PWM AND DC MOTOR CONTROL

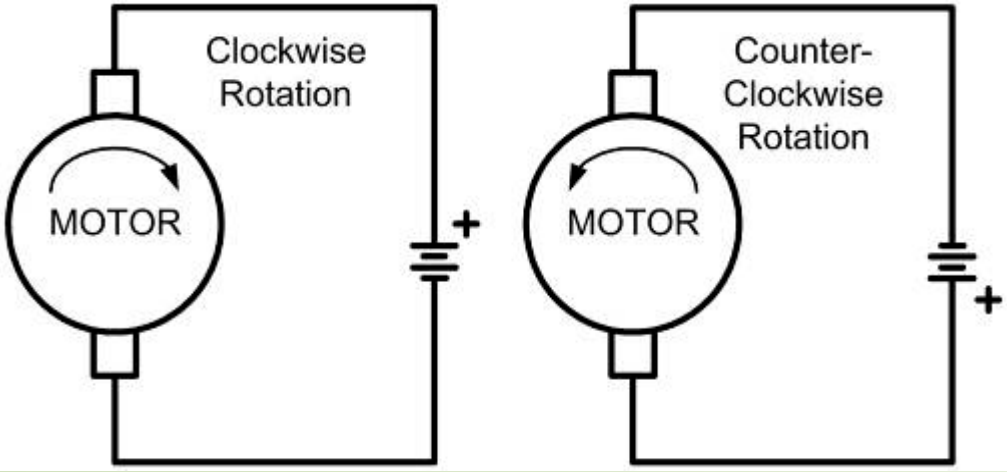
Taken/modified from:
Mazidi/Causey: “HCS12 Microcontroller and Embedded Systems,” chapter 17

TOPICS

- PWM to control motor speed.
- PWM features of the HCS12.
- Code HCS12 create PWM pulses.

DC Motor Interfacing and PWM

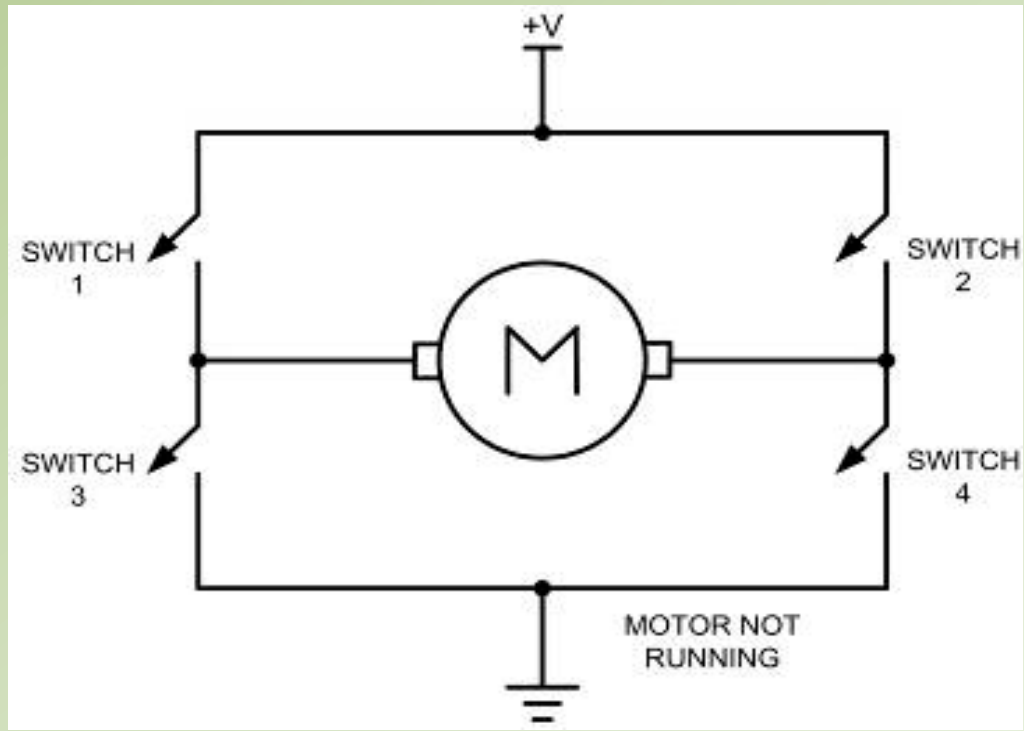
- DC motors have only + and – leads:
  - Connecting them to a DC voltage source → rotates the motor in one direction
  - Reversing the polarity, the DC motor will move in the opposite direction



DC Motor Rotation (Permanent Magnet Field)

DC Motor Interfacing and PWM:
Bidirectional control

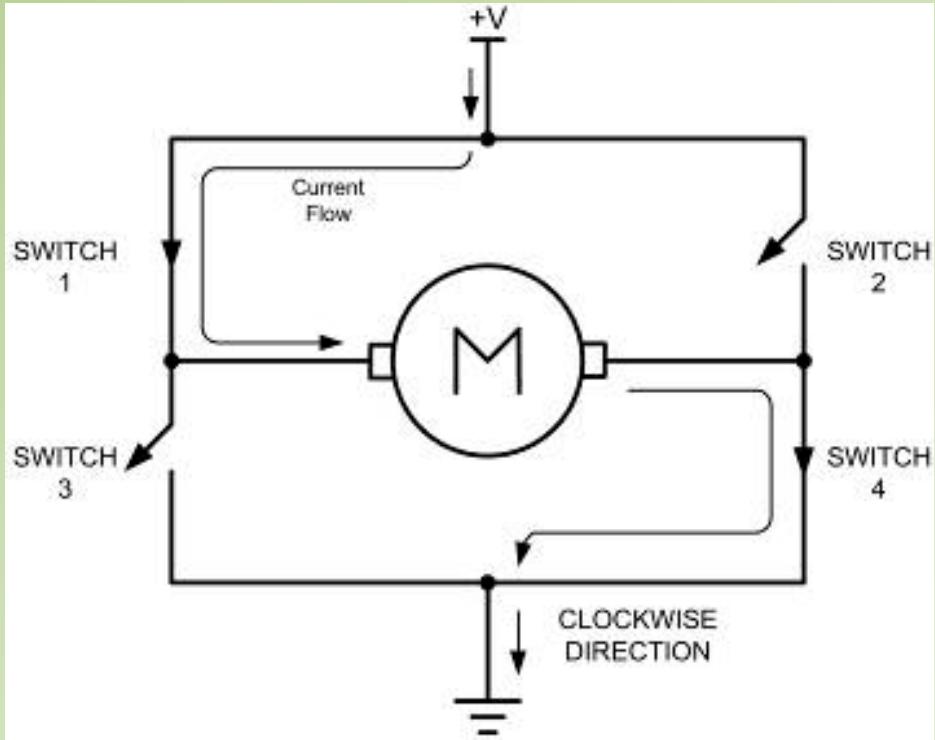
- H-Bridge using simple switches:
  - All switches are open → does not allow the motor to turn



H-Bridge Motor Configuration

... bidirectional control

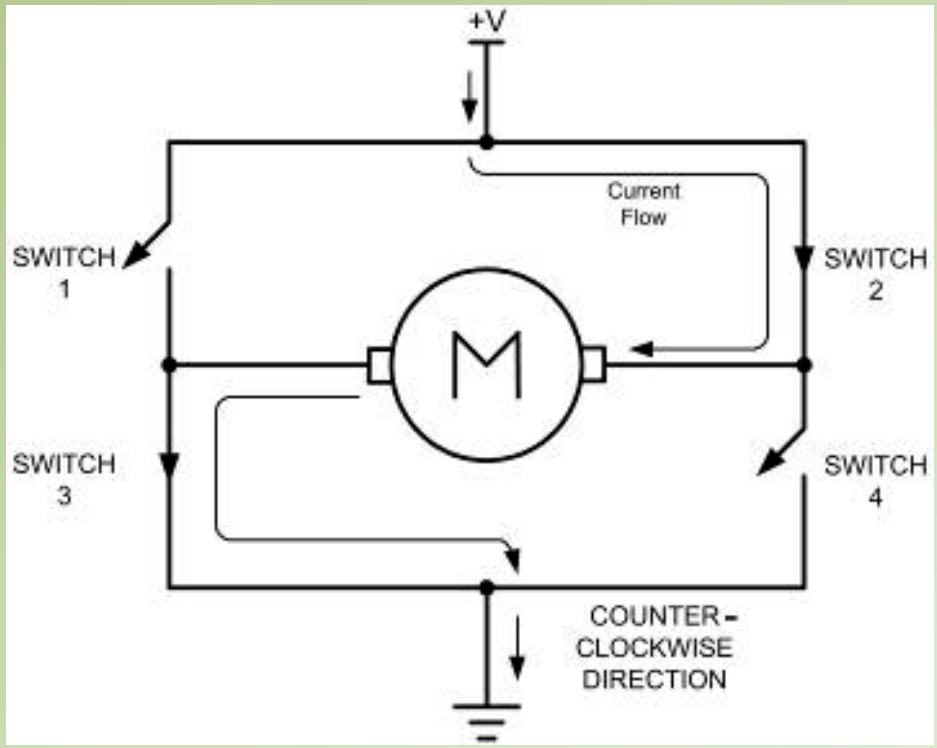
- Switch configuration for turning the motor in one direction.
  - when switches 1 and 4 are closed, current is allowed to pass through the motor



H-Bridge Motor Clockwise Configuration

... DC Motor Interfacing and PWM bidirectional control

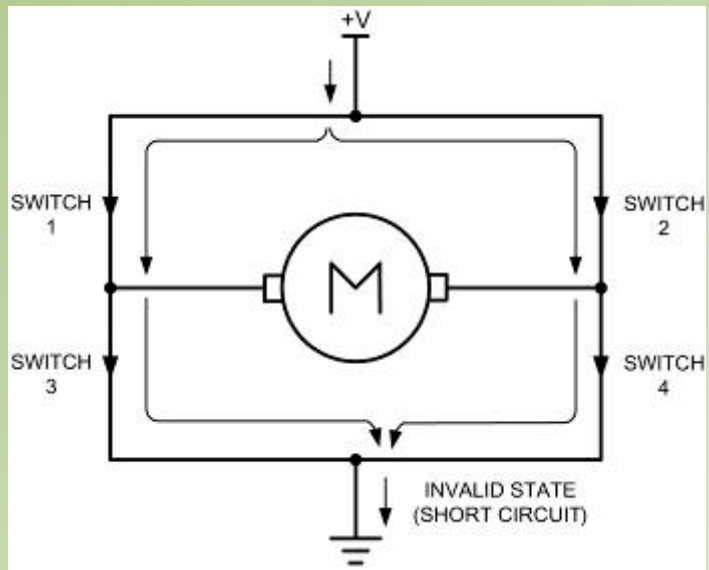
Motor Operation	SW1	SW2	SW3	SW4
Off	Open	Open	Open	Open
Clockwise	Closed	Open	Open	Closed
Counterclockwise	Open	Closed	Closed	Open
Invalid	Closed	Closed	Closed	Closed



- When switches 2 & 3 are closed, current is allowed to pass through the motor

... DC Motor Interfacing and PWM: bidirectional control

- H-Bridge controls can be created using relays, transistors, or a single IC such as the L293.
  - you must ensure that invalid configurations do not occur, as shown below.

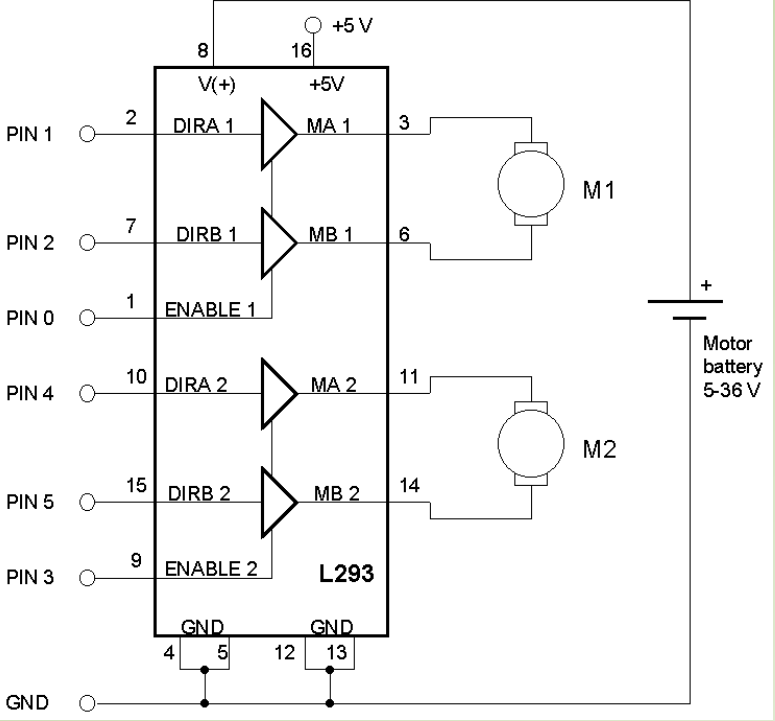


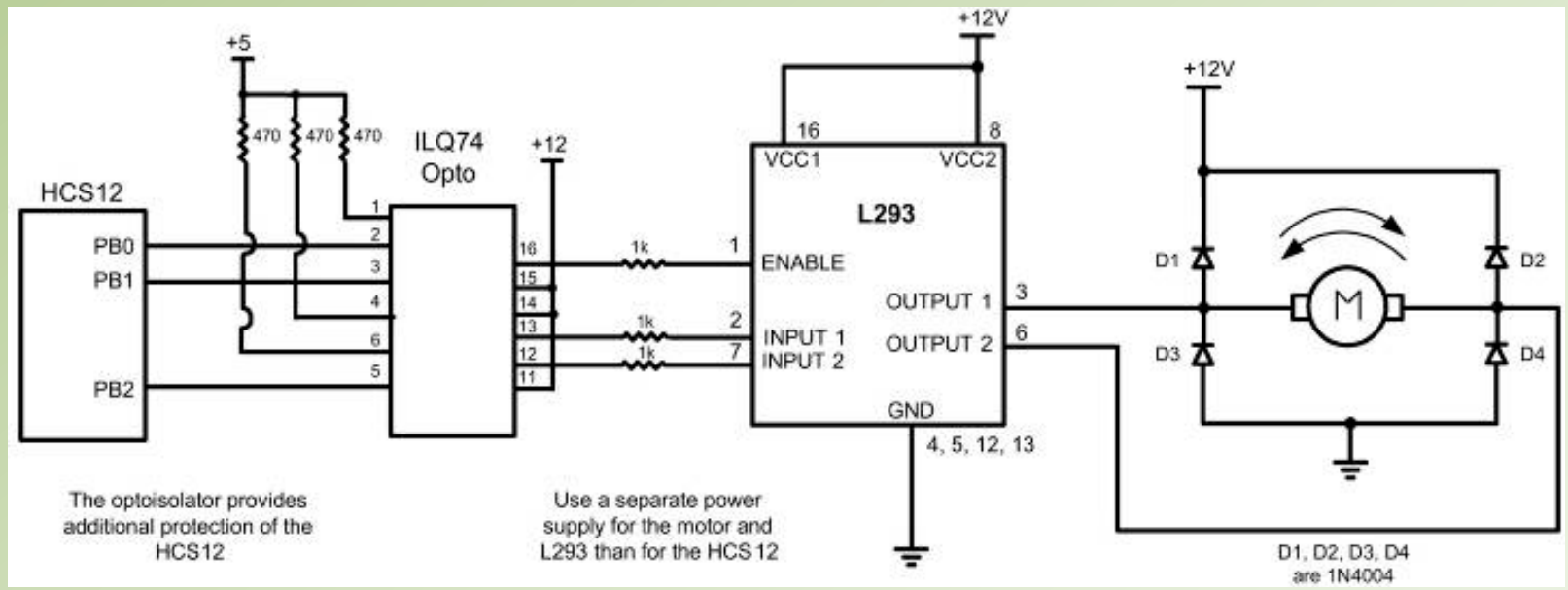
H-Bridge in an Invalid Configuration

- *Current flows directly to ground, creating a short circuit*
- *same effect as when switches 1 and 3 or switches 2 and 4 are closed*

... DC Motor bidirectional control

- Connection of the L293 chip to an HCS12.

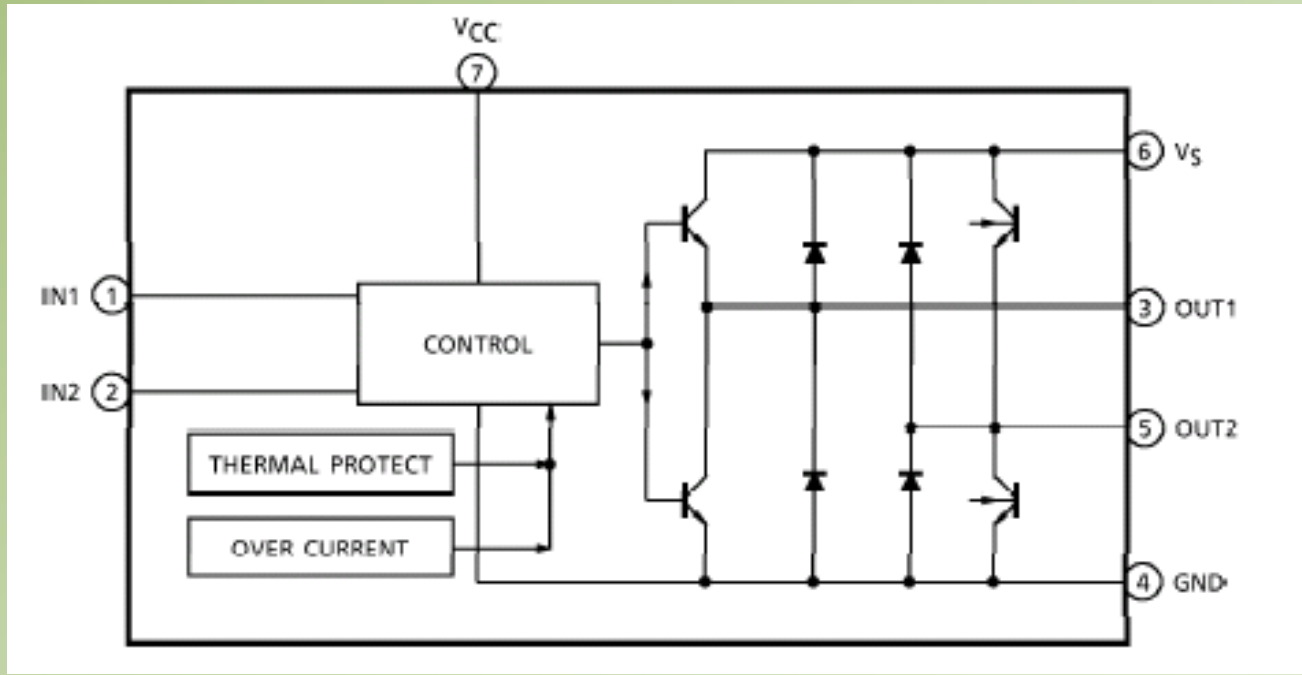




Bidirectional Motor Control Using an L293 Chip

TA7267 Bridge Driver

(used in robot arm in ENSC451 lab)



- Output Current Up to 1.0 A (AVE.), and 3.0 A (PEAK).
- 4 Function Modes (CW, CCW, STOP and Brake) are Controlled by 2 Logic Signals Fed Into 2 Input Terminals.
- Build in Over Current Protector and Thermal Shut Down Circuit.
- Operating Voltage Range : $V_{CC\,(opr.)}$ = 6~18 V, $V_{S\,(opr.)}$ = 0~18 V

Weight : 2.15 g (Typ.)

PIN FUNCTION

PIN No.	SYMBOL	FUNCTIONAL DESCRIPTION
1	IN1	Input terminal
2	IN2	Input terminal
3	OUT1	Output terminal
4	GND	GND terminal
5	OUT2	Output terminal
6	V_S	Voltage supply terminal
7	V_{CC}	Voltage supply terminal

FUNCTION

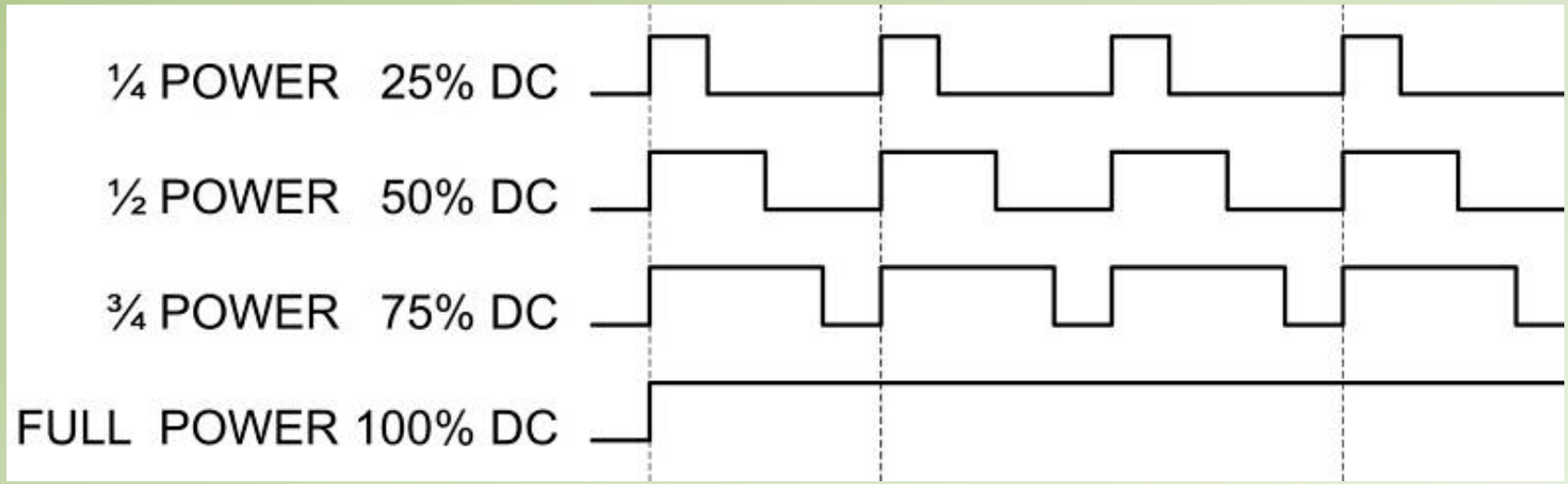
IN1	IN2	OUT1	OUT2	MODE
1	1	L	L	Brake
0	1	L	H	CW / CCW
1	0	H	L	CCW / CW
0	0	High Impedance		Stop

DC Motor Control using *pulse width modulation (PWM)*

- Speed of a motor depends on three factors:
  - load, voltage, current
- For a given fixed load we can maintain a steady speed using *pulse width modulation* (PWM).
- Changing (modulating) the width of the pulse applied to the motor can increase/decrease the amount of power provided increasing/decreasing the motor speed.
- PWM is widely used: Many microcontrollers come with the *PWM circuitry embedded in the chip*.

. . . PWM

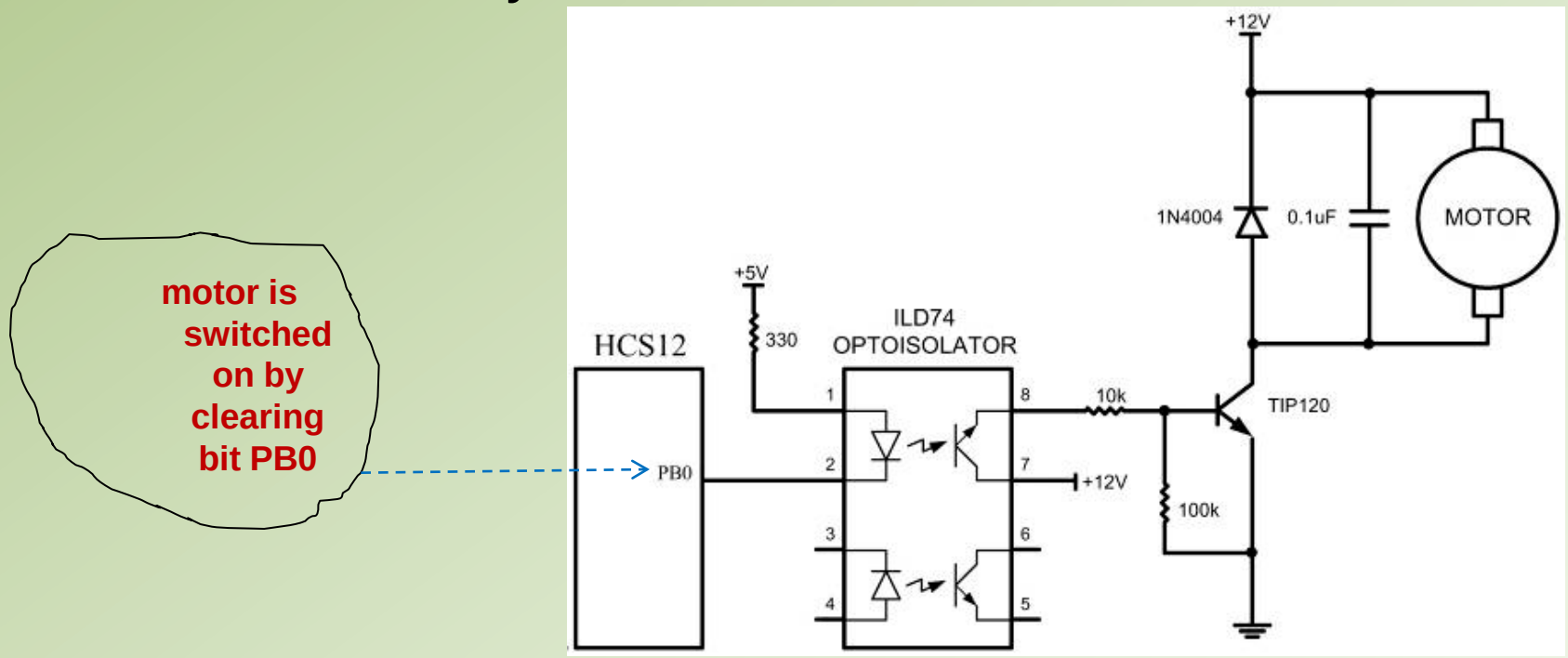
- Ability to control speed using PWM is one reason DC motors are preferable over AC motors.



Pulse Width Modulation

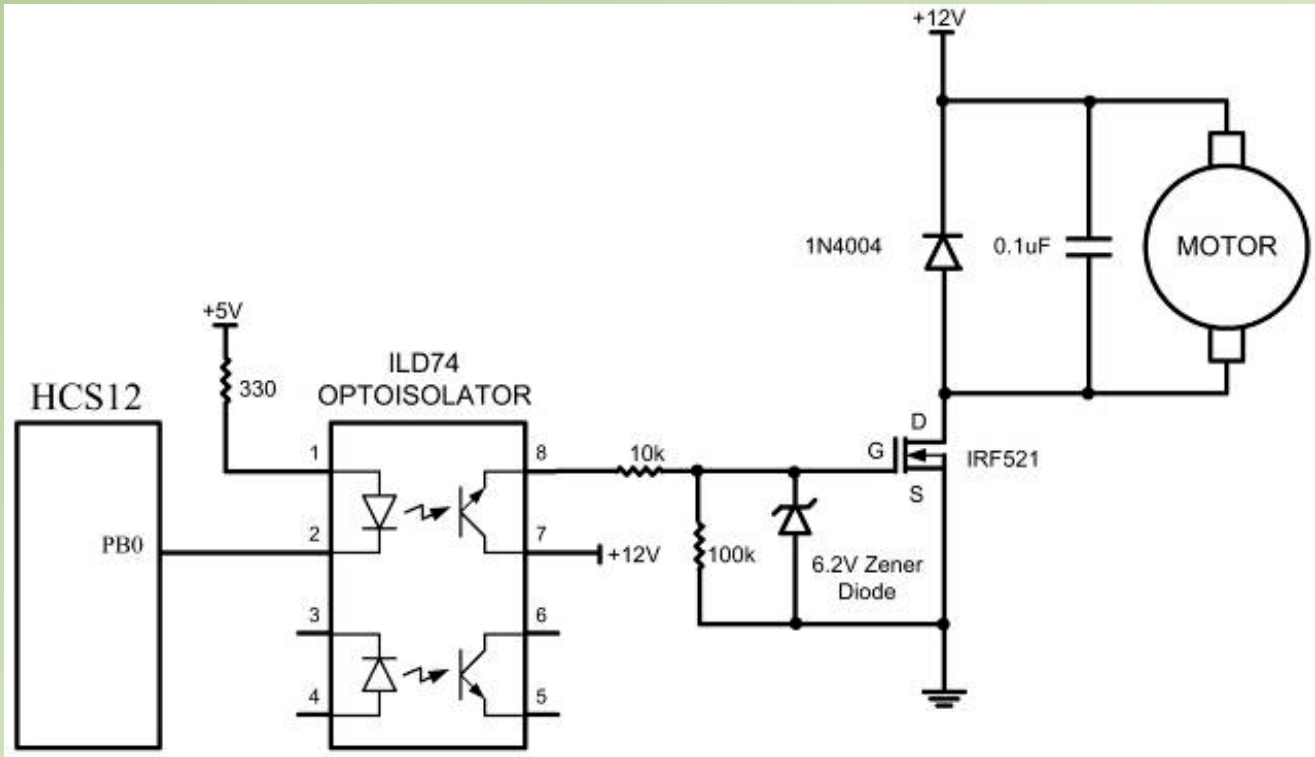
DC motor control with an opto-isolator

- Connection of a bipolar transistor to a motor:
  - control circuit protected by the optoisolator
  - motor & chip use separate power supplies
  - capacitor across the motor used to reduce EMI created by the motor



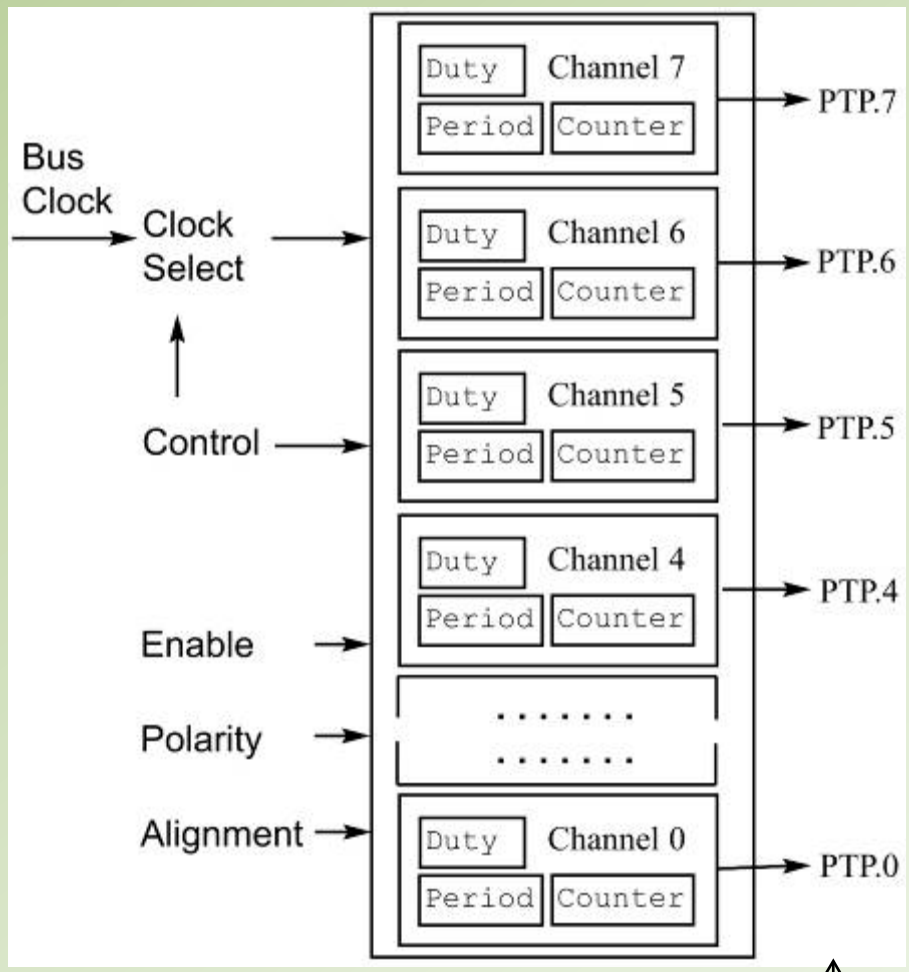
… DC motor control with an optoisolator

- Using a MOSFET
  - The optoisolator protects the HCS12 from EMI
  - The zener diode clamps gate voltage (< rated maximum value)



Programming PWM in HCS12:
PWM channels

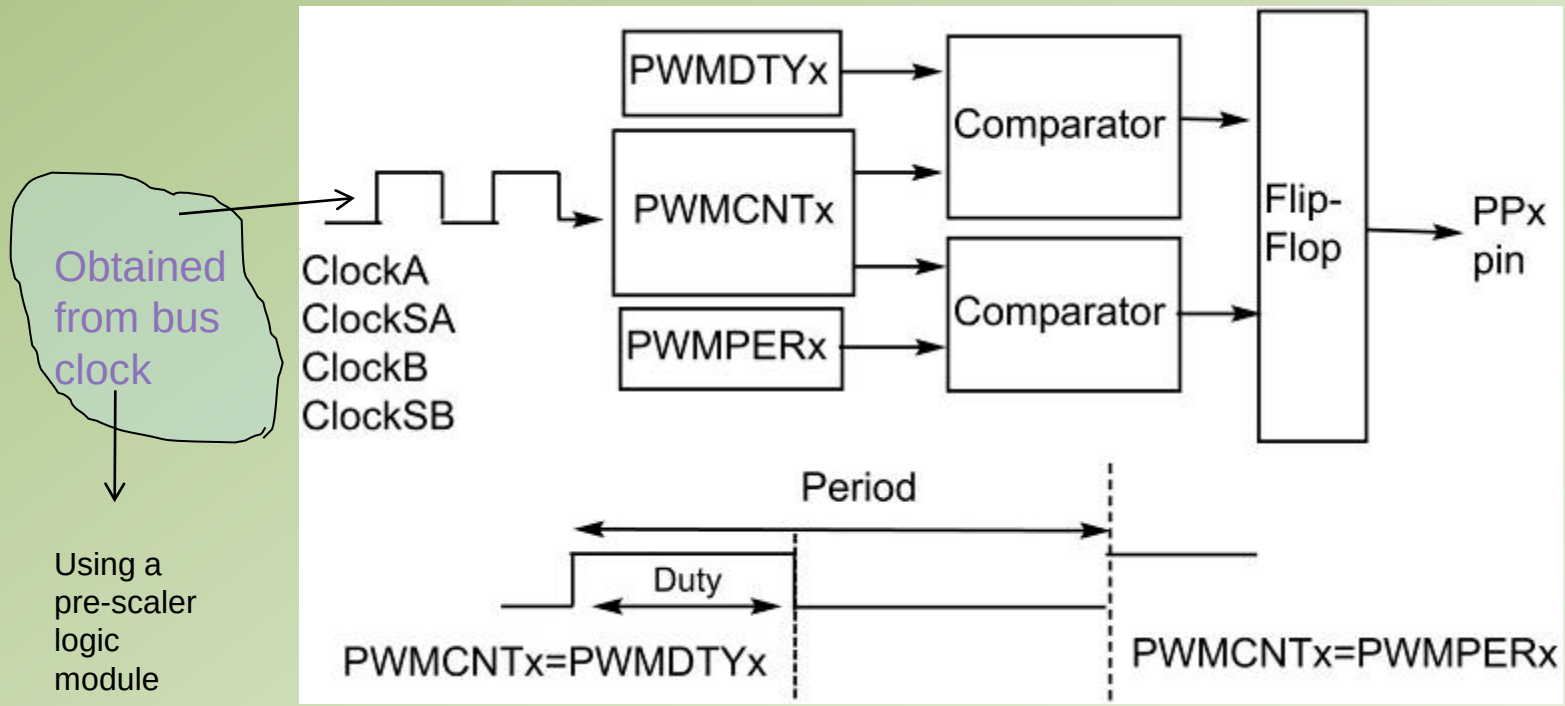
- HCS12 has *8 channels of 8-bit PWM*
- Two can combine and create a 16-bit PWM channel.
- HCS12 uses the PORTP pins for the PWM channels



PWM Channels and PORTP

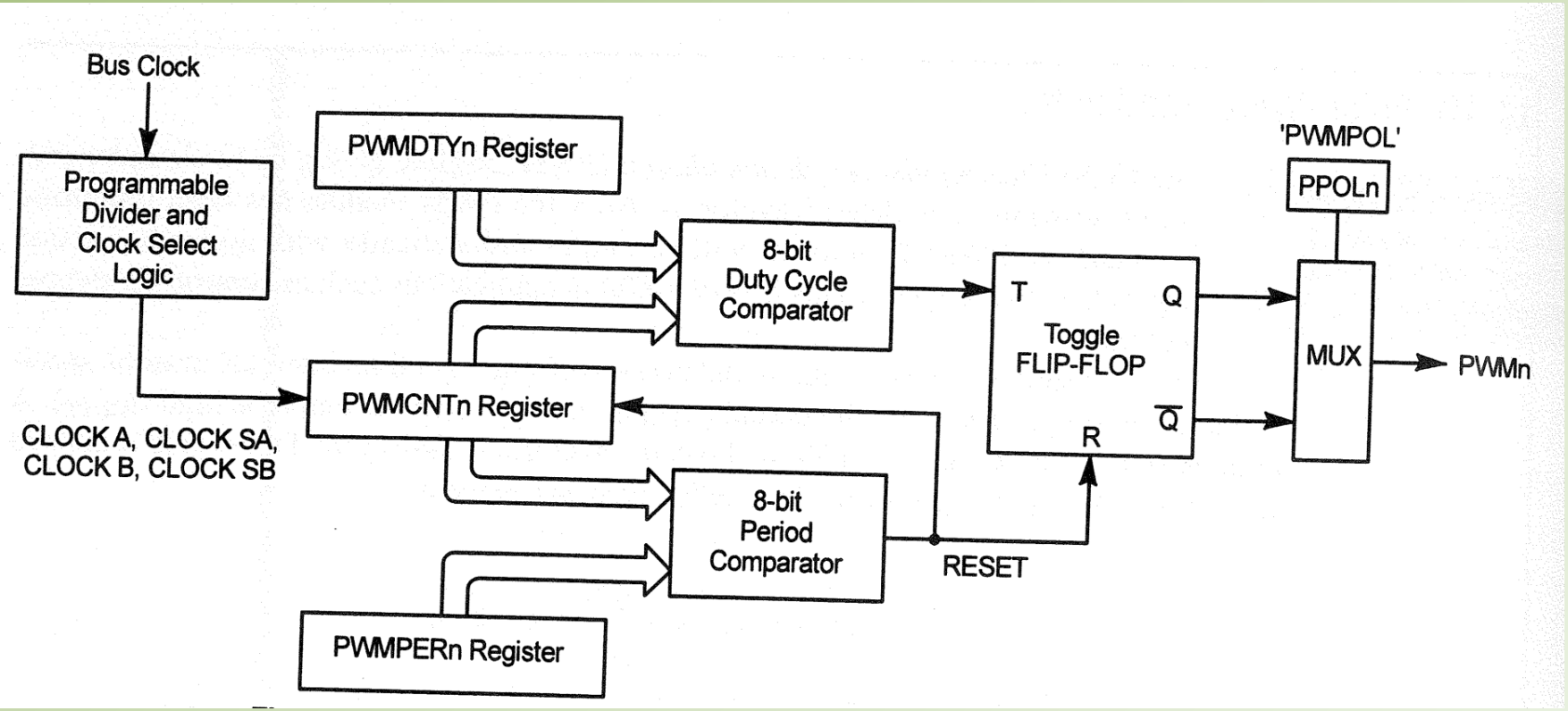
How PWM works (HCS12- other)

- Value in period register (PWMPERx) determines the frequency of the pulses out of the PORTP pin.



PWMCNT, PWMPER, and PWMDTY Registers used to generate PWM

HCS12 PWM Blocks

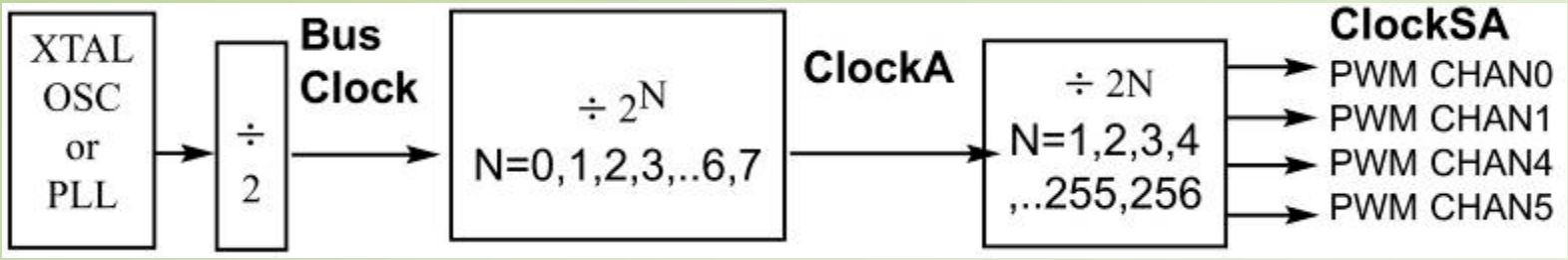


Programming PWM in HCS12: Clock sources

- PWM channels get their clocks from the bus clock
- Two input clocks:
  - ClockA is used for input to Channels 0, 1, 4, and 5
  - ClockB is used for input to Channels 2, 3, 6, and 7
- Using the pre-scaler, ClockA can be divided further, with the PWMPRCLK register
  - also true for ClockB

Generating the PWM clock source

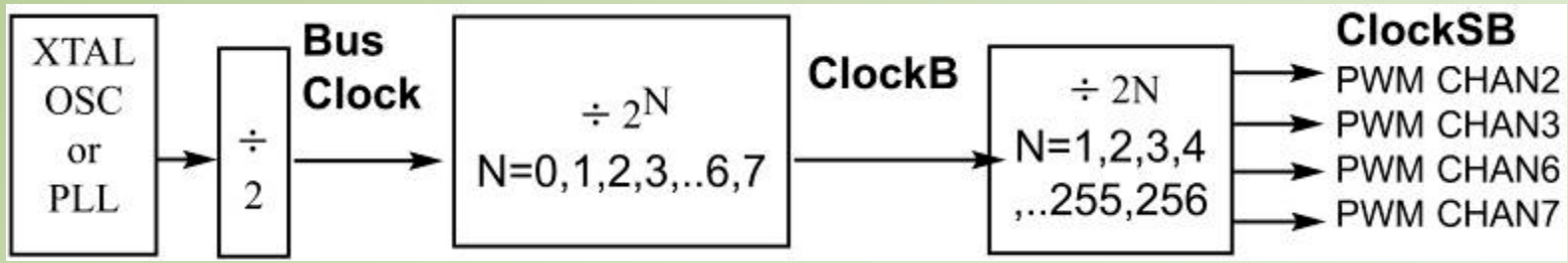
- ***ClockSA* -** To get a lower frequency for PWM channels, divide ClockA further using **PWMSCLA**.
- **PWMSCLA** registers are used to ***scale*** ClockA to ClockSA (scaled A).
  - the *highest number* by which ClockA is divisible is *512*
  - the *lowest number* is *2*.



ClockA and ClockSA Frequencies

… Programming PWM in HCS12: clock sources

- ***ClockSB* -** same way as ClockSA, except PWMSCLB is used.
  - similar rates are available for ClockSB as for ClockSA
  - in many applications we do not need to divide the clock



ClockB and ClockSB Frequencies (similar to ClockA/ClockSA but different channels)

Programming PWM in HCS12: PWMCLK used for clock source selection

PCLK7	PCLK6	PCLK5	PCLK4	PCLK3	PCLK2	PCLK1	PCLK0

PCLK7–PCLK0 PWM Clock Selection

PCLK7: D7 PWM Channel 7 Clock Selection
1 = ClockSB is the clock source for PWM channel 7.
0 = ClockB is the clock source for PWM channel 7.

PCLK6: D6 PWM Channel 6 Clock Selection
1 = ClockSB is the clock source for PWM channel 6.
0 = ClockB is the clock source for PWM channel 6.

PCLK5: D5 PWM Channel 5 Clock Selection
1 = ClockSA is the clock source for PWM channel 5.
0 = ClockA is the clock source for PWM channel 5.

PCLK4: D4 PWM Channel 4 Clock Selection
1 = ClockSA is the clock source for PWM channel 4.
0 = ClockA is the clock source for PWM channel 4.

PCLK3: D3 PWM Channel 3 Clock Selection
1 = ClockSB is the clock source for PWM channel 3.
0 = ClockB is the clock source for PWM channel 3.

PCLK2: D2 PWM Channel 2 Clock Selection
1 = ClockSB is the clock source for PWM channel 2.
0 = ClockB is the clock source for PWM channel 2.

PCLK1: D1 PWM Channel 1 Clock Selection
1 = ClockSA is the clock source for PWM channel 1.
0 = ClockA is the clock source for PWM channel 1.

PCLK0: D0 PWM Channel 0 Clock Selection
1 = ClockSA is the clock source for PWM channel 0.
0 = ClockA is the clock source for PWM channel 0.

PWMCLK (PWM Clock Selection) Register

- PWM channels **0, 1, 4, and 5,** can choose **ClockA or ClockSA**
- Also true for ClockB and ClockSB when using PWM channels **2, 3, 6, and 7**
- This is done using the **PWMCLK** register, as shown at left

Programming PWM in HCS12: Channel polarity

- Use the **PWMPOL** register to choose the polarity, as shown below

PPOL7	PPOL6	PPOL5	PPOL4	PPOL3	PPOL2	PPOL1	PPOL0

PPOL7–PPOL0 PWM Polarity Selection

PPOL7: PWM Channel 7 Polarity Selection
1 = PWM channel 7 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 7 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL6: PWM Channel 6 Polarity Selection
1 = PWM channel 6 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 6 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL5: PWM Channel 5 Polarity Selection
1 = PWM channel 5 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 5 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL4: PWM Channel 4 Polarity Selection
1 = PWM channel 4 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 4 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL3: PWM Channel 3 Polarity Selection
1 = PWM channel 3 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 3 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL2: PWM Channel 2 Polarity Selection
1 = PWM channel 2 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 2 output is low at the beginning of the period, then goes high when the duty count is reached.

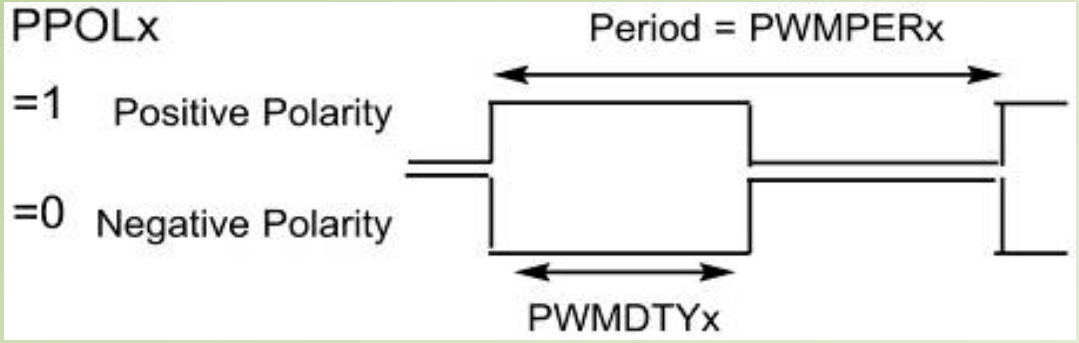
PPOL1: PWM Channel 1 Polarity Selection
1 = PWM channel 1 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 1 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL0: PWM Channel 0 Polarity Selection
1 = PWM channel 0 output is high at the beginning of the period, then goes low when the duty count is reached.
0 = PWM channel 0 output is low at the beginning of the period, then goes high when the duty count is reached.

PWMPOL (PWM Polarity Selection) Register

… Programming PWM in HCS12:
Left aligned PWM output

- To generate pulses for the PWM output, we have the choice of ***making the pulse go high first and then low***, or the other way around.
- Also- Center Aligned outputs.



PWM Left Alignment Output with Polarity

… left/center aligned PWM output

- Use the **PWMCAE** register to choose the alignment, as shown below.

PWMCAE (PWM Center Aligned Enable) Register

CAE7	CAE6	CAE5	CAE4	CAE3	CAE2	CAE1	CAE0

CAE7–CAE0 PWM Center Aligned Enable

CAE7: D7 PWM Channel 7 Center Aligned output mode
1 = Channel 7 operates in Center Aligned output mode.
0 = Channel 7 operates in Left Aligned output mode.

CAE6: D6 PWM Channel 6 Center Aligned output mode
1 = Channel 6 operates in Center Aligned output mode.
0 = Channel 6 operates in Left Aligned output mode.

CAE5: D5 PWM Channel 5 Center Aligned output mode
1 = Channel 5 operates in Center Aligned output mode.
0 = Channel 5 operates in Left Aligned output mode.

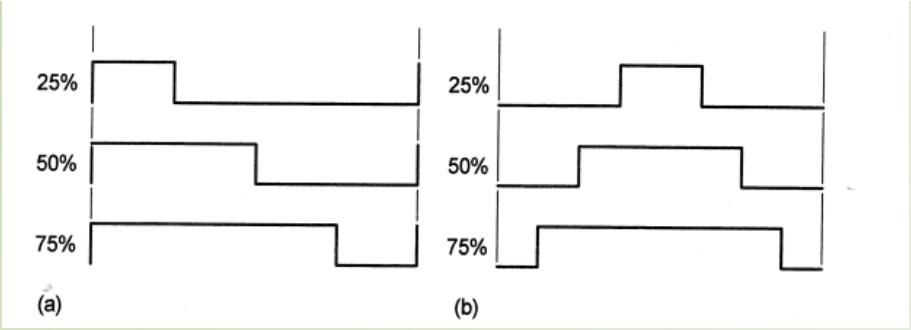
CAE4: D4 PWM Channel 4 Center Aligned output mode
1 = Channel 4 operates in Center Aligned output mode.
0 = Channel 4 operates in Left Aligned output mode.

CAE3: D3 PWM Channel 3 Center Aligned output mode
1 = Channel 3 operates in Center Aligned output mode.
0 = Channel 3 operates in Left Aligned output mode.

CAE2: D2 PWM Channel 2 Center Aligned output mode
1 = Channel 2 operates in Center Aligned output mode.
0 = Channel 2 operates in Left Aligned output mode.

CAE1: D1 PWM Channel 1 Center Aligned output mode
1 = Channel 1 operates in Center Aligned output mode.
0 = Channel 1 operates in Left Aligned output mode.

CAE0: D0 PWM Channel 0 Center Aligned output mode
1 = Channel 0 operates in Center Aligned output mode.
0 = Channel 0 operates in Left Aligned output mode.



Programming PWM in HCS12: Enabling (turning on) the PWM channel

- Upon reset, PWM channels are disabled***, i.e., no clock is provided to them.***
  - Use the **PWME** (PWM Enable) register to enable (turn on) a given channel, as shown below.

PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0

PWME7–PWME0 PWM Channel Enable

PWME7: D7 PWM Channel 7 Enable
1 = PWM channel 7 is enabled.
0 = PWM channel 7 is disabled.

PWME6: D6 PWM Channel 6 Enable
1 = PWM channel 6 is enabled.
0 = PWM channel 6 is disabled.

PWME5: D5 PWM Channel 5 Enable
1 = PWM channel 5 is enabled.
0 = PWM channel 5 is disabled.

PWME4: D4 PWM Channel 4 Enable
1 = PWM channel 4 is enabled.
0 = PWM channel 4 is disabled.

PWME3: D3 PWM Channel 3 Enable
1 = PWM channel 3 is enabled.
0 = PWM channel 3 is disabled.

PWME2: D2 PWM Channel 2 Enable
1 = PWM channel 2 is enabled.
0 = PWM channel 2 is disabled.

PWME1: D1 PWM Channel 1 Enable
1 = PWM channel 1 is enabled.
0 = PWM channel 1 is disabled.

PWME0: D0 PWM Channel 0 Enable
1 = PWM channel 0 is enabled.
0 = PWM channel 0 is disabled.

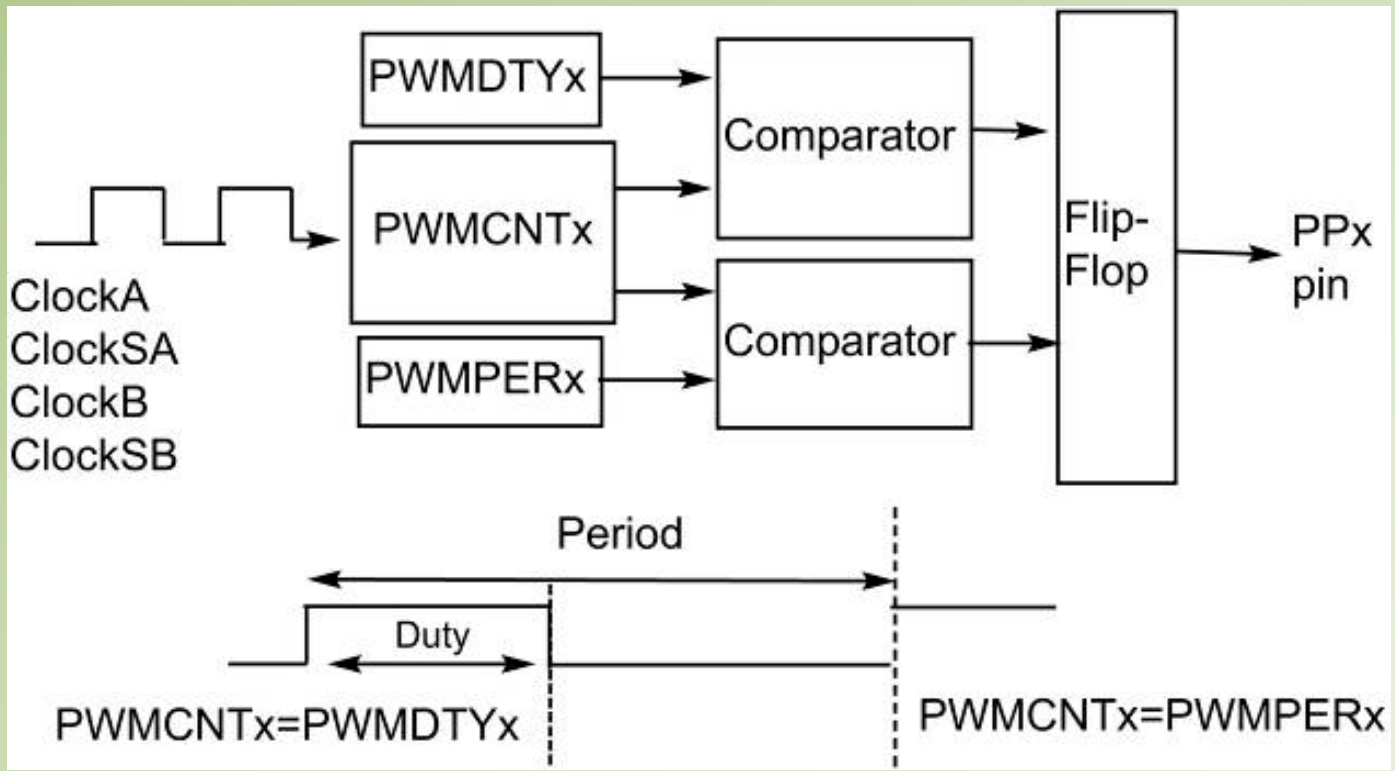
PWME (PWM Enable) Register

Programming PWM in HCS12: Counter, Period, Duty Cycle Registers

- Three 8-bit registers associated with each channel (8 channels):
  - **Counter (PWMCNTx)** registers: PWMCNT0, PWMCNT1, PWMCNT2, etc.
  - **Period (PWMPERx)** registers: PWMPER0, PWMPER1, PWMPER2, etc
  - **Duty Cycle (PWMDTYx)** registers: PWMDTY0, PWMDTY1, PWMDTY2, etc.

Programming PWM in HCS12: Counter, period, duty cycle registers

- Value in period register (PWMPERx) determines the frequency of the pulses out of the PORTP pin.



PWMCNT, PWMPER, and PWMDTY in Creating the Duty Cycle

Programming PWM in HCS12: Counter, period, duty cycle registers

- The **clock source** (ClockA, ClockB, Clock SA, or Clock SB) is **connected to the PWMCNTx** registers.
- Upon enabling the PWM channel, the PWMCNTx register starts to count up from zero.
- As it counts up, its value is compared with the period register (PWMPERx) value and when they match, the output pin changes state.
- The value in the duty cycle register determines the duty cycle of the PWM pulses coming out of the PORTP pin.

Programming PWM in HCS12: PWM control register (8 or 16 bit PWM resolution)

- The **PWM control register** can be used to select 8- or 16-bit PWM channels, which are 8-bit by default.
  - **PWMCTL** is also used to stop PWM output during Wait and Freeze modes of the CPU operation

CON67	CON45	CON23	CON01	PSWAI	PFRZ	0	0

CON67: D7 Concatenate channels 6 and 7 to create a 16-bit PWM channel.
1 = Channels 6 and 7 are concatenated to create a 16-bit PWM channel. Channel 6 is the high byte and channel 7 is the low byte of the 16 bits. All the options (output pin, clock source, and so on) for channel 7 are used for the 16-bit PWM.
0 = Channels 6 and 7 are separate 8-bit PWMs.

CON45: D6 Concatenate channels 4 and 5 to create a 16-bit PWM channel.
1 = Channels 4 and 5 are concatenated to create a 16-bit PWM channel. Channel 4 is the high byte and channel 5 is the low byte of the 16-bit. All the options (output pin, clock source, and so on) for channel 5 are used for the 16-bit PWM.
0 = Channels 4 and 5 are separate 8-bit PWMs.

CON23: D5 Concatenate channels 2 and 3 to create a 16-bit PWM channel.
1 = Channels 2 and 3 are concatenated to create a 16-bit PWM channel. Channel 2 is the high byte and channel 3 is the low byte of the 16-bit. All the options (output pin, clock source, and so on) for channel 3 are used for the 16-bit PWM.
0 = Channels 2 and 3 are separate 8-bit PWMs.

CON01: D4 Concatenate channels 0 and 1 to create a 16-bit PWM channel.
1 = Channels 0 and 1 are concatenated to create a 16-bit PWM channel. Channel 0 is the high byte and channel 1 is the low byte of the 16-bit. All the options (output pin, clock source, and so on) for channel 1 are used for the 16-bit PWM.
0 = Channels 0 and 1 are separate 8-bit PWMs.

PSWAI: D1 PWM stops in Wait mode.
1 = Stop the input clock to the prescaler whenever the CPU is in Wait mode.
0 = Allow the clock to the prescaler to continue while in Wait mode.

PFRZ: D0 PWM counter stops in Freeze mode.
1 = Disable PWM input clock to the prescaler whenever the part is in Freeze mode.
0 = Allow PWM to continue while in Freeze mode.

Note: Upon reset, the default value value for this register is 00000000.

PWMCTL (PWM Control) Register

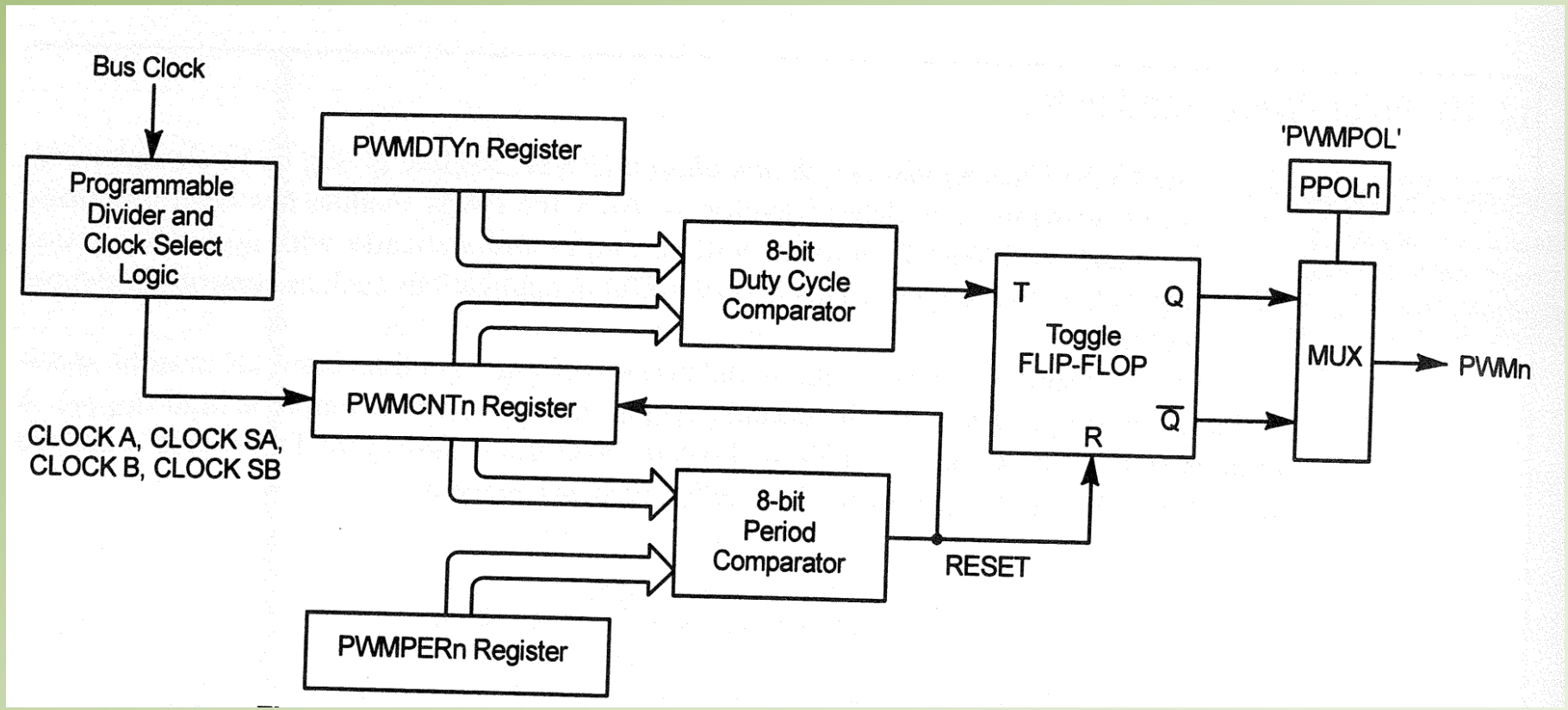
Steps in programming PWM 8-bit channels in HCS12

- Use **PWMPRCLK** to set prescaler value to bring Bus frequency (Fbus) to the desired value of ClockA (or ClockB).
- If needed, use **PWMSCLA** (or **PWMSCLB**) to bring down ClockA (or ClockB) even further to get ClockSA (or ClockSB).
- Use **PWMCLK** to select clock source of ClockA or ClockB (or ClockSA/ClockSB) for desired channel.
- Use **PWMPOL** to select **polarity** of the PWM output.
- Use **PWMCAE** to select Left/Center Aligned output.

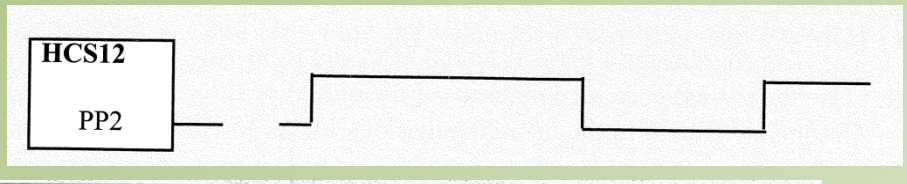
… steps in programming PWM

- Use **PWMCTL** register to **select the 8-bit channel** and PWM operations during the **wait and freeze modes** of the CPU.
- **Load** the value for the **period of the PWM** pulses into the **PWMPERx** register.
- **Load** the value for the **duty cycle** of the PWM pulses into the **PWMDTYx** register.
- **Clear** the **PWMCNTx** register.
- **Enable** (turn on) the desired PWM channel using the **PWMEN** register.

→ Several PWM registers have to be initialized before operation



PWM programming



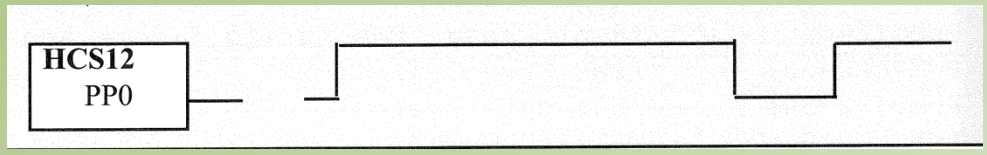
Assume XTAL = 4 MHz (Fbus = 2 MHz). Using channel 2, write a C program to create the PWM pulses with a frequency of 50 Hz and 60% duty cycle. This is a C version Example 17-11.

Solution:

We need to set PWMPRCLK = 30 for ClockB since ClockB = 2 MHz / 2^3 = 2 MHz/8 = 250 kHz. PWMSCLB = 50 since ClockSB = 250 kHz/(2 × 50) = 250 kHz/100 = 2,500 Hz. Since PWM_Freq = ClockSB/PWM_PER we need PWM_Freq = 2,500 Hz/50 = 50 Hz, which means we must have PWMPER2 = 50. For the duty cycle of 60%, we need PWMDTY2 = 30 since 50 × 60% = 30.

```
#include <mc9s12dp512.h>      /* derivative information */
void main()
  {
      PWMPRCLK=0x30;    //ClockSB=Fbus/8
      PWMSCLB=0x32;     //ClockSB/(2x50)
      PWMCLK=0x04;      //use ClockSB for channel 2
      PWMPOL=0x04;      //high, then low for polarity
      PWMCAE=0;         //left aligned
      PWMCTL=0;         //8-bit chan, PWM during freeze and wait
      PWMPER2=50;       //PWM_Freq=ClockSB/50
      PWMDTY2=30;       //60% duty cycle
      PWMCNT2=0;        //clear PWMCNT2
      PWME=0x04;        //turn on (start) PWM2
      while(1);
  }
```

PWM programming



Assume XTAL = 4 MHz (Fbus = 2 MHz). Using channel 0, write a program to create the PWM pulses with a frequency of 2 kHz and 80% duty cycle. This is a C version of Example 17-12.

Solution:

We need to set PWMPRCLK = 02 for ClockA since 2 MHz / 2^2 = 2 MHz/4 = 500 kHz. We bypass the use of ClockA since we can get the PWM_Freq = ClockA/PWM_PER. PWM_Freq = 500 kHz/ 250 = 2 kHz, which means we must have PWMPER0 = 250. For the duty cycle of 80% we have PWMDTY0 = 200 since 250 × 80% = 200.

```
#include <mc9s12dp512.h>      /* derivative information */
void main()
  {
      PWMPRCLK=0x2;     //ClockA=Fbus/4
      PWMCLK=0x00;      //use ClockA for channel 0
      PWMPOL=0x01;      //high, then low for polarity
      PWMCAE=0;         //left aligned
      PWMCTL=0;         //8-bit chan,PWM during freeze and wait
      PWMPER0=250;      //PWM_Freq=ClockA/250
      PWMDTY0=200;      //80% duty cycle
      PWMCNT0=0;        //clear PWMCNT0
      PWME=0x01;        //turn on (start) PWM0
      while(1);
  }
```