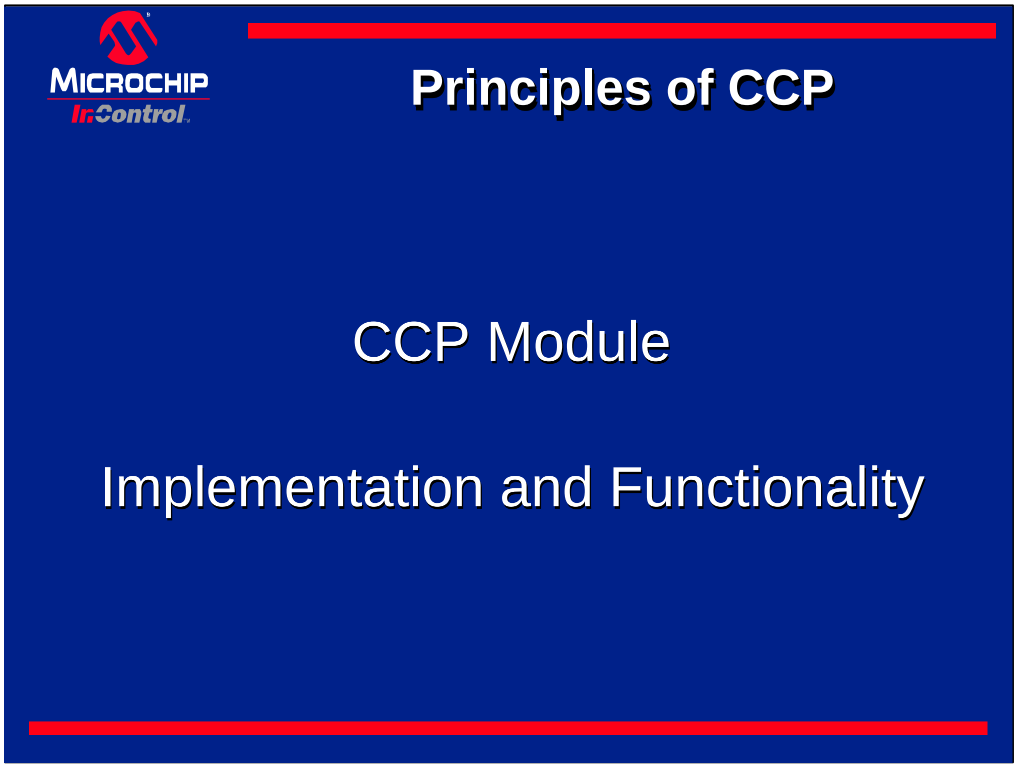
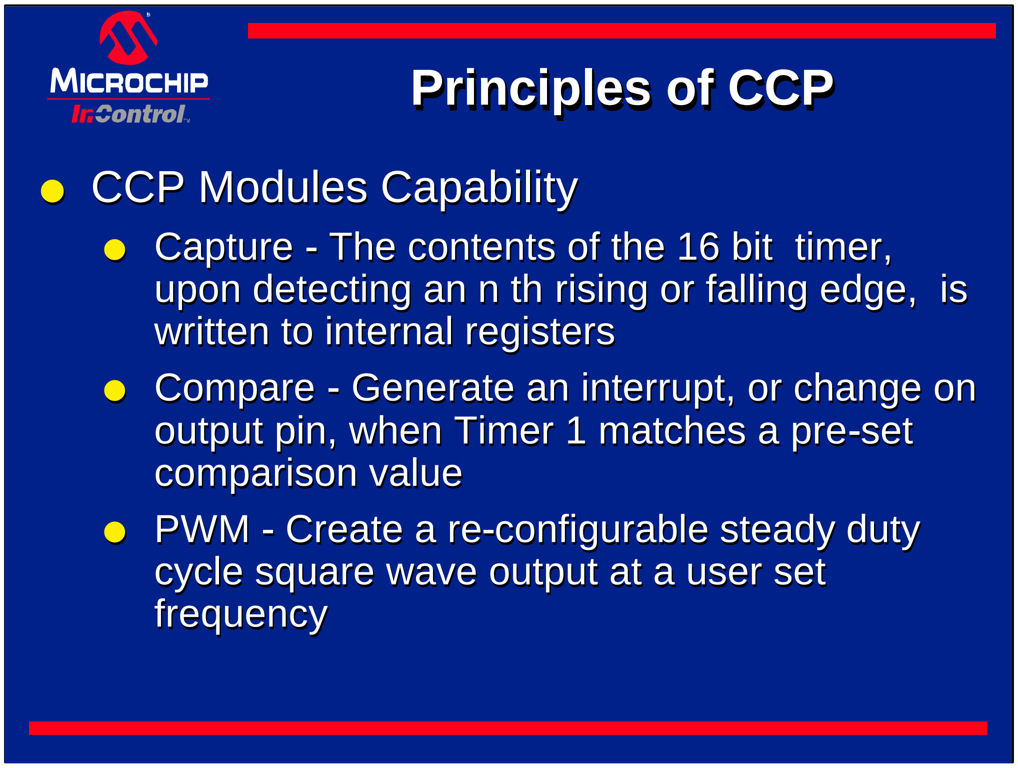


Principles of CCP

CCP Module

Implementation and Functionality

Welcome to Getting Started module on Capture, Compare and Pulse Width Modulation feature, abbreviated CCP. The CCP module will discuss initialization and function of on-board CCP resources



Principles of CCP

- CCP Modules Capability
  - Capture - The contents of the 16 bit timer, upon detecting an n th rising or falling edge, is written to internal registers
  - Compare - Generate an interrupt, or change on output pin, when Timer 1 matches a pre-set comparison value
  - PWM - Create a re-configurable steady duty cycle square wave output at a user set frequency

A capture, compare, and PWM, module, or CCP for short, is designed into the PicMicro to assist with measurement or control of time based pulse signals.

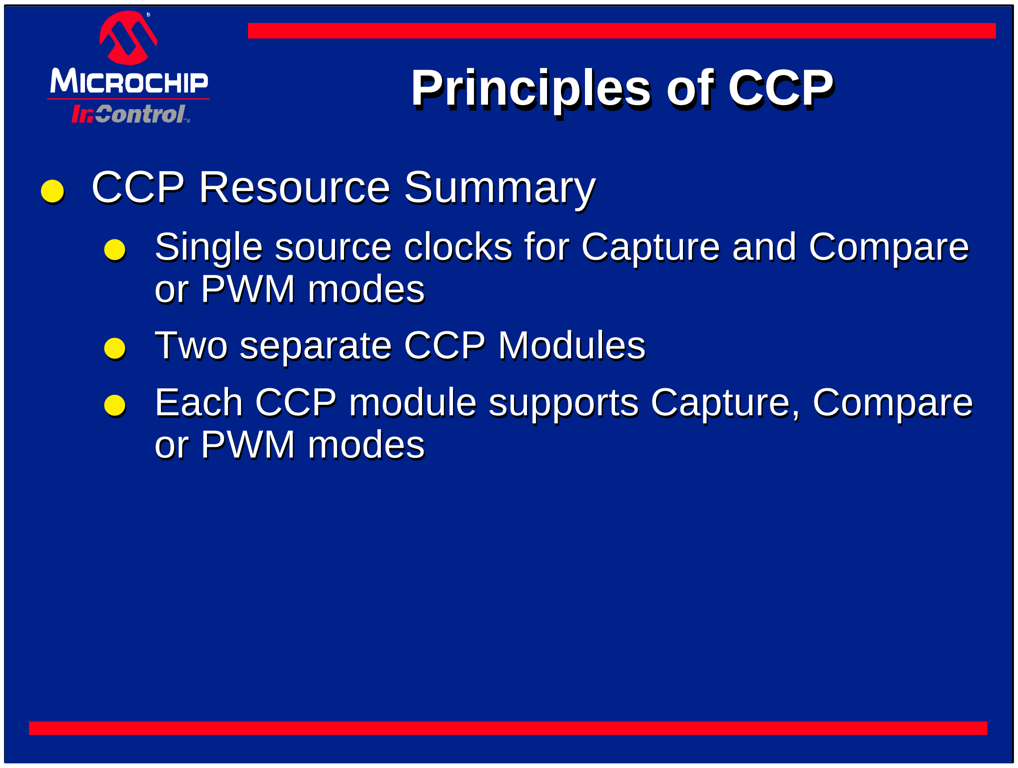
Capture mode causes the contents of an internal 16 bit timer, upon detecting an n th rising or falling edge, to be written to on-board special function registers

Compare mode generates an interrupt, or change on output pin, when Timer 1 matches a pre-set comparison value

PWM, or formally pulse width modulation, mode creates a re-configurable square wave duty cycle output at a user set frequency. The application software can change the duty cycle or period by modifying the value written to specific special function register.

Future slides in the Principles of CCP module describes how the different modes are set and operate.

Note, the CCP module integrates with Timer one and two to perform operation. Therefore, a review Timer slide may be helpful after viewing the CCP module.



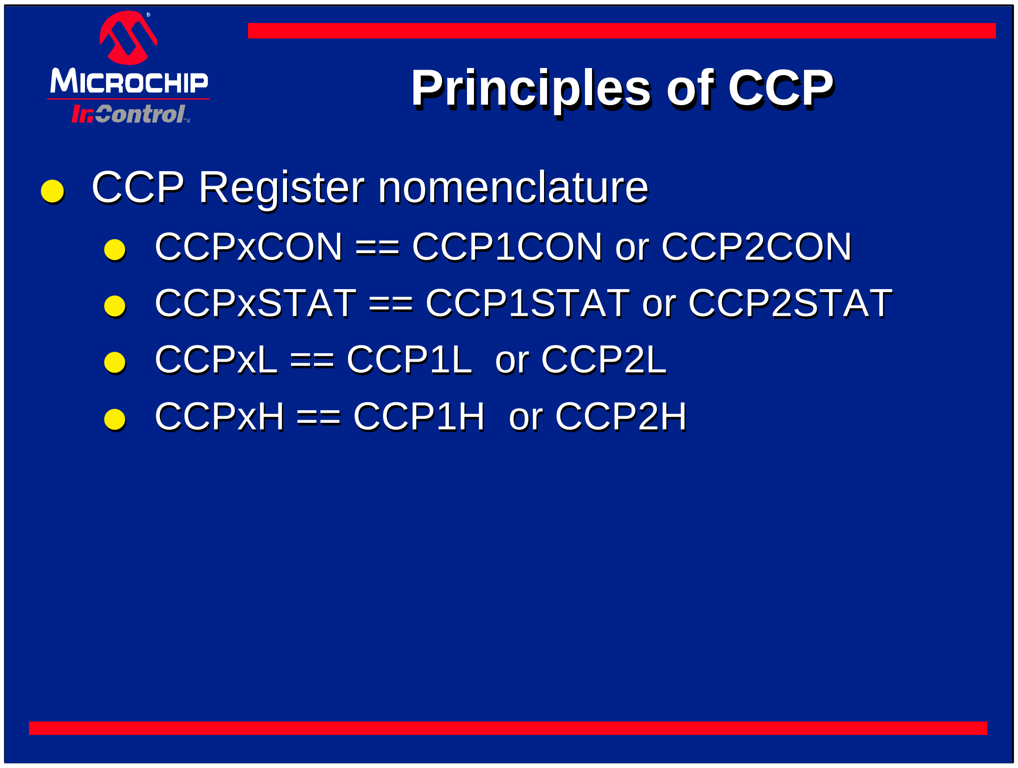
Principles of CCP

- CCP Resource Summary
  - Single source clocks for Capture and Compare or PWM modes
  - Two separate CCP Modules
  - Each CCP module supports Capture, Compare or PWM modes

The mid-range Pic micro-controller can contain up to two CCP modules. Each module operates independently, and supports capture, compare or PWM mode. Operation is synchronized to internal clocks. A single clock source is shared between all available Capture and Compare or PWM CCP modules on chip.

A CCP Module's mode is determined by the designer's specified options in the CCPxCON register, where 'x' will be '1' or '2' . We will begin with examining CCPxCON configurations, and will continue to the initialization, operation and design notes for each of the CCP modes.

When starting into any new design and chip selection, please check the Microchip web site(www.microchip.com) for errata sheets discussing known deviations from the original datasheet.



Principles of CCP

- CCP Register nomenclature
  - CCPxCON == CCP1CON or CCP2CON
  - CCPxSTAT == CCP1STAT or CCP2STAT
  - CCPxL == CCP1L or CCP2L
  - CCPxH == CCP1H or CCP2H

A quick overview on nomenclature used in the CCP Module and other documents supplied by Microchip.

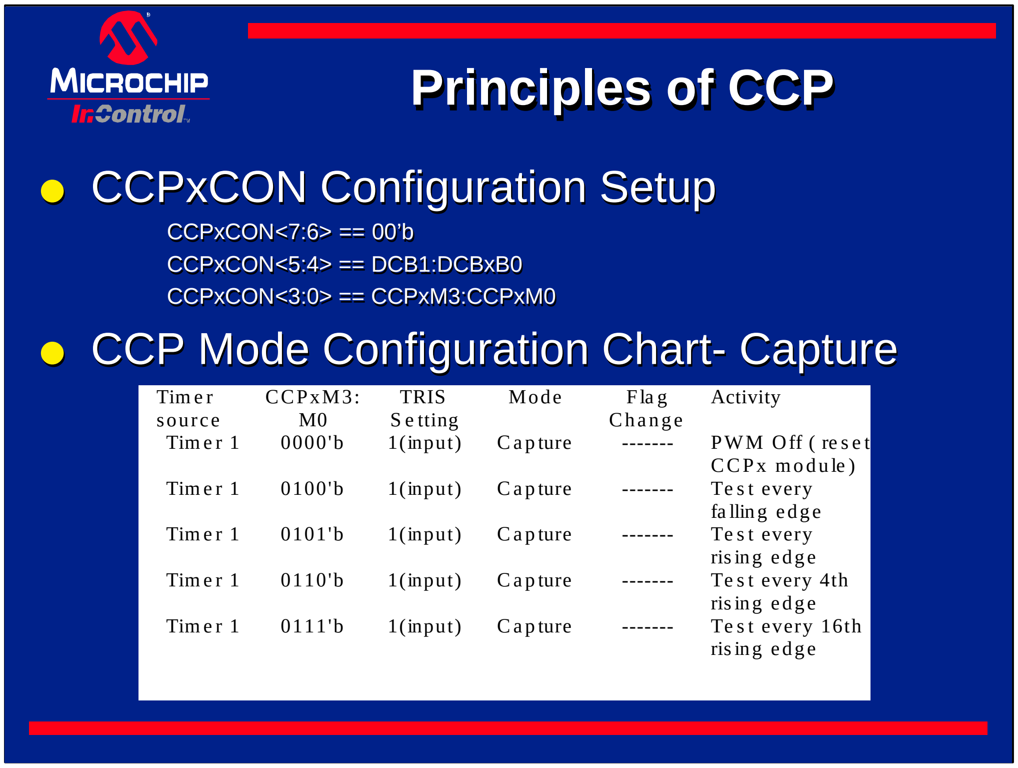
As a general rule, if the part contains two or more modules with duplicate functionality and special function registers, like TXREG1 and TXREG2, the use of an index variable 'x' with be used in documentation to refer to module 1 or module 2 register. To re-examine our TXREG1 and TXREG2 example, the data sheet will commonly refer to TXREGx as either TXREG1 or TXREG2.

Other examples centered toward CCP dual module documentation:

CCPxCON - refers to CCP one or CCP two control register, supporting the mode selection and access to the two least significant bits..

CCPxL - refers to CCP one or CCP two is used to store low byte value in Capture or compare operation and as entry register in PWM mode.

CCPxH - refers to CCP one or CCP two used to store high Byte value in capture or compare mode and as buffer in PWM mode.



Principles of CCP

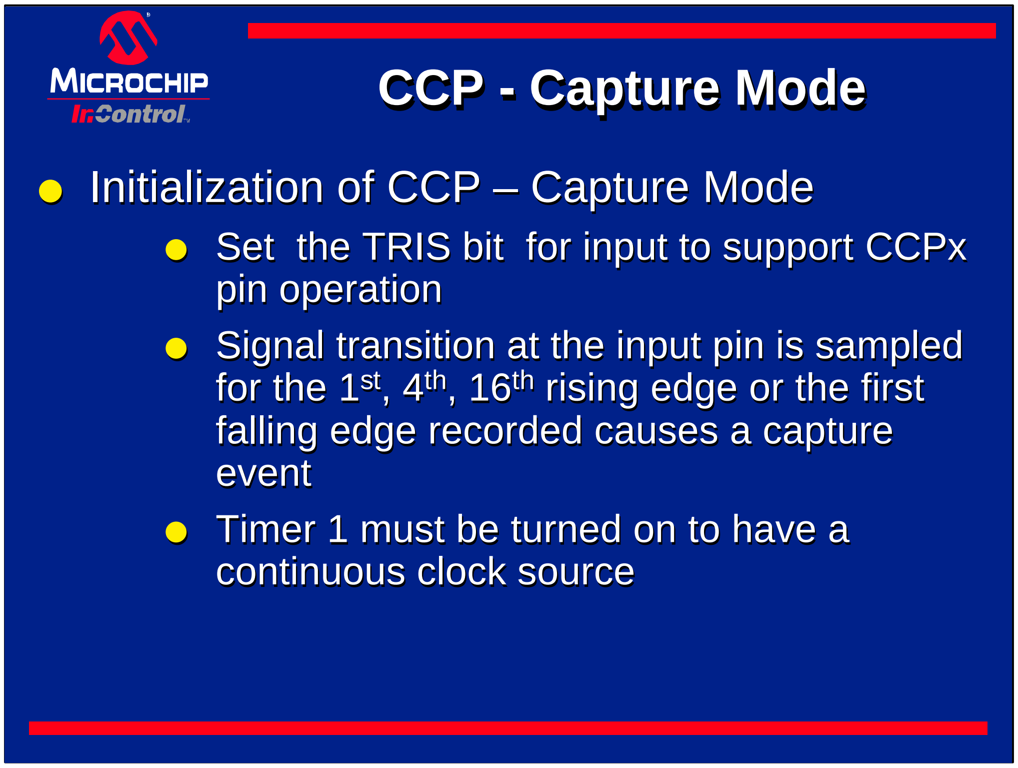
- CCPxCON Configuration Setup
  - CCPxCON<7:6> == 00'b
  - CCPxCON<5:4> == DCB1:DCBxB0
  - CCPxCON<3:0> == CCPxM3:CCPxM0
- CCP Mode Configuration Chart- Capture

Timer source	CCPxM3: M0	TRIS Setting	Mode	Flag Change	Activity
Timer 1	0000'b	1(input)	Capture	-------	PWM Off (reset CCPx module)
Timer 1	0100'b	1(input)	Capture	-------	Test every falling edge
Timer 1	0101'b	1(input)	Capture	-------	Test every rising edge
Timer 1	0110'b	1(input)	Capture	-------	Test every 4th rising edge
Timer 1	0111'b	1(input)	Capture	-------	Test every 16th rising edge

Each CCPxCON register contains mode control bits(CCPxM3:CCPxM0) and the two least significant bits of the PWM pulse.

For all cases, CCPxCON bit 7 and 6 are unused and zero is written to these locations when doing a byte write to CCPxCON. CCPxCON bits 5 and 4 are least significant bits for PWM mode, and are "don't care" in capture or compare mode. By writing to bits CCPxM3:CCPxM0, a designer can select the desired mode of operation.

A more detailed explanation for each of modes is discussed later in the CCP training module



CCP - Capture Mode

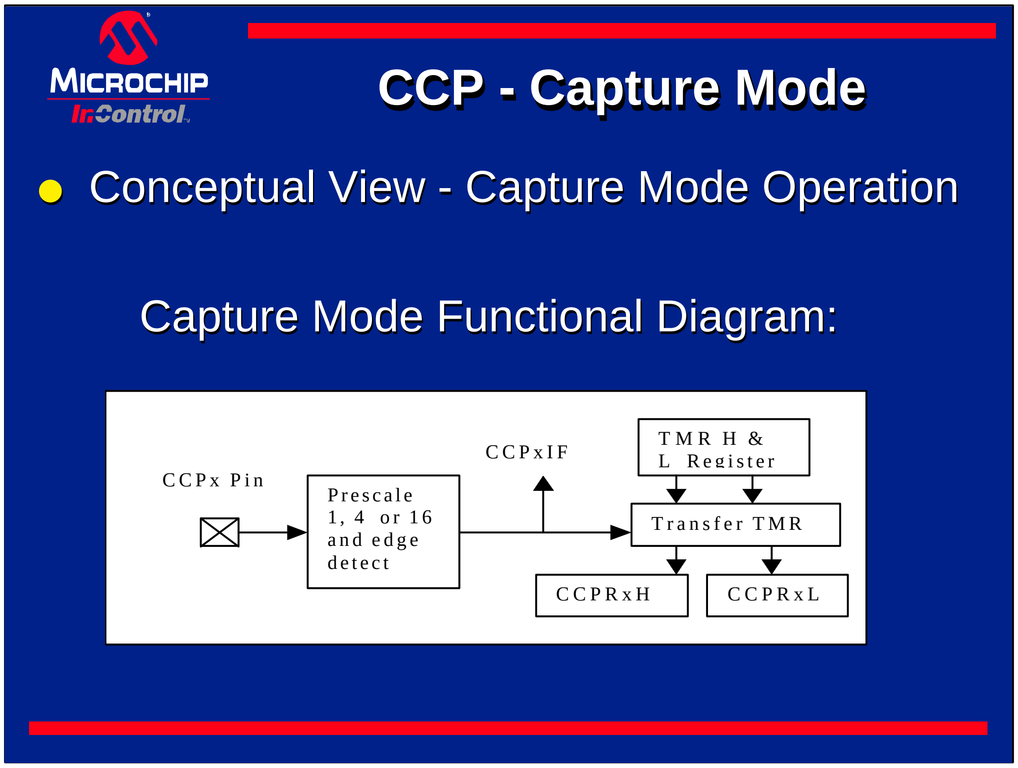
- Initialization of CCP – Capture Mode
  - Set the TRIS bit for input to support CCPx pin operation
  - Signal transition at the input pin is sampled for the 1st, 4th, 16th rising edge or the first falling edge recorded causes a capture event
  - Timer 1 must be turned on to have a continuous clock source

Before using the CCP module for operation, initialization of CCPx pin data direction, selection of CCP mode, and timer 1 or timer 2 setup and is required.

When configuring for capture mode, the TRIS bit, corresponding to the CCPx pin, must be “SET” to support detection of incoming pulses.

Capture mode is selected when CCPxM3:CCPxM2 =b’ 01, and the number of required rising edges, divided by 1,4 or 16, or a single falling edge required before the capture event will trigger can be set from CCPxM1:CCPxM0.

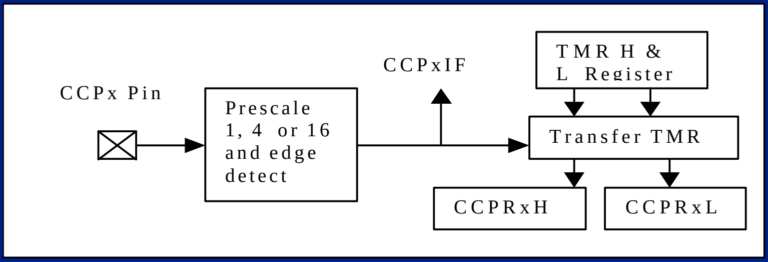
Since capture mode depends on Timer one as its reference clock, timer one must be enabled and configured for continuous clock source.



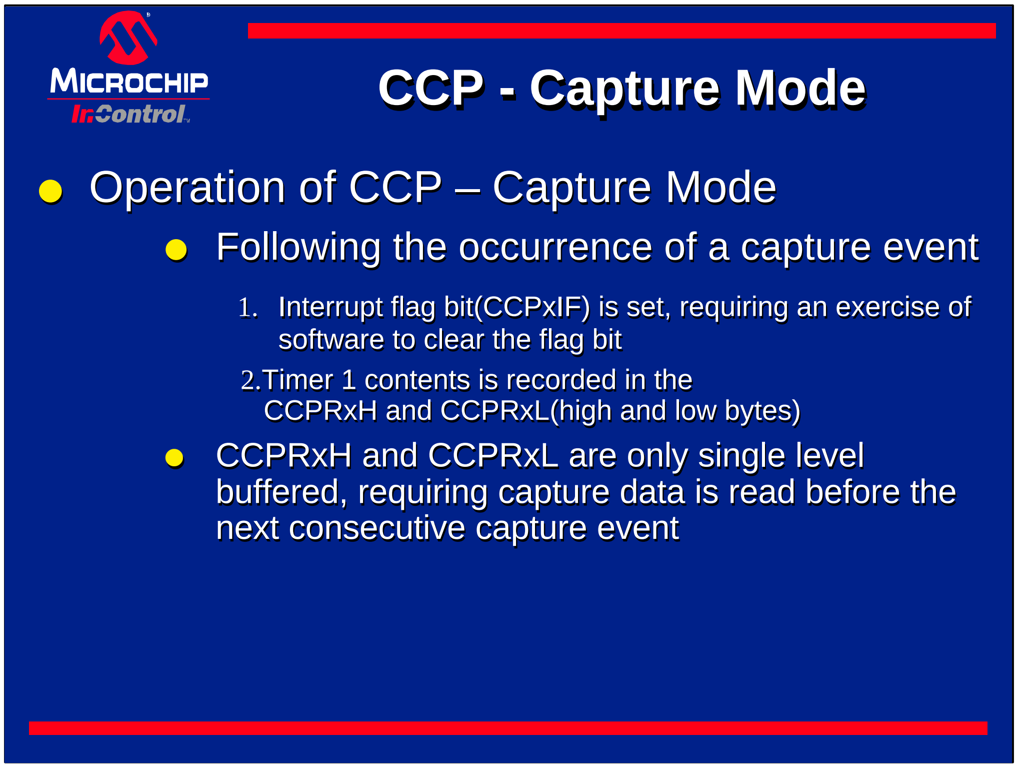
CCP - Capture Mode

- Conceptual View - Capture Mode Operation

Capture Mode Functional Diagram:



The capture mode functional diagram illustates how the incoming pulses are divided by the pre-scaler value before setting the CCPxIF flag, indicating a positive capture condition/event. The signal evaluated by the pre-scaler originates from the external CCPx pin and routed to the pre-scale circuitry. A pre-scaler is used to increase the number of edge counts required before sending the CCPxIF and transferring Timer high and low contents to CCPRxH and CCPRxL registers.



CCP - Capture Mode

- Operation of CCP – Capture Mode
  - Following the occurrence of a capture event
    1. Interrupt flag bit(CCPxIF) is set, requiring an exercise of software to clear the flag bit
    2. Timer 1 contents is recorded in the CCPRxH and CCPRxL(high and low bytes)
  - CCPRxH and CCPRxL are only single level buffered, requiring capture data is read before the next consecutive capture event

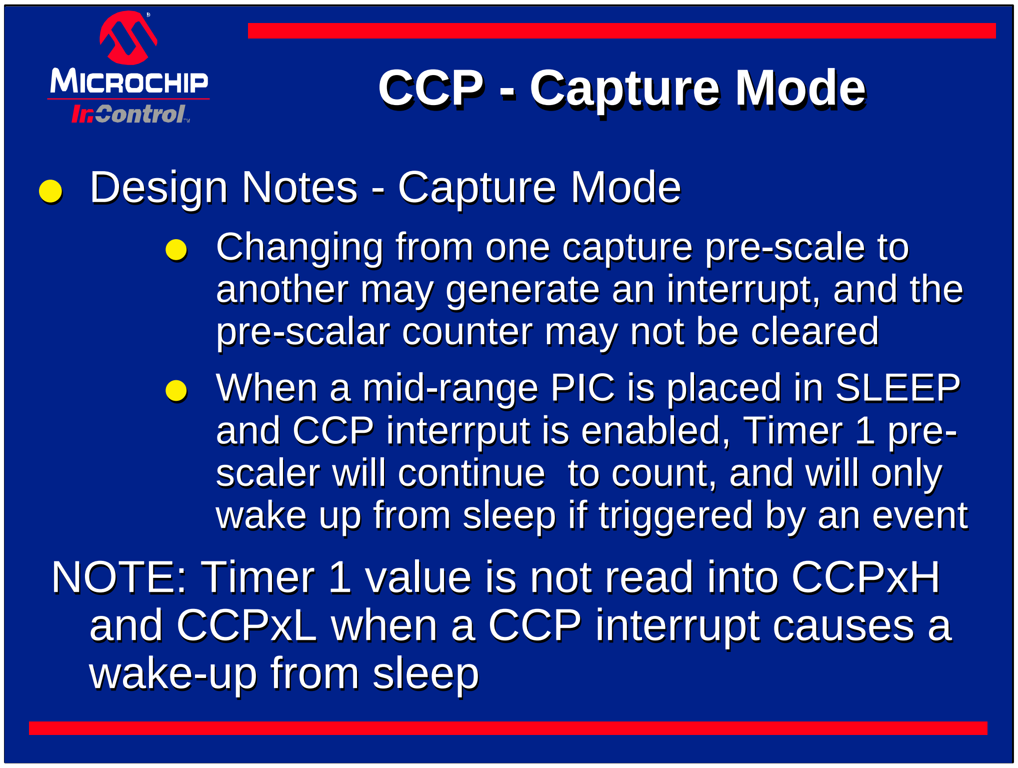
Following the occurrence of a capture event, CCPx module generates two actions:

Initially, timer 1 contents, high and low byte, is recorded in CCPRxH, high byte, and CCPRxL, low byte.

Secondly, the interrupt flag bit, CCPxIF, is set.

If CCPxIF is set and CCPxIE is enabled, the main code execution is paused and the interrupt handler is called. The interrupt handler must clear CCPxIF bit to allow detection of the next capture event condition.

Independent of the capture register pre-scaler, CCPRxH and CCPRxL are only single level buffered, requiring the CCPRx register data be saved to another register set before the next consecutive capture event.



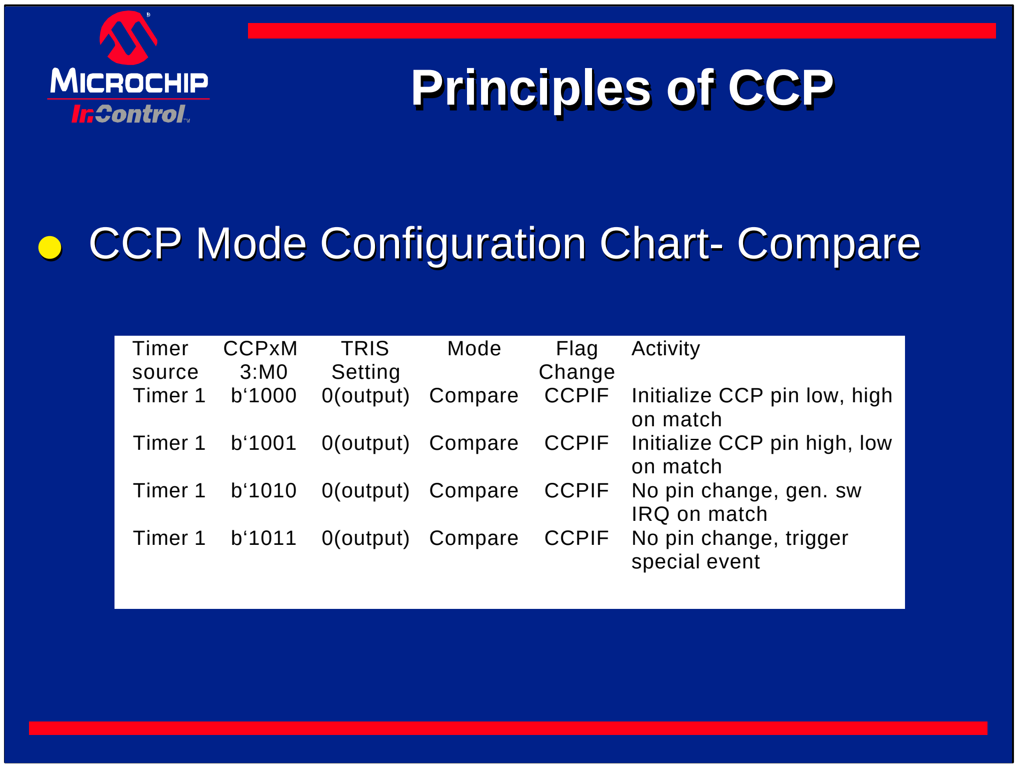
CCP - Capture Mode

- Design Notes - Capture Mode
  - Changing from one capture pre-scale to another may generate an interrupt, and the pre-scalar counter may not be cleared
  - When a mid-range PIC is placed in SLEEP and CCP interrput is enabled, Timer 1 pre-scaler will continue to count, and will only wake up from sleep if triggered by an event

NOTE: Timer 1 value is not read into CCPxH and CCPxL when a CCP interrupt causes a wake-up from sleep

Changing from one capture pre-scale to another may generate an interrupt, and the pre-scaler counter may not be cleared. To prevent this from occurring, disable the CCP module before setting a new pre-scaler and returning back to capture mode.

When a mid-range PIC is placed in SLEEP and timer 1 uses an external oscillator, Timer 1 pre-scaler will continue to count, but the core will only wake up from sleep after a trigger event .

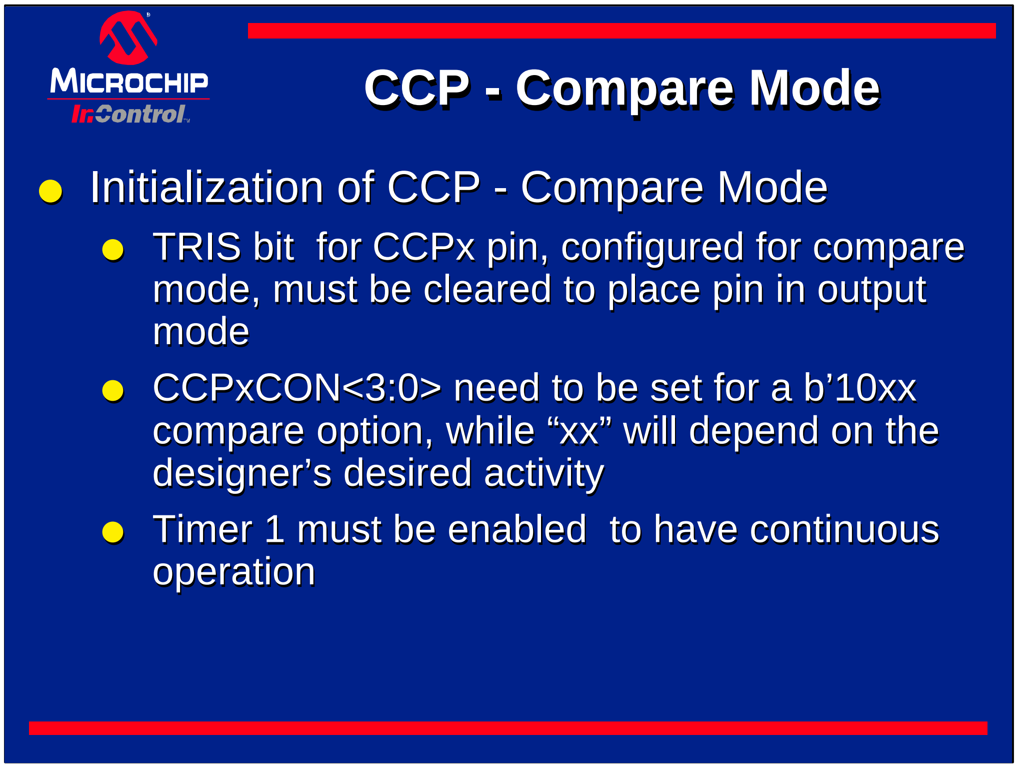


Principles of CCP

- CCP Mode Configuration Chart- Compare

Timer source	CCPxM 3:M0	TRIS Setting	Mode	Flag Change	Activity
Timer 1	b'1000	0(output)	Compare	CCPIF	Initialize CCP pin low, high on match
Timer 1	b'1001	0(output)	Compare	CCPIF	Initialize CCP pin high, low on match
Timer 1	b'1010	0(output)	Compare	CCPIF	No pin change, gen. sw IRQ on match
Timer 1	b'1011	0(output)	Compare	CCPIF	No pin change, trigger special event

CCPxCON bits<3:2> equal to 10'b specifies compare mode, while bits<1:0> will determine the activity. The activity options consist of a low to high or high to low CCPx pin state change on match, software IRQ or special trigger on match. In Compare mode, for all activities, CCPIF flag will be set and timer 1 is used in free running mode.



CCP - Compare Mode

- Initialization of CCP - Compare Mode
  - TRIS bit for CCPx pin, configured for compare mode, must be cleared to place pin in output mode
  - CCPxCON<3:0> need to be set for a b'10xx compare option, while "xx" will depend on the designer's desired activity
  - Timer 1 must be enabled to have continuous operation

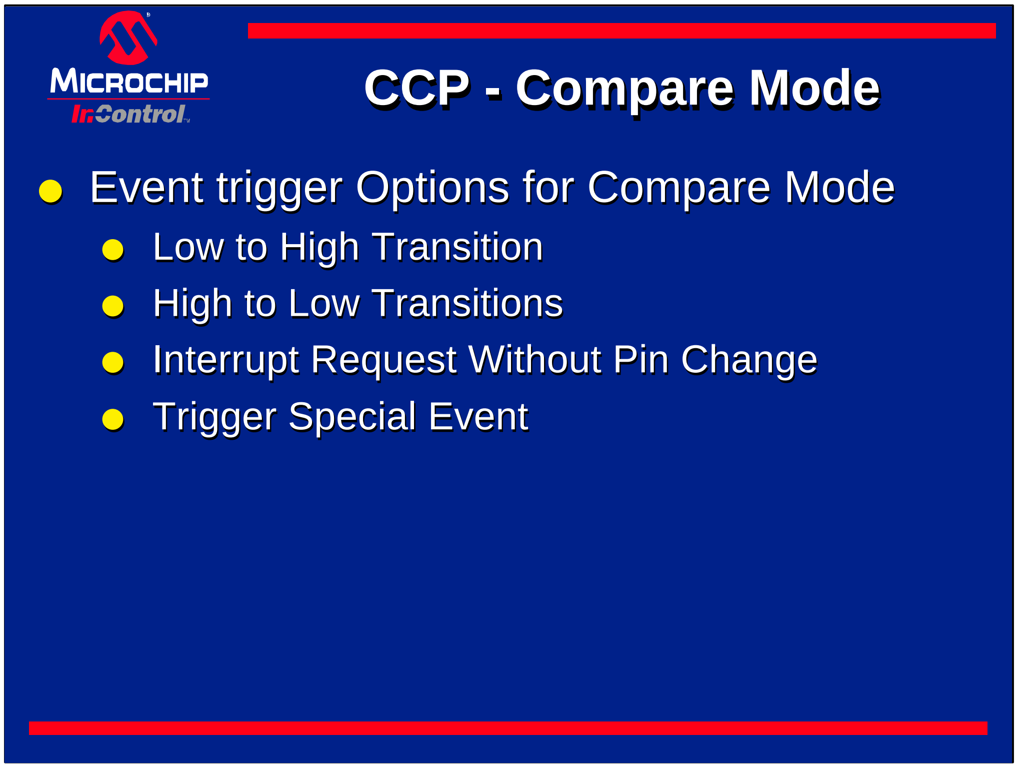
As mentioned earlier in the first slide, CCP module supports the three modes, and the mode we will examine next is "compare."

Initialization of CCP for compare consist of three parts:

First, the TRIS bit for the CCPx pin must be cleared to configure the pin as an output.

Second, CCPxCON<3:0> needs to be set for b'10xx, where xx are two bits select the desired event output. The four distinct compare mode events possible include high to low and low to high transition, interrupt request without pin change, or special trigger event.

Lastly, identical to the capture mode, timer one is used as a reference clock and must be enabled for continuous operation.



CCP - Compare Mode

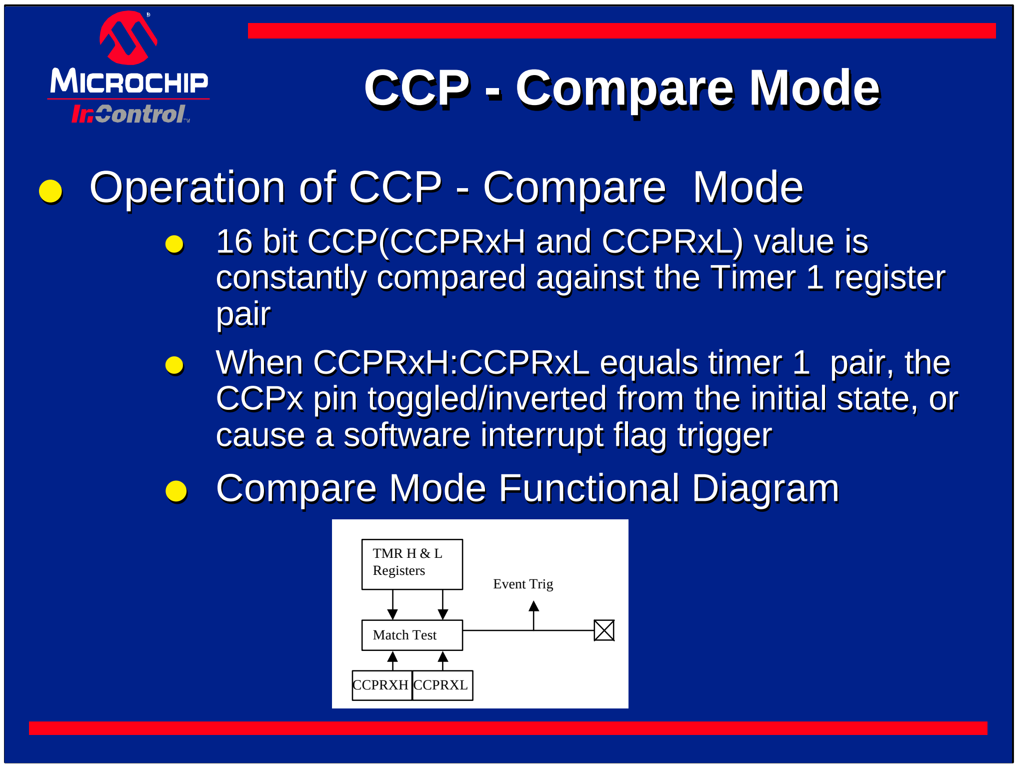
- Event trigger Options for Compare Mode
  - Low to High Transition
  - High to Low Transitions
  - Interrupt Request Without Pin Change
  - Trigger Special Event

The four distinct compare mode events possible.

High to low and low to high transition: specifies the initial CCPx pin output state and the final logic state indicating the compare match occurred.

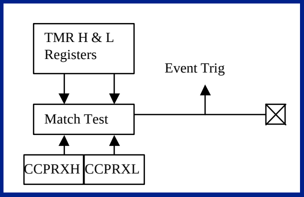
Interrupt request without pin change: causes the start of an interrupt handler, and execution continues from the interrupt vector until executing a RETFIE

Special event trigger used to reset the timer 1 register pair when detecting a compare event, CCPRxH and CCPRxL match, resulting with a CCPxIF flag set and the option to start an ADC conversion



CCP - Compare Mode

- Operation of CCP - Compare Mode
  - 16 bit CCP(CCPRxH and CCPRxL) value is constantly compared against the Timer 1 register pair
  - When CCPRxH:CCPRxL equals timer 1 pair, the CCPx pin toggled/inverted from the initial state, or cause a software interrupt flag trigger
  - Compare Mode Functional Diagram

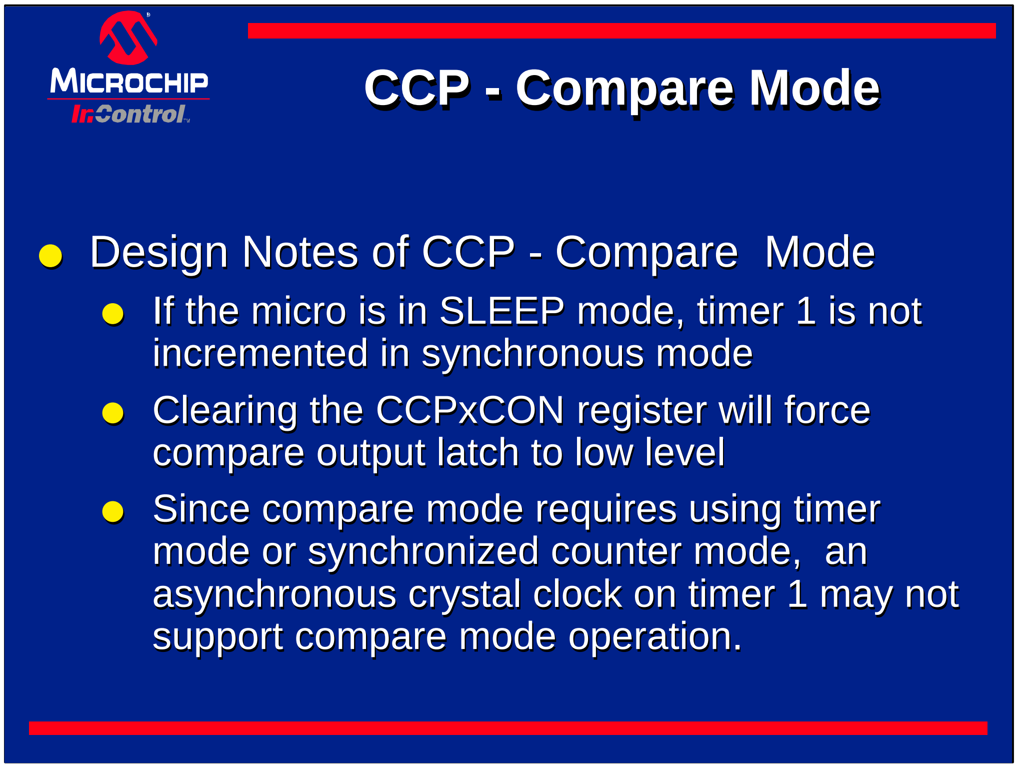


CCPx pin toggle event is driven by the match of a moving 16 bit counter, timer 1, against a known 16 bit value, cascaded CCPRxH and CCPxL value. When timer 1 equals CCPRx, a CCPx bit toggles, an interrupt or special event trigger occurs. The special event trigger output of CCP will cause a reset of timer 1 high and low bytes, and offer a means to start a conversion cycle in many mid-range parts containing an ADC module.

If software interrupt request or special event trigger mode is selected, there will be no change on CCPx pin. When interrupt request is selected, a CCP event will cause CCPxIF to a set condition, while special event trigger from CCPx will reset timer one.

Important to compare mode, clearing CCPxCON register will force compare output latch to logic low and not the I/O Port Latch register contents.

A mix of older PicMicros experienced a variation in compare mode pin out operation. Consequently, and as good practice, please review the ERRATA document for the part of consideration before final selection.

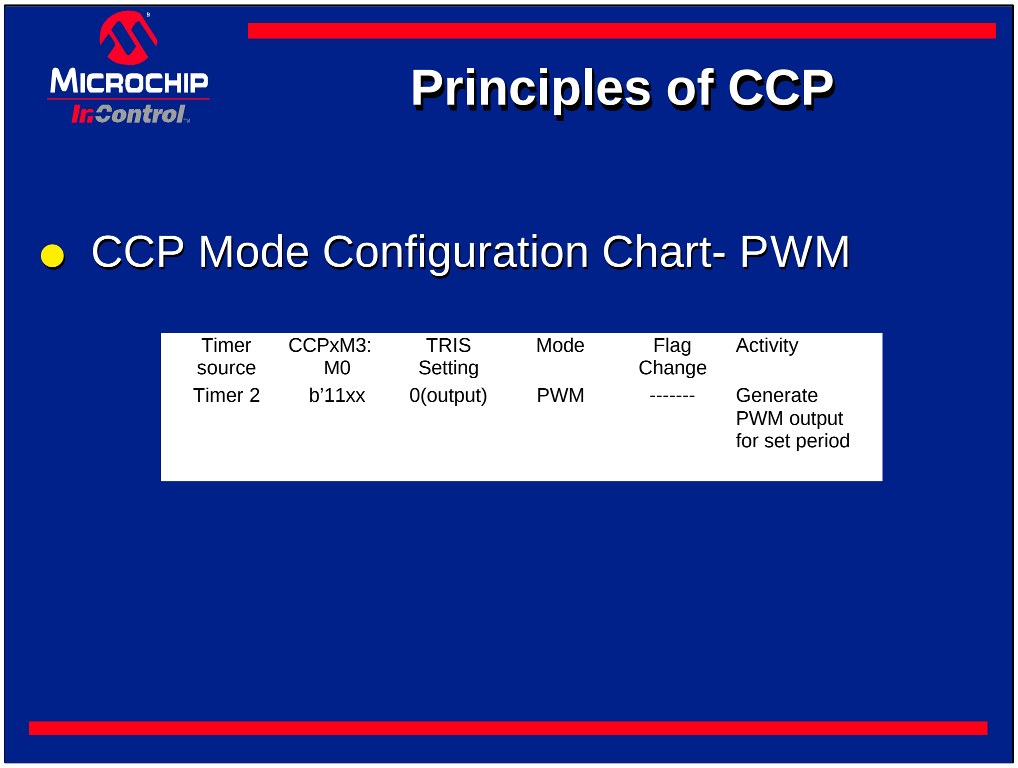


CCP - Compare Mode

- Design Notes of CCP - Compare Mode
  - If the micro is in SLEEP mode, timer 1 is not incremented in synchronous mode
  - Clearing the CCPxCON register will force compare output latch to low level
  - Since compare mode requires using timer mode or synchronized counter mode, an asynchronous crystal clock on timer 1 may not support compare mode operation.

Since compare mode depends on timer one in synchronous mode, and timer one depends on system clock in synchronous mode, timer one will not increment, preventing compare mode from operating while PIC micro is in SLEEP mode.

Since compare mode requires using timer mode or synchronized counter mode, an asynchronous crystal clock on timer 1 may not support compare mode operation.

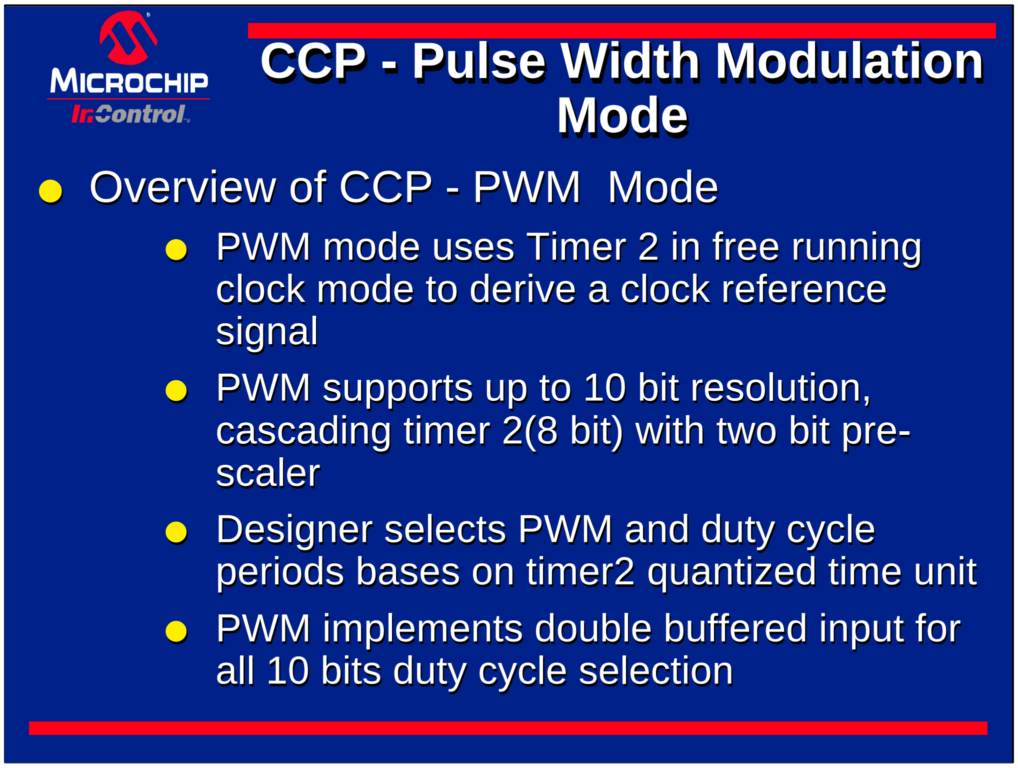

Principles of CCP

- CCP Mode Configuration Chart- PWM

Timer source	CCPxM3: M0	TRIS Setting	Mode	Flag Change	Activity
Timer 2	b'11xx	0(output)	PWM	-------	Generate PWM output for set period

CCP mode settings for Pulse Width Modulation require CCPxCON<3:0> is set to 11xx'b, where "xx" are don't cares, there is no flag state change on CCPxIF, and the PWM output port is generated at the CCPx pin.

Details on the frequency and duty cycle settings, discussed later, describes how the designer can specify the PWM signal period and duty cycle duration via special function registers PR2 and CCPxL.



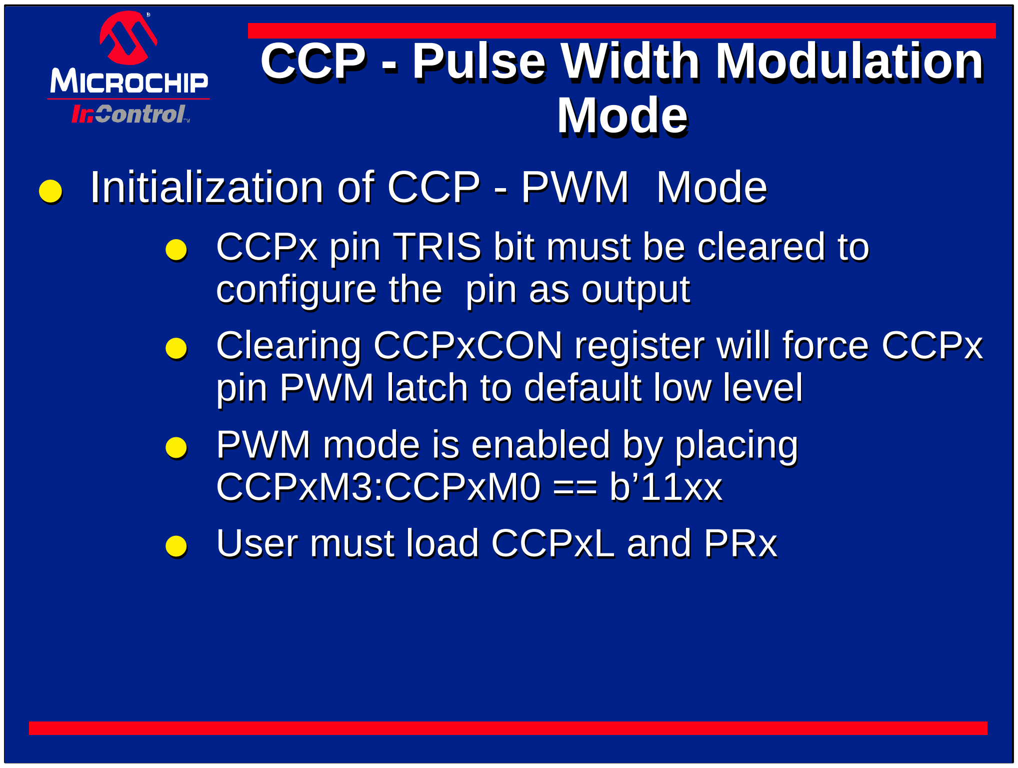
CCP - Pulse Width Modulation Mode

- Overview of CCP - PWM Mode
  - PWM mode uses Timer 2 in free running clock mode to derive a clock reference signal
  - PWM supports up to 10 bit resolution, cascading timer 2(8 bit) with two bit pre-scaler
  - Designer selects PWM and duty cycle periods bases on timer2 quantized time unit
  - PWM implements double buffered input for all 10 bits duty cycle selection

The CCP module supports a third mode, pulse width modulation(PWM). PWM module uses timer two in free running clock mode to derive a reference clock.

The PWM period is set by the contents of PR2, while the duty cycle is set by CCPRxL. Real time period and duty cycle duration's are functions of oscillator frequency and prescaler, and determining the smallest unit of time.

PWM output supports up to 10 bit resolution, cascading timer two with a two bit prescalar. CCPs PWM implements double buffered input for all 10 bits to allow the selection of a new duty cycle value without interrupting an uncompleted PWM cycle period



CCP - Pulse Width Modulation Mode

- Initialization of CCP - PWM Mode
  - CCPx pin TRIS bit must be cleared to configure the pin as output
  - Clearing CCPxCON register will force CCPx pin PWM latch to default low level
  - PWM mode is enabled by placing CCPxM3:CCPxM0 == b'11xx
  - User must load CCPxL and PRx

Initialization of CCP module for PWM operation required four steps:

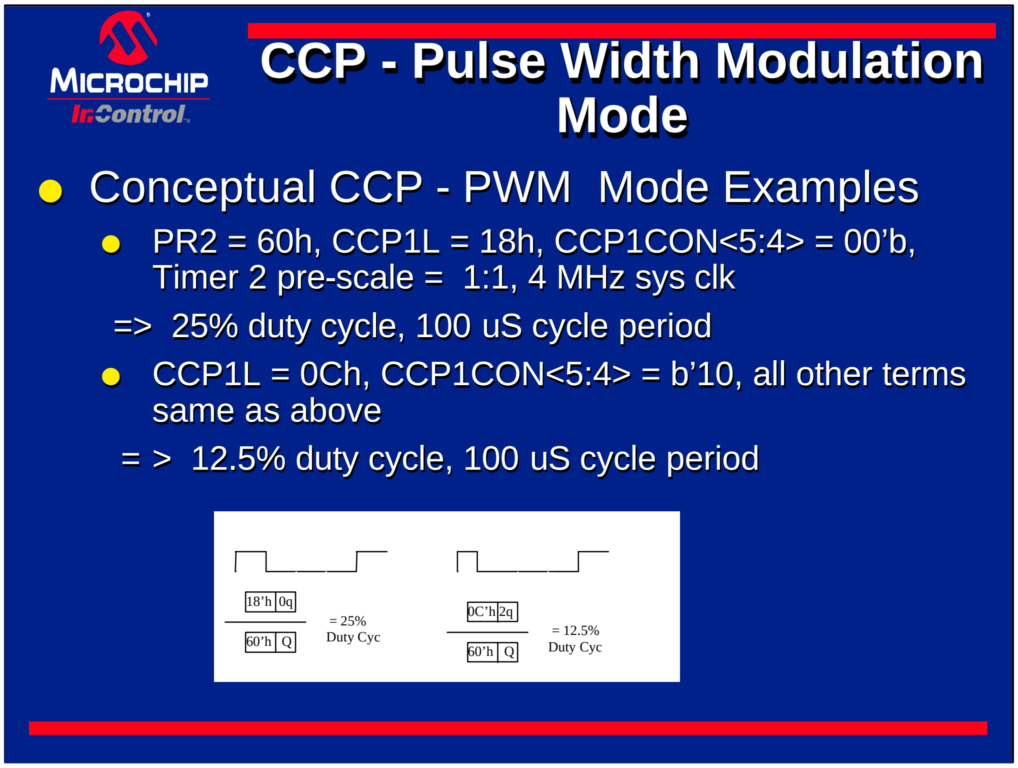
First, CCPx pin TRIS bit must be cleared to configure the pin as an output.

Second, resetting CCPx module by clearing CCPxCON register, and forcing PWM pin Latch to a logic low default

Third, PWM mode is enabled by placing CCPxM3:CCPxM0 = b'11xx, where 'xx' are don't cares.

Last, user must load CCPxL and PR2 registers. The next slide will discuss details on selecting PWM input parameters.

Note, Timer two must be enabled for the PWM mode to operate.

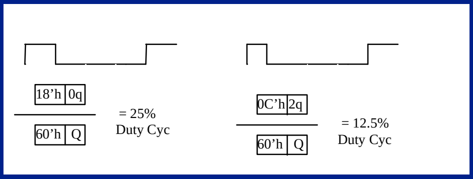


CCP - Pulse Width Modulation Mode

- Conceptual CCP - PWM Mode Examples
  - PR2 = 60h, CCP1L = 18h, CCP1CON<5:4> = 00'b, Timer 2 pre-scale = 1:1, 4 MHz sys clk

  => 25% duty cycle, 100 uS cycle period
  - CCP1L = 0Ch, CCP1CON<5:4> = b'10, all other terms same as above

  = > 12.5% duty cycle, 100 uS cycle period

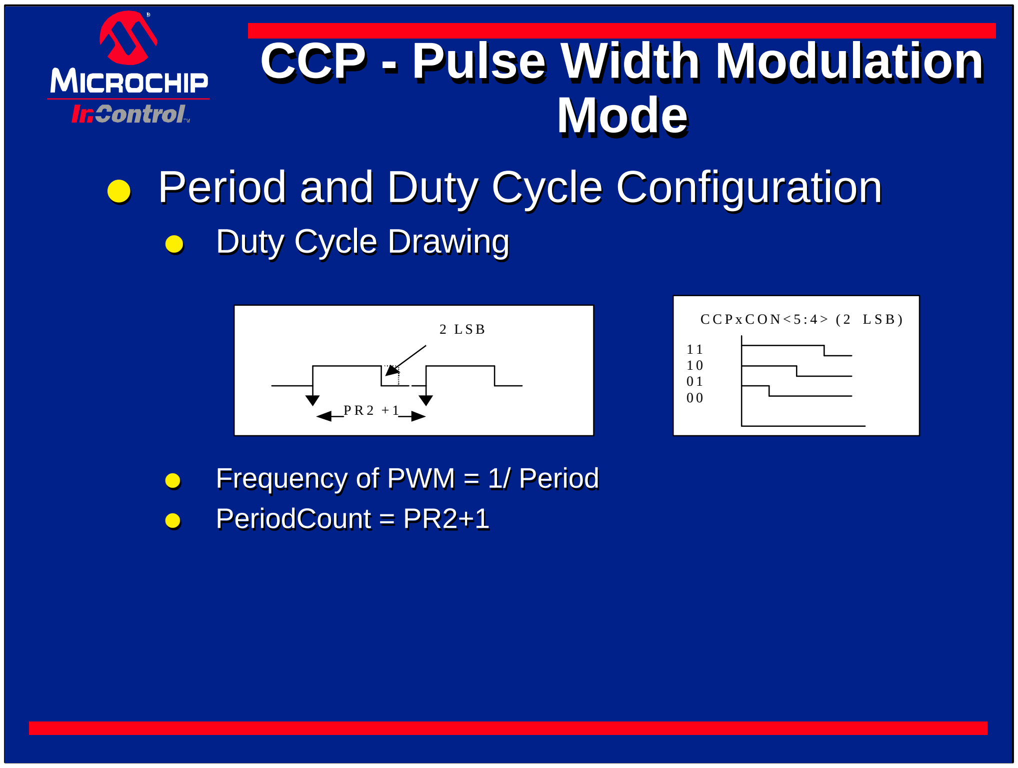


Conceptually, the proportional percentage of the duty cycle output is reflected by CCPxL to PR2.

As the period gets smaller, lower value in PR2, the percent of weight per counter value increases; effectively placing a greater dependency on the value of the two least significant bits.

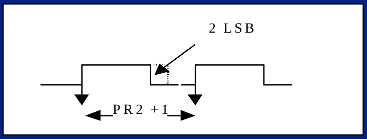
Similarly, the emphasis on the two least significant bits will also increase with the timer 2 pre-scaler. Therefore, when a pre-scale of 16:1 is in use, designer will want to put greater consideration to the values of CCPxCON<5:4>.

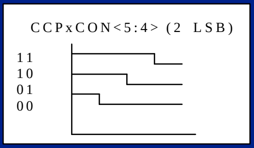
The present slide demonstrates the principle of pulse width modulation and how the duty cycle is proportional to the ratio CCP1L verses PR2. In the example, the two desired outputs are twenty-five and twelve point five percent. By modifying the two least significant bits, configurable from CCP1CON<5:4>, a higher precision PWM output is possible(ie. 25% and 12.5% produced).



CCP - Pulse Width Modulation Mode

- Period and Duty Cycle Configuration
  - Duty Cycle Drawing



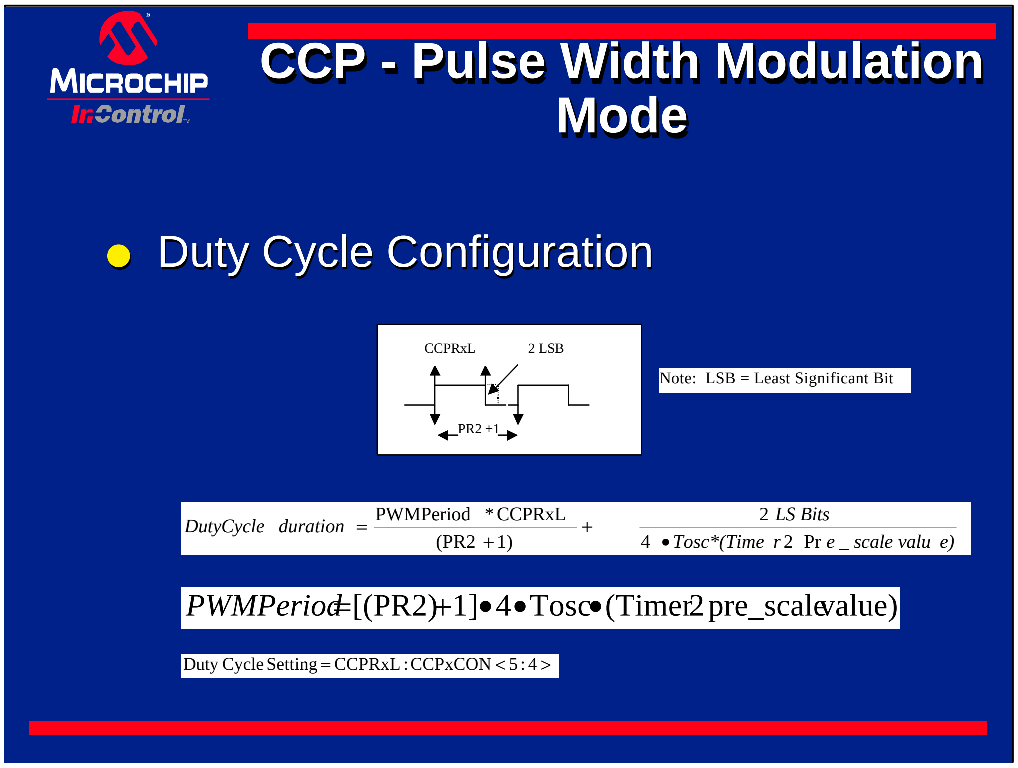


  - Frequency of PWM = 1/ Period
  - PeriodCount = PR2+1

Let us examine period for PWM operation.

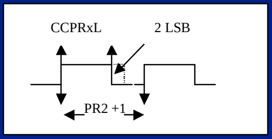
Looking at the illustration of a square wave, we see the definition of the period term equals PR2 + 1.

A second drawing illustates the effect of variation in the two least significant bits, widening the duty cycle by a two bit multiple of Tosc



CCP - Pulse Width Modulation Mode

- Duty Cycle Configuration



Note: LSB = Least Significant Bit

$$\text{DutyCycle duration} = \frac{\text{PWMPeriod} * \text{CCPRxL}}{(\text{PR2} + 1)} + \frac{2\ \textit{LS Bits}}{4 \bullet \textit{Tosc*(Timer2 Pre\_scale value)}}$$

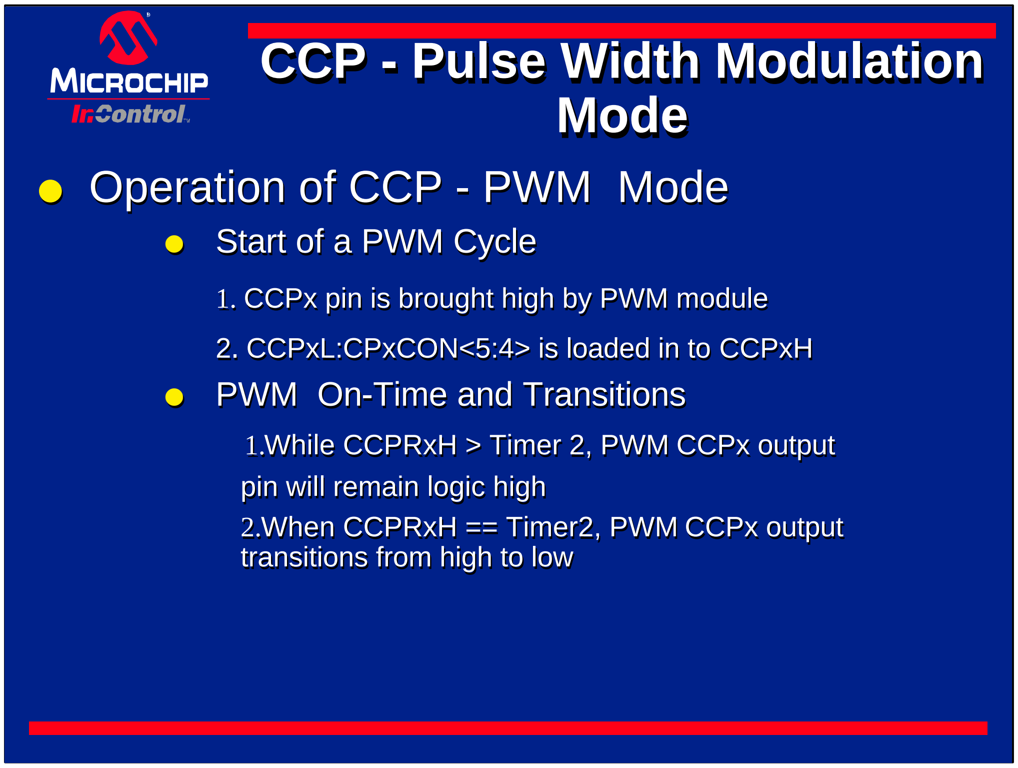
$$\textit{PWMPeriod} = [(\text{PR2}) + 1] \bullet 4 \bullet \text{Tosc} \bullet (\text{Timer2 pre\_scale value})$$

$$\text{Duty Cycle Setting} = \text{CCPRxL} : \text{CCPxCON} < 5:4 >$$

Duty cycle configuration for PWM operation.

PWMPeriod equals [(PR2+1)]***4**Tosc*(timer 2 prescale value), and the resultant PWM frequency equals 1/ PWM_Period.

Duty Cycle is based on CCPRxL, most significant byte, and CCPxCON<5:4>, least significant two bits. CCPRxL functions as a comparative value with timer 2 and a scaling factor to determine the number of timers counts the CCPx PWM logic remains high, without considering CCPxCON<5:4>. The two least significant bits, CCPxCON<5:4>, determine the percentage of the maximum resolution the PWM duty cycle is extended. Therefore, the practical real time “on” duration is PWM_Period*CCPRxL/(PR2+1) + [2 LSBits value]/4*Tosc*(timer 2 pre-scale value).



CCP - Pulse Width Modulation Mode

- Operation of CCP - PWM Mode
  - Start of a PWM Cycle

    1. CCPx pin is brought high by PWM module

    2. CCPxL:CPxCON<5:4> is loaded in to CCPxH
  - PWM On-Time and Transitions

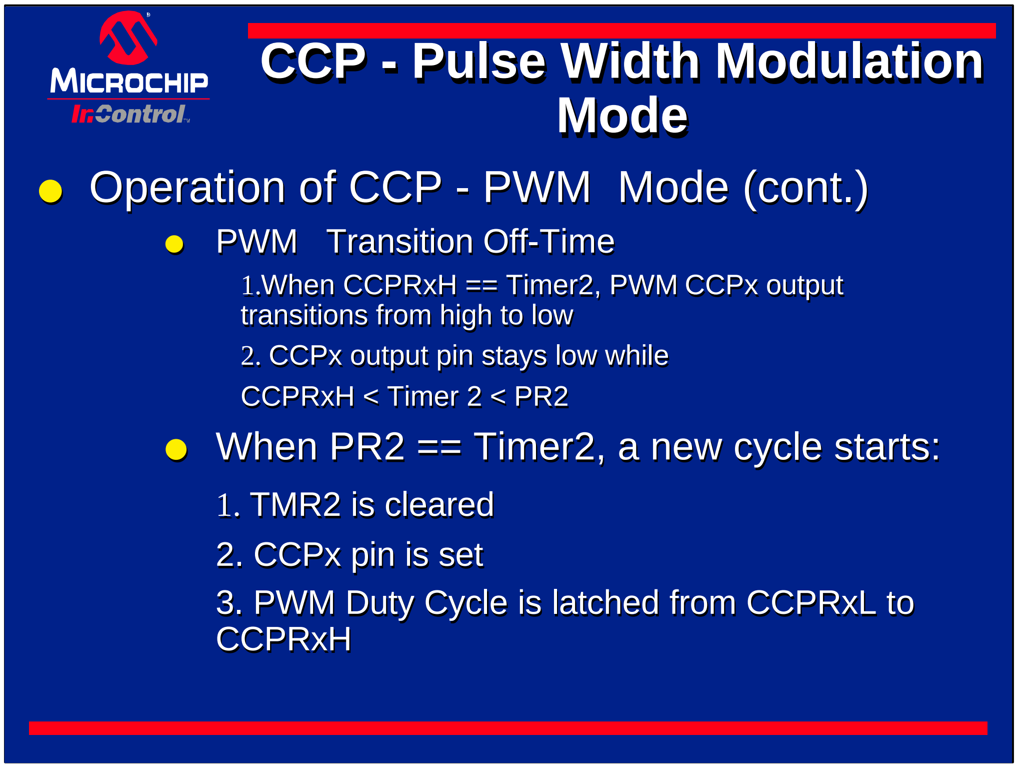
    1.While CCPRxH > Timer 2, PWM CCPx output pin will remain logic high

    2.When CCPRxH == Timer2, PWM CCPx output transitions from high to low

Operation of CCP module in PWM mode uses three main cycles: Start, Transition, and Restart

Start Cycle, CCPx pin brought high by the PWM module, and the value contained in CCPxL is loaded into CCPxH. The value contained in CCPxH is used for comparison of timer2 for transition detection.

On-Transition cycle examines for CCPRxH is greater than timer2, CCPRxH equal to timer2 and timer 2 grater than CCPRxH, and CCPRxH is less than PR2. As a system, when CCPRxH is greater than timer2, PWM CCPx output pin is set, and remains logic high until CCPRxH equals timer 2.



CCP - Pulse Width Modulation Mode

- Operation of CCP - PWM Mode (cont.)
  - PWM   Transition Off-Time

    1.When CCPRxH == Timer2, PWM CCPx output transitions from high to low

    2. CCPx output pin stays low while

    CCPRxH < Timer 2 < PR2
  - When PR2 == Timer2, a new cycle starts:

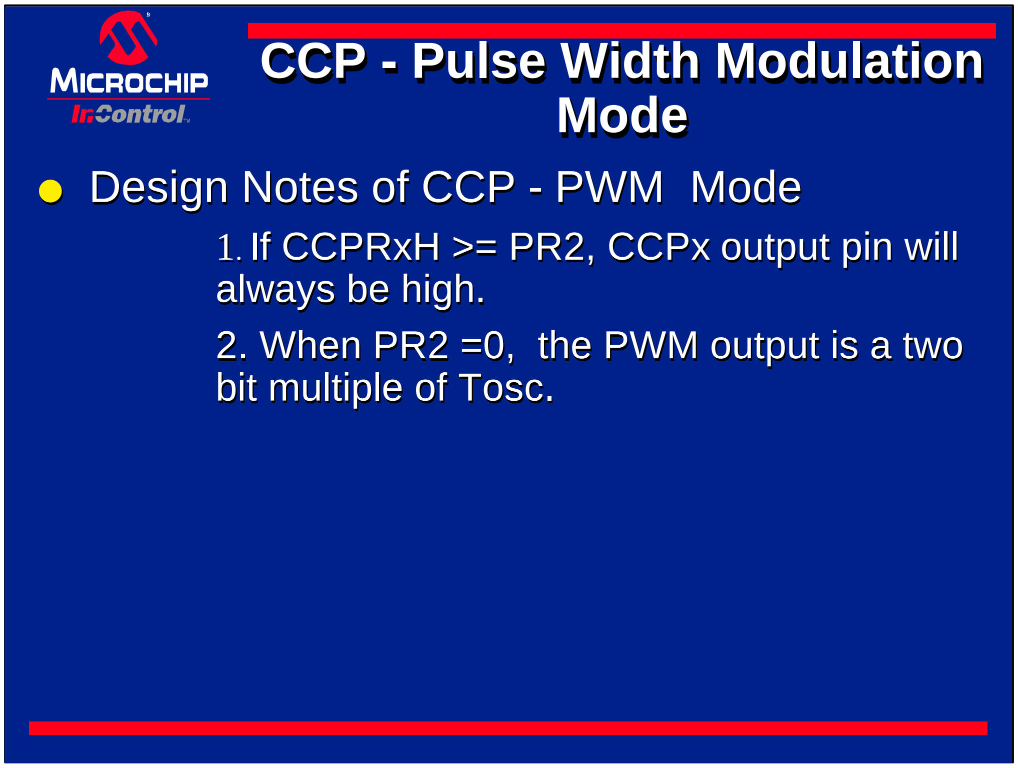
    1. TMR2 is cleared

    2. CCPx pin is set

    3. PWM Duty Cycle is latched from CCPRxL to CCPRxH

Reset cycle, started by CCPRxH equal to timer 2, causes PWM CCPx output to transition from high to low. CCPx pin will remain low while timer 2 is between CCPRxH and PR2 values.

When timer 2 equals PR2, a new cycle starts. Therefore, timer 2 is cleared, CCPx pin is set to one, and CCPRxH value is reloaded from CCPRxL as the new PWM duty cycle.



CCP - Pulse Width Modulation Mode

- Design Notes of CCP - PWM Mode
  1. If CCPRxH >= PR2, CCPx output pin will always be high.
  2. When PR2 =0, the PWM output is a two bit multiple of Tosc.

1. If CCPRxH >= PR2, CCPx output pin will always be high

2. When PR2 =0, the PWM output is a two bit multiple of Tosc and has a period of 4*Tosc, or one instruction cycle