Microprocessors & Interfacing

Analog Input/Output (I)

Lecturer : Annie Guo

Lecture Overview

- Analog output
  - PWM

PWM Analog Output

模拟

脉冲宽度调制

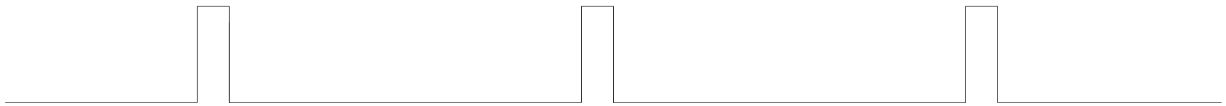
- PWM (Pulse Width Modulation) is a way of digitally encoding analog signal levels.
  - By using high-resolution counters, the duty cycle (pulse width/period) of a pulse wave is modulated 调节 to encode a specific analog signal level.
- PWM is a powerful technique for controlling analog circuits/devices with the processor's digital output.
- It is used in a wide variety of applications
  - E.g. motor speed control

PWM Analog Output (cont.)

- The PWM signal is still digital
  - Its value is either full high or full low.
- A low-pass filter is required to smooth the input signal and eliminate the inherent noise components in PWM signal.
  排除 内在的，固有的
- The output voltage is directly proportional to the pulse width.
  电压 成比例的
  - By changing the pulse width of the PWM waveform, we can control the output value.

PWM Signal Examples

Duty cycle=10%



Duty cycle=50%

Duty cycle=90%

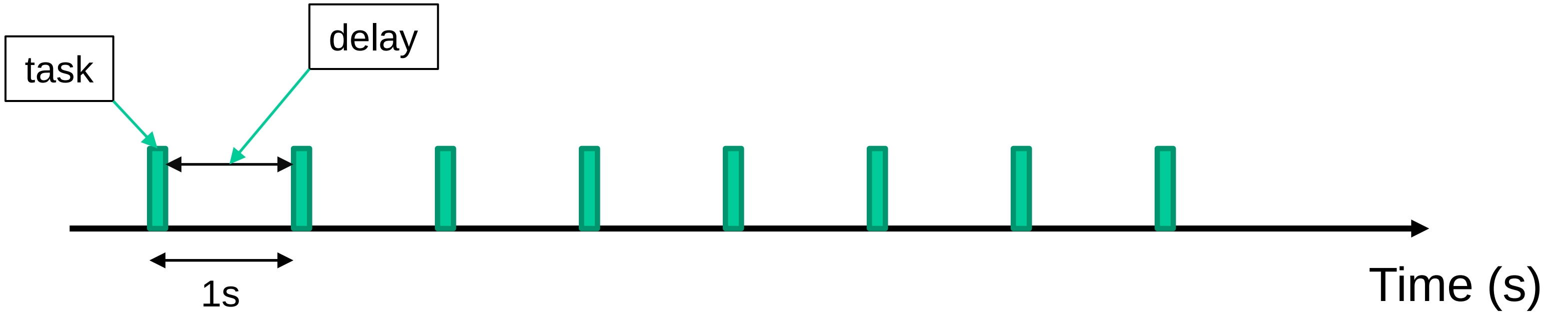
•Duty cycle: (pulse width)/period

PWM Generation in AVR

- PWM can be obtained through the provided timers.

Recall: Example 1 (Mon. Week 7)

- Implement a scheduler that can execute a task every one second.
  - Can be realized with
    - software design,
      - Software generates the delay
        - » With nop instructions
        - » With other tasks of known execution time
    - hardware design
      - Used here and solution is given in the next slides

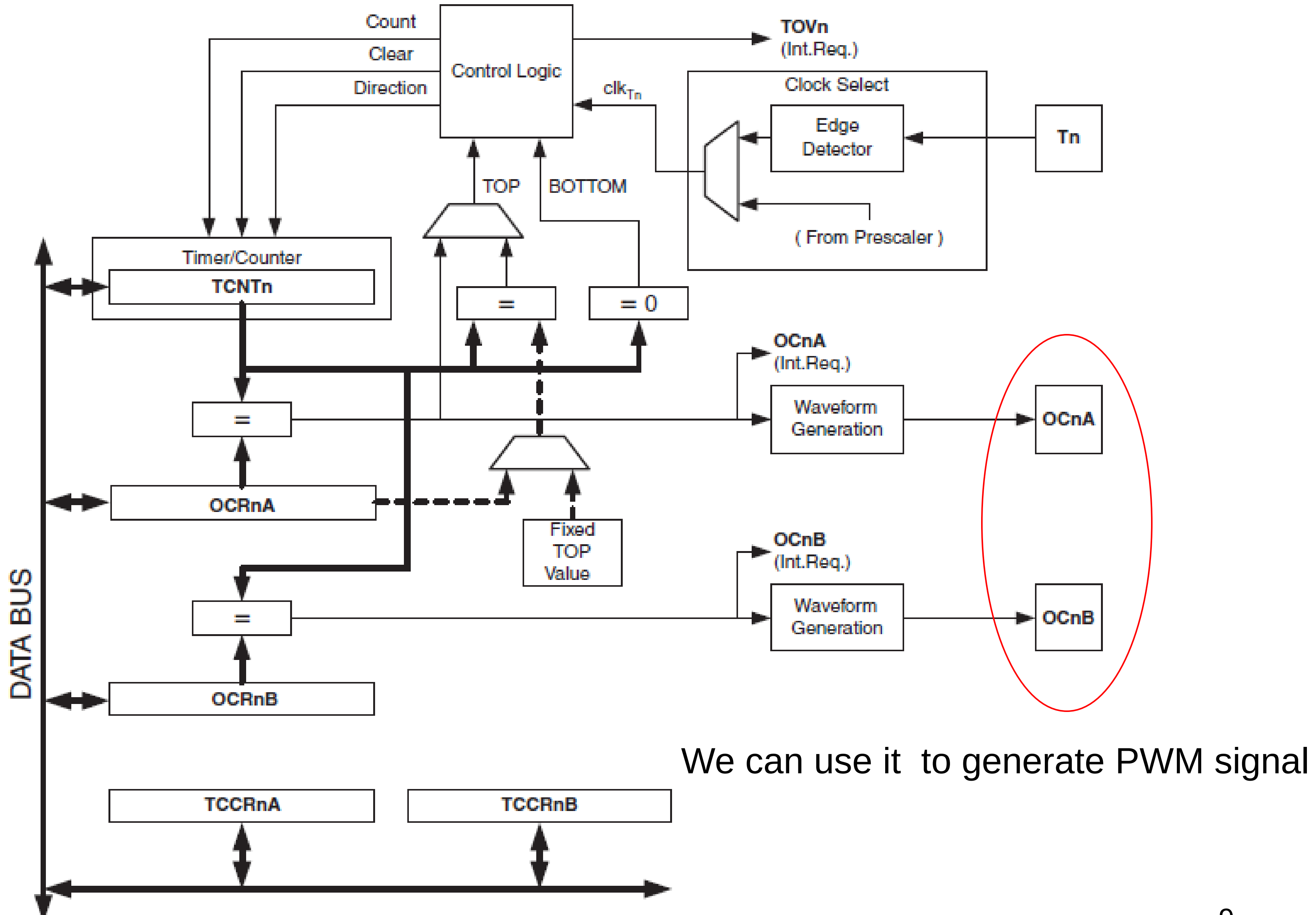


Recall: Example 1 Solution

- Use **8-bit** Timer0 to “count” the time
  - Let’s set Timer0 prescaler to /64 (i.e. the system frequency is divided by 64)
    - The full counting duration (time-out) for the setting should be
      - 256x(clock period) = 256x64/(16 MHz)

        = 1024 us
        - » Namely, we can set the Timer0 overflow interrupt that is to occur every 1024 us.
        - » Note, clock period = 1/16 MHz (obtained from the data sheet); the 8-bit counter can count 256 clock cycles.
    - For one second, there are
      - 1000000/1024 =~ 1000 interrupts

Recall: Timer0



We can use it to generate PWM signal

Configuration for PWM

- TCCR0A/B

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Mode	WGM2	WGM1	WGM0	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on[1][2]
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	TOP	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Configuration for PWM (cont.)

- TCCR0A/B
  - Phase Correct PWM

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Table 16-4. Compare Output Mode, Phase Correct PWM Mode[1]

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected WGM02 = 1: Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match when up-counting. Set OC0A on Compare Match when down-counting
1	1	Set OC0A on Compare Match when up-counting. Clear OC0A on Compare Match when down-counting

Configuration for PWM (cont.)

- TCCR0A/B
  - Fast PWM

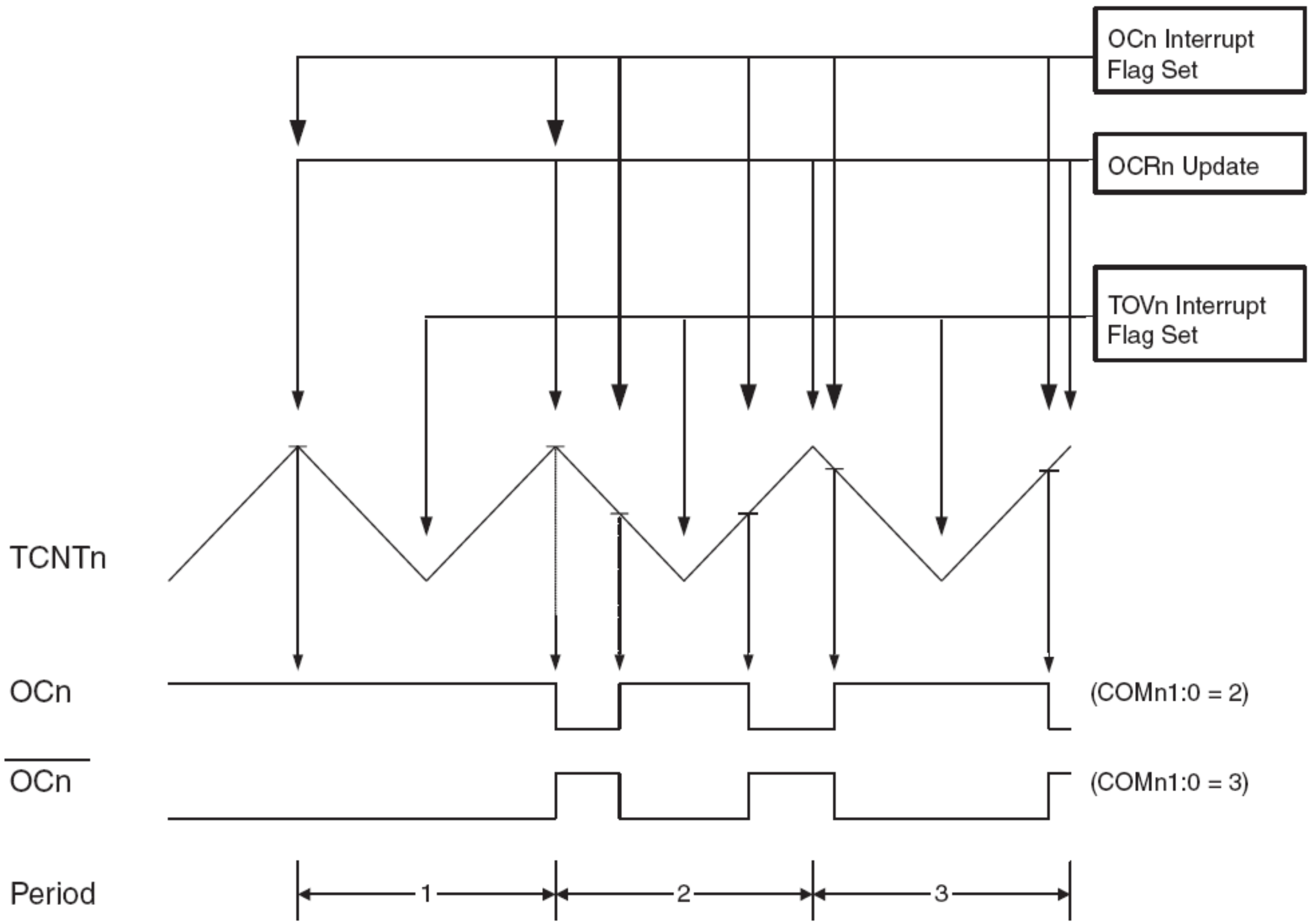
Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

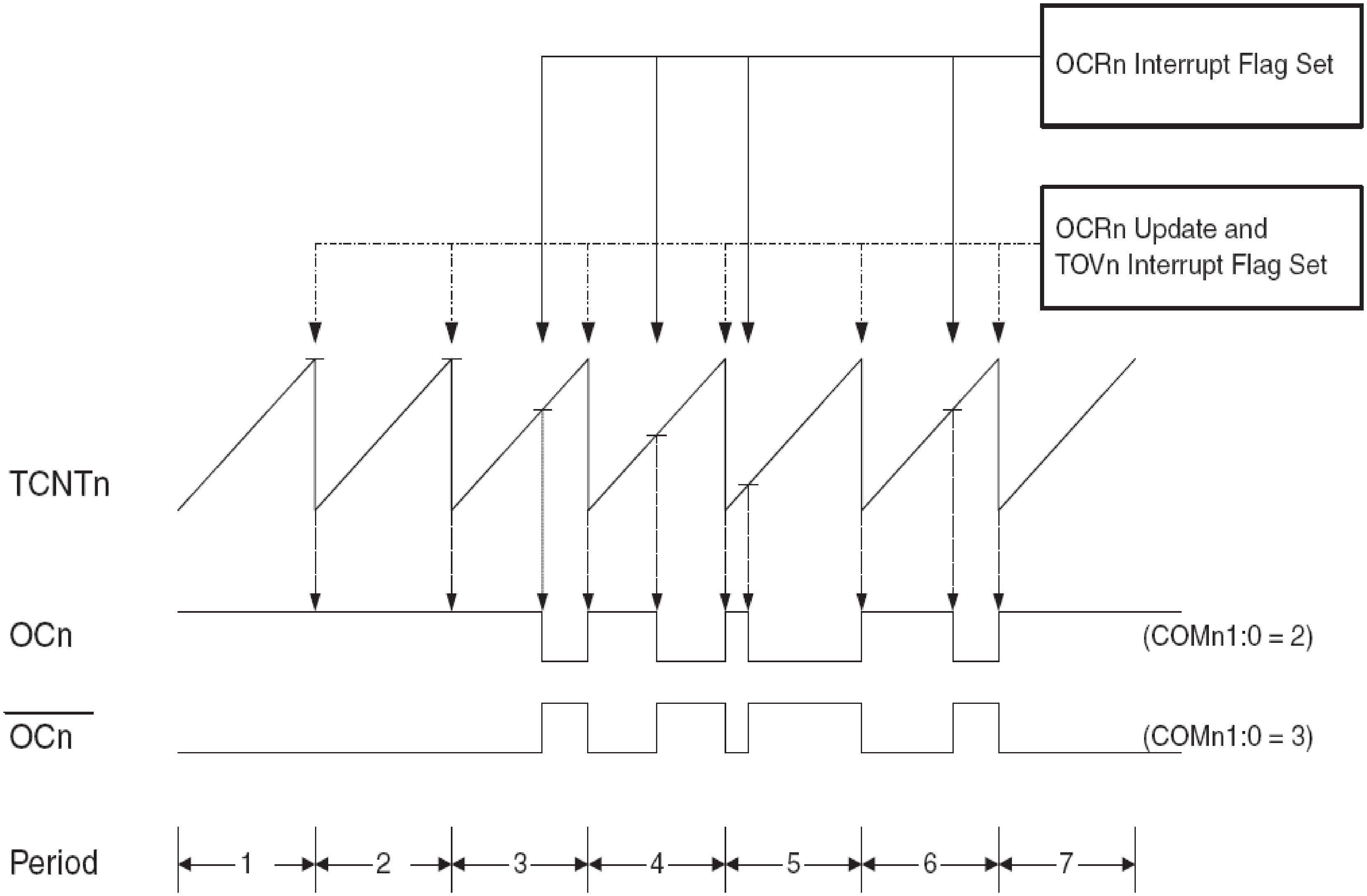
Table 16-3. Compare Output Mode, Fast PWM Mode[(1)]

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected WGM02 = 1: Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match, set OC0A at BOTTOM (non-inverting mode)
1	1	Set OC0A on Compare Match, clear OC0A at BOTTOM (inverting mode)

Phase Correct PWM

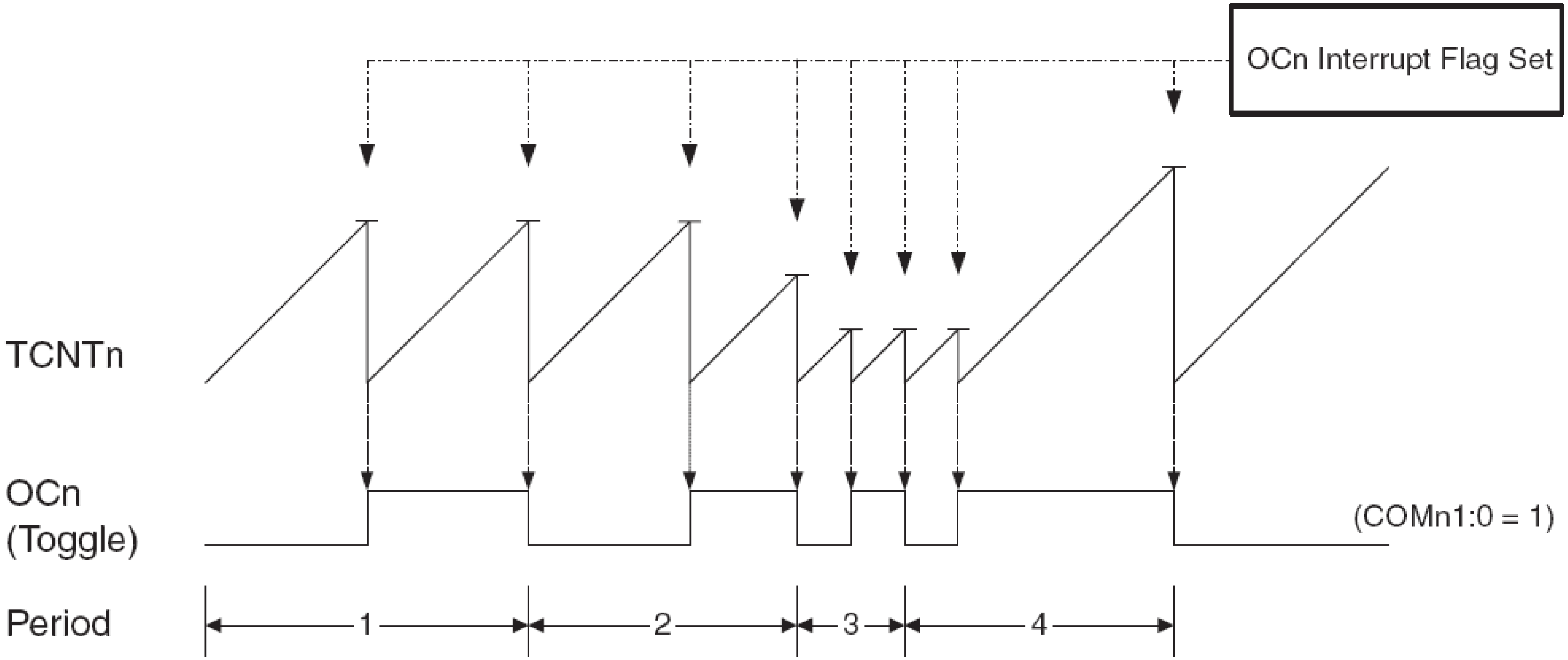


Fast PWM


OCRn Interrupt Flag Set
OCRn Update and
TOVn Interrupt Flag Set
TCNTn
OCn
(COMn1:0 = 2)
OCn
(COMn1:0 = 3)
Period
1
2
3
4
5
6
7

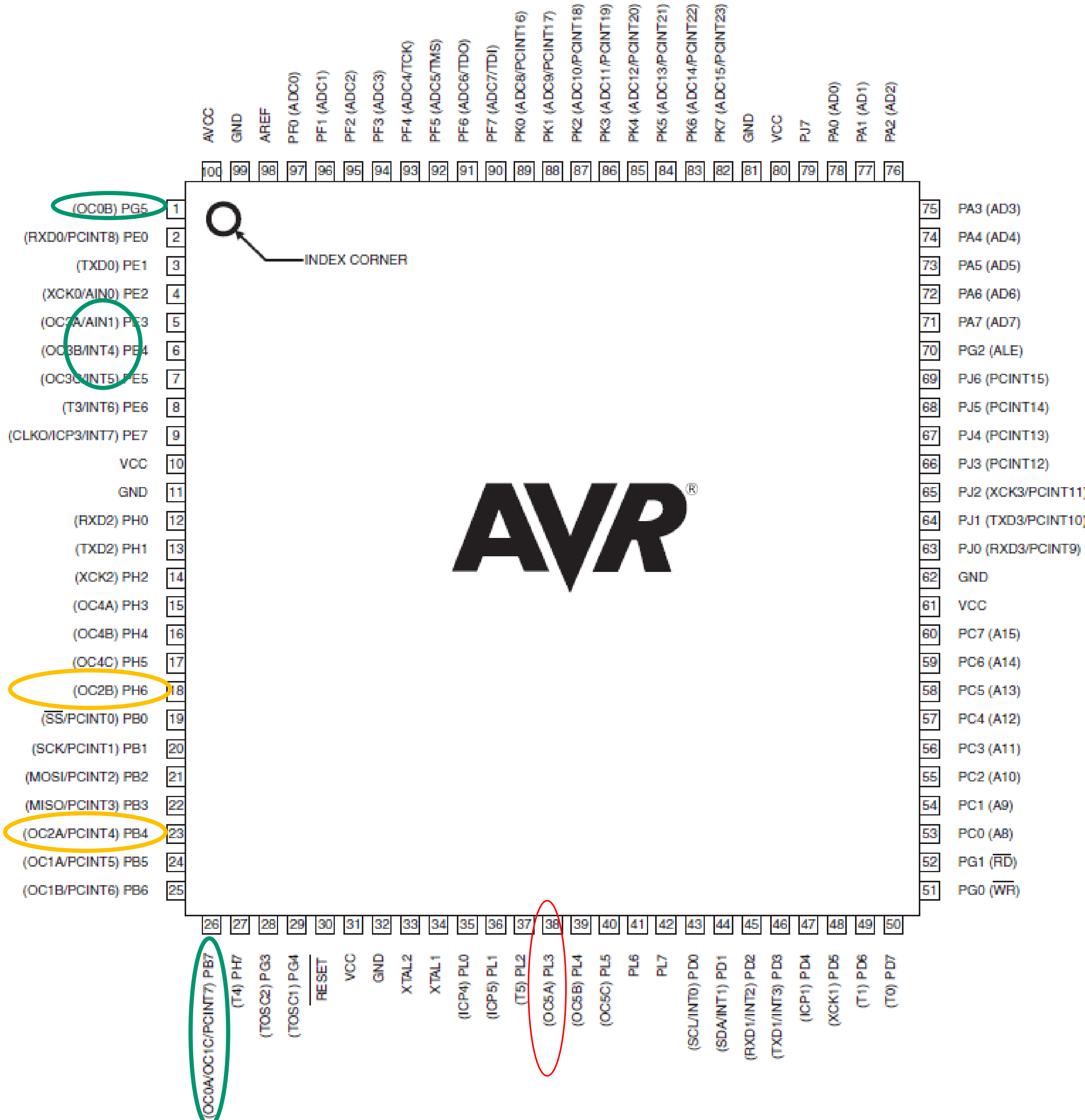
CTC*

- Clear Timer on Compare Match



Example

- Generate a PWM waveform.



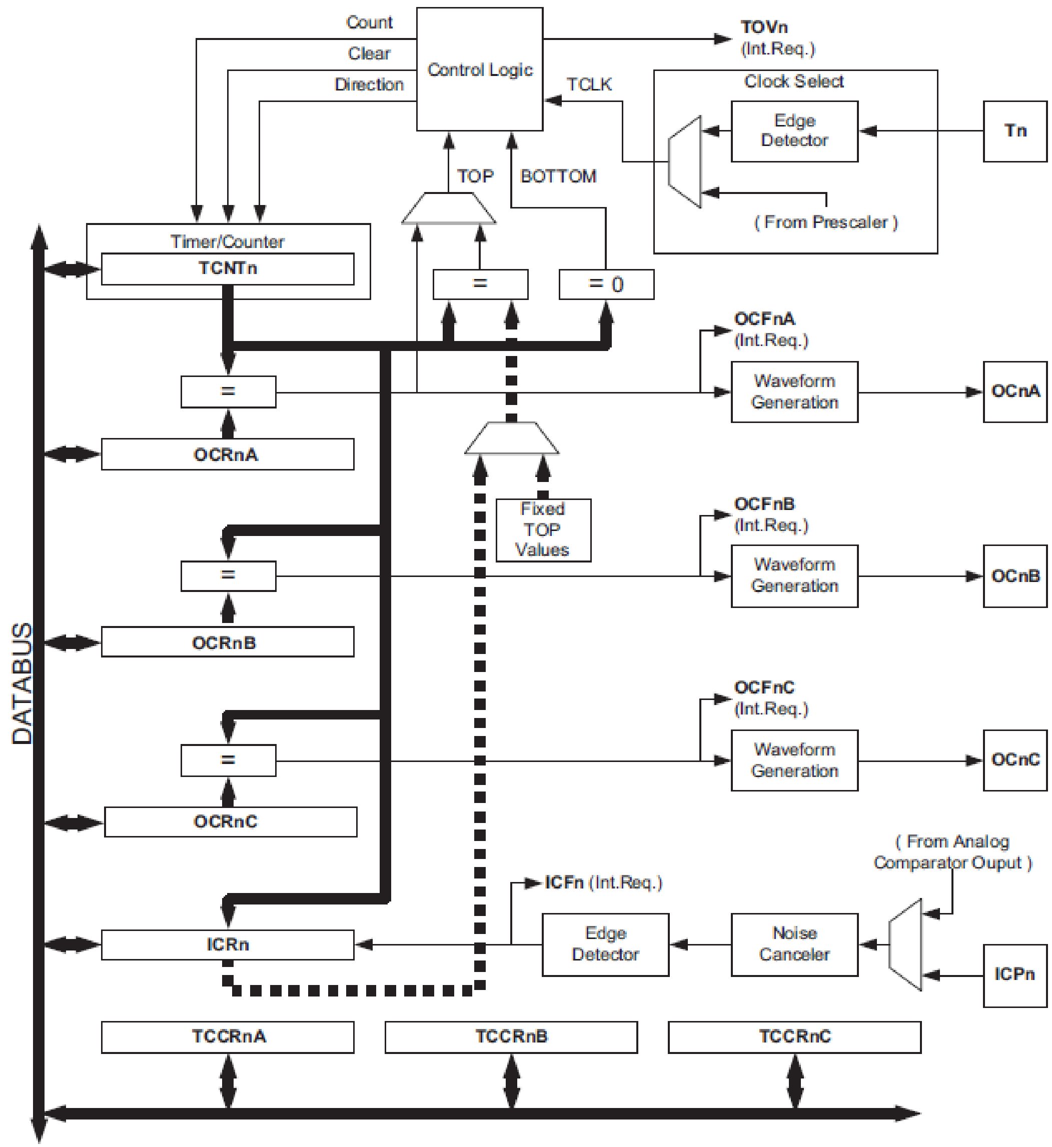
AVCC 100
GND 99
AREF 98
PF0 (ADC0) 97
PF1 (ADC1) 96
PF2 (ADC2) 95
PF3 (ADC3) 94
PF4 (ADC4/TCK) 93
PF5 (ADC5/TMS) 92
PF6 (ADC6/TDO) 91
PF7 (ADC7/TDI) 90
PK0 (ADC8/PCINT16) 89
PK1 (ADC9/PCINT17) 88
PK2 (ADC10/PCINT18) 87
PK3 (ADC11/PCINT19) 86
PK4 (ADC12/PCINT20) 85
PK5 (ADC13/PCINT21) 84
PK6 (ADC14/PCINT22) 83
PK7 (ADC15/PCINT23) 82
GND 81
VCC 80
PJ7 79
PA0 (AD0) 78
PA1 (AD1) 77
PA2 (AD2) 76
(OC0B) PG5 1
(RXD0/PCINT8) PE0 2
(TXD0) PE1 3
(XCK0/AIN0) PE2 4
(OC3A/AIN1) PE3 5
(OC3B/INT4) PE4 6
(OC3C/INT5) PE5 7
(T3/INT6) PE6 8
(CLKO/ICP3/INT7) PE7 9
VCC 10
GND 11
(RXD2) PH0 12
(TXD2) PH1 13
(XCK2) PH2 14
(OC4A) PH3 15
(OC4B) PH4 16
(OC4C) PH5 17
(OC2B) PH6 18
(SS/PCINT0) PB0 19
(SCK/PCINT1) PB1 20
(MOSI/PCINT2) PB2 21
(MISO/PCINT3) PB3 22
(OC2A/PCINT4) PB4 23
(OC1A/PCINT5) PB5 24
(OC1B/PCINT6) PB6 25
INDEX CORNER
AVR®
75 PA3 (AD3)
74 PA4 (AD4)
73 PA5 (AD5)
72 PA6 (AD6)
71 PA7 (AD7)
70 PG2 (ALE)
69 PJ6 (PCINT15)
68 PJ5 (PCINT14)
67 PJ4 (PCINT13)
66 PJ3 (PCINT12)
65 PJ2 (XCK3/PCINT11)
64 PJ1 (TXD3/PCINT10)
63 PJ0 (RXD3/PCINT9)
62 GND
61 VCC
60 PC7 (A15)
59 PC6 (A14)
58 PC5 (A13)
57 PC4 (A12)
56 PC3 (A11)
55 PC2 (A10)
54 PC1 (A9)
53 PC0 (A8)
52 PG1 (RD)
51 PG0 (WR)
26 (OC0A/OC1C/PCINT7) PB7
27 (T4) PH7
28 (TOSC2) PG3
29 (TOSC1) PG4
30 RESET
31 VCC
32 GND
33 XTAL2
34 XTAL1
35 (ICP4) PL0
36 (ICP5) PL1
37 (T5) PL2
38 (OC5A) PL3
39 (OC5B) PL4
40 (OC5C) PL5
41 PL6
42 PL7
43 (SCL/INT0) PD0
44 (SDA/INT1) PD1
45 (RXD1/INT2) PD2
46 (TXD1/INT3) PD3
47 (ICP1) PD4
48 (XCK1) PD5
49 (T1) PD6
50 (T0) PD7

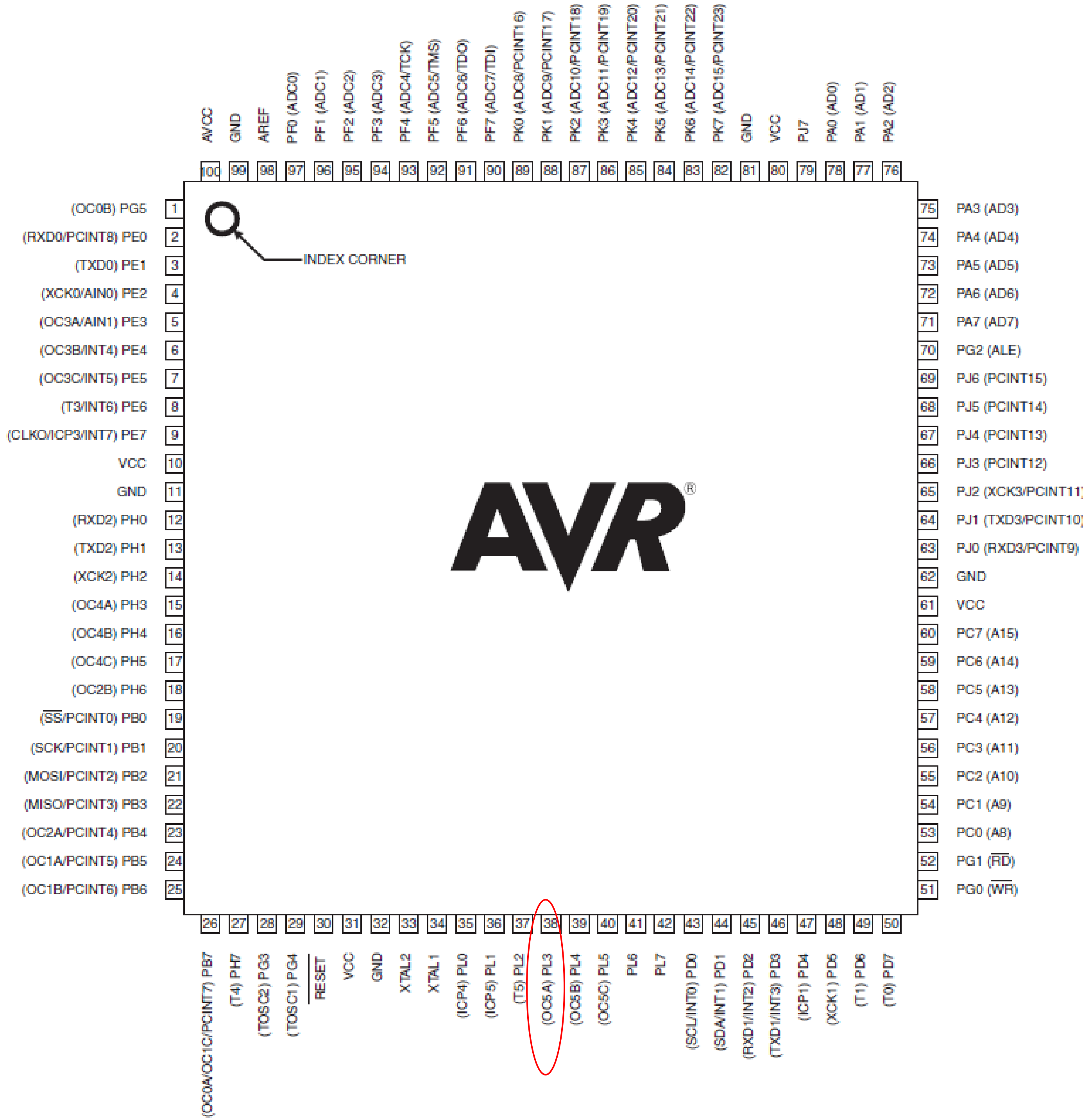
Example (solution)

- Use Timer5
  - Set OC5A as output
  - Set the Timer5 operation mode as Phase Correct PWM mode
  - Set the timer clock

16-bit Timer Block Diagram*

16-bit Timer/Counter Block Diagram[1]




AVCC
GND
AREF
PF0 (ADC0)
PF1 (ADC1)
PF2 (ADC2)
PF3 (ADC3)
PF4 (ADC4/TCK)
PF5 (ADC5/TMS)
PF6 (ADC6/TDO)
PF7 (ADC7/TDI)
PK0 (ADC8/PCINT16)
PK1 (ADC9/PCINT17)
PK2 (ADC10/PCINT18)
PK3 (ADC11/PCINT19)
PK4 (ADC12/PCINT20)
PK5 (ADC13/PCINT21)
PK6 (ADC14/PCINT22)
PK7 (ADC15/PCINT23)
GND
VCC
PJ7
PA0 (AD0)
PA1 (AD1)
PA2 (AD2)
INDEX CORNER
(OC0B) PG5
(RXD0/PCINT8) PE0
(TXD0) PE1
(XCK0/AIN0) PE2
(OC3A/AIN1) PE3
(OC3B/INT4) PE4
(OC3C/INT5) PE5
(T3/INT6) PE6
(CLKO/ICP3/INT7) PE7
VCC
GND
(RXD2) PH0
(TXD2) PH1
(XCK2) PH2
(OC4A) PH3
(OC4B) PH4
(OC4C) PH5
(OC2B) PH6
(SS/PCINT0) PB0
(SCK/PCINT1) PB1
(MOSI/PCINT2) PB2
(MISO/PCINT3) PB3
(OC2A/PCINT4) PB4
(OC1A/PCINT5) PB5
(OC1B/PCINT6) PB6
PA3 (AD3)
PA4 (AD4)
PA5 (AD5)
PA6 (AD6)
PA7 (AD7)
PG2 (ALE)
PJ6 (PCINT15)
PJ5 (PCINT14)
PJ4 (PCINT13)
PJ3 (PCINT12)
PJ2 (XCK3/PCINT11)
PJ1 (TXD3/PCINT10)
PJ0 (RXD3/PCINT9)
GND
VCC
PC7 (A15)
PC6 (A14)
PC5 (A13)
PC4 (A12)
PC3 (A11)
PC2 (A10)
PC1 (A9)
PC0 (A8)
PG1 (RD)
PG0 (WR)
(OC0A/OC1C/PCINT7) PB7
(T4) PH7
(TOSC2) PG3
(TOSC1) PG4
RESET
VCC
GND
XTAL2
XTAL1
(ICP4) PL0
(ICP5) PL1
(T5) PL2
(OC5A) PL3
(OC5B) PL4
(OC5C) PL5
PL6
PL7
(SCL/INT0) PD0
(SDA/INT1) PD1
(RXD1/INT2) PD2
(TXD1/INT3) PD3
(ICP1) PD4
(XCK1) PD5
(T1) PD6
(T0) PD7

Example Code

```
.include "m2560def.inc"
.def temp=r16

        ldi temp, 0b00001000
        sts DDRL, temp              ; Bit 3 will function as OC5A.

        clr temp                    ; the value controls the PWM duty cycle
        sts OCR5AH, temp
        ldi temp, 0x4A
        sts OCR5AL, temp
                                    ; Set Timer5 to Phase Correct PWM mode.
        ldi temp, (1 << CS50)       ; Set Timer clock frequency
        sts TCCR5B, temp.
        ldi temp, (1<< WGM50)|(1<<COM5A1)
        sts TCCR5A, temp
end:    rjmp end
```

Exercise

- The motor on the lab board is a DC motor that is driven by an input signal. The higher the input voltage, the faster the motor spins. How to use the PWM signal generated from the code shown in the previous slide to drive the motor?