NVIC and PULSE WIDTH MODULATION
(REFER LPC 1678 MANUAL FOR CLEAR FIGURES)

Basic terminologies on Interrupts

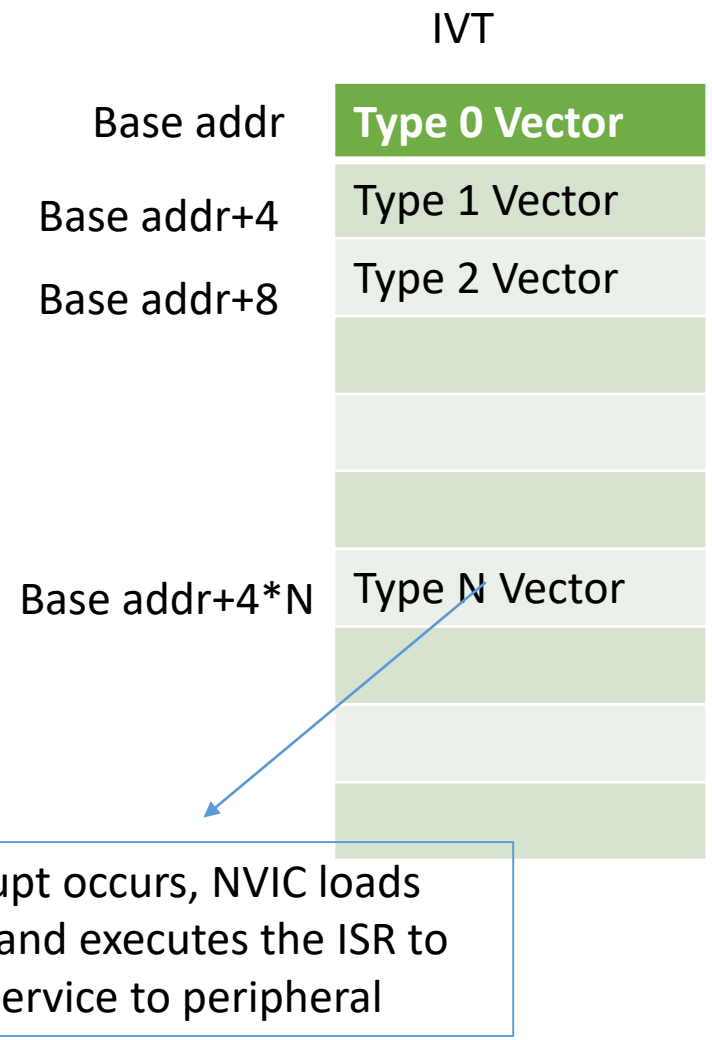
- Interrupt
- Interrupt Service Subroutine
- Interrupt Vector
- Interrupt Vector Table
- Priority of interrupts

Nested Vectored Interrupt Controller(NVIC)

- Controls system exceptions and peripheral interrupts
- In the LPC176x, the NVIC supports 35 vectored interrupts

- Each peripheral device may have one or more interrupt lines to the Vectored Interrupt Controller.
- Interrupt numbers relate to where entries are stored in the Interrupt vector table.

- Interrupt Vector – Is the address of Interrupt Service Subroutine (ISR)
- Interrupt vector of Interrupt Type N is stored at an offset N*4 from the base address of Interrupt Vector Table(IVT)
- If the peripheral device is enabled to generate Interrupt when some event



Nested Vectored Interrupt Controller(NVIC)

- If the peripheral device is enabled to generate Interrupt when some event occurs, the INTR request is sent to NVIC
- If NVIC is enabled to service the INTR request from the peripheral, it services the INTR request by executing ISR pertaining to the peripheral.(i.e Save the return address, Get the Interrupt Vector from IVT and load that address to PC. Upon completion of ISR execution resumes the calling function)

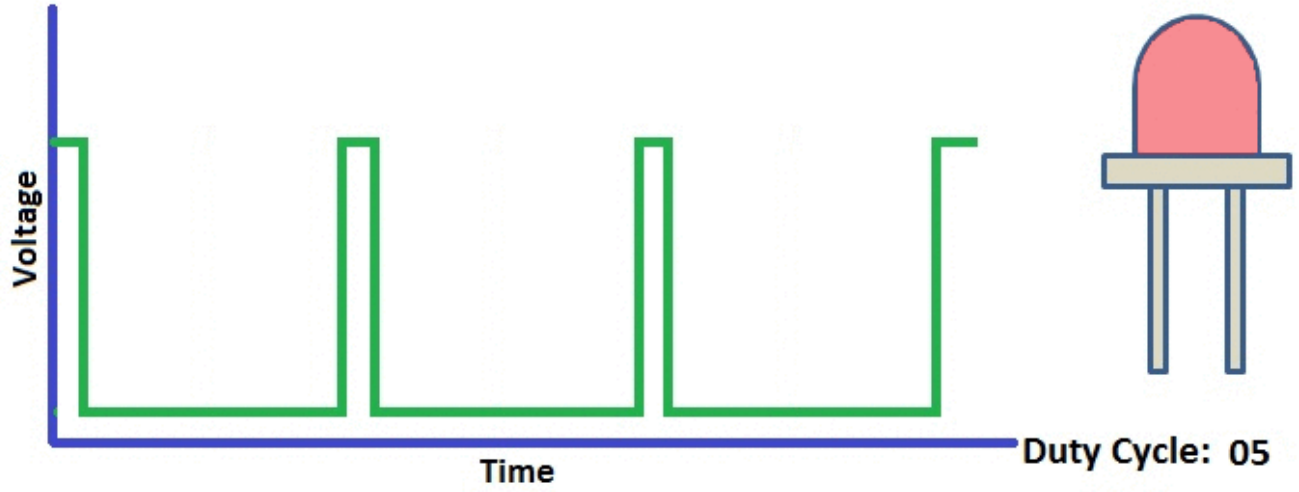
Nested Vectored Interrupt Controller(NVIC)

- In case of Timer, there are 6 INTR enable flags (4 in MCR and 2 in CCR. i.e 4 Match events and 2 Capture events can generate the Interrupt when the event occurs)
- When the event occurs the corresponding bit is set automatically in the IR register. This indicates the NVIC about the event
- If the NVIC is enabled to service the Timer Interrupt, it executes the corresponding ISR and gives the desired service to the Timer.
- In the ISR, clear the corresponding bit, by writing back 1.

- C statement to Enable the interrupt at NVIC , written in the main program.
  - Syntax: **NVIC_EnableIRQ**(Int_name**_IRQn**);// red colored content is //fixed/keyword
    - Ex:to enable the timer 0 interrupt
      - NVIC_EnableIRQ (TIMER0_IRQn);
      - To enable the PWM interrupt: NVIC_EnableIRQ(PWM1_IRQn);

- C statement to write the interrupt Handler:
  - Syntax: void INT_NAME_**IRQHandler()**
  - **{**
  - **}**
  - **Ex: void TIMER0_IRQHandler();**
    - void PWM1_IRQHandler();

PWM



Pavg = Pmax Ton/(Ton+Toff)

Types of PWM

- Single Edge PWM:
  - Two match registers are used to generate single edge controlled PWM output. PWMMR0 decides the cycle rate by resetting the count after match.
  - All single edge controlled PWM outputs go high at the beginning of a PWM cycle unless their match value is equal to 0.
  - Each PWM output will go low when its match value is reached. If no match occurs (i.e. the match value is greater than the PWM rate), the PWM output remains continuously high.
  - Ex: MR0=100, MR1=40, output at PWM1.1=?

Types of PWM contd..

- Double edge PWM
  - Both edges can be modulated/moved which changes the width of the pulse.
  - Pulse can be placed anywhere in the period.

  - Three match registers are used to generate single edge controlled PWM output. PWMMR0 decides the cycle rate by resetting the count after match.
  - Ex: PWM1.2 uses MR0, MR1 & MR2 registers

Features of LC1768 PWM

- Uses a 32-bit timer (features same as discussed before under timers)
- Supports 6 PWM outputs : P1.1,…P1.5,P1.6

The PWM is based on the standard Timer block and inherits all of its features, although only the PWM function is pinned out on the LPC176x/5x. The Timer is designed to count cycles of the peripheral clock (PCLK) and optionally generate interrupts or perform other actions when specified timer values occur, based on seven match registers. The PWM function is in addition to these features, and is based on match register events.

PWM timer diagram

- Ref Fig. 120 PWM block diagram in LPC 1768 reference manual

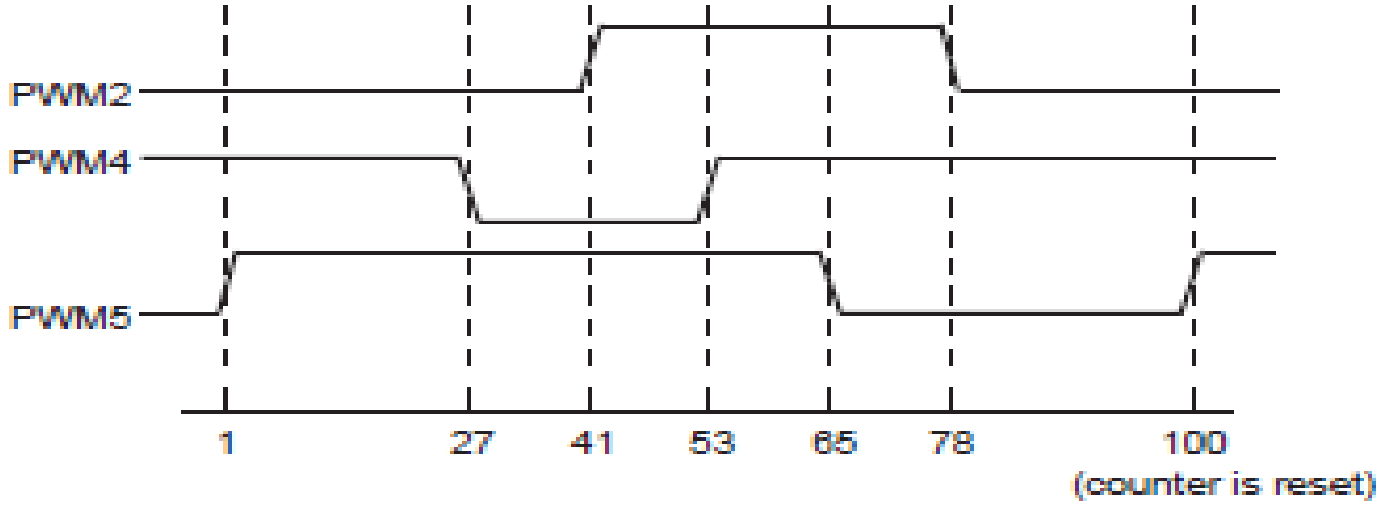
Single and double edge controlled PWM

Two match registers can be used to provide a single edge controlled PWM output. One match register (PWMMR0) controls the PWM cycle rate, by resetting the count upon match. The other match register controls the PWM edge position. Additional single edge controlled PWM outputs require only one match register each, since the repetition rate is the same for all PWM outputs. Multiple single edge controlled PWM outputs will all have a rising edge at the beginning of each PWM cycle, when an PWMMR0 match occurs.

Three match registers can be used to provide a PWM output with both edges controlled. Again, the PWMMR0 match register controls the PWM cycle rate. The other match registers control the two PWM edge positions. Additional double edge controlled PWM outputs require only two match registers each, since the repetition rate is the same for all PWM outputs.

With double edge controlled PWM outputs, specific match registers control the rising and falling edge of the output. This allows both positive going PWM pulses (when the rising edge occurs prior to the falling edge), and negative going PWM pulses (when the falling edge occurs prior to the rising edge).

Sample PWM waveforms



The waveforms show one PWM cycle and demonstrate PWM outputs under the following conditions:

The timer is configured for PWM mode (counter resets to 1).

Match 0 is configured to reset the timer/counter when a match event occurs.

Control bits PWMSEL2 and PWMSEL4 are set.

The Match register values are as follows:

MR0 = 100 (PWM rate)

MR1 = 41, MR2 = 78 (PWM2 output)

MR3 = 53, MR4 = 27 (PWM4 output)

MR5 = 65 (PWM5 output)

SFRs OF PWM

Table 446. PWM1 register map

Generic Name	Description
IR	Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of eight possible interrupt sources are pending.
TCR	Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.
TC	Timer Counter. The 32-bit TC is incremented every PR+1 cycles of PCLK. The TC is controlled through the TCR.
PR	Prescale Register. The TC is incremented every PR+1 cycles of PCLK.
PC	Prescale Counter. The 32-bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented. The PC is observable and controllable through the bus interface.
MCR	Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.

PCR	PWM Control Register. Enables PWM outputs and selects PWM channel types as either single edge or double edge controlled.	R/W
LER	Load Enable Register. Enables use of new PWM match values.	R/W
CTCR	Count Control Register. The CTCR selects between Timer and Counter mode, and in Counter mode selects the signal and edge(s) for counting.	R/W

MRO TO MR4 :MATCH REGISTERS
CRO TO CR1: CAPTURE REGISTERS

PWM IR

The PWM Interrupt Register consists of 9 INTR Flags (7Match,2Capture) If an interrupt is generated then the corresponding bit in the PWMIR will be high. Otherwise, the bit will be low. Writing a logic 1 to the corresponding IR bit will reset the interrupt.

Bit	Symbol	Description
0	PWMMR0 Interrupt	Interrupt flag for PWM match channel 0.
1	PWMMR1 Interrupt	Interrupt flag for PWM match channel 1.
2	PWMMR2 Interrupt	Interrupt flag for PWM match channel 2.
3	PWMMR3 Interrupt	Interrupt flag for PWM match channel 3.
4	PWMCAP0 Interrupt	Interrupt flag for capture input 0
5	PWMCAP1 Interrupt	Interrupt flag for capture input 1.
7:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
8	PWMMR4 Interrupt	Interrupt flag for PWM match channel 4.
9	PWMMR5 Interrupt	Interrupt flag for PWM match channel 5.
10	PWMMR6 Interrupt	Interrupt flag for PWM match channel 6.
31:11	-	Reserved, user software should not write ones to reserved bits.

SFRs OF PWM: TCR

The PWM Timer Control Register (PWMTCR) is used to control the operation of the PWM Timer Counter.

Bit 0	Counter Enable	When 1, PWM Timer Counter and Prescale Counter are enabled.
Bit 1	Counter Reset	When 1, PWM Timer Counter and Prescale Counter are reset on next positive edge of PCLK.
Bit 3	PWM Enable	When 1, PWM Mode is enabled. TC will reset to 1.

```
LPC_PWM2->TCR = ?  ; //RESET
LPC_PWM2->TCR = ?  ; //ENABLE COUNTER AND PWM
```

SFRs OF PWM: TCR

- Bit 0 – Counter Enable
  - This bit is used to Enable or Disable the PWM Timer and PWM Prescalar Counters
  - 0- Disable the Counters
  - 1- Enable the Counter incrementing.
- Bit 1 – Counter reset
  - This bit is used to clear the PWM Timer and PWM Prescalar Counter values.
  - 0- Do not Clear.
  - 1- The PWM Timer Counter and the PWM Prescale Counter are synchronously reset on the next positive edge of PCLK.

- Bit 3 – PWM Enable
  - Used to Enable or Disable the PWM Block.
  - 0- PWM Disabled
  - 1- PWM Enabled

PWM CTCR

Bit	Symbol	Value	Description
1:0	Counter/ Timer Mode	00	Timer Mode: the TC is incremented when the Prescale Counter matches the Prescale Register.
		01	Counter Mode: the TC is incremented on rising edges of the PCAP input selected by bits 3:2.
		10	Counter Mode: the TC is incremented on falling edges of the PCAP input selected by bits 3:2.
		11	Counter Mode: the TC is incremented on both edges of the PCAP input selected by bits 3:2.
3:2	Count Input Select		When bits 1:0 of this register are not 00, these bits select which PCAP pin which carries the signal used to increment the TC.
		00	PCAP1.0
		01	PCAP1.1 (Other combinations are reserved)
31:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.

PWM: PCR

Bit	Symbol	Value	Description
1:0	Unused		Unused, always zero.
2	PWMSEL2	1	Selects double edge controlled mode for the PWM2 output.
		0	Selects single edge controlled mode for PWM2.
3	PWMSEL3	1	Selects double edge controlled mode for the PWM3 output.
		0	Selects single edge controlled mode for PWM3.
4	PWMSEL4	1	Selects double edge controlled mode for the PWM4 output.
		0	Selects single edge controlled mode for PWM4.
5	PWMSEL5	1	Selects double edge controlled mode for the PWM5 output.
		0	Selects single edge controlled mode for PWM5.
6	PWMSEL6	1	Selects double edge controlled mode for the PWM6 output.
		0	Selects single edge controlled mode for PWM6.
8:7	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.

Bit	Symbol	Value	Description
9	PWMENA1	1	The PWM1 output enabled.
		0	The PWM1 output disabled
10	PWMENA2	1	The PWM2 output enabled.
		0	The PWM2 output disabled
11	PWMENA3	1	The PWM3 output enabled.
		0	The PWM3 output disabled
12	PWMENA4	1	The PWM4 output enabled.
		0	The PWM4 output disabled
13	PWMENA5	1	The PWM5 output enabled.
		0	The PWM5 output disabled
14	PWMENA6	1	The PWM6 output enabled.
		0	The PWM6 output disabled
31:15	Unused		Unused, always zero.

PWM:PCR

- **PWMSELx**
  This bit is used to select the single edged and double edge mode form PWMx (x:2-6)
  0- Single Edge mode for PWMx
  1- Double Edge Mode for PWMx.
- **PWMENAx**
  This bit is used to enable/disable the PWM output for PWMx(x:1-6)
  0- PWMx Disable.
  1- PWMx Enabled.

```
LPC_PWM2->PCR = ?  ; //TO ENABLE O/P AT PWM2
```

PWM :MCR

The PWM Match Control Register is used to specify what operations can be done when the value in a particular Match register equals the value in TC.

For each Match Register we have 3 options : Either generate an Interrupt , or Reset the TC , or Stop .. which stops the counters and disables PWM. Hence this register is divided into group of 3 bits. The first 3 bits are for Match Register 0 i.e PWMMR0 , next 3 for PWMMR1 , and so on.

- **1) Bit 0 :** Interrupt on PWM1MR0 Match – If set to 1 then it will generate an Interrupt else disable if set to 0.
- **2) Bit 1 :** Reset on PWM1MR0 Match – If set to 1 it will reset the PWM Timer Counter i.e. PWM1TC else disabled if set to 0.
- **3) Bit 2 :** Stop on PWM1MR0 Match – If this bit is set 1 then both PWM1TC and PWM1PC will be stopped & PWMTCR[0] is reset which disables the Counter.

PWM : MCR

Bit	Symbol	Value	Description
0	PWMMR0I	1	Interrupt on PWMMR0: an interrupt is generated when PWMMR0 matches the value in the PWMTC.
		0	This interrupt is disabled.
1	PWMMR0R	1	Reset on PWMMR0: the PWMTC will be reset if PWMMR0 matches it.
		0	This feature is disabled.
2	PWMMR0S	1	Stop on PWMMR0: the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR0 matches the PWMTC.
		0	This feature is disabled
3	PWMMR1I	1	Interrupt on PWMMR1: an interrupt is generated when PWMMR1 matches the value in the PWMTC.
		0	This interrupt is disabled.
4	PWMMR1R	1	Reset on PWMMR1: the PWMTC will be reset if PWMMR1 matches it.
		0	This feature is disabled.
5	PWMMR1S	1	Stop on PWMMR1: the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR1 matches the PWMTC.
		0	This feature is disabled.
6	PWMMR2I	1	Interrupt on PWMMR2: an interrupt is generated when PWMMR2 matches the value in the PWMTC.
		0	This interrupt is disabled.
7	PWMMR2R	1	Reset on PWMMR2: the PWMTC will be reset if PWMMR2 matches it.
		0	This feature is disabled.

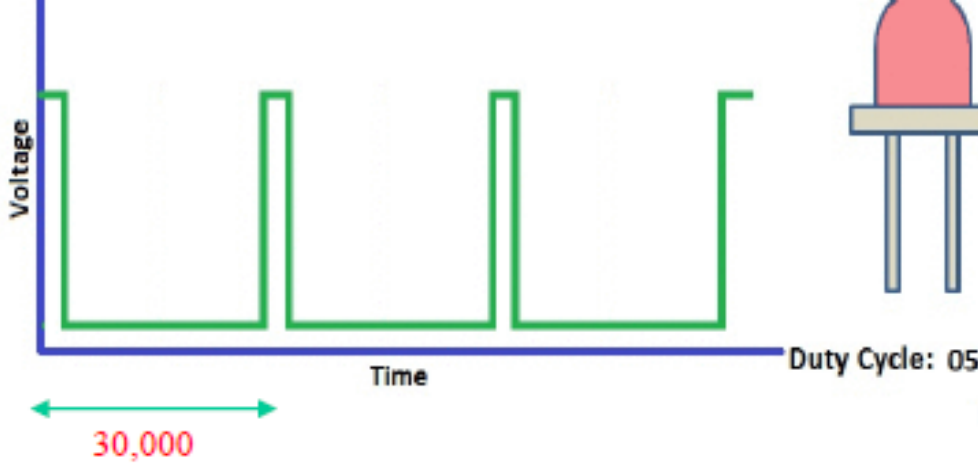
Bit	Symbol	Value	Description
8	PWMMR2S	1	Stop on PWMMR2: the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR2 matches the PWMTC.
		0	This feature is disabled
9	PWMMR3I	1	Interrupt on PWMMR3: an interrupt is generated when PWMMR3 matches the value in the PWMTC.
		0	This interrupt is disabled.
10	PWMMR3R	1	Reset on PWMMR3: the PWMTC will be reset if PWMMR3 matches it.
		0	This feature is disabled
11	PWMMR3S	1	Stop on PWMMR3: The PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR3 matches the PWMTC.
		0	This feature is disabled
12	PWMMR4I	1	Interrupt on PWMMR4: An interrupt is generated when PWMMR4 matches the value in the PWMTC.
		0	This interrupt is disabled.
13	PWMMR4R	1	Reset on PWMMR4: the PWMTC will be reset if PWMMR4 matches it.
		0	This feature is disabled.
14	PWMMR4S	1	Stop on PWMMR4: the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR4 matches the PWMTC.
		0	This feature is disabled
15	PWMMR5I	1	Interrupt on PWMMR5: An interrupt is generated when PWMMR5 matches the value in the PWMTC.
		0	This interrupt is disabled.
16	PWMMR5R	1	Reset on PWMMR5: the PWMTC will be reset if PWMMR5 matches it.
		0	This feature is disabled.
17	PWMMR5S	1	Stop on PWMMR5: the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR5 matches the PWMTC.
		0	This feature is disabled
18	PWMMR6I	1	Interrupt on PWMMR6: an interrupt is generated when PWMMR6 matches the value in the PWMTC.
		0	This interrupt is disabled.
19	PWMMR6R	1	Reset on PWMMR6: the PWMTC will be reset if PWMMR6 matches it.
		0	This feature is disabled.
20	PWMMR6S	1	Stop on PWMMR6: the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR6 matches the PWMTC.
31:21	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.

PWM LER

Bit	Symbol	Description
0	Enable PWM Match 0 Latch	Writing a one to this bit allows the last value written to the PWM Match 0 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
1	Enable PWM Match 1 Latch	Writing a one to this bit allows the last value written to the PWM Match 1 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
2	Enable PWM Match 2 Latch	Writing a one to this bit allows the last value written to the PWM Match 2 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
3	Enable PWM Match 3 Latch	Writing a one to this bit allows the last value written to the PWM Match 3 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
4	Enable PWM Match 4 Latch	Writing a one to this bit allows the last value written to the PWM Match 4 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
5	Enable PWM Match 5 Latch	Writing a one to this bit allows the last value written to the PWM Match 5 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
6	Enable PWM Match 6 Latch	Writing a one to this bit allows the last value written to the PWM Match 6 register to be become effective when the timer is next reset by a PWM Match event. See Section 24.6.4 "PWM Match Control Register (PWM1MCR - 0x4001 8014)".
31:7	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.

Write PWM program to change the intensity of LED connected to (PWM1.4)(P1.23, Funct 2) continuously using single edge controlled

```
unsigned long int i;
unsigned char flag,flag1;
 int main(void)
{
                SystemInit();
                SystemCoreClockUpdate();
                pwm_init();
                while(1);
                 }
 void pwm_init(void)
{               LPC_PINCON->PINSEL3 = 2<<14;                        //pwm1.4 is selecte
                LPC_PWM1->PR  = 0x00000000;     //Count frequency : Fpclk
                LPC_PWM1->PCR = 0x00001000;     //select PWM1 single edge
                LPC_PWM1->MCR = 0x00000003;     //Reset and interrupt on PWMMR0
                LPC_PWM1->MR0 = 30000;         //setup match register 0 count
                LPC_PWM1->MR4 = 50;     //setup match register MR4
                LPC_PWM1->LER = 0x0000007F;     //enable shadow copy register
                LPC_PWM1->TCR = 0x00000002;     //RESET TC and PC
                LPC_PWM1->TCR = 0x00000009;     //enable PWM and counter
                NVIC_EnableIRQ(PWM1_IRQn);
                return;
}
```



```
void PWM1_IRQHandler(void)
{
        LPC_PWM1->IR = 0x01; //clear the interrupts
        if(flag == 0x00)
  {
                LPC_PWM1->MR4 += 50;
                LPC_PWM1->LER = 0x0000007F;
                if(LPC_PWM1->MR4 > 29900)
                {
                 flag = 0xff;
        LPC_PWM1->LER = 0x000000FF;
                }
        }
  else if(flag == 0xff)
  {
                LPC_PWM1->MR4 -= 50;
                LPC_PWM1->LER = 0x0000007F;
                if(LPC_PWM1->MR4 < 100)
                {
                 flag  = 0x00;

                }
        }
}
```