RV College of Engineering®

Go, change the world

Course Code:CS344AI

IOT and Embedded Computing

Programming LPC 2148 PWM,DC Motor Speed Control

What is PWM..

Pulse rate / Pulse Period

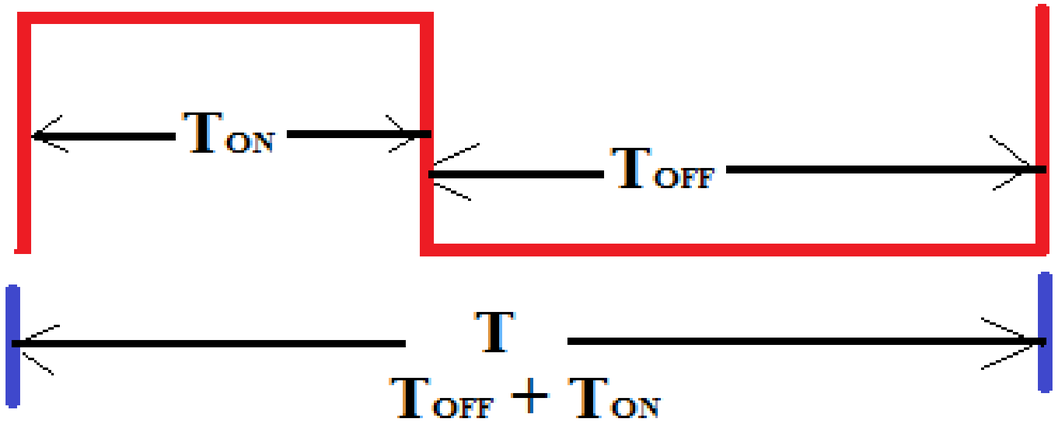
Period of one cycle- Ton + Toff time

Pulse Width

Ton - (On time)

Varying Duty Cycle

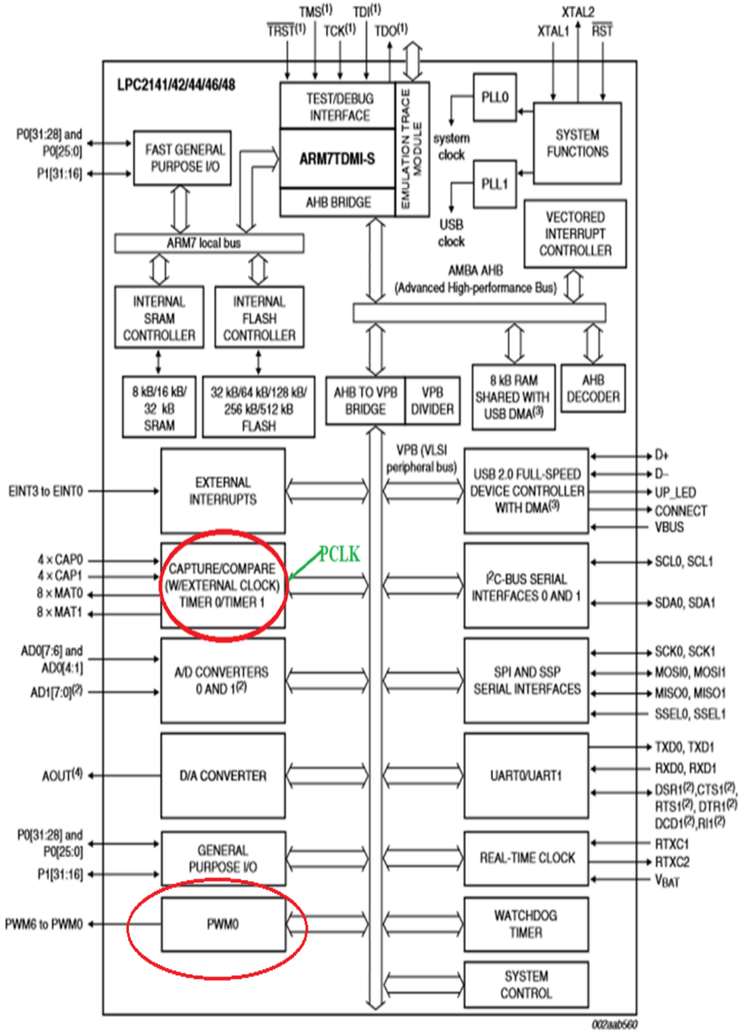
By Programming Pule Period and Pulse width, hence achieving different duty cycles ratio of ON time to ON+OFF time.

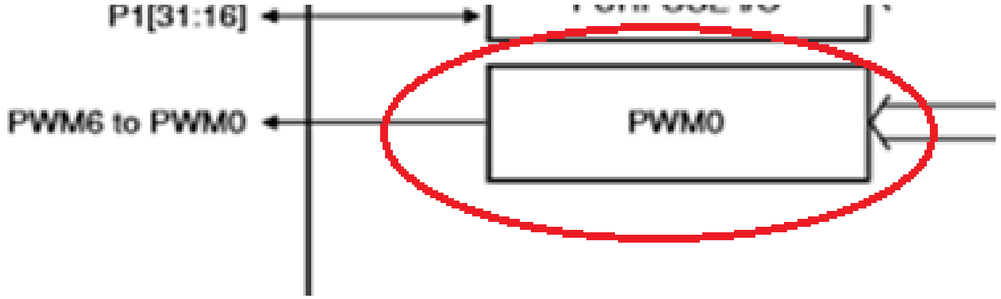


$$\text{DUTY CYCLE} = \frac{T_{ON}}{T_{OFF} + T_{ON}} = \frac{T_{ON}}{T}$$

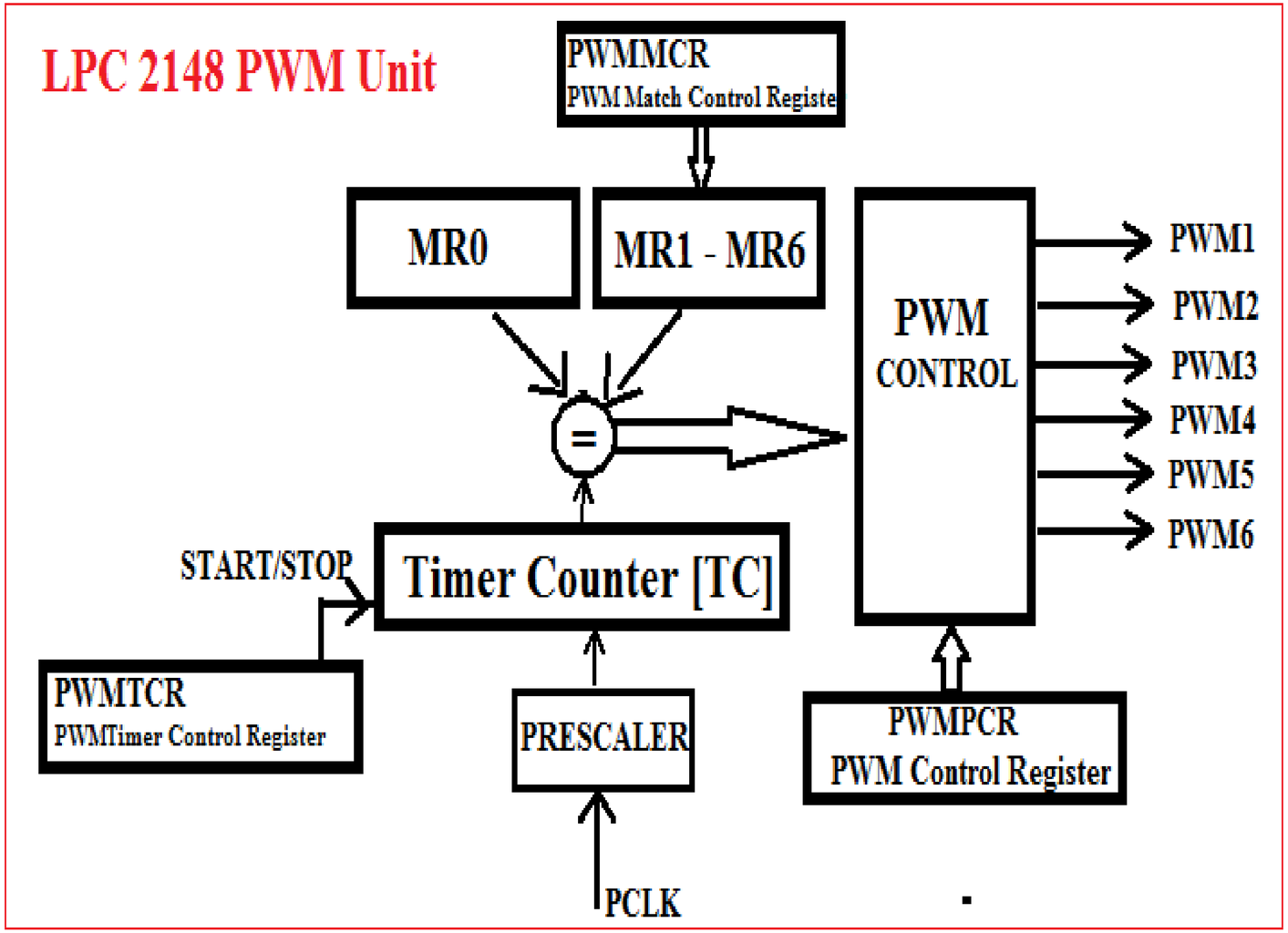
Duty cycle = [Ton / (Ton + Toff)] x 100 in%

LPC 2148 PWM Unit...

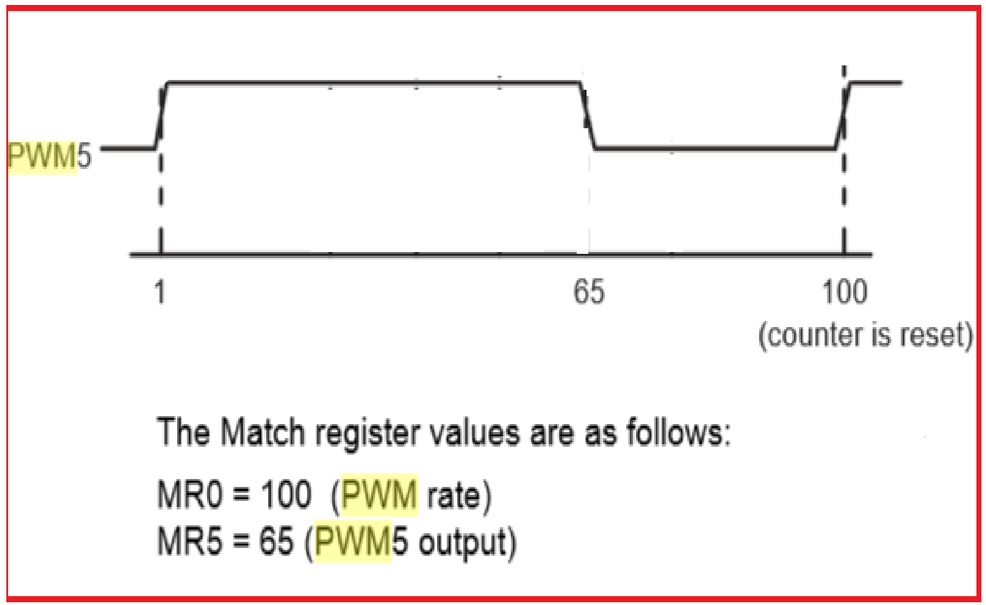




PWM Block Diagram ...



Defining Pule Period & Pulse Width



Pulse period and width can be any number of timer counts and all PWM outputs will occur at the same repetition rate.

Meaning of MR0:

One Match Register is required to define the number of PCLK cycles required to make one T, Pulse period (Ton + Toff), MR0 is used to define this value.

(Pulse period and width can be any number of timer counts and all PWM outputs will occur at the same repetition rate.)

Meaning Of MR1-MR6 :

To define, different pulse periods for six different PWM channels, PWM1 TO PWM6, six Match Registers are provided, MR1 to MR6. The number, which is loaded into these registers, define number of PCLK cycles will constitute Ton time.

PWM Registers..

PWMTCR - PWM Timer Control Register

Bit	Symbol	Description	Reset value
0	Counter Enable	When one, the PWM Timer Counter and PWM Prescale Counter are enabled for counting. When zero, the counters are disabled.	0
1	Counter Reset	When one, the PWM Timer Counter and the PWM Prescale Counter are synchronously reset on the next positive edge of PCLK. The counters remain reset until TCR[1] is returned to zero.	0
2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	PWM Enable	When one, PWM mode is enabled. PWM mode causes shadow registers to operate in connection with the Match registers. A program write to a Match register will not have an effect on the Match result until the corresponding bit in PWMLER has been set, followed by the occurrence of a PWM Match 0 event. Note that the PWM Match register that determines the PWM rate (PWM Match 0) must be set up prior to the PWM being enabled. Otherwise a Match event will not occur to cause shadow register contents to become effective. **Remark:** The PWM Enable bit needs to be always set to 1 for PWM operation. Otherwise, the PWM will behave as standard timer .	0
7:4	-	Reserved, user software should not write ones to	NA

PWMPCR – PWM control register

used to selectively enable / disable PWM channels PWM1 – PWM6. Also this register is used to configure the PWM channels as

a) single edge

b) double edge.

Bits: D15 D14 D13……D9.. D6 D5 …..D2 D1 D0

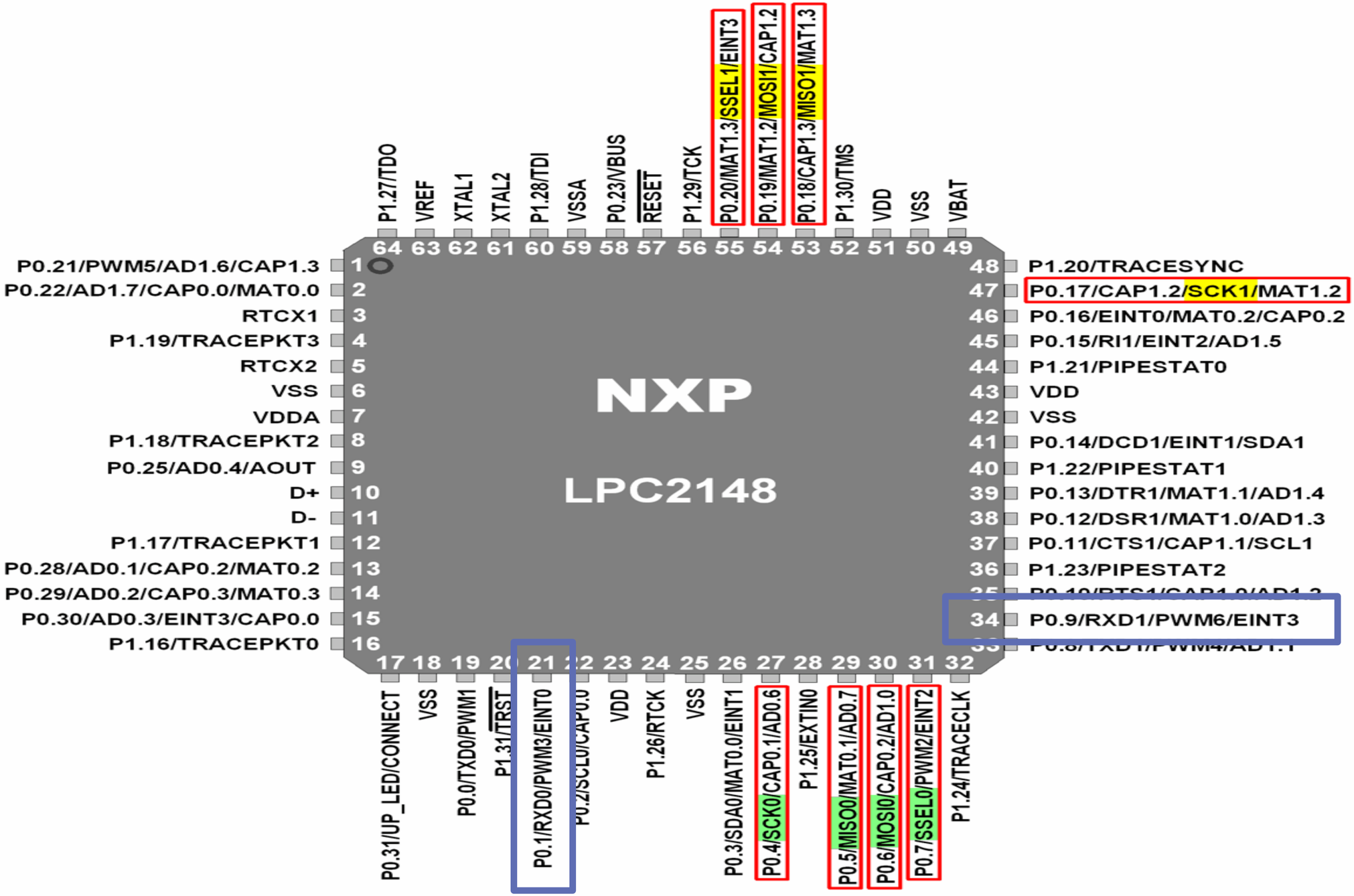
(Bits D9 to D14 are used to enable the channels PWM1 to PWM6)

PWMLER – PWM Latch Enable Register,

Bits: D7 D6 D5 D4 D3 D2 D1 D0

MR6 MR5 MR4 MR3 MR2 MR1 MR0

‘1’ – Writing 1 to this bit allows the last value written to the Match register to become effective, when the timer is next by a PWM Match event.


NXP
LPC2148
1 P0.21/PWM5/AD1.6/CAP1.3
2 P0.22/AD1.7/CAP0.0/MAT0.0
3 RTCX1
4 P1.19/TRACEPKT3
5 RTCX2
6 VSS
7 VDDA
8 P1.18/TRACEPKT2
9 P0.25/AD0.4/AOUT
10 D+
11 D-
12 P1.17/TRACEPKT1
13 P0.28/AD0.1/CAP0.2/MAT0.2
14 P0.29/AD0.2/CAP0.3/MAT0.3
15 P0.30/AD0.3/EINT3/CAP0.0
16 P1.16/TRACEPKT0
17 P0.31/UP_LED/CONNECT
18 VSS
19 P0.0/TXD0/PWM1
20 P1.31/TRST
21 P0.1/RXD0/PWM3/EINT0
22 P0.2/SCL0/CAP0.0
23 VDD
24 P1.26/RTCK
25 VSS
26 P0.3/SDA0/MAT0.0/EINT1
27 P0.4/SCK0/CAP0.1/AD0.6
28 P1.25/EXTIN0
29 P0.5/MISO0/MAT0.1/AD0.7
30 P0.6/MOSI0/CAP0.2/AD1.0
31 P0.7/SSEL0/PWM2/EINT2
32 P1.24/TRACECLK
33 P0.8/TXD1/PWM4/AD1.1
34 P0.9/RXD1/PWM6/EINT3
35 P0.10/RTS1/CAP1.0/AD1.2
36 P1.23/PIPESTAT2
37 P0.11/CTS1/CAP1.1/SCL1
38 P0.12/DSR1/MAT1.0/AD1.3
39 P0.13/DTR1/MAT1.1/AD1.4
40 P1.22/PIPESTAT1
41 P0.14/DCD1/EINT1/SDA1
42 VSS
43 VDD
44 P1.21/PIPESTAT0
45 P0.15/RI1/EINT2/AD1.5
46 P0.16/EINT0/MAT0.2/CAP0.2
47 P0.17/CAP1.2/SCK1/MAT1.2
48 P1.20/TRACESYNC
49 VBAT
50 VSS
51 VDD
52 P1.30/TMS
53 P0.18/CAP1.3/MISO1/MAT1.3
54 P0.19/MAT1.2/MOSI1/CAP1.2
55 P0.20/MAT1.3/SSEL1/EINT3
56 P1.29/TCK
57 RESET
58 P0.23/VBUS
59 VSSA
60 P1.28/TDI
61 XTAL2
62 XTAL1
63 VREF
64 P1.27/TDO

Generate the PWM waveform of 25% duty cycle at the PWM channel 3

```
#include <LPC214x.h>
int main()
{
PWM_Init();
while(1)
  {
   PWMMR3  =   500; // value which decides pulse ON time, ¼ of 2000
   PWMLER  = 0X08;
  }
}
```

```
void PWM_Init(void)
 {
   //P0.1 pin has second alternate function as PWM3 channel, so using PINSEL0 register
   // Select  P0.1 as PWM output ,  bits D2 & D3 are for P0.1
   PINSEL0 |=  0x00000008;

   //load the value to MR0 to fix the pulse rate
   PWMMR0 = 2000; // any other value, I could have taken

   //Configure PWM channel 3  as single edge type and enable the channel
   //bit D3 to select single edge(make it  0), bit D11 is for enabling PWM3 channel(make it 1)
   PWMPCR = 0x00000800;

   // enable PWM unit of LPC2148 and start the timer
   PWMTCR = 0x00000009;   // bit D3 = 1 (enable  PWM), bit D0=1 (start the timer)
 }
```

Calculation of duty cycle in the above program:

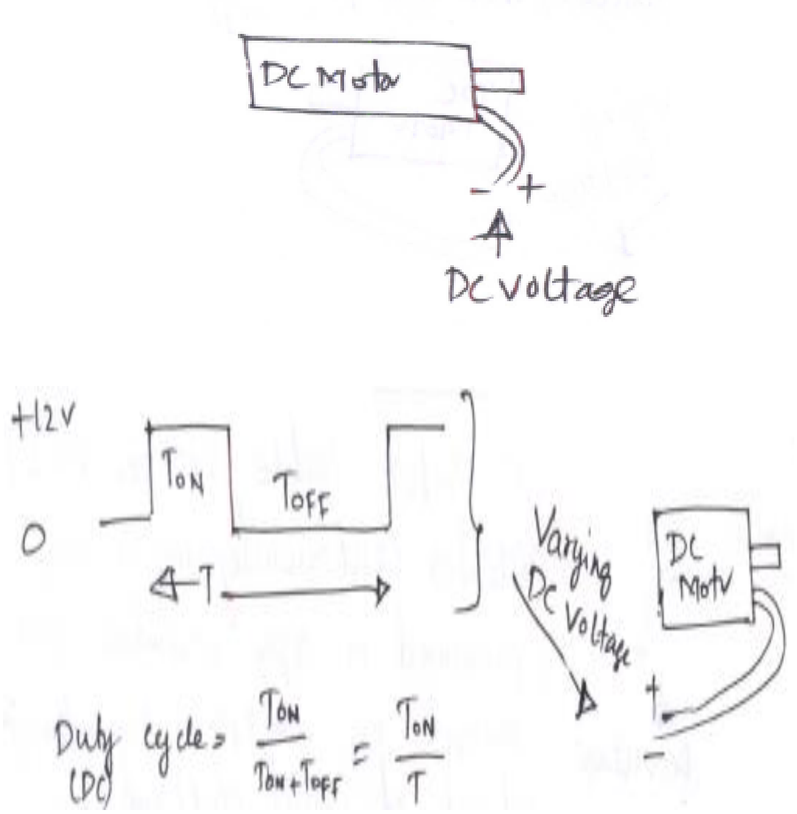
if PCLK is 15MHz, then T= 0.067µsec,

Pulse rate/Period, T = ON+OFF = 2000 * T

Puse width, Ton = 500 * T

Duty cycle = Ton/T = 500*T/2000*T = ¼= 0.25 x 100 = 25 %

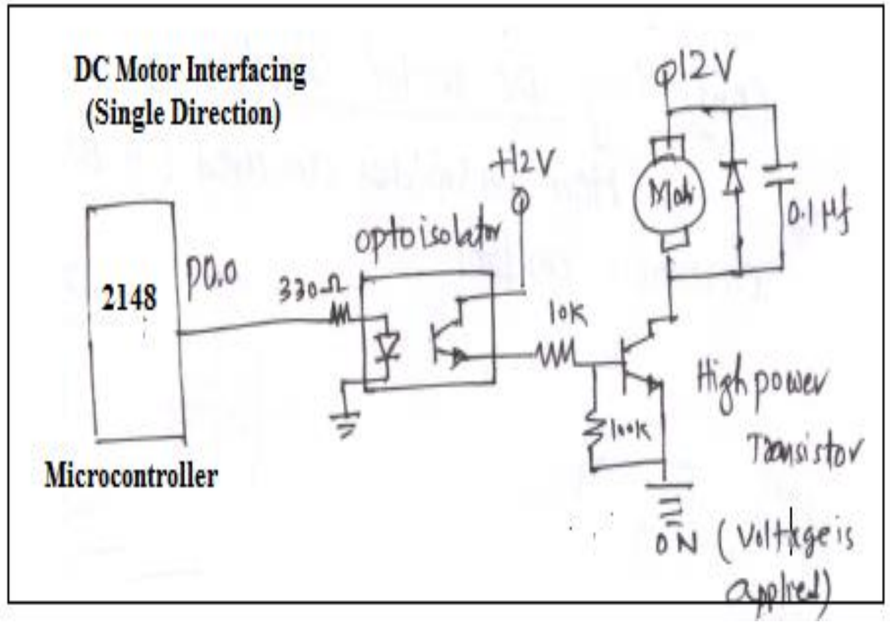
DC Motor Speed Control

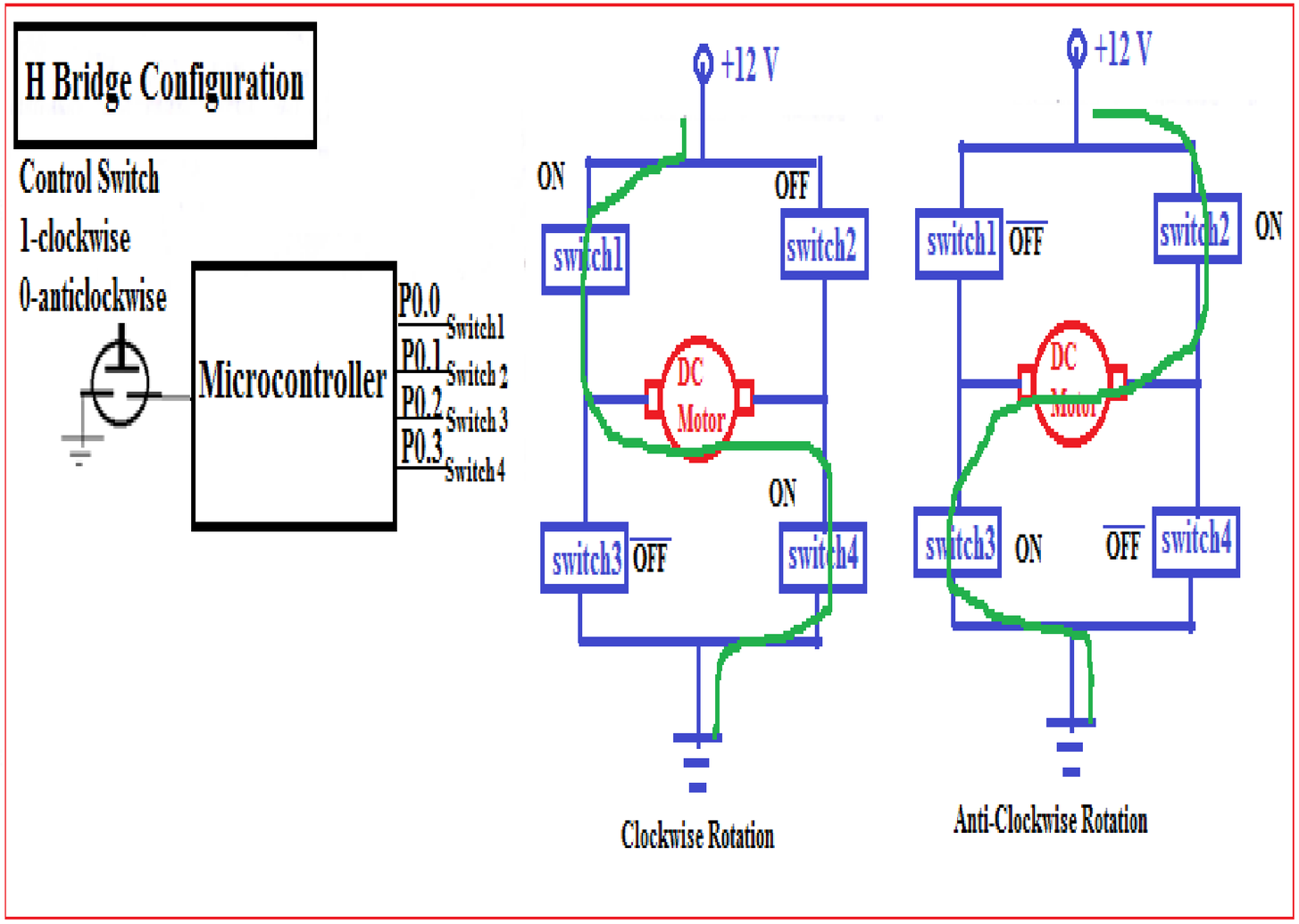


Effective DC Voltage Applied to DC Motor
= (Ton/T) x Supply voltage (say 12V)

If Ton = T, Full Power i.e 12V
If Ton = T/2, 50 % DC (1/2 Power)

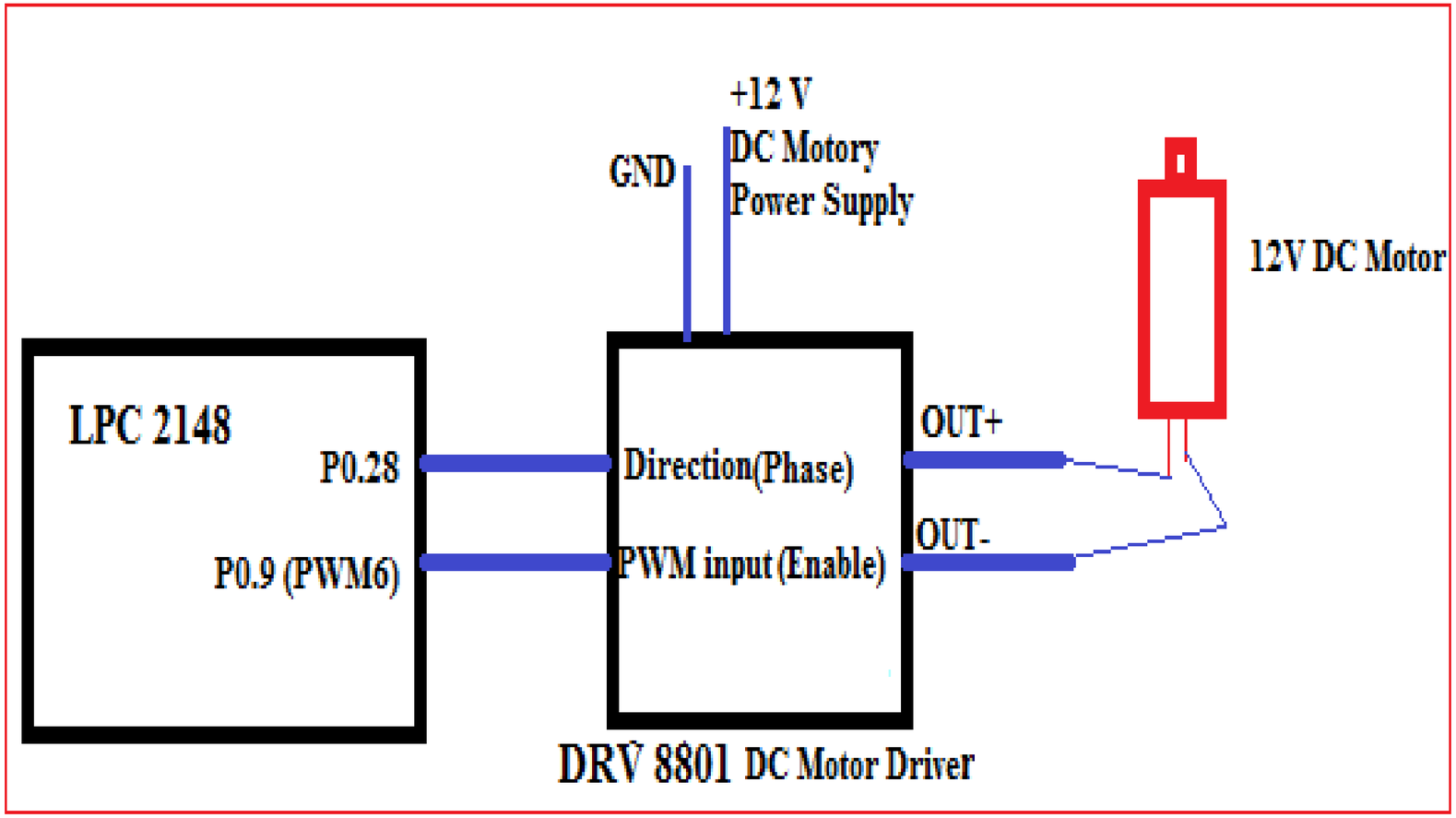
H Bridge arrangement To Drive DC Motor





Interface DC Motor to LPC 2148 and write an Embedded C program to generate PWM wave to control speed of DC motor.

```
//DC Motor Speed Control
//P0.28 - used for direction control
//P0.9  - used for speed,generated by PWM6
//duty cycle - 0 to 100 controlled by PWM,
```



```
#include <lpc214x.h>
void runDCMotor(int direction, int dutycycle) ;

int main()
{
    IO0DIR |= 1U << 28;     //set P0.28 as output pin
    PINSEL0 |= 2 << 18;    //select P0.9 as PWM6 (option 2)
    runDCMotor(2,10);    // run at 10% duty cycle
    while(1);  // do other jobs
}
```

```
void runDCMotor(int direction, int dutycycle)
{
    if (direction == 1)
        IO0SET = 1 << 28;   //set to 1, to choose anti-clockwise direction
    else
        IO0CLR = 1 << 28;  //set to 0, to choose clockwise direction
    PWMMR0 = 1000;       // set PULSE rate to value suitable for DC Motor operation
    PWMMR6 = (1000U*dutycycle)/100;  // set PULSE period
    PWMPCR = (1 << 14);   // enable PWM6 channel
    PWMTCR = 0x00000009;  // bit D3 = 1 (enable  PWM), bit D0=1 (start the timer)
    PWMLER = 0X70;  // load the new values to PWMMR0 and PWMMR6 registers
}
```

//Control the duty cycle by analog input fed from potentiometer.
//Duty cycle - 0 to 100 controlled by PWM, fed from Potentiometer connected to ADC

```
#include <lpc214x.h>
void runDCMotor(int direction,int dutycycle);
unsigned int adc(int no,int ch);  // refer UNIT 4 Notes, for this function implementation
int main()
{
    int dig_val;
    IO0DIR |= 0x00FF0000 | 1U << 30; // to set P0.16 to P0.23 as o/ps
    do{
        dig_val = adc(1,2) / 10;
        if(dig_val > 100) dig_val =100;
        runDCMotor(1,dig_val); // run at 10% duty cycle
    }
    while(1);
}
```