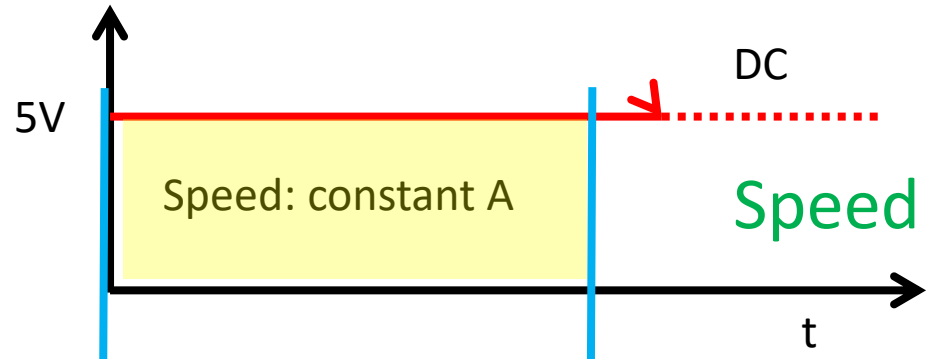
CET 241: Day 15
Motors and PWM

Dr. Noori Kim

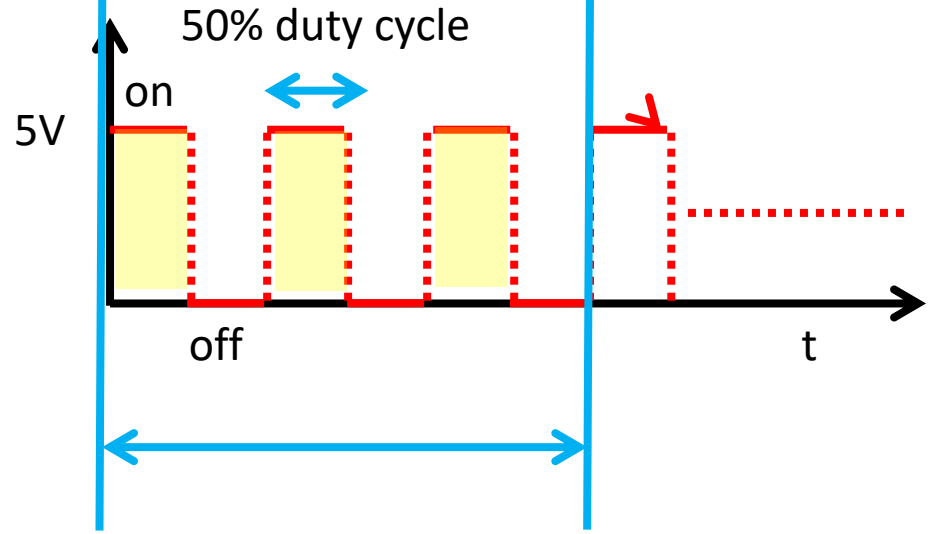
(A)



Speed ∝ supplied voltage

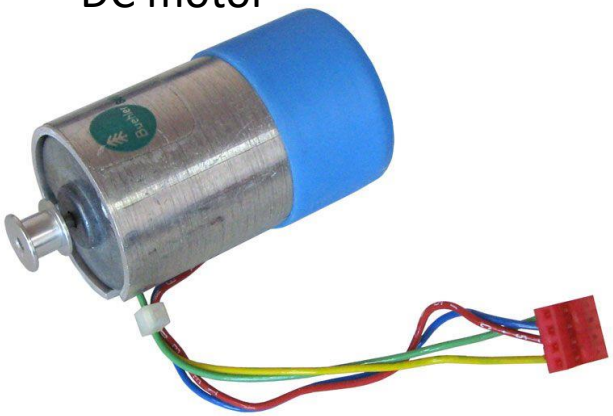
➢ I want to reduce the speed in a half

(B)

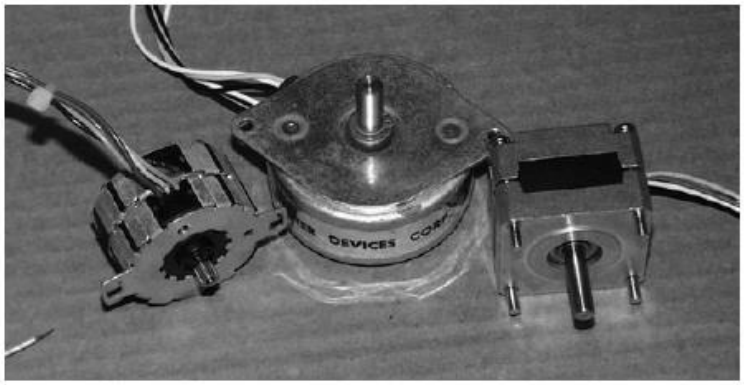


Average voltage supplied to the motor given is half of A

DC motor

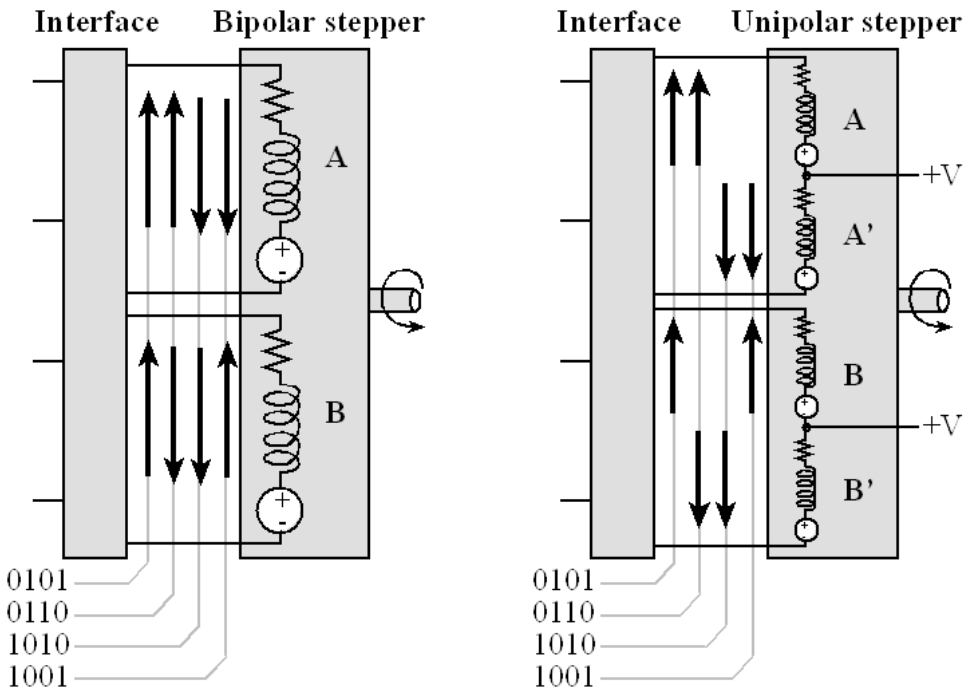


Brushless DC motor



Recap: Stepper motors

- Two coils, labeled **A** and **B**



By GPIO?? By PWM??

Choosing PWM Frequency

- Application dependant.
- Not too low:
  - Audible frequencies
  - At least twice the inverse of device time constant (NST)
- Not too high:
  - Transistors generate more heat at higher frequencies
  - Some loads will not respond at higher frequencies

PWM Modules in the TM4C123GH6PM MCU System

Datasheet 1230

- The TM4C123GH6PM MCU contains **two PWM modules**.
  - PWM0: 0x4002.8000
  - PWM1: 0x4002.9000
- Each module has **four** PWM generator blocks and a control block and each generator block can create **two** PWM output signals ➔total of **16** PWM output signals

- The output signals, pwmA' and pwmB', (of the PWM generation blocks) are managed by the output control block before being passed to the device pins as
  - MnPWM0 and MnPWM1 or
  - MnPWM2 and MnPWM3, and so on.

IO	Ain	0	1	2	3	4	5	6	7	8	9	14
PA2		Port		SSI0Clk								
PA3		Port		SSI0Fss								
PA4		Port		SSI0Rx								
PA5		Port		SSI0Tx								
PA6		Port			I_2C1SCL		M1PWM2					
PA7		Port			I_2C1SDA		M1PWM3					
PB0		Port	U1Rx						T2CCP0			
PB1		Port	U1Tx						T2CCP1			
PB2		Port			I_2C0SCL				T3CCP0			
PB3		Port			I_2C0SDA				T3CCP1			
PB4	Ain10	Port		SSI2Clk		M0PWM2			T1CCP0	CAN0Rx		
PB5	Ain11	Port		SSI2Fss		M0PWM3			T1CCP1	CAN0Tx		
PB6		Port		SSI2Rx		M0PWM0			T0CCP0			
PB7		Port		SSI2Tx		M0PWM1			T0CCP1			
PC4	C1-	Port	U4Rx	U1Rx		M0PWM6		IDX1	WT0CCP0	U1RTS		
PC5	C1+	Port	U4Tx	U1Tx		M0PWM7		PhA1	WT0CCP1	U1CTS		
PC6	C0+	Port	U3Rx					PhB1	WT1CCP0	USB0epen		
PC7	C0-	Port	U3Tx						WT1CCP1	USB0pflt		

- The two PWM signals that share the same timer and frequency and can either be programmed with independent actions or as a single pair of complementary signals

Figure 20-1. PWM Module Diagram

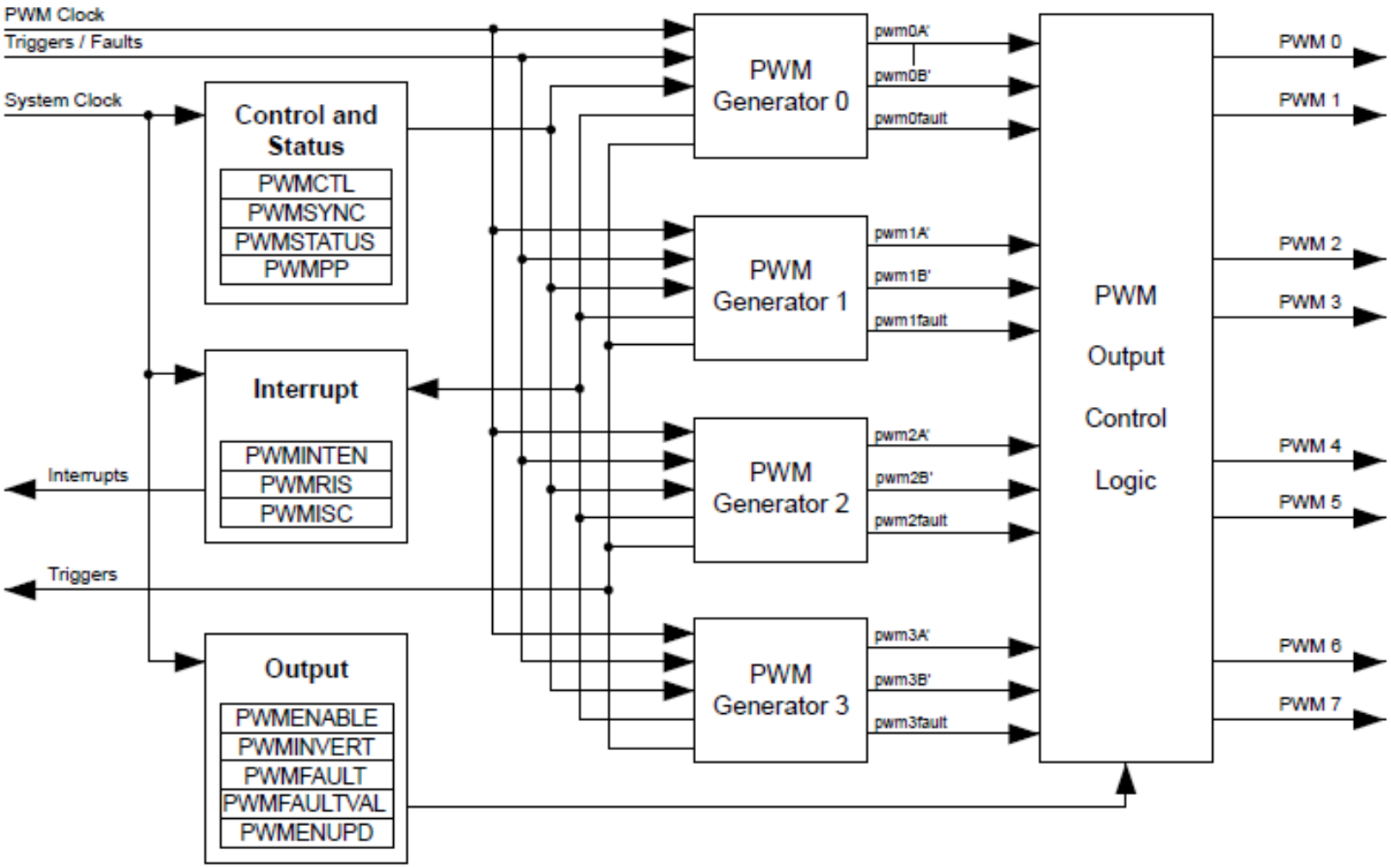
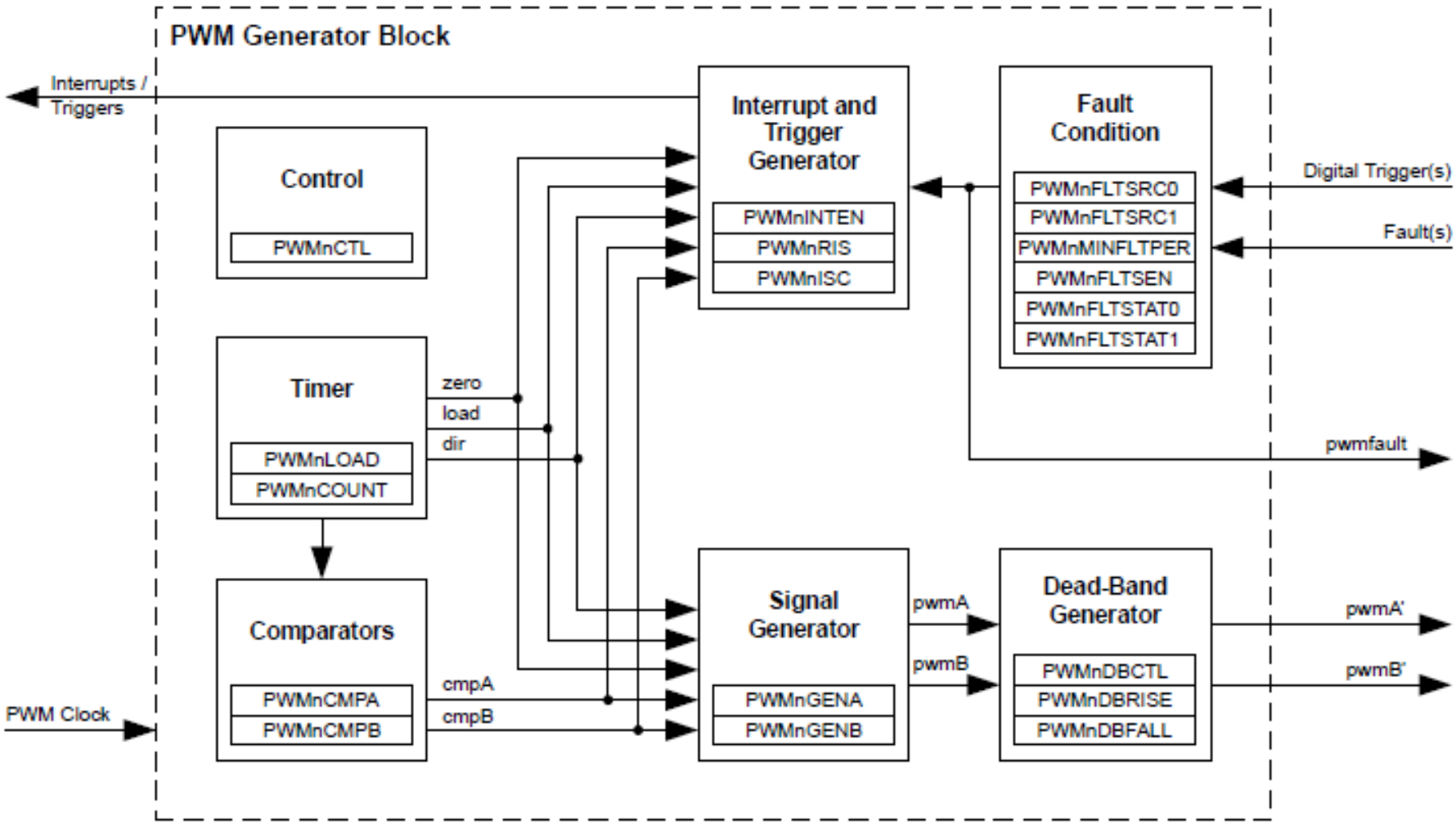


Figure 20-2. PWM Generator Block Diagram



Datasheet 1239

Initialization and Configuration
The following example shows how to initialize PWM Generator 0 with a 25-kHz frequency, a 25% duty cycle on the MnPWM0 pin, and a 75% duty cycle on the MnPWM1 pin. This example assumes the system clock is 20 MHz.

1. Enable the PWM clock by writing a value of 0x0010.0000 to the **RCGC0** register in the System Control module (see page 456, or RCGCPWM can)

Register 134: Run Mode Clock Gating Control Register 0 (RCGC0), offset 0x100

This register controls the clock gating logic in normal Run mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes. Note that there must be a delay of 3 system clocks after a module clock is enabled before any registers in that module are accessed.

Important: This register is provided for legacy software support only.

The peripheral-specific Run Mode Clock Gating Control registers (such as **RCGCWD**) should be used to reset specific peripherals. A write to this legacy register also writes the corresponding bit in the peripheral-specific register. Any bits that are changed by writing to this register can be read back correctly with a read of this register. Software

Run Mode Clock Gating Control Register 0 (RCGC0)

Base 0x400F.E000
Offset 0x100
Type RO, reset 0x0000.0040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			WDT1	reserved		CAN1	CAN0	reserved			PWM0	reserved		ADC1	ADC0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				MAXADC1SPD		MAXADC0SPD		reserved	HIB	reserved		WDT0	reserved		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

20	PWM0	RO	0x0	PWM Clock Gating Control This bit controls the clock gating for the PWM module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

2. Enable the clock to the appropriate GPIO module via the **RCGC2** register in the System Control module (see page 464 or RCGCGPIO can).

3. In the GPIO module, enable the appropriate pins for their alternate function using the **GPIOAFSEL** register. To determine which GPIOs to configure, see Table 23-4 on page 1344.
4. Configure the PMCn fields in the **GPIOPCTL** register to assign the PWM signals to the appropriate pins (see page 688 and Table 23-5 on page 1351).

IO	Ain	0	1	2	3	4	5	6	7	8	9	14
PA2		Port		SSI0Clk								
PA3		Port		SSI0Fss								
PA4		Port		SSI0Rx								
PA5		Port		SSI0Tx								
PA6		Port			I_2C1SCL		M1PWM2					
PA7		Port			I_2C1SDA		M1PWM3					
PB0		Port	U1Rx						T2CCP0			
PB1		Port	U1Tx						T2CCP1			
PB2		Port			I_2C0SCL				T3CCP0			
PB3		Port			I_2C0SDA				T3CCP1			
PB4	Ain10	Port		SSI2Clk		M0PWM2			T1CCP0	CAN0Rx		
PB5	Ain11	Port		SSI2Fss		M0PWM3			T1CCP1	CAN0Tx		
PB6		Port		SSI2Rx		M0PWM0			T0CCP0			
PB7		Port		SSI2Tx		M0PWM1			T0CCP1			
PC4	C1-	Port	U4Rx	U1Rx		M0PWM6		IDX1	WT0CCP0	U1RTS		
PC5	C1+	Port	U4Tx	U1Tx		M0PWM7		PhA1	WT0CCP1	U1CTS		
PC6	C0+	Port	U3Rx					PhB1	WT1CCP0	USB0epen		
PC7	C0-	Port	U3Tx						WT1CCP1	USB0pflt		
PD0	Ain7	Port	SSI3Clk	SSI1Clk	I_2C3SCL	M0PWM6	M1PWM0		WT2CCP0			
PD1	Ain6	Port	SSI3Fss	SSI1Fss	I_2C3SDA	M0PWM7	M1PWM1		WT2CCP1			
PD2	Ain5	Port	SSI3Rx	SSI1Rx		M0Fault0			WT3CCP0	USB0epen		
PD3	Ain4	Port	SSI3Tx	SSI1Tx				IDX0	WT3CCP1	USB0pflt		
PD6		Port	U2Rx			M0Fault0		PhA0	WT5CCP0			
PD7		Port	U2Tx					PhB0	WT5CCP1	NMI		
PE0	Ain3	Port	U7Rx									
PE1	Ain2	Port	U7Tx									
PE2	Ain1	Port										
PE3	Ain0	Port										
PE4	Ain9	Port	U5Rx		I_2C2SCL	M0PWM4	M1PWM2			CAN0Rx		
PE5	Ain8	Port	U5Tx		I_2C2SDA	M0PWM5	M1PWM3			CAN0Tx		
PF0		Port	U1RTS	SSI1Rx	CAN0Rx		M1PWM4	PhA0	T0CCP0	NMI	C0o	
PF1		Port	U1CTS	SSI1Tx			M1PWM5	PhB0	T0CCP1		C1o	TRD1
PF2		Port		SSI1Clk		M0Fault0	M1PWM6		T1CCP0			TRD0
PF3		Port		SSI1Fss	CAN0Tx		M1PWM7		T1CCP1			TRCLK
PF4		Port					M1Fault0	IDX0	T2CCP0	USB0epen		

5. Configure the **Run-Mode Clock Configuration (RCC)** register in the System Control module to use the PWM divide (USEPWMDIV) and set the divider (PWMDIV) to divide by 2 (000).

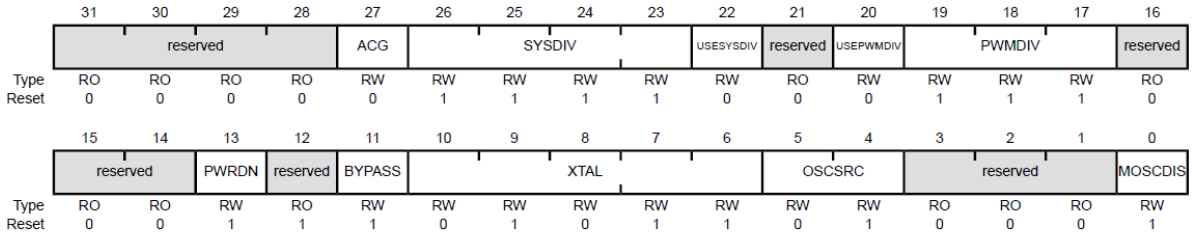
Register 8: Run-Mode Clock Configuration (RCC), offset 0x060

The bits in this register configure the system clock and oscillators.

Important: Write the **RCC** register prior to writing the **RCC2** register.

Run-Mode Clock Configuration (RCC)

Base 0x400F.E000
Offset 0x060
Type RW, reset 0x078E.3AD1



20	USEPWMDIV	RW	0	Enable PWM Clock Divisor

Value	Description
0	The system clock is the source for the PWM clock.
1	The PWM clock divider is the source for the PWM clock.

Note that when the PWM divisor is used, it is applied to the clock for both PWM modules.

19:17	PWMDIV	RW	0x7	PWM Unit Clock Divisor

This field specifies the binary divisor used to predivide the system clock down for use as the timing reference for the PWM module. The rising edge of this clock is synchronous with the system clock.

Value	Divisor
0x0	/2
0x1	/4
0x2	/8
0x3	/16
0x4	/32
0x5	/64
0x6	/64
0x7	/64 (default)

6. Configure the PWM generator for countdown mode with immediate updates:

   a. Write the **PWM0CTL** register with a value of **0x0000.0000**
      - **bit 1: count down (0)**
      - **bit 0: disable PWM while init (0)**
   b. Write the **PWM0GENA** register with a value of **0x0000.008C**
   c. Write the **PWM0GENB** register with a value of **0x0000.080C**.

Register 12: PWM0 Control (PWM0CTL), offset 0x040

Register 13: PWM1 Control (PWM1CTL), offset 0x080

Register 14: PWM2 Control (PWM2CTL), offset 0x0C0

Register 15: PWM3 Control (PWM3CTL), offset 0x100

These registers configure the PWM signal generation blocks (PWM0CTL controls the PWM generator 0 block, and so on). The Register Update mode, Debug mode, Counting mode, and Block Enable mode are all controlled via these registers. The blocks produce the PWM signals, which can be either two independent PWM signals (from the same counter), or a paired set of PWM signals with dead-band delays added.

The PWM0 block produces the MnPWM0 and MnPWM1 outputs, the PWM1 block produces the MnPWM2 and MnPWM3 outputs, the PWM2 block produces the MnPWM4 and MnPWM5 outputs, and the PWM3 block produces the MnPWM6 and MnPWM7 outputs.

PWMn Control (PWMnCTL)

PWM0 base: 0x4002.8000
PWM1 base: 0x4002.9000
Offset 0x040
Type RW, reset 0x0000.0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved													LATCH	MINFLTPER	FLTSRC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBFALLUPD		DBRISEUPD		DBCTLUPD		GENBUPD		GENAUPD		CMPBUPD	CMPAUPD	LOADUPD	DEBUG	MODE	ENABLE
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1	MODE	RW	0	Counter Mode

Value	Description
0	The counter counts down from the load value to 0 and then wraps back to the load value (Count-Down mode).
1	The counter counts up from 0 to the load value, back down to 0, and then repeats (Count-Up/Down mode).

0	ENABLE	RW	0	PWM Block Enable

Note: Disabling the PWM by clearing the ENABLE bit does not clear the COUNT field of the **PWMnCOUNT** register. Before re-enabling the PWM (ENABLE = 0x1), the COUNT field should be cleared by resetting the PWM registers through the **SRPWM** register in the System Control Module.

Value	Description
0	The entire PWM generation block is disabled and not clocked.
1	The PWM generation block is enabled and produces PWM signals.

PWM0_0_GENA_R = 0x8C

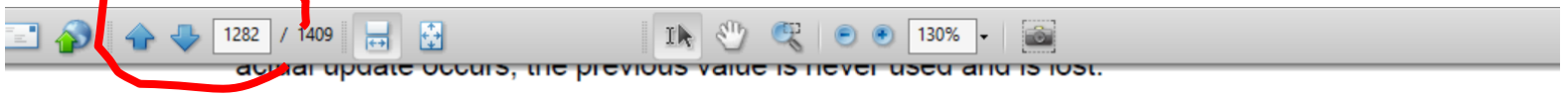
Register 44: PWM0 Generator A Control (PWM0GENA), offset 0x060

Register 45: PWM1 Generator A Control (PWM1GENA), offset 0x0A0

Register 46: PWM2 Generator A Control (PWM2GENA), offset 0x0E0

Register 47: PWM3 Generator A Control (PWM3GENA), offset 0x120

These registers control the generation of the pwmA signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators (**PWM0GENA** controls the PWM generator 0 block, and so on). When the counter is running in Count-Down mode, only four of these events occur; when running in Count-Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the resulting PWM signal.



actual update occurs, the previous value is never used and is lost.

PWMn Generator A Control (PWMnGENA)

PWM0 base: 0x4002.8000
PWM1 base: 0x4002.9000
Offset 0x060
Type RW, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
Type	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
7:6	ACTCMPAD	RW	0x0	Action for Comparator A Down This field specifies the action to be taken when the counter matches comparator A while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
5:4	ACTCMPAU	RW	0x0	Action for Comparator A Up This field specifies the action to be taken when the counter matches comparator A while counting up. This action can only occur when the MODE bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
3:2	ACTLOAD	RW	0x0	Action for Counter=LOAD This field specifies the action to be taken when the counter matches the value in the **PWMnLOAD** register. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
1:0	ACTZERO	RW	0x0	Action for Counter=0 This field specifies the action to be taken when the counter is zero. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.

- **PWM is set to count down mode**
- **When CMP matches counter, drive pwm low**
- **When where is load on the counter, drive pwm high**

PWM0_0_GENB_R = 0x80C

Register 48: PWM0 Generator B Control (PWM0GENB), offset 0x064

Register 49: PWM1 Generator B Control (PWM1GENB), offset 0x0A4

Register 50: PWM2 Generator B Control (PWM2GENB), offset 0x0E4

Register 51: PWM3 Generator B Control (PWM3GENB), offset 0x124

These registers control the generation of the pwmB signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators (**PWM0GENB** controls the PWM generator 0 block, and so on). When the counter is running in Count-Down mode, only four of these events occur; when running in Count-Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the resulting PWM signal.

PWMn Generator B Control (PWMnGENB), offset 0x064

PWM0 base: 0x4002.8000
PWM1 base: 0x4002.9000
Offset 0x064
Type RW, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
Type	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0000: do nothing

Bit/Field	Name	Type	Reset	Description
11:10	ACTCMPBD	RW	0x0	Action for Comparator B Down This field specifies the action to be taken when the counter matches comparator B while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
9:8	ACTCMPBU	RW	0x0	Action for Comparator B Up This field specifies the action to be taken when the counter matches comparator B while counting up. This action can only occur when the MODE bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
3:2	ACTLOAD	RW	0x0	Action for Counter=LOAD This field specifies the action to be taken when the counter matches the load value. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
1:0	ACTZERO	RW	0x0	Action for Counter=0 This field specifies the action to be taken when the counter is 0. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.

- **PWM is set to count down mode**
- **When CMP matches counter, drive pwm low**

- **When there is load on the counter, drive pwm high**

7. Set the period for the counter. For a **5KHz** input frequency, the period = 1/5000, or **200 µs**.
   - The PWM clock source is **20 MHz**.
   - Thus there are 20 MHz / 5 KHz = **4000** clock ticks per period.
   - Use this value to set the **PWM0LOAD** register (**PWM0LOAD=** 4000 – 1 = **3999=**0x0000.018F).

Load➔4000

For compA

- **When where is load on the counter, drive pwmA high**
- **When CMP matches counter, drive pwmA low**

For compB

- **When where is load on the counter, drive pwmB high**
- **When CMP matches counter, drive pwmB low**

8. Set pulse width of a **25%** duty cycle (PWM0). Write the **PWM0CMPA** register with a value of **3000**.
9. Set pulse width of a **75%** duty cycle (PWM1). Write the **PWM1CMPB** register with a value of **1000**.

Register 36: PWM0 Compare A (PWM0CMPA), offset 0x058

These registers contain a value to be compared against the counter (**PWM0CMPA** controls the PWM generator 0 block, and so on). When this value matches the counter, a pulse is output which can be configured to drive the generation of the pwmA and pwmB signals (via the **PWMnGENA** and **PWMnGENB** registers) or drive an interrupt or ADC trigger (via the **PWMnINTEN** register). If the value of this register is greater than the **PWMnLOAD** register (see page 1278), then no pulse is ever output.

If the comparator A update mode is locally synchronized (based on the CMPAUPD bit in the **PWMnCTL** register), the 16-bit COMPA value is used the next time the counter reaches zero. If the update mode is globally synchronized, it is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 1244). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

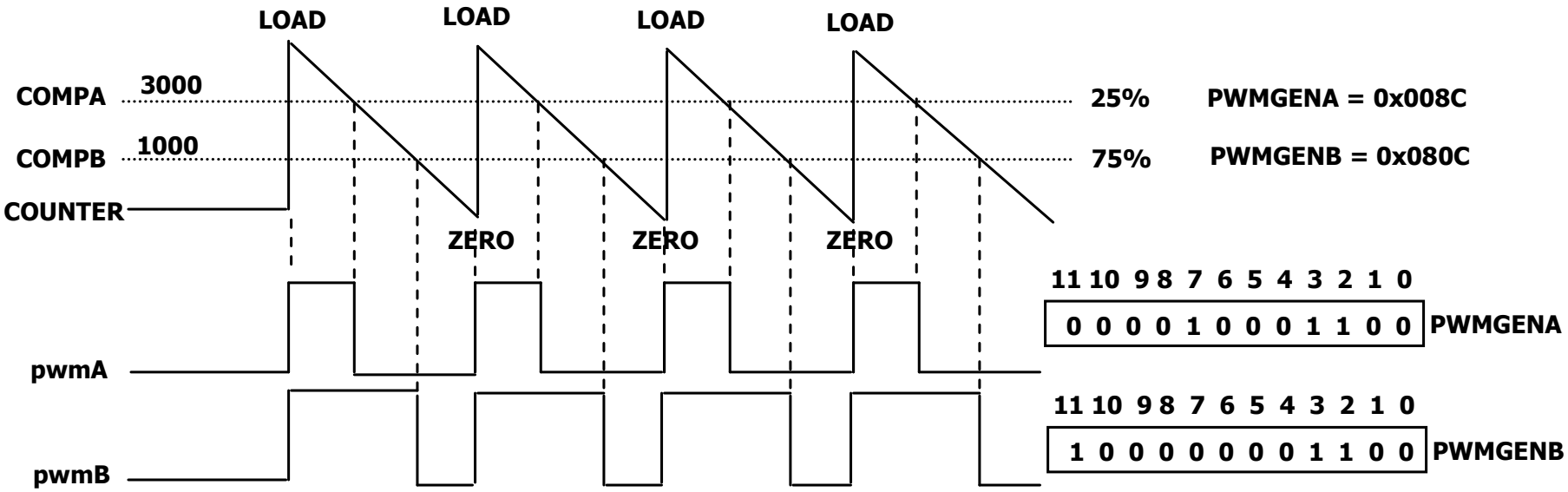
PWMn Compare A (PWMnCMPA)

PWM0 base: 0x4002.8000
PWM1 base: 0x4002.9000
Offset 0x058
Type RW, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COMPA															
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **When there is load on the counter, drive pwm high**



The initialization and configuration process for PWM Module0.

For steps **8, 9**, a **25%** duty cycle is calculated as **4000 * 0.75** = **3000**. A **75%** duty cycle is **4000 * 0.25** = **1000**. Refer to Figure 7.46, the higher the duty cycle value, the lower the setup value installed in the **PWM1CMPA** or **PWM1CMPB** registers. (**This is opposite to the common sense, due to the setting that we have)**

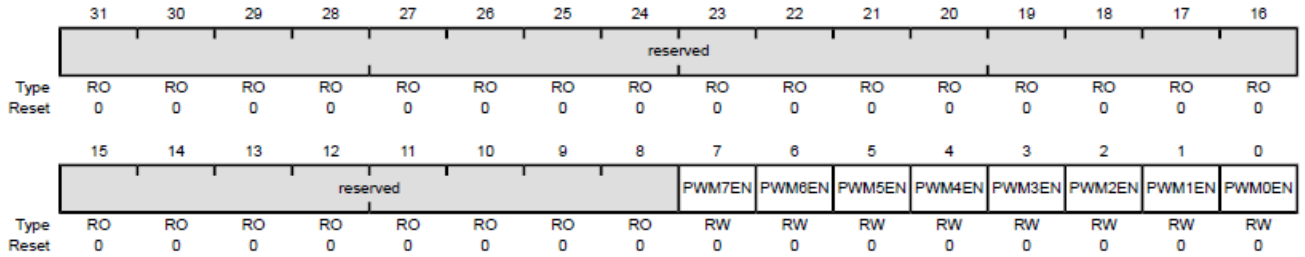
10. Start the counters in generator 0. Write the **PWM0CTL** register with a value of **0x0000.0001**. (refer to step 6)
11. Enable PWM outputs. Write the **PWMENABLE** register with **0x0000.0003**.

Register 3: PWM Output Enable (PWMENABLE), offset 0x008

This register provides a master control of which generated pwmA' and pwmB' signals are output to the MnPWMn pins. By disabling a PWM output, the generation process can continue (for example, when the time bases are synchronized) without driving PWM signals to the pins. When bits in this register are set, the corresponding pwmA' or pwmB' signal is passed through to the output stage. When bits are clear, the pwmA' or pwmB' signal is replaced by a zero value which is also passed to the output stage. The **PWMINVERT** register controls the output stage, so if the corresponding bit is set in that register, the value seen on the MnPWMn signal is inverted from what is configured by the bits in this register. Updates to the bits in this register can be immediate or locally or globally synchronized to the next synchronous update as controlled by the ENUPDn fields in the **PWMENUPD** register.

PWM Output Enable (PWMENABLE)

PWM0 base: 0x4002.8000
PWM1 base: 0x4002.9000
Offset 0x008
Type RW, reset 0x0000.0000



1	PWM1EN	RW	0	MnPWM1 Output Enable

Value	Description
0	The MnPWM1 signal has a zero value.
1	The generated pwm0B' signal is passed to the MnPWM1 pin.

0	PWM0EN	RW	0	MnPWM0 Output Enable

Value	Description
0	The MnPWM0 signal has a zero value.
1	The generated pwm0A' signal is passed to the MnPWM0 pin.

Program 6.8 in the book: Common sense example

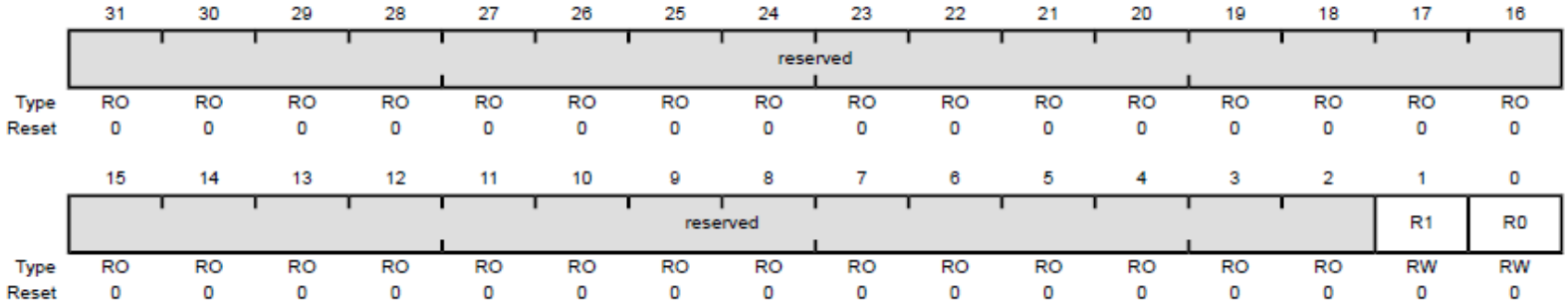
DC Motor PWM

```
void PWM0A_Init(uint16_t period, uint16_t duty){
  SYSCTL_RCGCPWM_R |= 0x01;          // 1) activate PWM0
  SYSCTL_RCGCGPIO_R |= 0x02;         // 2) activate port B
  while((SYSCTL_PRGPIO_R&0x02) == 0){};
  GPIO_PORTB_AFSEL_R |= 0x40;        // 3) alt funct on PB6
  GPIO_PORTB_PCTL_R =
   (GPIO_PORTB_PCTL_R&0xF0FFFFFF)+0x04000000; // 4) PWM0
  GPIO_PORTB_AMSEL_R &= ~0x40;       // disable analog on PB6
  GPIO_PORTB_DEN_R |= 0x40;          // enable digital I/O on PB6
  SYSCTL_RCC_R = 0x00100000 |        // 5) use PWM divider
      (SYSCTL_RCC_R & (~0x000E0000));   //  /2 divider
  PWM0_0_CTL_R = 0;          // 6-a) re-loading down-counting mode
  PWM0_0_GENA_R = 0xC8;//6-b) low on LOAD, high on CMPA down
  PWM0_0_LOAD_R = period - 1;    // 7) cycles count down to 0
  PWM0_0_CMPA_R = duty - 1;      // 8) count value output rises
  PWM0_0_CTL_R |= 0x00000001;    // 10) start PWM0
  PWM0_ENABLE_R |= 0x00000001;   // 11) enable PB6/M0PWM0
}
```

```
SYSCTL_RCGCPWM_R |= 0x01;        // 1) activate PWM0
```

Pulse Width Modulator Run Mode Clock Gating Control (RCGCPWM)

Base 0x400F.E000
Offset 0x640
Type RW, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	R1	RW	0	PWM Module 1 Run Mode Clock Gating Control Value Description 0 PWM module 1 is disabled. 1 Enable and provide a clock to PWM module 1 in Run mode.
0	R0	RW	0	PWM Module 0 Run Mode Clock Gating Control Value Description 0 PWM module 0 is disabled. 1 Enable and provide a clock to PWM module 0 in Run mode.

4) GPIO_PORTB_PCTL_R = (GPIO_PORTB_PCTL_R&0xF0FFFFFF)+0x04000000

IO	Ain	0	1	2	3	4	5	6	7	8	9	14
PA2		Port		SSI0Clk								
PA3		Port		SSI0Fss								
PA4		Port		SSI0Rx								
PA5		Port		SSI0Tx								
PA6		Port			I_2C1SCL		M1PWM2					
PA7		Port			I_2C1SDA		M1PWM3					
PB0		Port	U1Rx						T2CCP0			
PB1		Port	U1Tx						T2CCP1			
PB2		Port			I_2C0SCL				T3CCP0			
PB3		Port			I_2C0SDA				T3CCP1			
PB4	Ain10	Port		SSI2Clk		M0PWM2			T1CCP0	CAN0Rx		
PB5	Ain11	Port		SSI2Fss		M0PWM3			T1CCP1	CAN0Tx		
PB6		Port		SSI2Rx		M0PWM0			T0CCP0			
PB7		Port		SSI2Tx		M0PWM1			T0CCP1			
PC4	C1-	Port	U4Rx	U1Rx		M0PWM6		IDX1	WT0CCP0	U1RTS		
PC5	C1+	Port	U4Tx	U1Tx		M0PWM7		PhA1	WT0CCP1	U1CTS		
PC6	C0+	Port	U3Rx					PhB1	WT1CCP0	USB0epen		
PC7	C0-	Port	U3Tx						WT1CCP1	USB0pflt		
PD0	Ain7	Port	SSI3Clk	SSI1Clk	I_2C3SCL	M0PWM6	M1PWM0		WT2CCP0			
PD1	Ain6	Port	SSI3Fss	SSI1Fss	I_2C3SDA	M0PWM7	M1PWM1		WT2CCP1			
PD2	Ain5	Port	SSI3Rx	SSI1Rx		M0Fault0			WT3CCP0	USB0epen		
PD3	Ain4	Port	SSI3Tx	SSI1Tx				IDX0	WT3CCP1	USB0pflt		
PD6		Port	U2Rx			M0Fault0		PhA0	WT5CCP0			
PD7		Port	U2Tx					PhB0	WT5CCP1	NMI		
PE0	Ain3	Port	U7Rx									
PE1	Ain2	Port	U7Tx									
PE2	Ain1	Port										
PE3	Ain0	Port										
PE4	Ain9	Port	U5Rx		I_2C2SCL	M0PWM4	M1PWM2			CAN0Rx		
PE5	Ain8	Port	U5Tx		I_2C2SDA	M0PWM5	M1PWM3			CAN0Tx		
PF0		Port	U1RTS	SSI1Rx	CAN0Rx		M1PWM4	PhA0	T0CCP0	NMI	C0o	
PF1		Port	U1CTS	SSI1Tx			M1PWM5	PhB0	T0CCP1		C1o	TRD1
PF2		Port		SSI1Clk		M0Fault0	M1PWM6		T1CCP0			TRD0
PF3		Port		SSI1Fss	CAN0Tx		M1PWM7		T1CCP1			TRCLK
PF4		Port					M1Fault0	IDX0	T2CCP0	USB0epen		

```
5) SYSCTL_RCC_R = 0x00100000 | (SYSCTL_RCC_R & (~0x000E0000));
```

Register 8: Run-Mode Clock Configuration (RCC), offset 0x060

The bits in this register configure the system clock and oscillators.

Important: Write the **RCC** register prior to writing the **RCC2** register.

Run-Mode Clock Configuration (RCC)

Base 0x400F.E000
Offset 0x060
Type RW, reset 0x078E.3AD1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved				ACG	SYSDIV				USESYSDIV	reserved	USEPWMDIV	PWMDIV			reserved
Type	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		PWRDN	reserved	BYPASS	XTAL					OSCSRC		reserved			MOSCDIS
Type	RO	RO	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO	RO	RW
Reset	0	0	1	1	1	0	1	0	1	1	0	1	0	0	0	1

20	USEPWMDIV	RW	0	Enable PWM Clock Divisor

Value	Description
0	The system clock is the source for the PWM clock.
1	The PWM clock divider is the source for the PWM clock.

Note that when the PWM divisor is used, it is applied to the clock for both PWM modules.

19:17	PWMDIV	RW	0x7	PWM Unit Clock Divisor

This field specifies the binary divisor used to predivide the system clock down for use as the timing reference for the PWM module. The rising edge of this clock is synchronous with the system clock.

Value	Divisor
0x0	/2
0x1	/4
0x2	/8
0x3	/16
0x4	/32
0x5	/64
0x6	/64
0x7	/64 (default)

6-b) PWM0_0_GENA_R = 0xC8

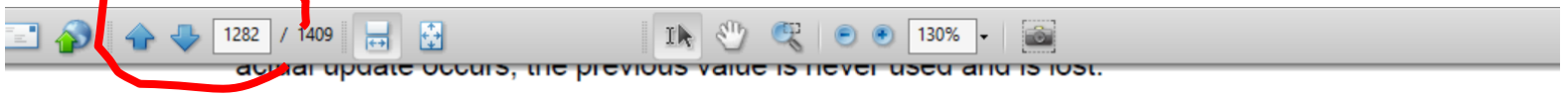
Register 44: PWM0 Generator A Control (PWM0GENA), offset 0x060

Register 45: PWM1 Generator A Control (PWM1GENA), offset 0x0A0

Register 46: PWM2 Generator A Control (PWM2GENA), offset 0x0E0

Register 47: PWM3 Generator A Control (PWM3GENA), offset 0x120

These registers control the generation of the pwmA signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators (**PWM0GENA** controls the PWM generator 0 block, and so on). When the counter is running in Count-Down mode, only four of these events occur; when running in Count-Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the resulting PWM signal.



actual update occurs, the previous value is never used and is lost.

PWMn Generator A Control (PWMnGENA)

PWM0 base: 0x4002.8000
PWM1 base: 0x4002.9000
Offset 0x060
Type RW, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
Type	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
7:6	ACTCMPAD	RW	0x0	Action for Comparator A Down This field specifies the action to be taken when the counter matches comparator A while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
5:4	ACTCMPAU	RW	0x0	Action for Comparator A Up This field specifies the action to be taken when the counter matches comparator A while counting up. This action can only occur when the MODE bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
3:2	ACTLOAD	RW	0x0	Action for Counter=LOAD This field specifies the action to be taken when the counter matches the value in the **PWMnLOAD** register. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
1:0	ACTZERO	RW	0x0	Action for Counter=0 This field specifies the action to be taken when the counter is zero. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.

- **PWM is set to count down mode**
- **When counter matches CMP, drive pwm high**

Common sense

- **When where is load on the counter, drive pwm low**

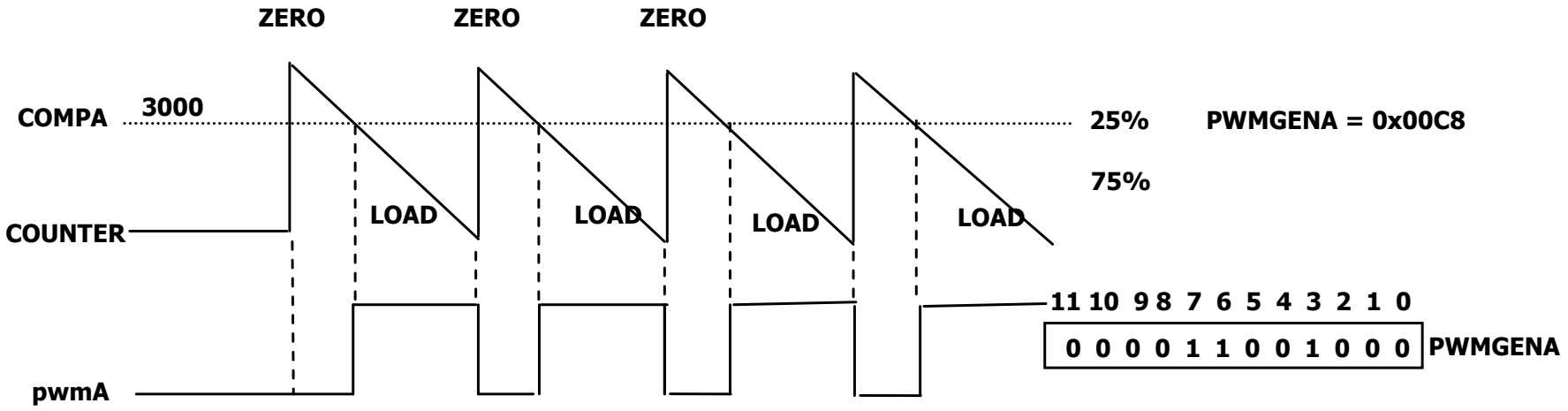
Period (1ms) = 40000

```
PWM0A_Init(40000, 30000);
// 1000 Hz, 75% duty
```

1. 30000 is 75% of 40000,
2. if system clock 80MHz, PWM clock is 40MHz
   40*1e6/(4e4)=1kHz

```
void PWM0A_Duty(uint16_t duty){
  PWM_0_CMPA_R = duty - 1;  // 6) count value output rises
}
```

Program 6.8 in the book: Common sense example



```
PWM0_0_GENA_R = 0xC8;   // low on LOAD, high on CMPA down
PWM0_0_LOAD_R = period - 1;   // 5) cycles count down to 0
PWM0_0_CMPA_R = duty - 1;
```

- CMPA and LOAD will count-up or down (based on your GENA configuration)
  - In this textbook example, when Counter ==CMPA, GENA generates high
  - when counter is loaded, GENA generates low
- The frequency and duty are computed by users considering its bus clock speed (period, duty)
  - Related registers are LOAD and COMP

DC Motor PWM

$$Dutycycle(\%) = \frac{\mathbf{High}}{\mathbf{High} + \mathbf{Low}} \times 100 = \frac{\mathbf{High}}{\mathbf{Period}} \times 100$$

```
PWM0A_Init(40000, 30000);          // 1000 Hz, 75% duty
```

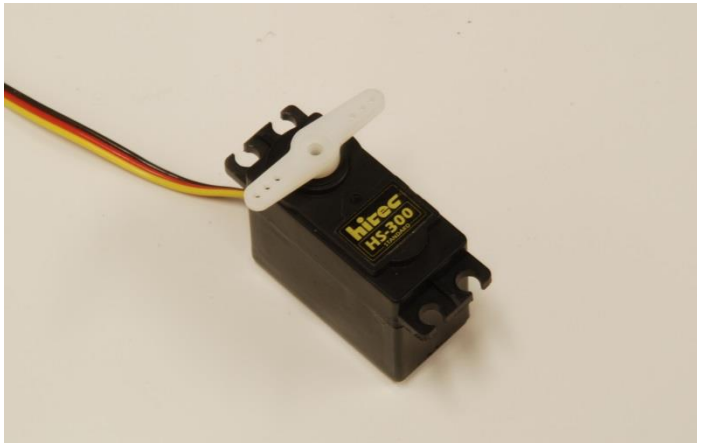
1. 30000 is 75% of 40000,
2. if system clock 80MHz, PWM clock is 40MHz 40*1e6/(4e4)=1kHz

Period (1ms) = 40000

```
void PWM0A_Duty(uint16_t duty){
  PWM_0_CMPA_R = duty - 1;  // 6) count value output rises
}
```

Servo Motor

- Simple digital interface (built in controller)
- Duty cycle controls angle



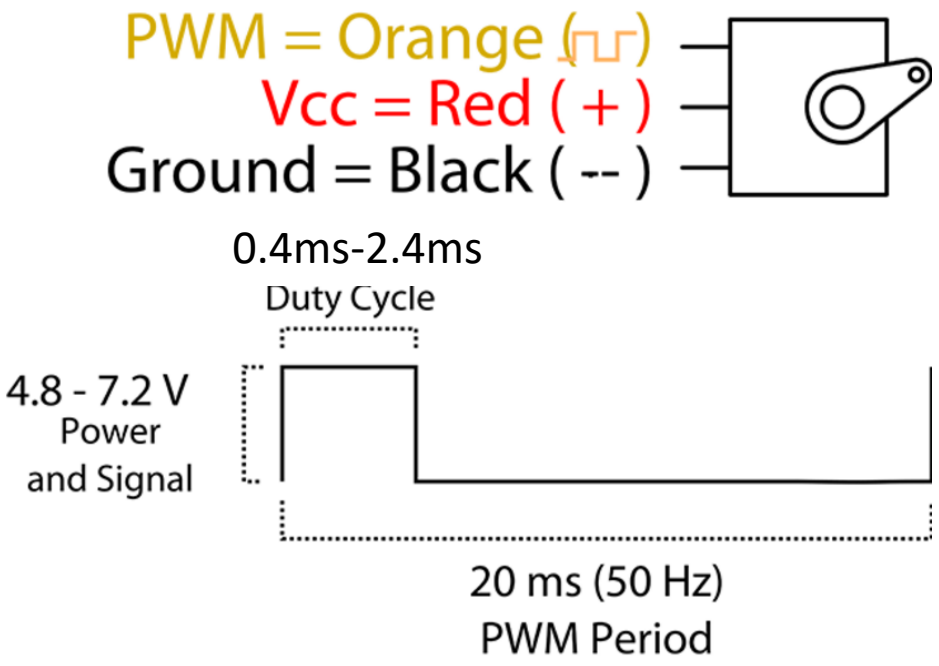
TowerPro MG90 - Micro Servo

Specifications

Modulation:	Analog
Torque:	**4.8V:** 30.60 oz-in (2.20 kg-cm) **6.0V:** 34.70 oz-in (2.50 kg-cm)
Speed:	**4.8V:** 0.11 sec/60° **6.0V:** 0.10 sec/60°
Weight:	0.49 oz (14.0 g)
Dimensions:	**Length:** 0.91 in (23.1 mm) **Width:** 0.48 in (12.2 mm) **Height:** 1.14 in (29.0 mm)
Motor Type:	(add)
Gear Type:	Metal
Rotation/Support:	Dual Bearings
Rotational Range:	180°
Pulse Cycle:	20 ms
Pulse Width:	400-2400 µs
Connector Type:	(add)

0.4ms-2.4ms

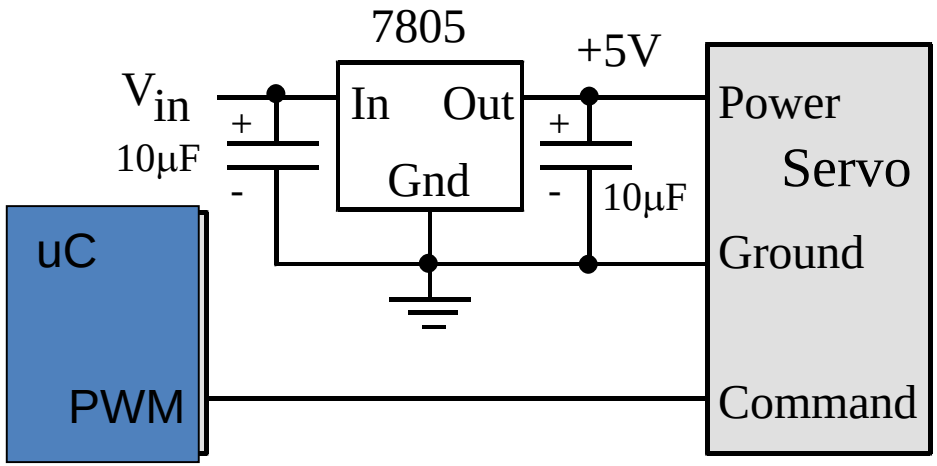
Brand:	Tower pro
Product Number:	(add)
Typical Price:	(add)
Compare:	add+



DC motor, controller (potentialmeter), gears

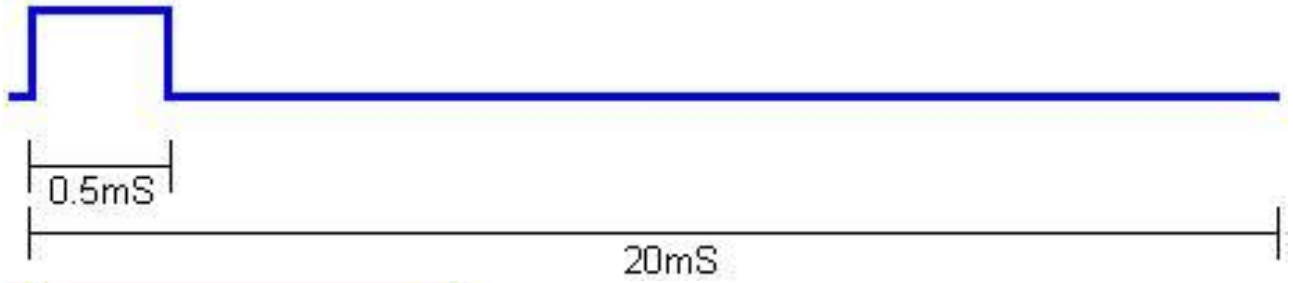
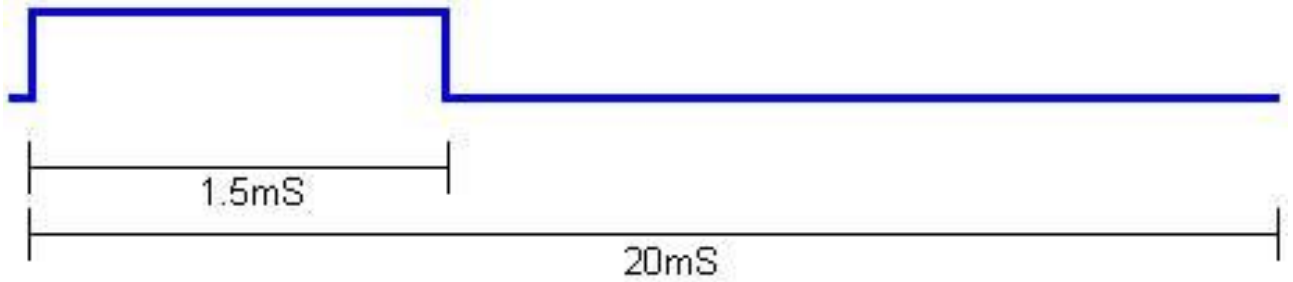
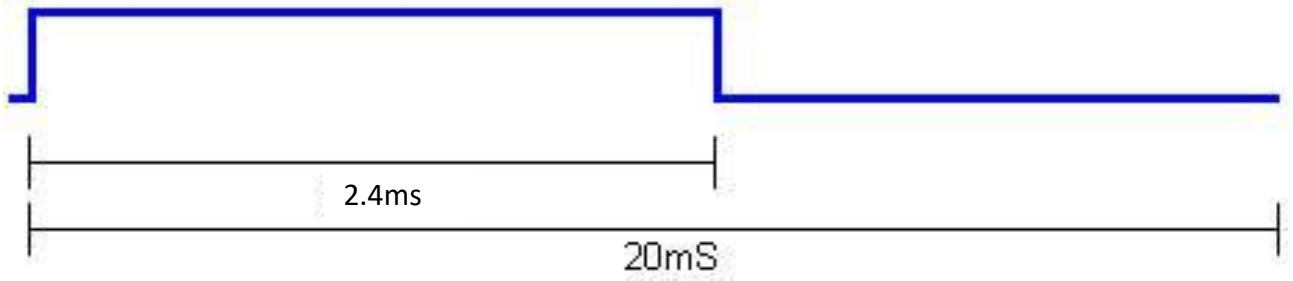
Interface of a Servo motor

- Needs its own +5V regulator
- Duty cycle controls angle



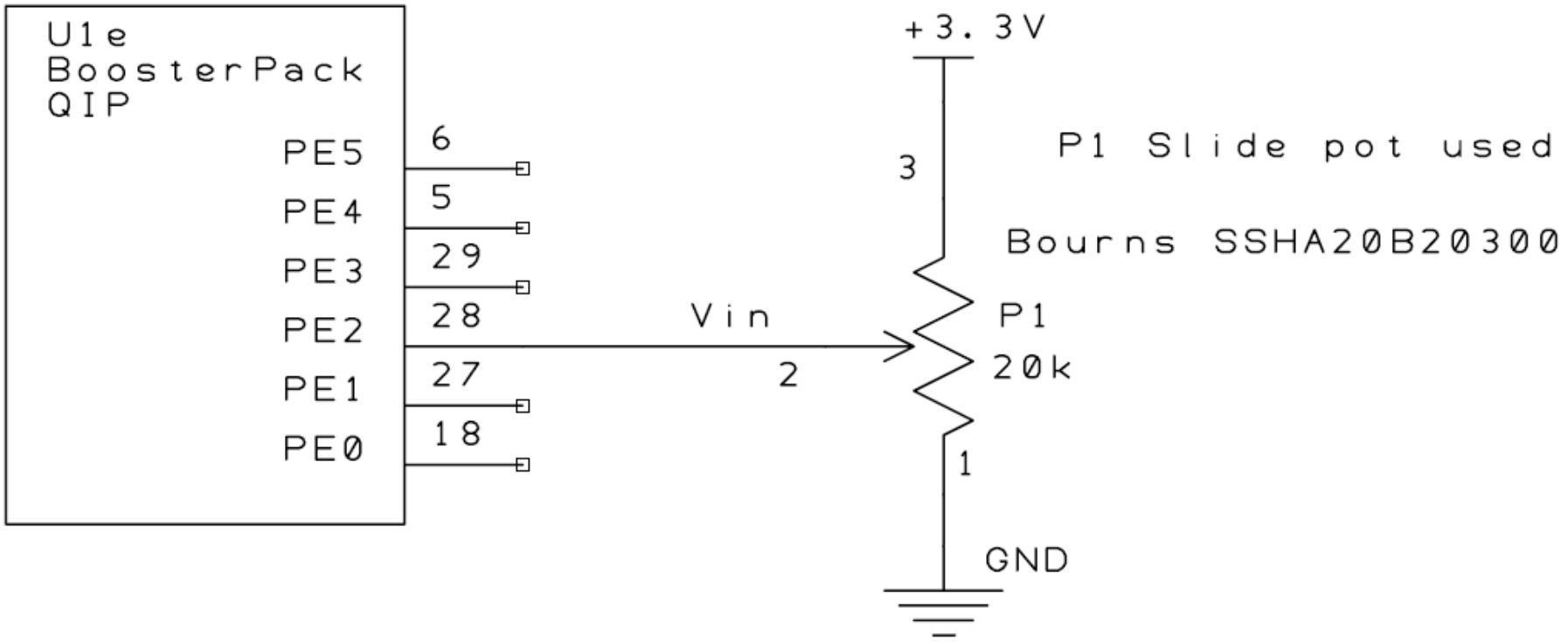
Servo Software

- Duty cycle controls angle (i.e.)

Position	Pulse Width	Example Pulse
Minimum	0.5ms	
Center	1.5ms	
Maximum	2.4ms	

Recall ADC

- Potentiometer and distance measurement
- Let’s use the potentiometer as a control switch of the servo motor


U1e
BoosterPack
QIP
PE5
PE4
PE3
PE2
PE1
PE0
6
5
29
28
27
18
Vin
2
+3.3V
3
P1
20k
1
GND
P1 Slide pot used
Bourns SSHA20B20300

RECALL

```
39  unsigned long Size = 6; // potentiomter range in cm
40  unsigned char String[10]; // string to output to screen
41  unsigned long Distance; // potentiometer distance in units of 0.001 cm
42  unsigned long ADCdata;  // 12-bit 0 to 4095 sample
43  unsigned long Flag;     // 1 means valid Distance, 0 means Distance is empty

45  unsigned long Convert(unsigned long sample){
46    // Converts ADC input to actual distance in units of 0.001cm
47    // Input: sample  12-bit ADC sample
48    // Output: 32-bit distance (resolution 0.001cm)
49    return (int)sample/4095.0 * Size * 1000;
50  }

void ConvertDistance(unsigned long n){
void SysTick_Init(unsigned long period){
void SysTick_Handler(void){
int main(void){
  // main function
  volatile unsigned long delay;
  ADC0_Init();
  SysTick_Init(2000000); // 40Hz (assuming 80MHz PLL)
  __enable_irq(); //enable interrupt

  while(1){
    if(Flag){
      Flag = 0;
      ConvertDistance(Distance);
    }
  }
}
```

```
// Slide pot pin 3 connected to +3.3V
// Slide pot pin 2 connected to PE2
// Slide pot pin 1 connected to ground
```

- Length ∝ Voltage receiving
- Based on voltage, MCU converts into digital value

ADC_In
→Returns “ADC inputs”

SysTick_Handler
→Calls Convert (ADC inputs)
→Updates “Distance”

main → ConvertDistance (Distance)
→Updates “String”

```
int main(void){
  volatile unsigned long delay, value;
  SysTick_Init(1500); //interrupts 250nsec*1500=375microsec fast enough to ADC data sampling
  ADC0_Init();
  delay = SYSCTL_RCGC2_R;         // allow time for clock to stabilize
  __enable_irq();

  PLL_Init(); // bus clock at 3.2 MHz
  PWM0_Init(32000,850);      // 32000: initialize PWM0, 50 Hz,  20 ms fixed perood for servo control
                             // considering /2 factor of PWM clock setting
                             // 850 for dcpwm0 to simulate 4-5% (min) duty cycle.
// PWM clock rate = processor clock rate/SYSCTL_RCC_PWMDIV
//                = BusClock/2
//                = 3.2 MHz/2 = 1.6MHz ==> 625nsec
// 32000*625nsec=0.02sec

 while(1){
    // read mailbox
    if(Flag==1){
      value= ADCdata*2600/4095;
                             //4096 is 12 bit ADC
                             //3840=2.4ms/625ns ==>3550 is the experimental max
                             // 800=0.5ms/625ns ==>950 is the experimental min
                             // +950 is for minimum postion of servo
      delay = SYSCTL_RCGC2_R;
      delay = SYSCTL_RCGC2_R;
      value= value+950;
      PWM0_Period(value);
    }
  Flag=0; // Flag reset to send the signal that last measure has been diplayed
  }
}
```

```
void PWM0_Period(unsigned long dcpwm0){
        PWM0_0_CMPA_R =dcpwm0;
}
```

Reading

Vol.1	Vol.2
Ch.8 (8.7)	Ch.6 (6.3, 6.5, 6.6…)