EEC 417/517
Embedded Systems
Cleveland State University

Lab 5

CCP Module (PWM Mode),
Program Memory Paging

Dan Simon
Rick Rarick

Spring 2018

Lab 5 Outline

1. CCP Module (PWM mode)
2. Lab 5 Settings
3. Program Memory Paging
4. Electrical Characteristics
5. Sleep and Standalone Modes

CCP Module - PWM Mode

1. The CCP modules can operate in three modes:

   a) Capture (Timer1)
   b) Compare (Timer1)
   c) PWM (Timer2)

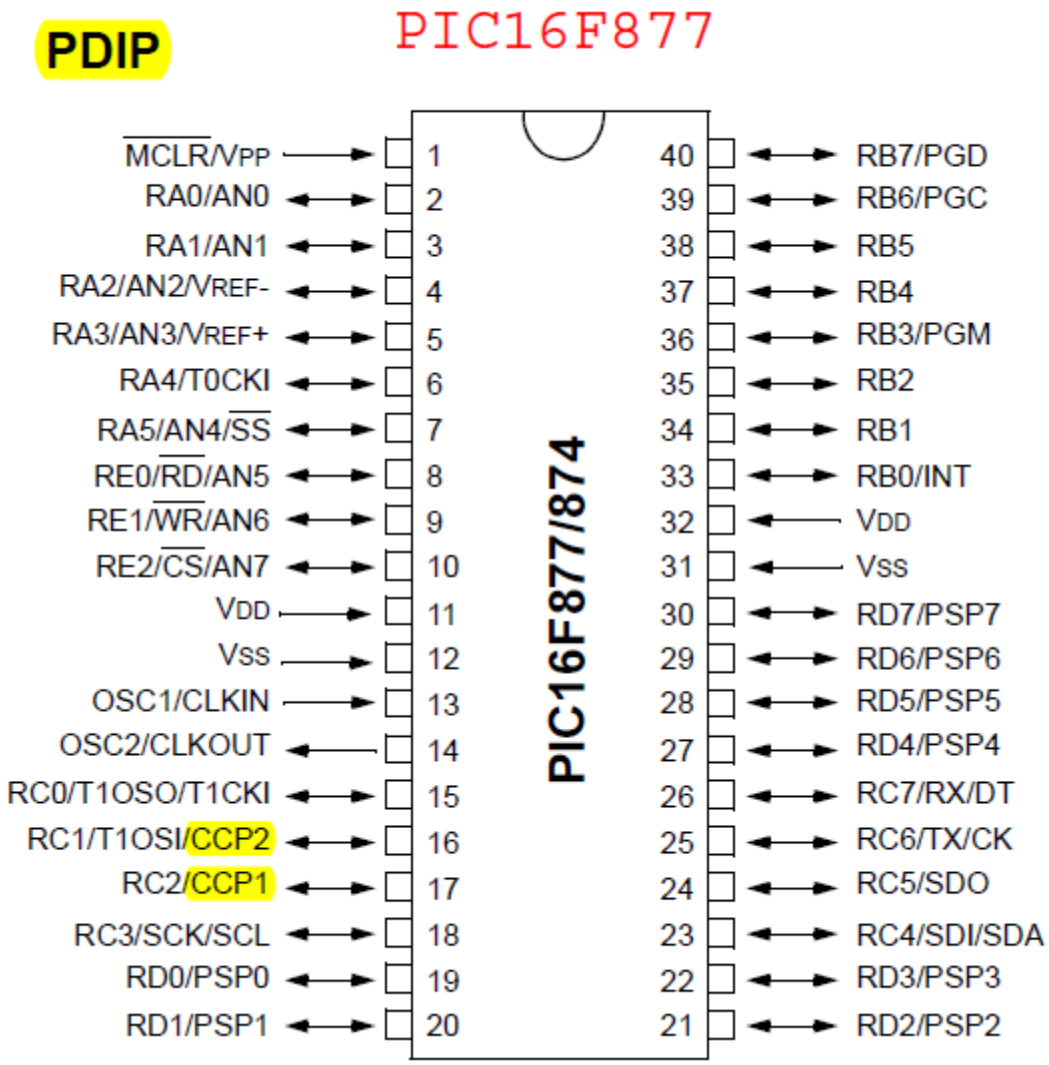
2. In PWM mode, the PWM output is on pin

   a) RC2 (CCP1) for the CCP1 module
   b) RC1 (CCP2) for the CCP2 module

3. TRISC<1> or TRISC<2> must be configured as outputs.

CCP Module - PWM Mode

PWM Output



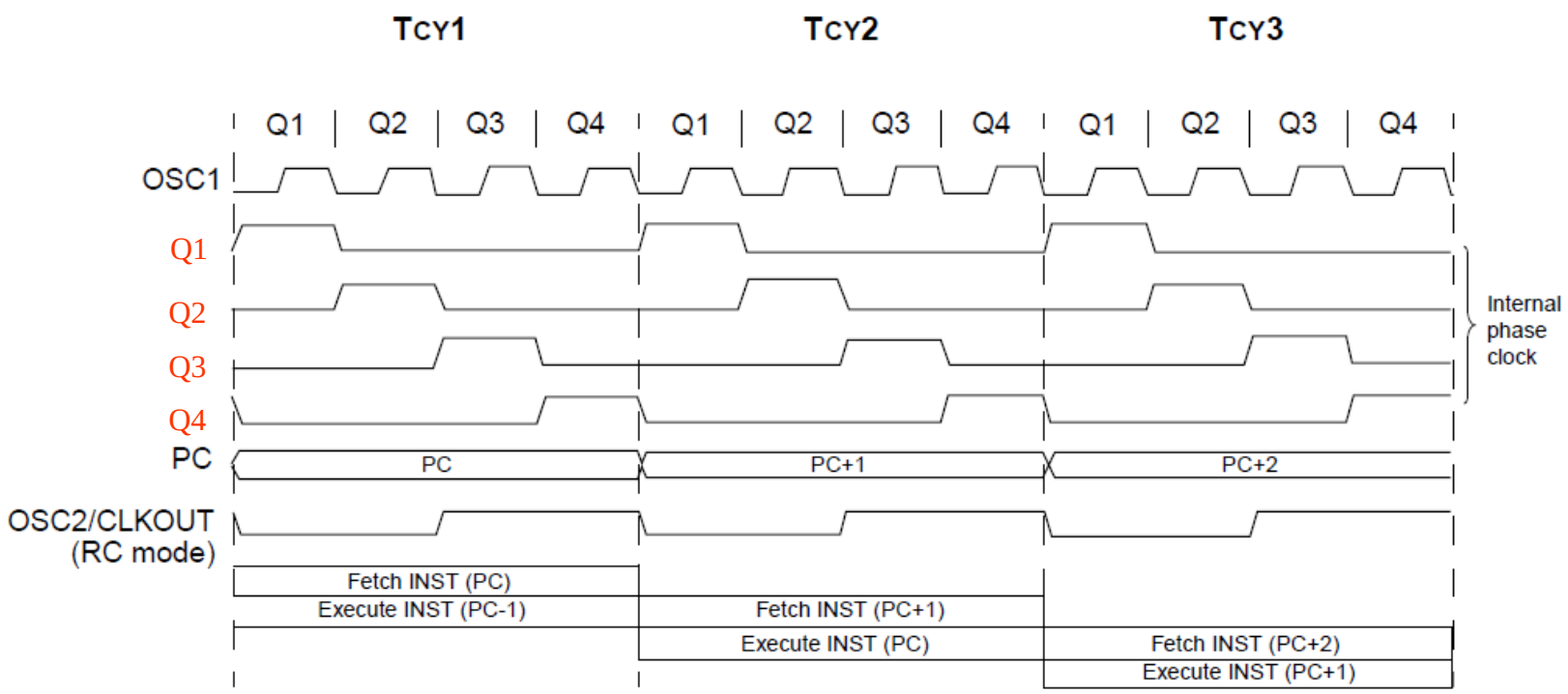
Quadrature Clocks

1. The oscillator input (from OSC1) is internally divided into four non-overlapping **quadrature clock** signals, called Q1, Q2, Q3, and Q4.

2. Internally, the program counter (PC) is incremented every Q1, and the instruction is fetched from the program memory and latched into the instruction register in Q4.

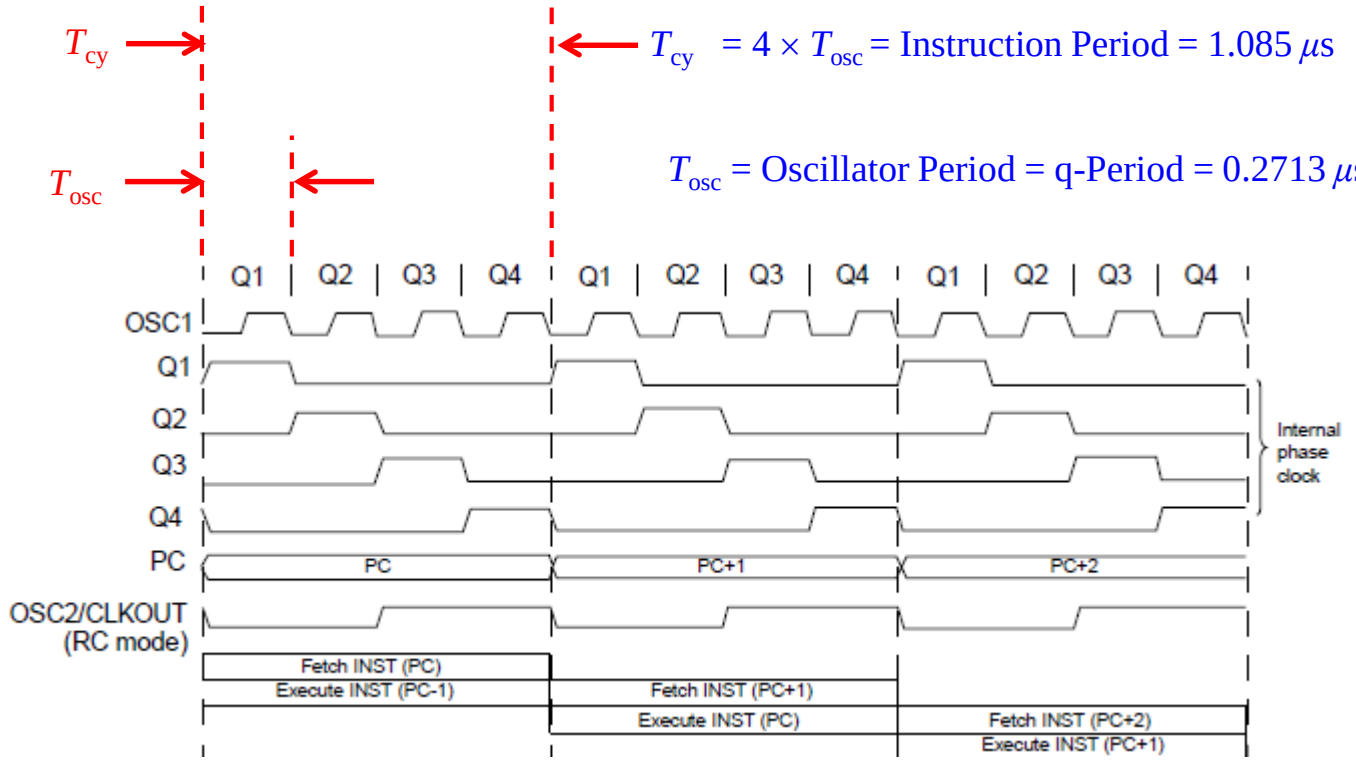
3. The instruction is decoded and executed during the following Q1 through Q4.

Figure 4-3: Clock/Instruction Cycle



Mid-Range Reference Manual, p. 4-5

Quadrature Clocks

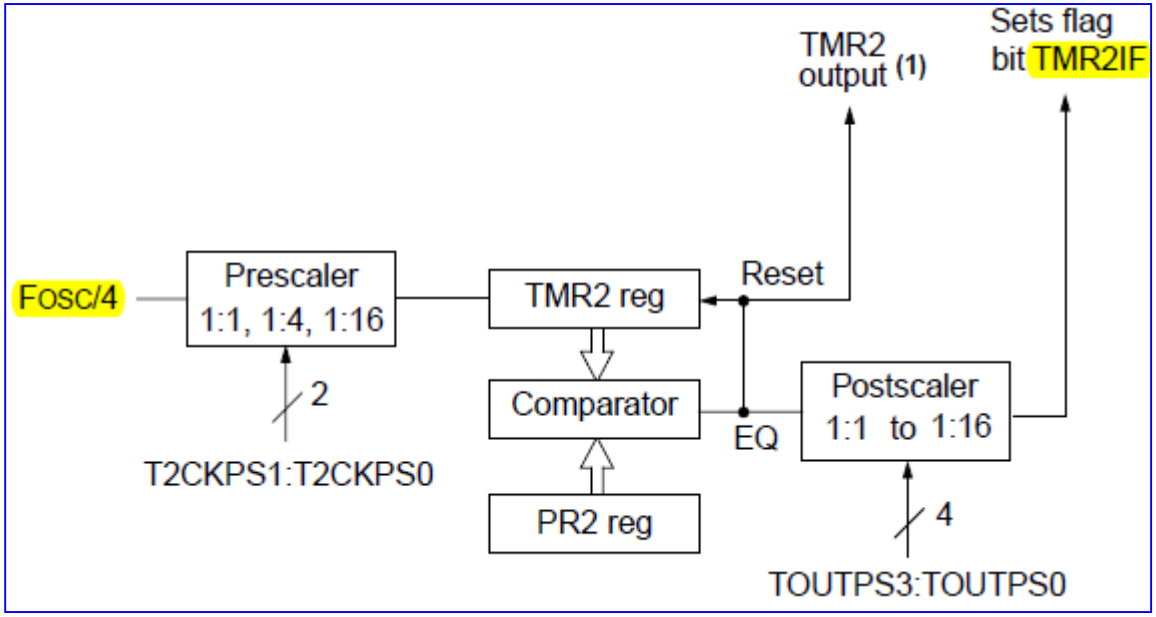


1. There are four internal quadrature clocks (or Q-clocks or phase clocks).
2. The Q-clocks can be decoded to create a 2-bit timer which ticks on Q1, Q2, Q3, Q4.
3. The 2-bit timer ticks every oscillator cycle (T_{osc}) = 0.2713 μs.

Mid-Range Reference Manual, p. 4-5

For 2, see Microchip DS39758D, page 131

Timer2 Block Diagram

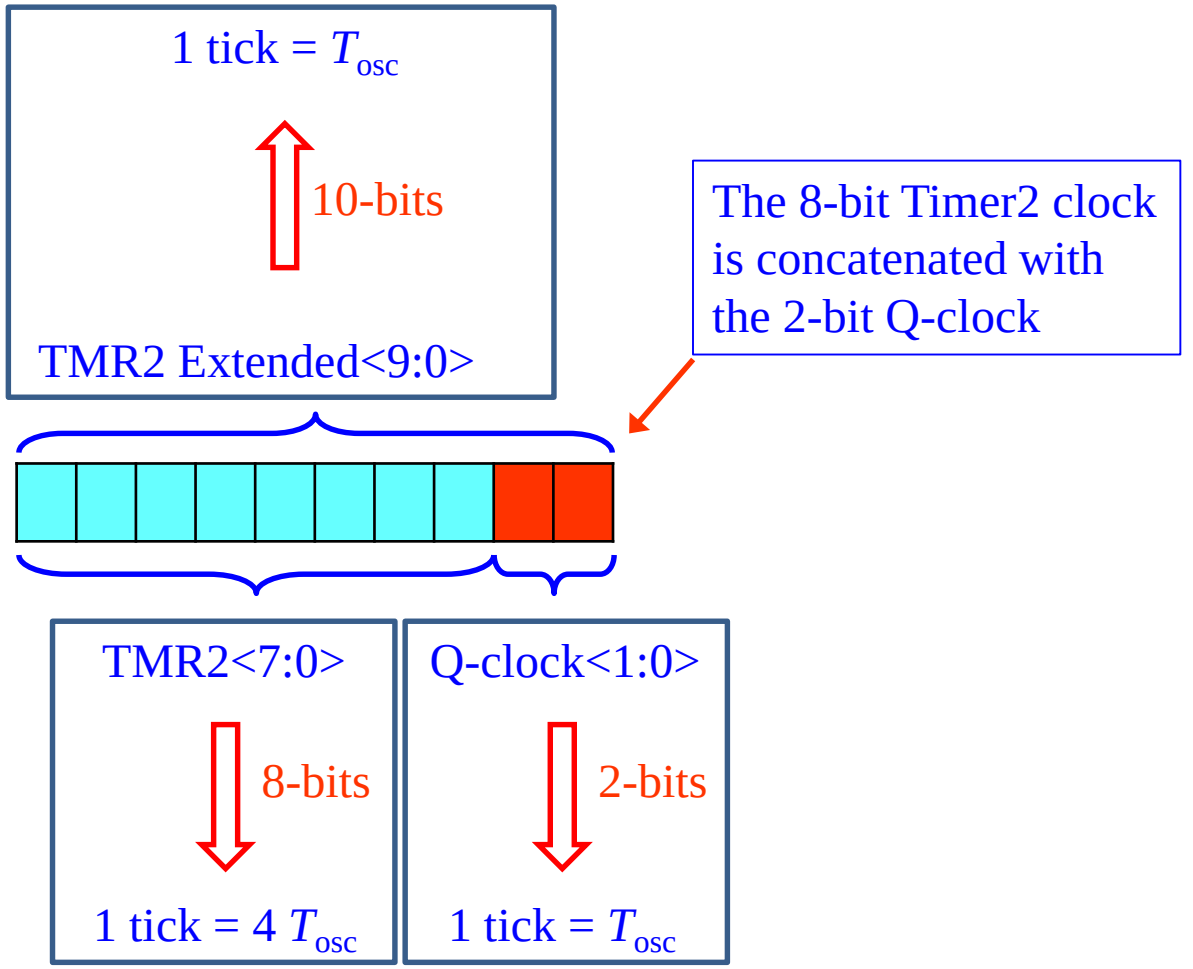


For Lab05, *Prescale* = 1, so the TMR2 register increments once every four oscillator cycles:

1 TMR2 tick = 4 T_{osc}

Mid-Range Reference Manual, p. 13-2

Extended Timer2



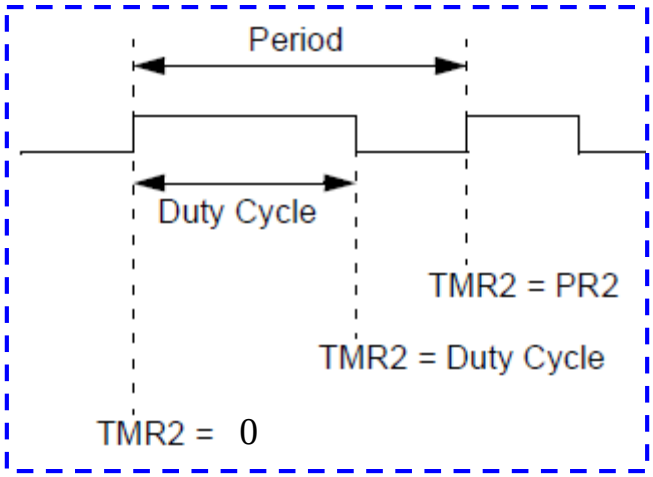
The TMR2 Extended register increments once every oscillator cycle:

1 TMR2 Extended tick = T_{osc}.

CCP Module - PWM Mode

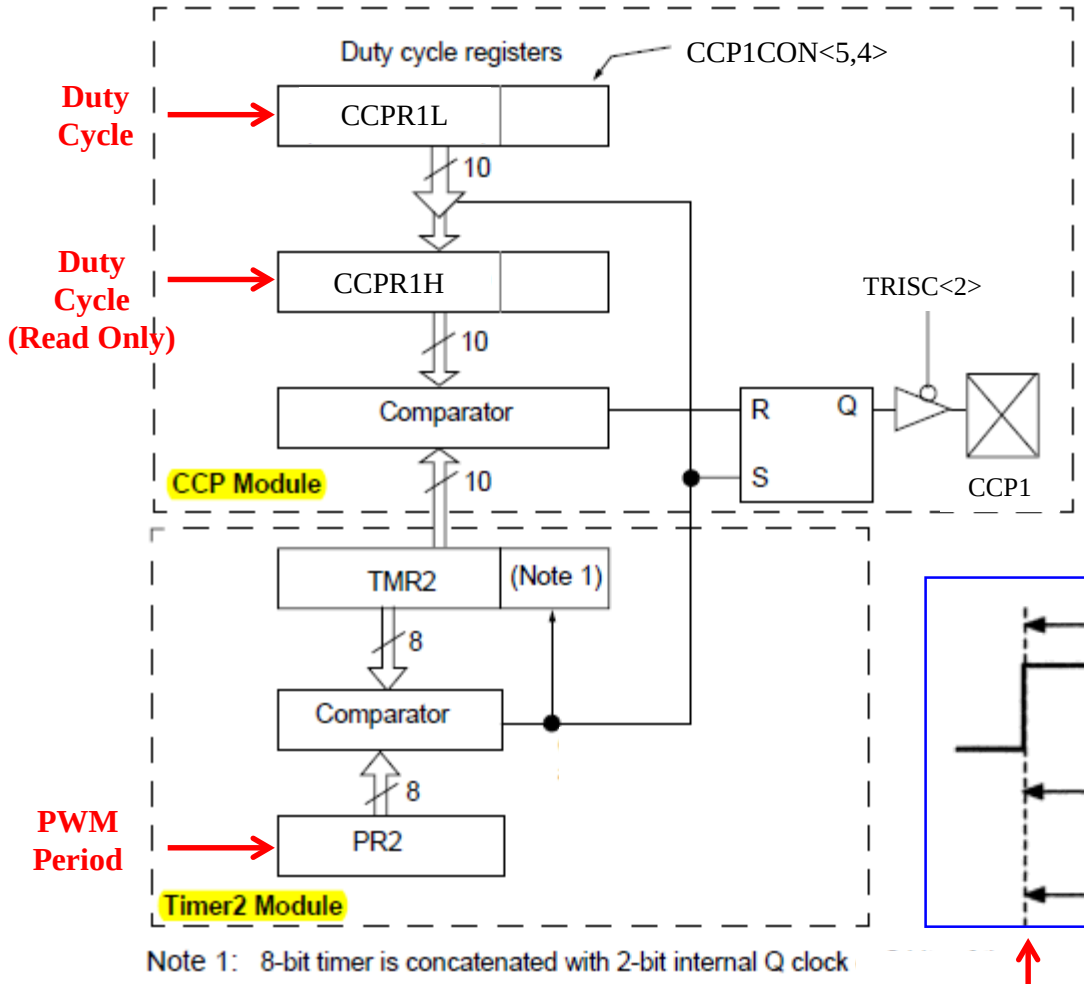
Recall: The term "duty cycle" is used loosely in the datasheet:

1. Definition: duty cycle = on-time / PWM period (e. g., DC = 0.75 = 75%).
2. In the datasheet, “duty cycle” = on-time (in sec, e.g., DC = 200 μs).
3. In the datasheet, “duty cycle” = on-time (in timer ticks, e. g., DC = 51).



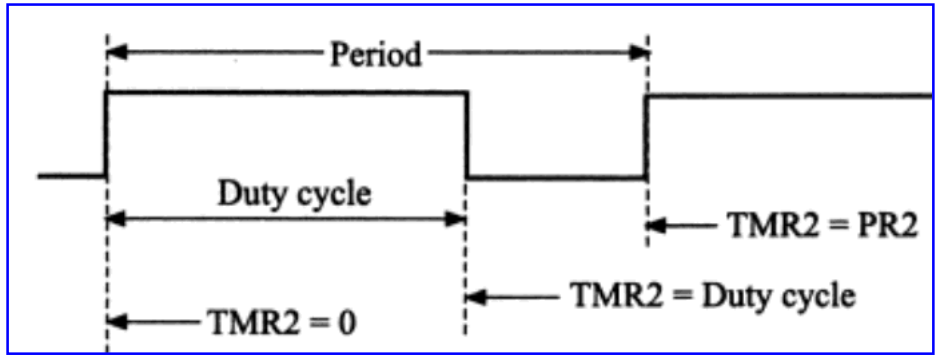
Data sheet, p. 58

CCP Module - PWM Mode



1. Load duty cycle into CCPR1L : CCP1CON<5:4>
2. Load PWM period into PR2
3. Start TMR2

Duty Cycle < PWM Period



Set latch, begin PWM period, begin duty cycle

Reset latch, end duty cycle

Set latch, end PWM period

Mid-Range Reference Manual, p. 14-8

CCP Module - PWM Mode

10-bit resolution:
CCPR1L : CCP1CON<5:4>

REGISTER 8-1: CCP1CON REGISTER/CCP2CON REGISTER (ADDRESS: 17h/1Dh)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CCPxX:CCPxY**: PWM Least Significant bits

Capture mode:
Unused

Compare mode:
Unused

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0 **CCPxM3:CCPxM0**: CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCPxIF bit is set)
1001 = Compare mode, clear output on match (CCPxIF bit is set)
1010 = Compare mode, generate software interrupt on match (CCPxIF bit is set, CCPx pin is unaffected)
1011 = Compare mode, trigger special event (CCPxIF bit is set, CCPx pin is unaffected); CCP1 resets TMR1; CCP2 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

Data sheet, p. 58

CCP Module - PWM Mode Calculations

1. PWM period = (Timer2 Prescale) (PR2 + 1) (instruction period)
   = (Timer2 Prescale) (PR2 + 1) (4 Tosc)

2. PWM on-time = (CCPR1L : CCP1CON<5:4>) (Timer2 Prescale) (Tosc)

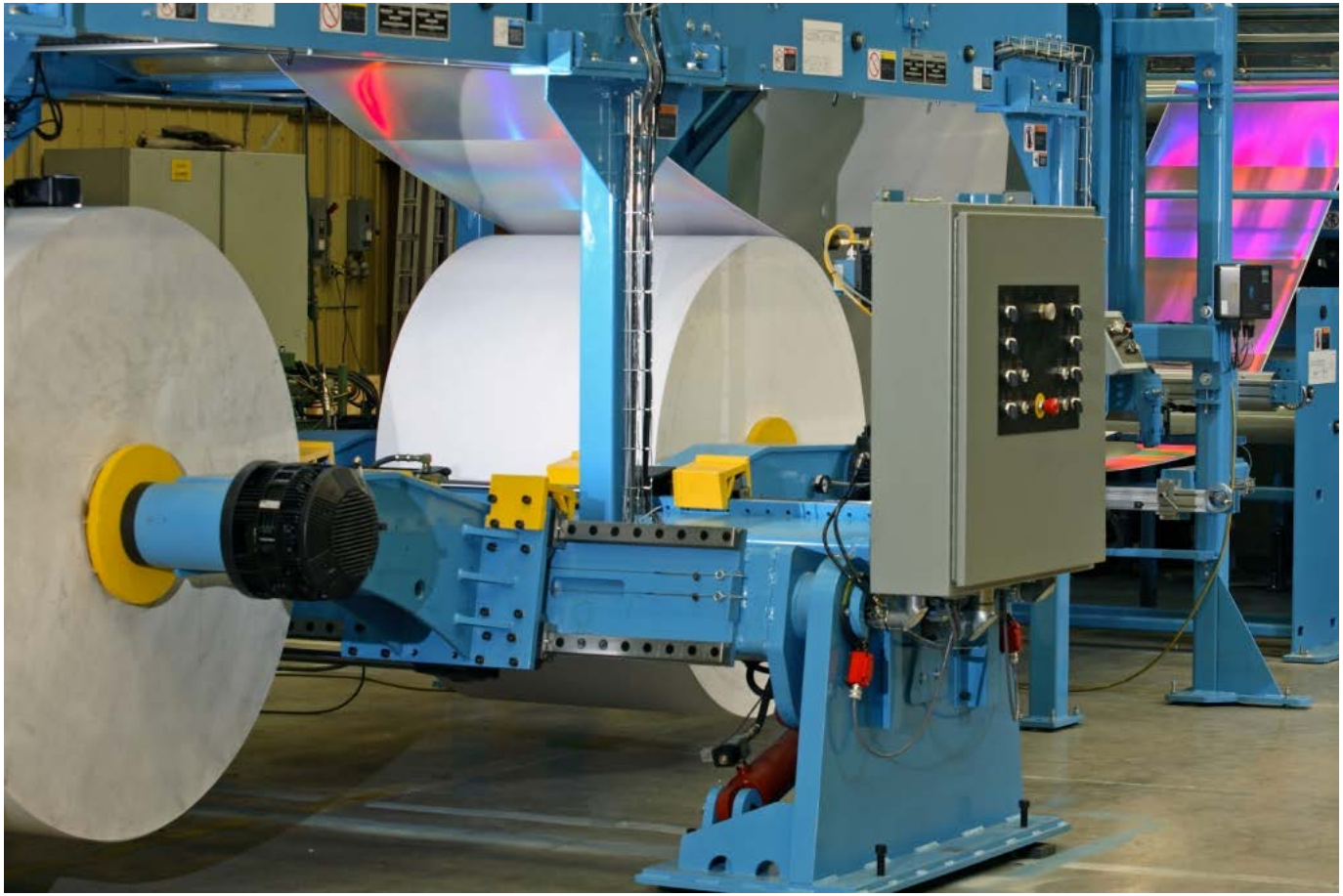
   "duty cycle" (in sec)

   "duty cycle" (in timer ticks converted to decimal)

3. The on-time has a resolution of 10 bits, so the duty cycle has a finer resolution than the period of the PWM.

4. If (on-time > period), then duty cycle = 100%

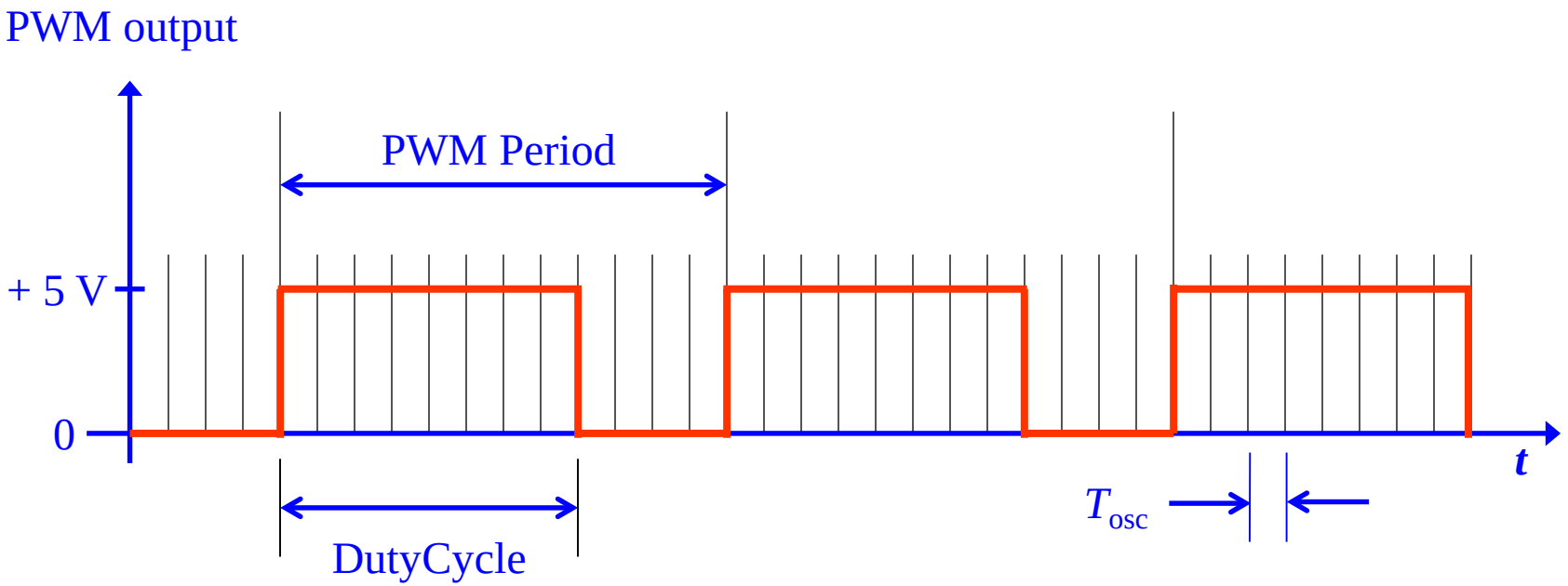
PWM Resolution



Paper Mill

PWM resolution: How finely can the motor speed (duty cycle) be varied?

PWM Resolution

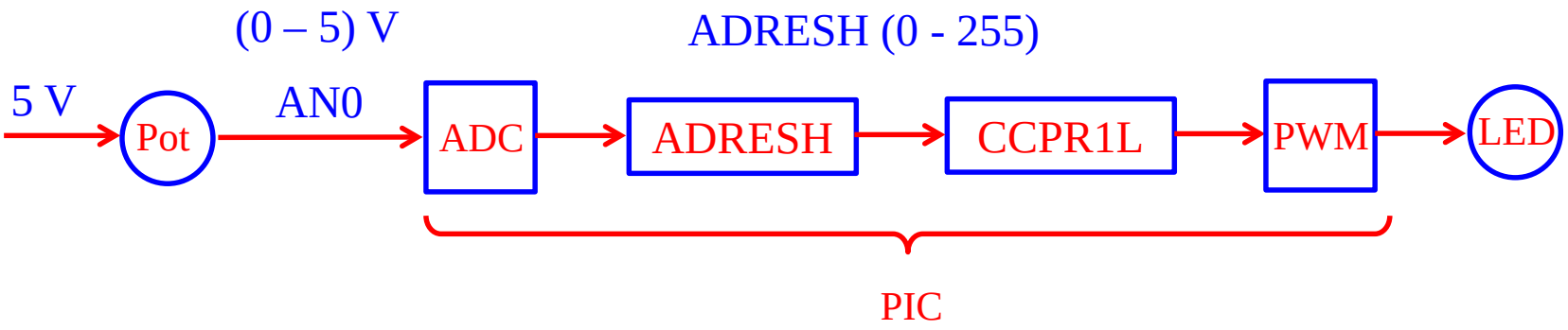


1. We can only make discrete (step) changes in the duty cycle.
2. Result: can only make discrete (step) changes in average voltage.
3. Minimum step change in duty cycle = T_{osc}

Lab 5 Outline

1. CCP Module (PWM mode)
2. Lab 5 Settings
3. Program Memory Paging
4. Electrical Characteristics
5. Sleep and Standalone Modes

lab05 – PWM Scheme



The PWM period is a constant, and the on-time (duty cycle) is adjusted by the pot on the AN0 analog input.

The PWM output is on pin RC2/CCP1.

lab05 - Calculations

1. For a 3.6864 MHz oscillator

   a) 1 clock cycle = Tosc = 0.2713 μs
   b) 1 instr. cycle = 4 Tosc = 1.085 μs

2. Lab05 uses

   a) PR2 = 199
   b) Timer2 Prescale = 1
   c) CCP1CON<5:4> = 00

3. PWM period = (Timer2 Prescale) (PR2 + 1) (4 Tosc)
   = (1)(200)(1.085 μs) = 217.0 μs

4. PWM on-time = (CCPR1L : CCP1CON<5:4>) × Tosc

   10-bit "duty cycle"

   (Cont.)

lab05 - Calculations

4. PWM on-time = (CCPR1L : CCP1CON<5:4>) × 0.2713 μs

   10-bit "duty cycle"

5. So, if we only use the CCPR1L register for the "duty cycle" and we set CCP1CON<5:4> = 00, the on-time range is

   on-time range: (0000 0000 00) (0.2713 μs) to (1111 1111 00) (0.271 μs)

   0
   0 μs
   pot = 0.0 V

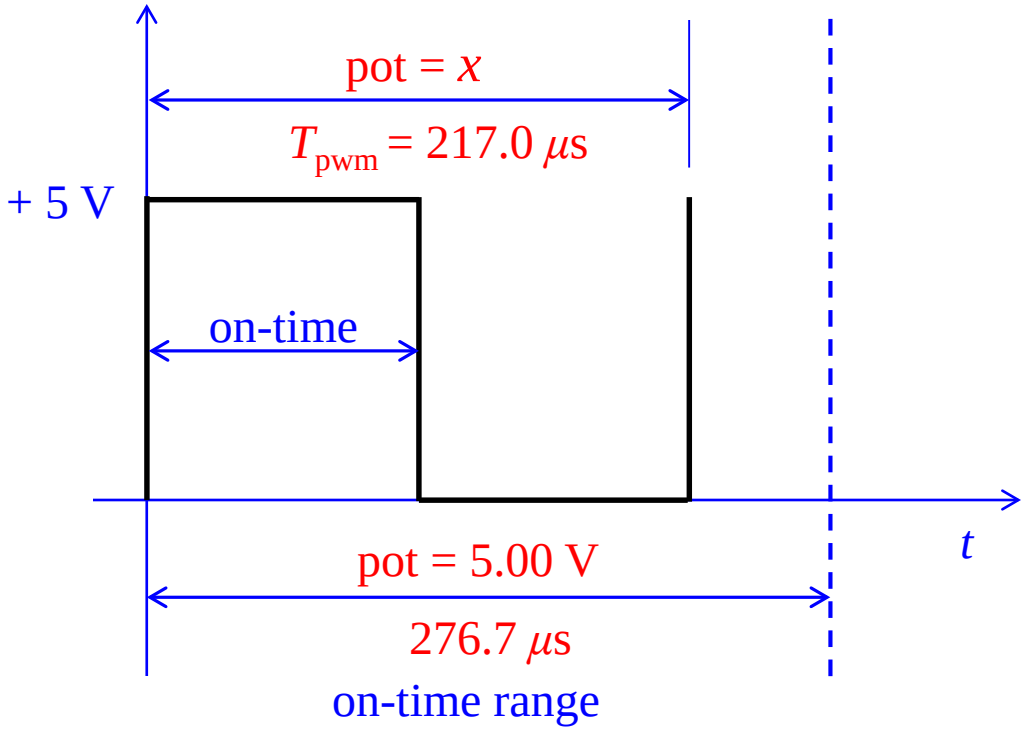
   1020
   276.7 μs
   pot = 5.0 V

   on-time range: 0 μs to 276.7 μs

6. Since the PWM period = 217.0 μs, it is possible for the on-time to exceed the period.

7. If on-time > PWM period, then the duty cycle = 100 %.

lab05 – Duty Cycle



As the pot varies from 0 to 5 volts, the on-time varies from 0 to 276.7 μs.

If T_{pwm} = 217 μs, the duty cycle reaches 100% when on-time = 217.0 μs. So,

$$\frac{217.0}{276.7} = \frac{x}{5.00} = \frac{3.931}{5.00}$$

Lab05 – Calculation Example

1. Assume a 3.6864 MHz oscillator

   a) 1 clock cycle = Tosc = 0.271.7 μs
   b) 1 instr. cycle = 4 Tosc = 1.085 μs

2. Assume

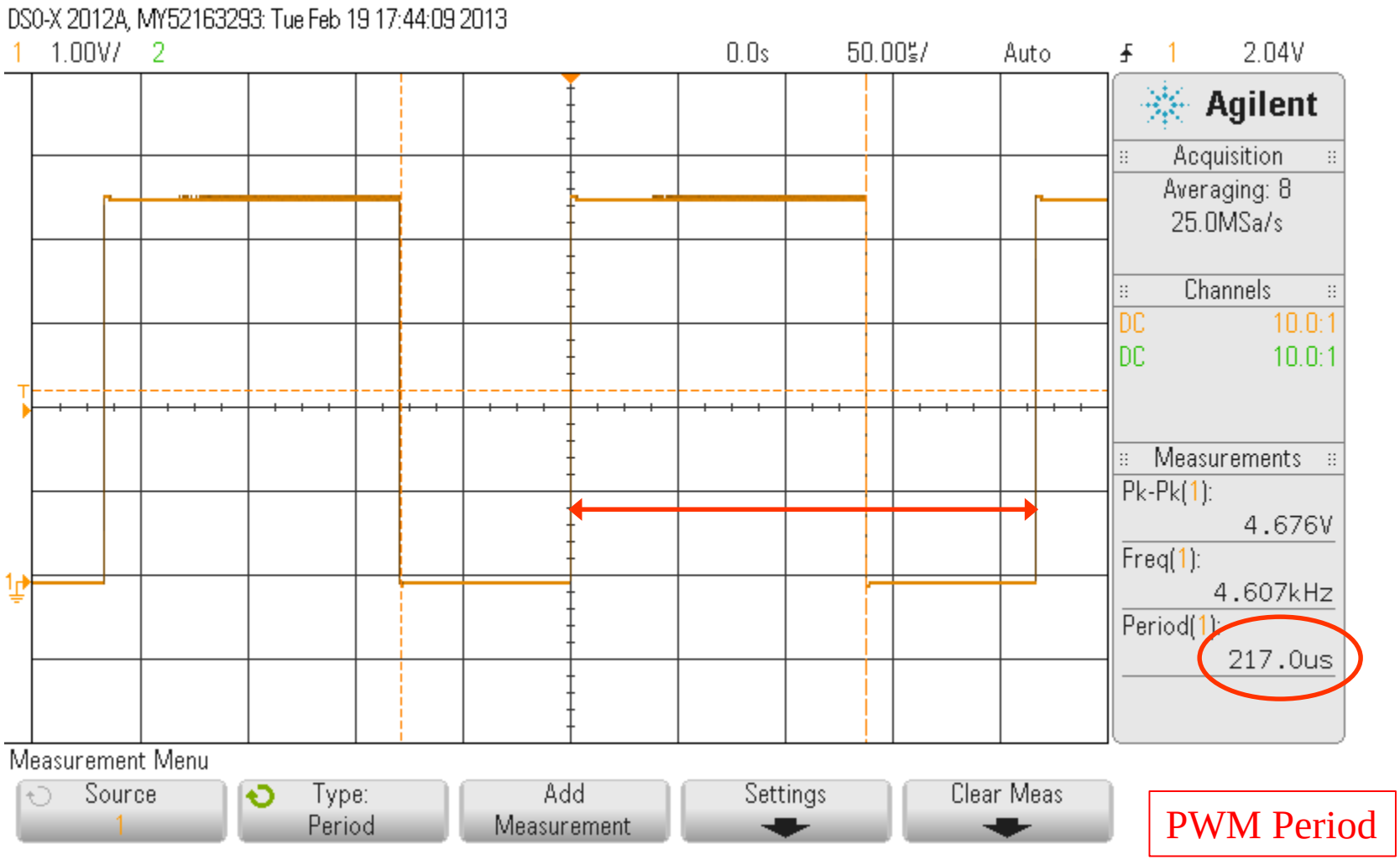
   a) PR2 = 199
   b) Prescaler = 1
   c) CCPR1L = 1000 0000 (= 128: pot set at 2.5 V)
   d) CCP1CON<5:4> = 00

3. PWM period = (PR2+1) × (Timer2 Prescale) × (4 Tosc)
   = 200 × 1 × 1.085 μs = 217.0 μs

4. PWM on-time = (CCPR1L : CCP1CON<5:4>) (Timer2 Prescale) (Tosc)
   = (1000 0000 00) (1) (0.271 μs)
   = (512) (0.2712 μs) = 139 μs

5. Duty cycle = on-time / period = 139 / 217 = 64 %

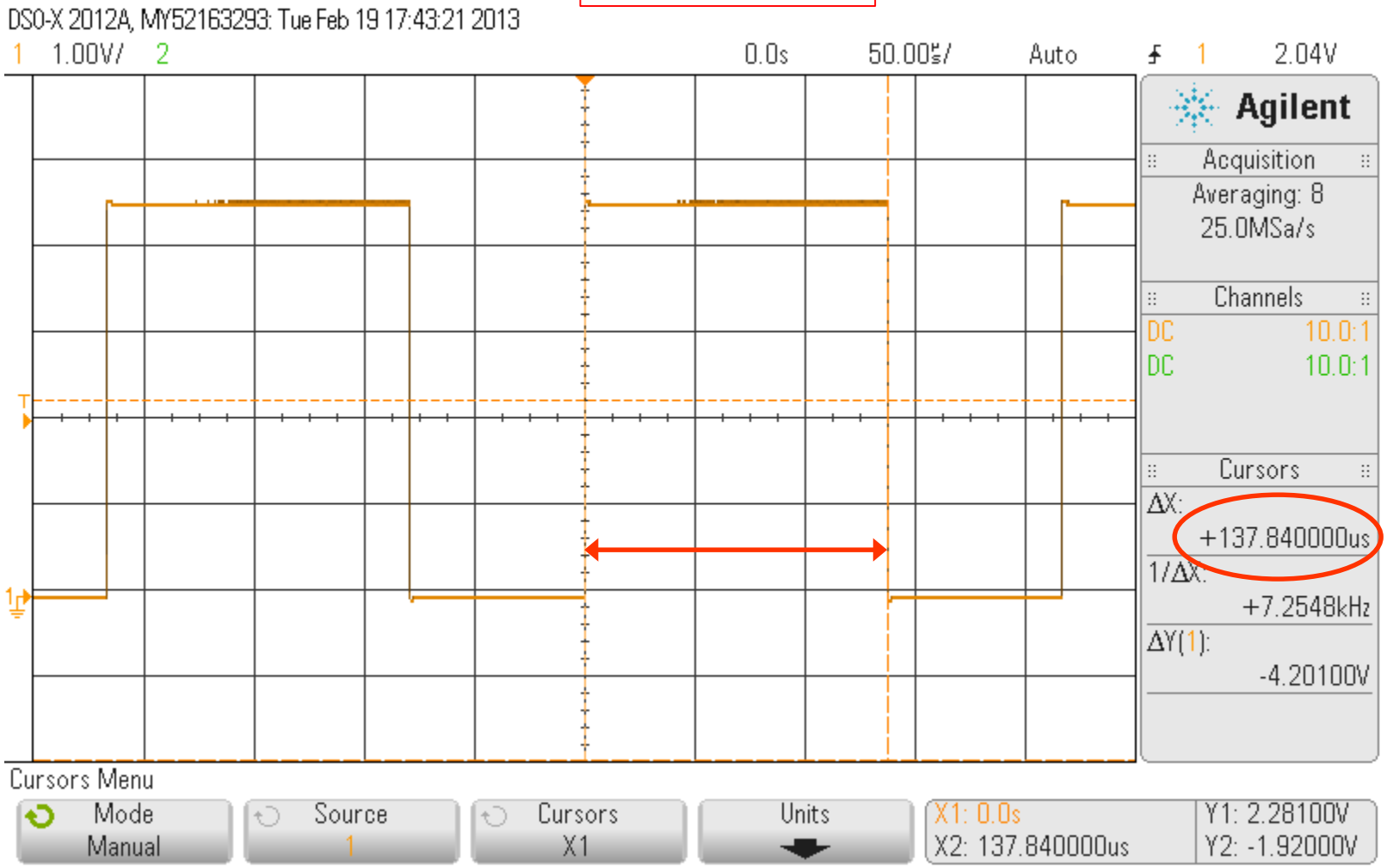
Lab05 – RC2 output @ 2.489 volts on AN0



PWM Period

Lab05 – RC2 output @ 2.489 volts on AN0

PWM On-Time



lab05 - Initialization

```
Init
    movlw   B'01000001' ; A/D enabled at a frequency of Fosc/8
    banksel ADCON0      ; ADCON0 in Bank 0
    movwf   ADCON0

    banksel ADCON1      ; ADCON1 in Bank 1
    movlw   B'00001110' ; Left justify A/D data, 1 analog channel
    movwf   ADCON1      ; Use VDD and VSS for A/D references

    movlw   D'199'      ; PWM period = (PR2+1)(Timer2 Prescale)(4 Tosc)
    movwf   PR2         ; PR2 in Bank 1

    movlw   B'10000000' ; PWM on-time = (DC)(Timer2 Prescale)(Tosc)
                        ; DC = CCPR1L : CCP1CON<5,4>
    banksel CCPR1L      ; CCPR1L in Bank 0
    movwf   CCPR1L      ; CCPR1L = 1000 0000

    movlw   B'00001100' ; CCP1CON<5,4> = 00 (CCP1COn in Bank 0)
    movwf   CCP1CON     ; DC = 10 0000 0000

    movlw   B'11111011' ; Set RC2 as output for PWM signal
    banksel TRISC       ; TRISC in Bank 1
    movwf   TRISC       ; PORTC = 1111 1011

    movlw   B'10000000' ; Set up Timer0 for A/D acquisition delay.
    movwf   OPTION_REG  ; Timer0 prescaler = 2, rollover = 556 usec

    movlw   B'00000100' ; Timer2 prescaler = 1
    banksel T2CON       ; T2CON in Bank 0
    movwf   T2CON
```

lab05 – PWM Code

```
Main
    ; Timer0 delay for A/D voltage acquisition

    btfss   INTCON,T0IF ; Test the TIMER0 interrupt flag bit (T0IF).
                        ; The INTCON register is in Bank 0.
                        ; If T0IF = 1 (TMR0 rollover), skip next
                        ; instruction (skip goto).
    goto    Main

    bcf     INTCON,T0IF ; Clear the T0IF bit for the next interrupt.

    banksel ADCON0      ; ADCON0 in Bank 0

    bsf     ADCON0,GO   ; Start the A/D conversion

WaitForConversion

    btfss   PIR1, ADIF  ; Wait for conversion to complete
    goto    WaitForConversion

    bcf     PIR1, ADIF
    movf    ADRESH, W   ; Get the A/D result
    movwf   CCPR1L      ; Use the A/D result for the PWM duty cycle
    goto    Main        ; Do it again
```

Lab 5 Outline

1. CCP Module (PWM mode)
2. Lab 5 Settings
3. Program Memory Paging
4. Electrical Characteristics
5. Sleep and Standalone Modes

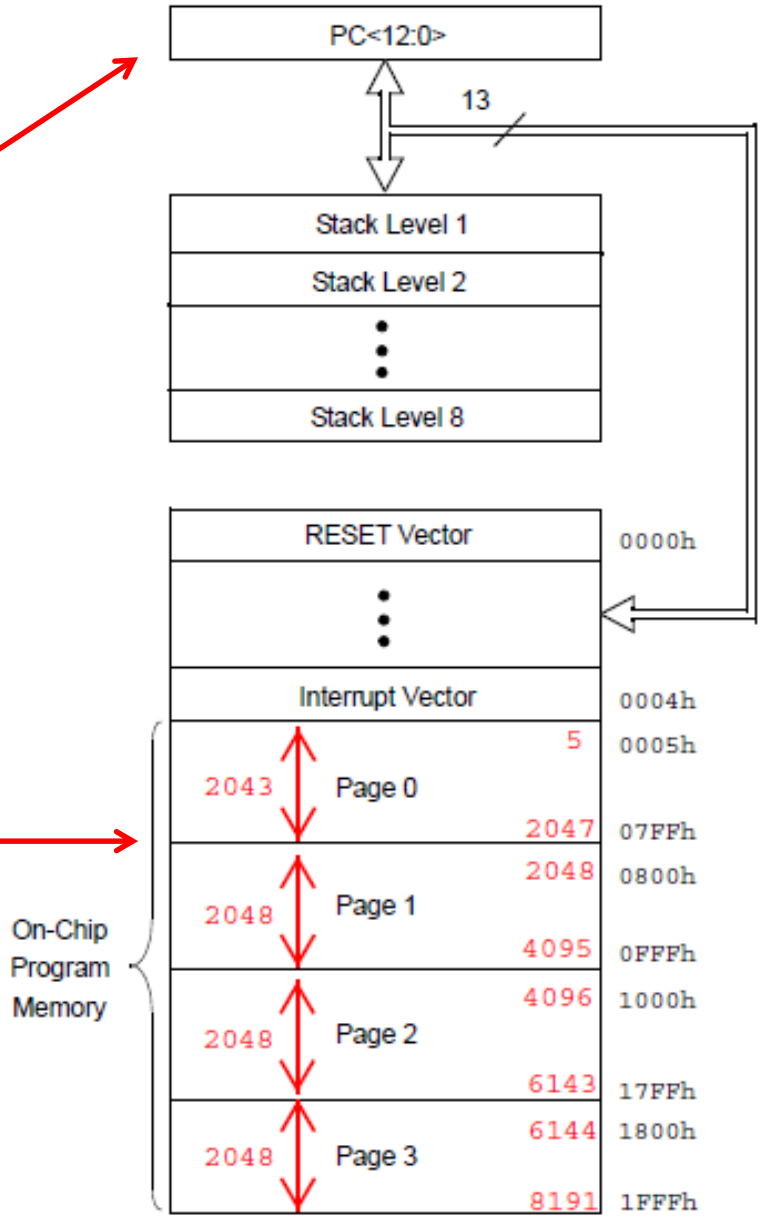
Program Memory Paging

PC = Program Counter (13 bits)

$2^{13} = 8192$

Memory Page Boundary

$2^{11} = 2048$



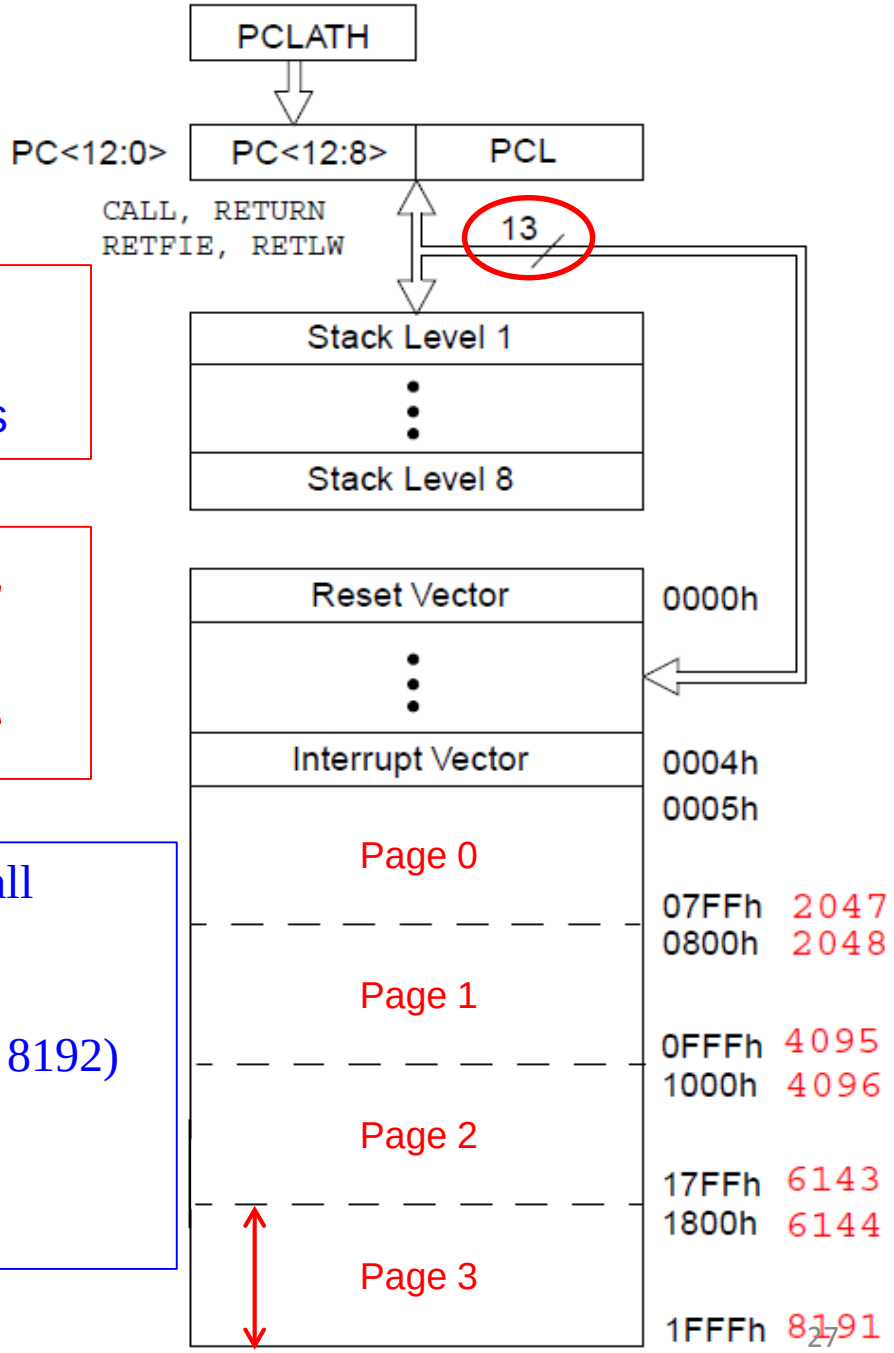
Program Memory Paging

```
call  <label>

<label> interpreted as address
```

```
call  k  ⇒  10 0kkk kkkk kkkk

goto  k  ⇒  10 1kkk kkkk kkkk
```

11-bits

1. There are 11 bits for the address in the call and goto instructions. ($2^{11} = 2048$)
2. Program memory requires 13 bits. ($2^{13} = 8192$)
3. 11 bits only specify the **relative** address on each page.



Program Memory Paging

Example:

```
0x0200    Call Sub1      ; Machine code: 10 0kkk kkkk kkkk
...                      ; 0x0900 = 2304
0x0900    Sub1           ; On Page 1: 2048 < 2304 < 4095
...                      ; 0x0900 =    0000 1001 0000 0000
...       Return         ; Machine code: 10 0001 0000 0000
```

1. There are not enough address bits in the call instruction to include all the bits of the Sub1 address.

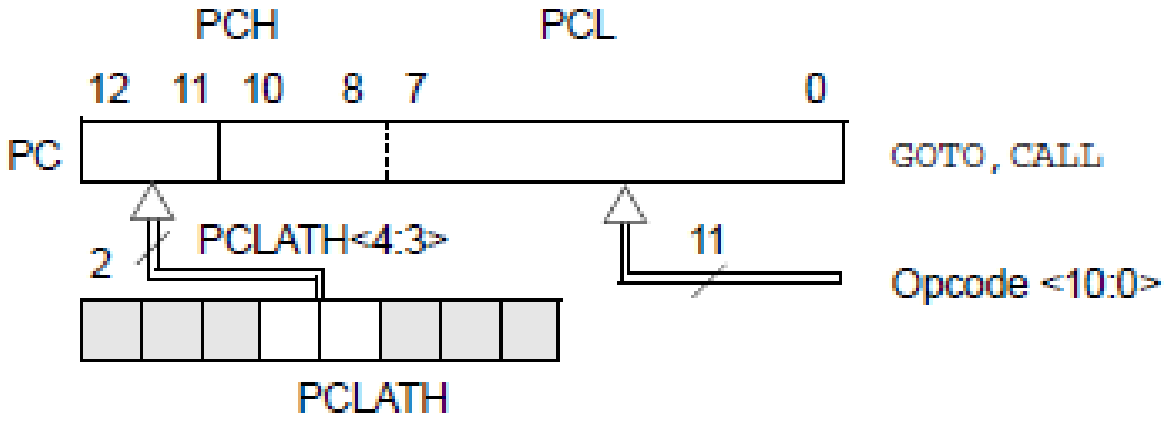
2. The call instruction will transfer the PC to address 0x0100.

3. We need two more bits to specify a 13-bit address – these two bits are called the "**page select bits**"

Program Memory Paging

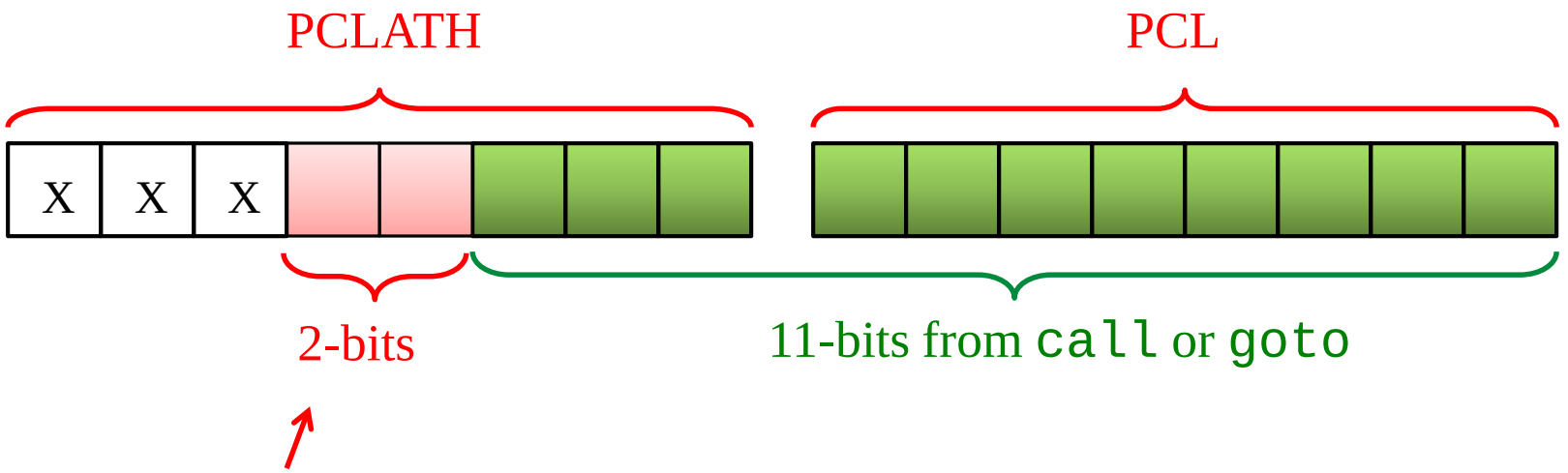
1. The page select bits PC<12:11> in the PC register cannot be accessed directly.

2. We must use the PCLATH<4:3> bits.



Page 26, Data Sheet

Program Memory Paging

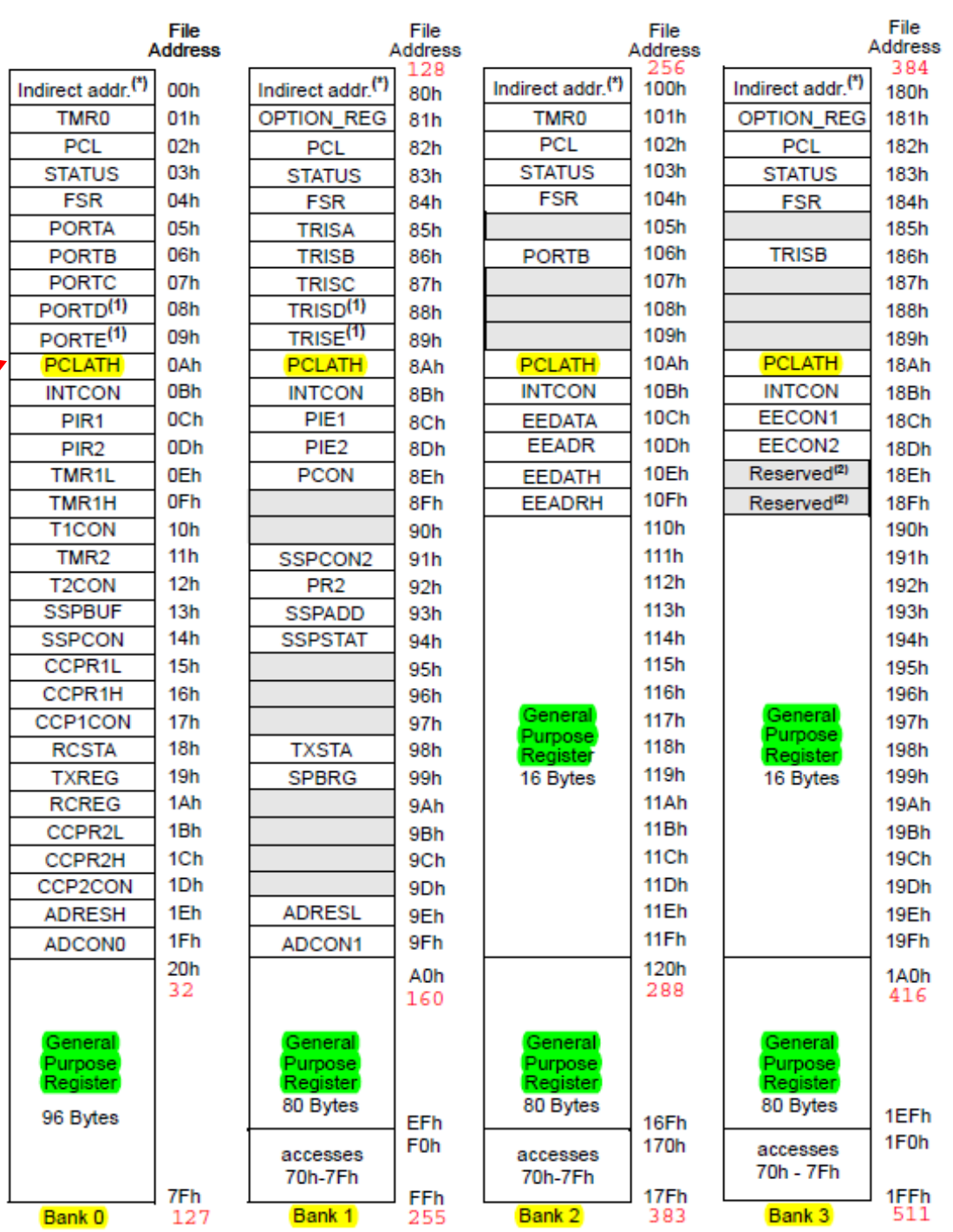


1. PCLATH<4:3> = page select bits

2. Before a `call` or `goto`, the PIC looks at the page select bits to decide which page to transfer to.

Program Memory Paging

Don't have to change banks to select PCLATH



Program Memory Paging

1. Page select bits

   00 = page 0 (starts at 0x0000)
   01 = page 1 (starts at 0x0800 = 2048)
   10 = page 2 (starts at 0x1000 = 4096)
   11 = page 3 (starts at 0x1800 = 6144)

2. Each page has an 11-bit range of addresses: $2^{11} = 2048$

3. Relative range: address 0 to 0x7FF (0 to 2047)

Program Memory Paging

```
0x0200    bcf  PCLATH, 4  ; Change from Page 0 to Page 1
0x0201    bsf  PCLATH, 3  ; PCLATH<4:3> = 01
0x0201    call Sub1       ; Machine code: 10 0kkk kkkk kkkk
...
0x0900    Sub1            ;    0x0900 = 0000 1001 0000 0000
...
...       Return
```

1. Sub1 = 0x0900 = 0000 1001 0000 0000

2. The `call` instruction will transfer the PC to 0x100 relative to the start of page 1 (0x0800), which is address 0x0900.

3. Do we need to change the page select bits before the return statement in a call subroutine? (Homework question)

Program Memory Paging

1. Instead of using `bcf` or `bsf` to set the page select bits in PCLATH, we can use the `pagesel` directive; for example, `pagesel SUB1`.

2. The `pagesel` directive results in two assembly code instructions:

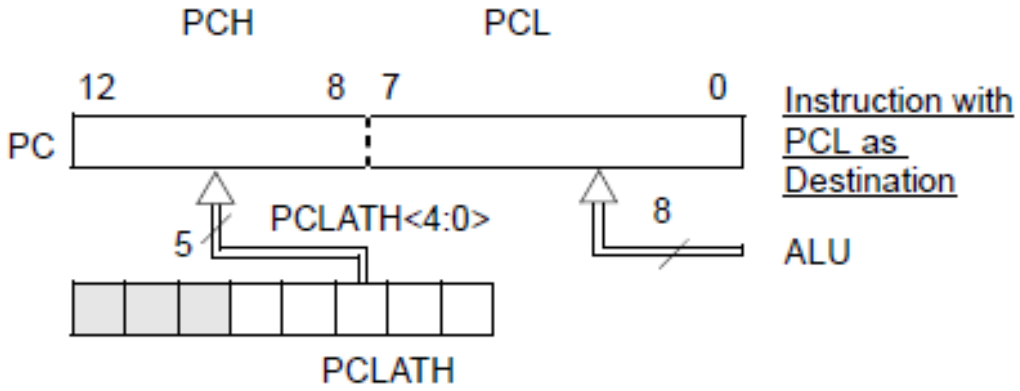
```
pagesel SUB1   ⇔      bcf   PCLATH, 4
                      bsf   PCLATH, 3
```

Program Memory Paging

Computed `goto`

A computed `goto` is any instruction which changes the program counter (PC) directly.

Any instruction **with PCL as the destination** (such as the `addwf PCL, F` instruction in LookupTable.asm) will load the PCH register with the 5 low bits PCLATH<4:0> from the PCLATH register.

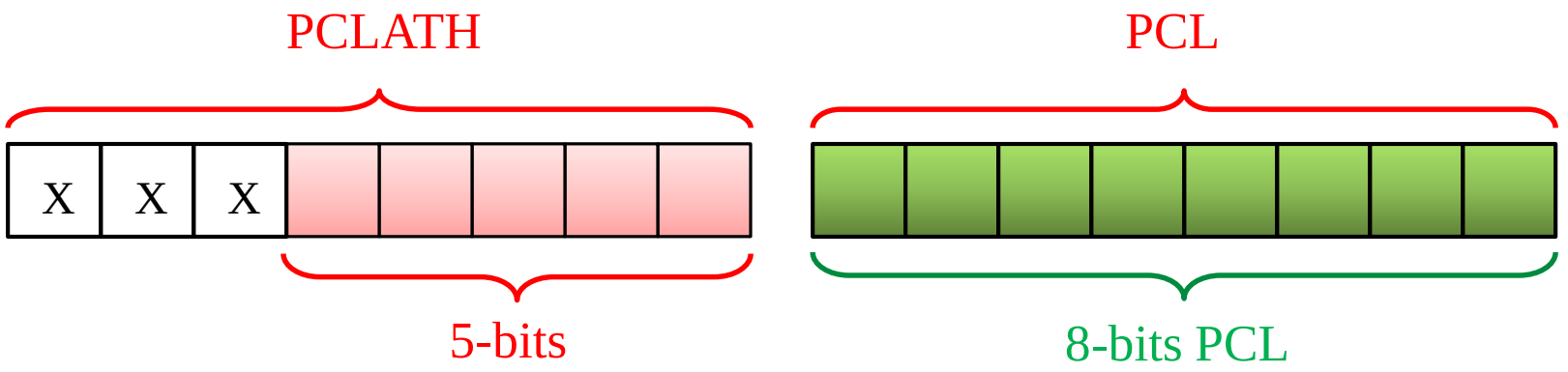


Page 26, Data Sheet

Program Memory Paging

Computed `goto`

When doing a table read using a computed `goto` method, care should be exercised if the table location crosses a PCL memory boundary (each 256 word block of program memory determined by the 8-bit PCL register). Refer to the application note, “Implementing a Table Read“ (AN556).



Computed `goto`: 256 byte program memory boundary

	Address	Program Memory
	⋮	
	0x0027	movlw D'4'
	0x0028	call Lookup
	0x0029	Instruction 29
	⋮	
Lookup	0x00FC	addwf PCL, F
W = 0	0x00FD	retlw B'00000001'
W = 1	0x00FE	retlw B'00000010'
W = 2	0x00FF	retlw B'00000100'
W = 3	0x0100	retlw B'00001000'
W = 4	0x0101	retlw B'00010000'
W = 5	0x0102	retlw B'00100000'

PCL = PCL + W + 1

PCL = 252 = 0xFC
= 000 1111 1100

256 word boundary

PCL = 252 + 4 + 1
= 0x0101
= 001 0000 0001

The example will not work because of the page boundary crossing. See datasheet.

Lab 5 Outline

1. CCP Module (PWM mode)
2. Lab 5 Settings
3. Program Memory Paging
4. Electrical Characteristics
5. Sleep and Standalone Modes

15.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

Ambient temperature under bias	-55 to +125°C
Storage temperature	-65°C to +150°C
Voltage on any pin with respect to V_{SS} (except V_{DD}, $\overline{MCLR}$, and RA4)	-0.3 V to (V_{DD} + 0.3 V)
Voltage on V_{DD} with respect to V_{SS}	-0.3 to +7.5 V
Voltage on $\overline{MCLR}$ with respect to V_{SS} **(Note 2)**	0 to +14 V
Voltage on RA4 with respect to Vss	0 to +8.5 V
Total power dissipation **(Note 1)**	1.0 W
Maximum current out of V_{SS} pin	300 mA
Maximum current into V_{DD} pin	250 mA
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD})	± 20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD})	± 20 mA
Maximum output current sunk by any I/O pin	25 mA
Maximum output current sourced by any I/O pin	25 mA
Maximum current sunk by PORTA, PORTB, and PORTE (combined) **(Note 3)**	200 mA
Maximum current sourced by PORTA, PORTB, and PORTE (combined) **(Note 3)**	200 mA
Maximum current sunk by PORTC and PORTD (combined) **(Note 3)**	200 mA
Maximum current sourced by PORTC and PORTD (combined) **(Note 3)**	200 mA

Note 1: Power dissipation is calculated as follows: $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$

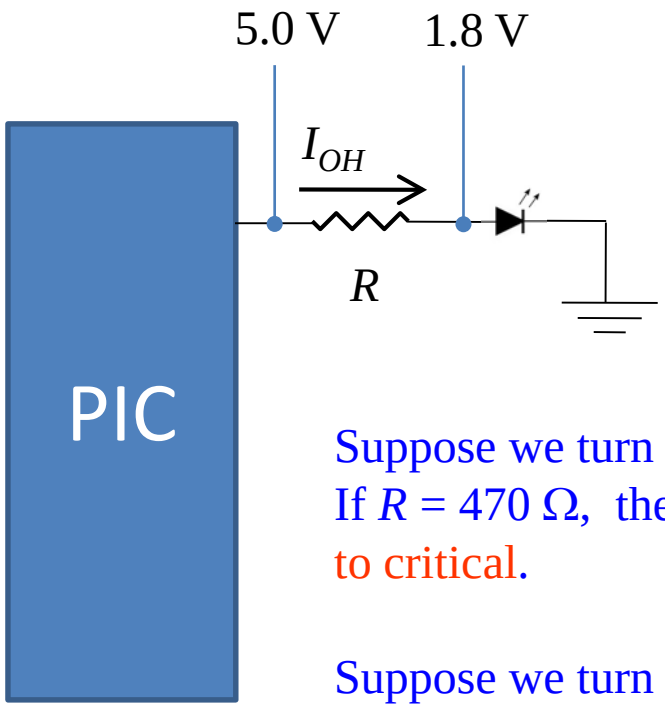
2: Voltage spikes below V_{SS} at the $\overline{MCLR}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$ pin, rather than pulling this pin directly to V_{SS}.

3: PORTD and PORTE are not implemented on PIC16F873/876 devices.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

LED Connections

Voltage drop across red LED ≈ 1.8 V



$$I_{OH} = \frac{5.0 - 1.8}{R} = \begin{cases} 6.8 & \text{mA if } R = 470\ \Omega \\ 68 & \text{mA if } R = 47\ \Omega \end{cases}$$

Suppose we turn on all 25 LEDs connected to Ports A, B, C, D. If R = 470 Ω, the total current sourced is 25(6.8) = 170 mA – close to critical.

Suppose we turn on all 33 LEDs connected to Ports A, B, C, D, and E. If R = 470 Ω, the current out of each I/O pin is 6.8 mA.

$P = 33\ I_{OH}\ V_{OH} = 33(0.0068)(5) = 1.12$ W,

exceeding the maximum 1.0 W.

PIC Power Dissipation

1. Nominal operating conditions:

   a) 4 MHz oscillator
   b) No current sunk or sources by I/O pins
   c) Timers off, ADC off
   d) Nominal current draw I_{DD} = 2 mA

2. PIC power consumed:

   $P = V_{DD} I_{DD} = (5\text{ V})\ (2\text{ mA}) = 10\text{ mW}$

3. How long will the PIC last on a 9 V battery?

PIC Power Dissipation

1. PIC power consumed = 10 mW.

2. 9 V Energizer 522 battery current capacity ≈ 600 mA-h.

3. Battery power capacity = (9 V)(600 mA-h) = 5400 mW-h

4. $$\text{Battery Life} \approx \frac{\text{Battery power capacity}}{\text{PIC power consumption}}$$
$$= \frac{5400 \text{ mW-h}}{10 \text{ mW}} = 540 \text{ hours}$$
$$= 22.5 \text{ days}$$

5. Conserve energy by going into sleep mode, also called power-down mode.

Lab 5 Outline

1. CCP Module (PWM mode)
2. Lab 5 Settings
3. Program Memory Paging
4. Electrical Characteristics
5. Sleep and Standalone Modes

15.1 DC Characteristics: PIC16F873/874/876/877-04 (Commercial, Industrial)
PIC16F873/874/876/877-20 (Commercial, Industrial)
PIC16LF873/874/876/877-04 (Commercial, Industrial)
(Continued)

PIC16LF873/874/876/877-04 (Commercial, Industrial)			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial				
PIC16F873/874/876/877-04 PIC16F873/874/876/877-20 (Commercial, Industrial)			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial				
Param No.	**Symbol**	**Characteristic/ Device**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	IPD	**Power-down Current(3,5)**					
D020		16LF87X	—	7.5	30	µA	VDD = 3.0V, WDT enabled, -40°C to +85°C
D020		16F87X	—	10.5	42	µA	VDD = 4.0V, WDT enabled, -40°C to +85°C
D021		16LF87X	—	0.9	5	µA	VDD = 3.0V, WDT enabled, 0°C to +70°C
D021		16F87X	—	1.5	16	µA	VDD = 4.0V, WDT enabled, -40°C to +85°C
D021A		16LF87X		0.9	5	µA	VDD = 3.0V, WDT enabled, -40°C to +85°C
D021A		16F87X		1.5	19	µA	VDD = 4.0V, WDT enabled, -40°C to +85°C
D023	ΔIBOR	**Brown-out Reset Current(6)**	—	85	200	µA	BOR enabled, VDD = 5.0V

Sleep mode

Sleep mode – Data sheet p. 154 – Current draw = 1.5 µA

Sleep Mode

1. Normal operating mode: I_{DD} = 2 mA,
   battery life ≈ 22.5 days.

2. Sleep mode: I_{DD} = 1.5 μA, so the battery will last
   2 mA/1.5 μA ≈ 1333 times longer, or

   battery life ≈ 1333 × 22.5 days ≈ 82 years.

3. The `sleep` instruction puts PIC in sleep mode.

4. Clock stops in sleep mode, instructions do not execute.

5. Note: Sleep mode might not work in debug mode

Standalone Mode / Release Mode

1. Select **Programmer → Select Programmer → PICkit3**
2. Program the PIC
3. Disconnect PIC from debugger
4. Reset or power cycle the PIC

End of Lab 5