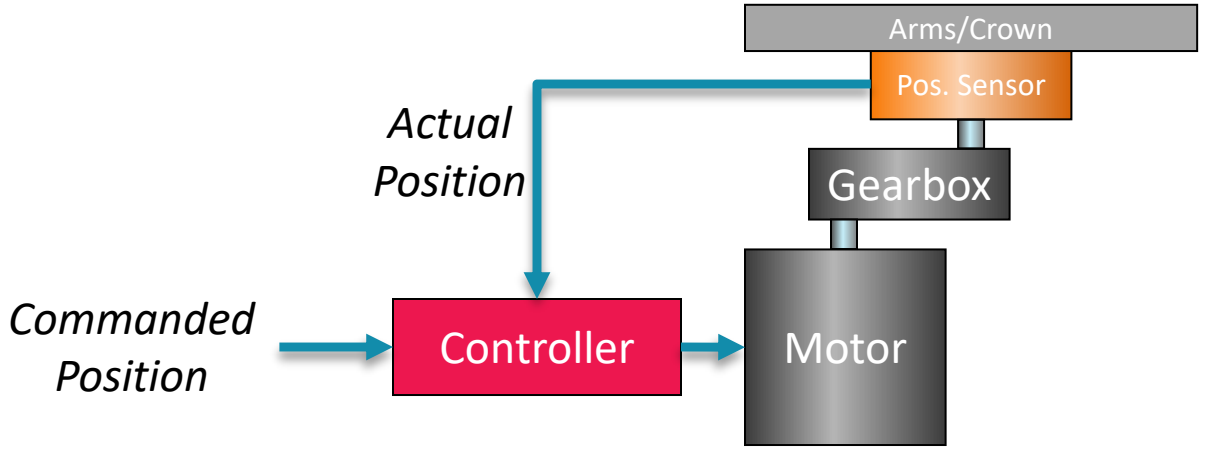
Center-Aligned PWM Mode



- Center-aligned - centers of signals from all PWM channels are aligned
  - Uses count up/down mode
  - Period = 2*MOD cycles. 0x0001 <= MOD <= 0x7FFF
  - Pulse width = 2*CnV cycles
- MSnB:MSnA = 10, CPWMS = 1
  - ELSnB:ELSnA = 10 - high-true pulses
  - ELSnB:ELSnA = x1 - low-true pulses



Servo Motor

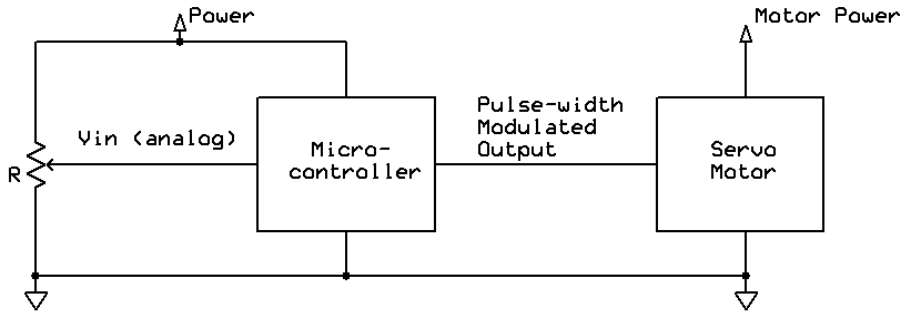


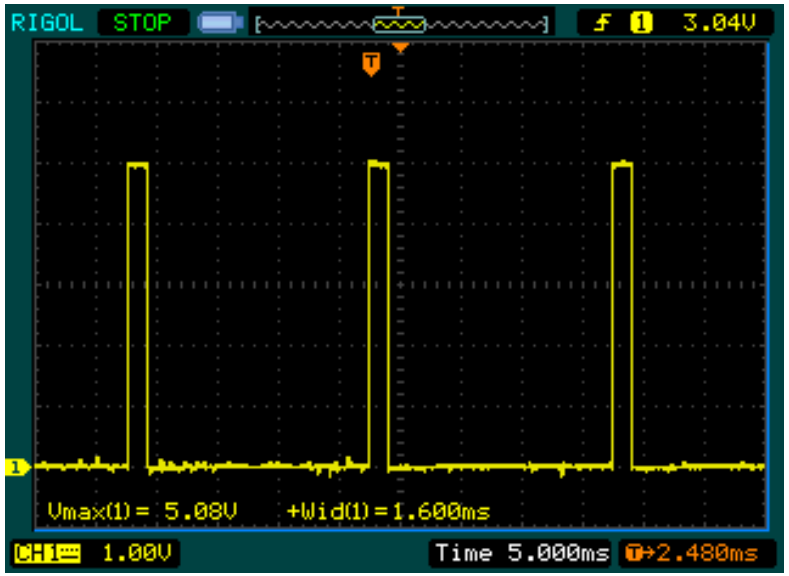
- Components
  - Motor
  - Gearbox
  - Output shaft with crown and position sensor
  - Control system
- Inputs
  - Power & Ground
  - PWM control signal
- Control system
  - Moves motor to commanded position
  - Uses position sensor for feedback
- Gearing from motor to output shaft
  - Increases torque, reduces speed
- Limited range of rotation (e.g. 90^0), can't do full rotation

Using PWM to Drive a Servo Motor

NC STATE UNIVERSITY

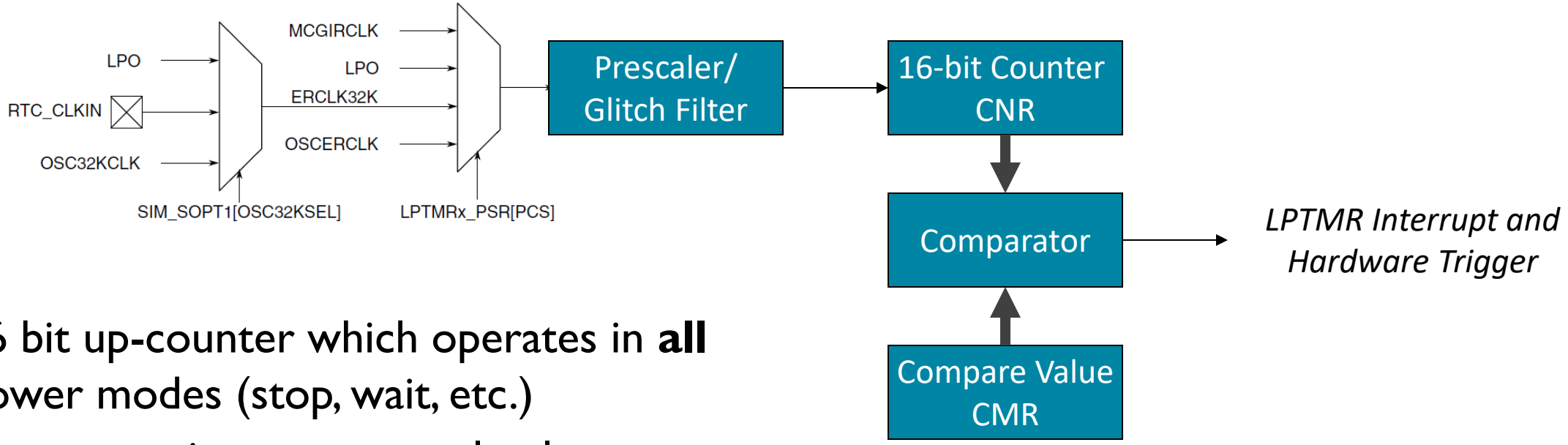
- PWM signal tells servo desired shaft angle
  - 20 ms period (50 Hz frequency)
  - 1 to 2 ms pulse width
- Position proportional to pulse width
  - 1.5 ms: centered (neutral)
  - <1.5 ms: counter-clockwise
  - >1.5 ms: clockwise





LOW POWER TIMER (LPTMR)

LPTMR Overview



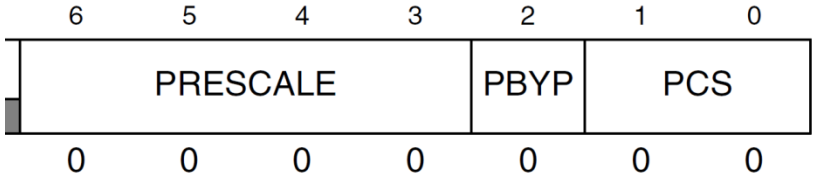
- 16 bit up-counter which operates in **all** power modes (stop, wait, etc.)
- Can count time or external pulses
- Glitch filter removes high-frequency noise from pulses
- Can generate interrupt when counter matches compare value
- Interrupt wakes MCU from any low power mode

- Registers
  - Counter register LPTMRx_CNR
  - Compare register LPTRMx_CMR
  - Prescale register LPTMRx_PSR
  - Control Status register LPTMRx_CSR

Prescale Register

NC STATE UNIVERSITY

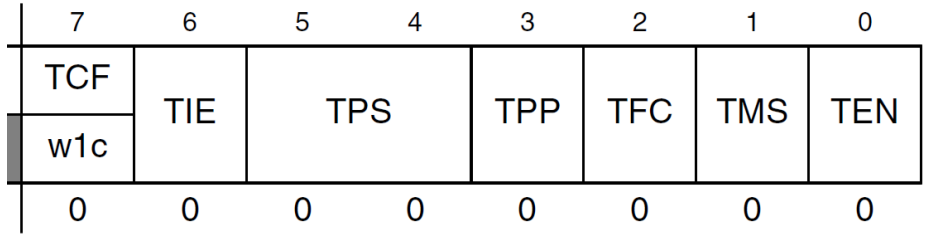
- PRESCALE: divide by 2 to 65536
  - Time counter mode: Divide input clock by $2^{PRESCALE+1}$
  - Pulse counter mode: Is glitch filter which recognizes input signal change after $2^{PRESCALE}$ rising clock cycles
- PBYP: Prescaler Bypass
  - 0: use prescaler
  - 1: bypass prescaler
- PCS: Prescaler Count Select
  - Inputs available depend on chip configuration, see KL25 SRM Chapter 3: Chip Configuration



PCS	Clock Source
00	MCGIRCLK — internal reference clock (not available in LLS and VLLS modes)
01	LPO — 1 kHz clock (not available in VLLS0 mode)
10	ERCLK32K (not available in VLLS0 mode when using 32 kHz oscillator)
11	OSCERCLK — external reference clock (not available in VLLS0 mode)

Control Status Register

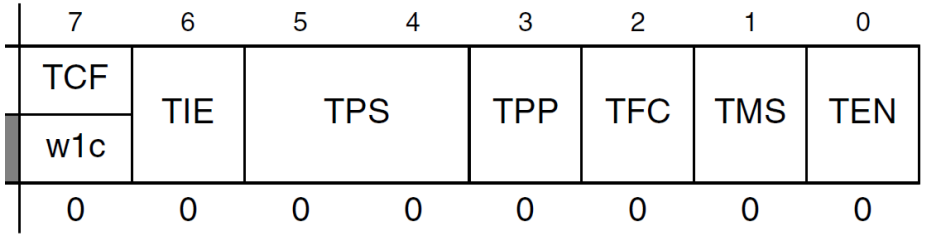
NC STATE UNIVERSITY



- TCF: Timer Compare Flag
  - 1 if CNR matches CMR and increments
- TIE: Timer Interrupt Enable
  - Set to 1 to enable interrupt when TCF == 1
- TPS: Timer Pin Select for pulse counter mode
  - Inputs available depend on chip configuration, see KL25 SRM Chapter 3: Chip Configuration

LPTMR_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0 output
01	1	LPTMR_ALT1 pin
10	2	LPTMR_ALT2 pin
11	3	LPTMR_ALT3 pin

Control Status Register



7	6	5	4	3	2	1	0
TCF	TIE	TPS		TPP	TFC	TMS	TEN
w1c							
0	0	0	0	0	0	0	0

- TPP: Timer Pin Polarity
  - 0: input is active high, increments CNR on rising edge
  - 1: input is active low, increments CNR on falling edge
- TFC: Timer Free-running Counter
  - 0: Reset CNR whenever TCF is set (on match)
  - 1: Reset CNR on overflow (wrap around)
- TMS: Timer Mode Select
  - 0: Time counter
  - 1: Pulse counter
- TEN: Timer Enable
  - 1: Enable LPTMR operation