Bulgarian Academy of Sciences. Space Research and Technology Institute.
Aerospace Research in Bulgaria. 29, 2017, Sofia

GENERATING A PULSE-WIDTH MODULATED SIGNAL BY MEANS OF TIMER MODULE INTERRUPT OF PIC18F2550 MCU

Konstantin Metodiev

Space Research and Technology Institute – Bulgarian Academy of Sciences
e-mail: komet@space.bas.bg

Abstract

The article hereby examines an approach towards pulse-width modulated (PWM) signal generation by means of PIC18F2550 microcontroller unit (MCU). The proposed technique employs the so-called internal interrupt by timer module. This solution may come into use in process automation as a terminal device, for instance putting in motion a servo motor. Although the MCU provides a hardware PWM module, it is dependant of the crystal oscillator frequency. Most of the time, it is difficult to generate a PWM signal at low frequencies. The proposed solution is shown to be versatile enough and suitable to actuate servomotors widely used in RC hobby activities.

Special attention is given to the MCU software peculiarities. Additional computer simulation has also been made. The used software was MikroC Pro for PIC and Proteus VMS. The proposed solution has been shown to operate with sufficient precision. The source code is also included in the present article.

1. Introduction

As the abstract suggests, the article topic is generating a PWM signal at low frequencies, precisely at 50 Hz. Most of the servomotors used by RC hobbyists nowadays operate at this frequency. The pulse width varies within 1 up to 2 ms. It is found to be difficult for the hardware to generate a PWM signal with aforementioned parameters. For instance, if the instruction clock is provided by a high-speed oscillator at 4 MHz, the PWM signal frequency will vary within 244 Hz to 1 MHz [1], which is quite above the designated value. In fact, the required PWM frequency of 50 Hz is achievable by means of the internal oscillator. The oscillator is capable of providing the MCU with a range of clock frequencies from 31 kHz to 4 MHz [2]. This clock source however is said to be quite temperature dependent and unstable. Given these impediments, developing an alternative technique to implement a PWM signal appears to be necessary.

The main purpose of the present article is to demonstrate the ability of a simple solution to generate a PWM signal at 50 Hz by triggering an internal interrupt from a Timer 1 module. In order to test the source code, a simulation has

been carried out by means of Proteus VMS. A real experiment is also implemented so as to make a comparison.

2. Materials and methods

The electronic circuit consists of minimum required parts that make the MCU running according to Fig. 1, i.e. a high-speed crystal of 20 MHz and 2 capacitors of 15 pF each. These are said to provide the MCU with stable instruction clock of 5 MHz [2]. In addition to it, three servomotors are connected to the MCU for verification reasons.

The program algorithm is following. In order to make sure interrupt occurs every 128 μs, Timer 1 module is initially set to 0xFD80. This value is repeatedly reset each time Timer 1 overflows. The PWM period is obtained approximately by getting the product of 156 interrupts times 128 μs equals 19 968 μs. This value is derived as close as possible for given oscillator clock of 20 MHz and prescaler value of 1:1.

The number of interrupts is committed to a counter. As soon as the counter reaches value of 156, start pulse edge is triggered. The edge is rising for all attached servomotors. The falling edge is triggered afterwards depending upon what amount each motor has been assigned to. For instance, falling edge might occur after 12 interrupts times 128 μs each or 1536 μs in total. This value denotes the pulse width.

Desired time may be calculated by means of formula:

$$T = \left(0xFFFF + 1 - TMR1\right)\frac{4}{F_{OSC}} Prescaler, \quad [s] \tag{1}$$

where TMR1 is initial Timer 1 value, hex, and F_{OSC} is crystal oscillator frequency, Hz. In current case study, TMR1 = 0xFD80, F_{OSC} = 20E+06 Hz, Prescaler = 1. Hence, for interrupt period it yields T = 128 μs.

The proposed circuit has been put to the test as follows. Three servomotors are connected to the MCU. Each of them is controlled separately. Therefore, each servo is expected to rotate at different angle according to duty cycle value of the fed PWM signal.

3. Results

Having carried out a simulation by means of Proteus VMS, the obtained results are depicted in Fig. 1. Each servo cycles through distinct angular positions in accordance with set value of variables motor 1, 2, 3 in function main, Appendix 1. This value accounts for how many interrupts (128 μs each) have been triggered. A delay of 500 ms is set between cycles so that each servo has enough time to complete its stroke. All servos operate the way they are expected.

In addition, a real experiment has been carried out by means of HXT 900 Micro Servo 1.6 kg/0.12 s/9 g. There is a full agreement between results obtained by both physical and simulated experiments.

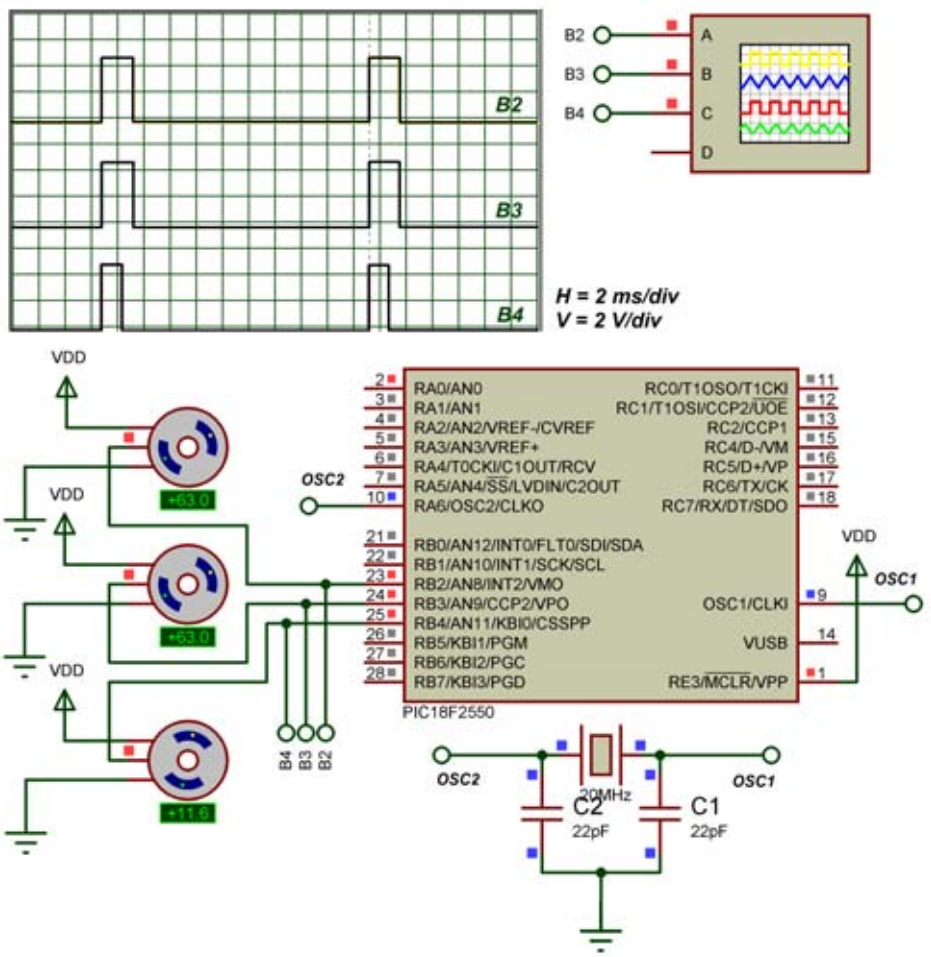

Fig. 1. Project simulation in Proteus VMS

4. Discussion

The described test case generates a PWM signal at 50 Hz. The signal at such a frequency is widely used in remotely controlled vehicles for hobbyists.

The proposed technique might be used in designing supplementary actuating devices for RC aircraft. In addition, this report might be found useful by developers who are less experienced in the interrupt technique.

References

1. PIC PWM Calculator & Code Generator. http://www.micro-examples.com/public/microex-navig/doc/097-pwm-calculator.html (date accessed 10 December 2017).
2. PIC18F2455/2550/4455/4550 Data Sheet, Microchip Technology Inc., 2006.

Appendix: Source code, MikroC Pro for PIC v.6.6.3

```
unsigned short counter = 0, motor1 = 0, motor2 = 0, motor3 = 0;

//Timer1: Prescaler 1:1; TMR1 Preload = 64896; Actual
//Interrupt Time: 128 us
void InitTimer1(void) {

     T1CON = 0x01;
     TMR1IF_bit = 0;
     TMR1H = 0xFD;
     TMR1L = 0x80;
     TMR1IE_bit = 1;
     INTCON = 0xC0;

     return;
}

void hereGoesMyISR(void) {
     if (TMR1IF_bit == 1) {       //Check if Timer1 has
//overflowed
        TMR1IF_bit = 0;        //Reset Timer1 flag
        TMR1H = 0xFD;
        TMR1L = 0x80;

        counter++;           //Increase counter by 1
        //156 interrupts * 128 us = 19968 usec
        if (counter == 156) { //Check if interrupt was
//triggered 156 times
           PORTB.F2 = 1;      //Start pulse to servo motor 1
           PORTB.F3 = 1;      //Start pulse to servo motor 2
           PORTB.F4 = 1;      //Start pulse to servo motor 3
           counter = 0;       //Reset counter
        }
        if (counter == motor1) //Check if time to end left
//servo pulse
           PORTB.F2 = 0;      //End pulse to left servo motor
        if (counter == motor2) //Check if time to end right
//servo pulse
           PORTB.F3 = 0;     //End pulse to right servo motor
```

```
        if (counter == motor3) //Check if time to end right
servo pulse
           PORTB.F4 = 0;     //End pulse to right servo motor
     }
     return;
}

void interrupt(void) {
     hereGoesMyISR();
}

void main() {

     CMCON = 0x07;  //disables comparators
     ADCON1 = 0x0F; //disables analogue functions
     GIE_bit = 1;   //Enable global interrupts
     TRISB.F2 = 0;  //Set up Port B2 as an output
     TRISB.F3 = 0;  //Set up Port B3 as an output
     TRISB.F4 = 0;  //Set up Port B4 as an output
     TRISA.F0 = 0;  //Set up Port A0 as an output
     PORTA.F0 = 1;

     InitTimer1();

     while(1) {           // Set up infinite loop
          motor1 = 6;    // Falling edge after 6 interrupts *
//128 us each = 768 us
          motor2 = 6;
          motor3 = 6;
          Delay_ms(500);
          motor1 = 18;   // Falling edge after 16 interrupts
//* 128 us each = 2048 us
          motor2 = 18;
          motor3 = 12;   // Falling edge after 12 interrupts
//* 128 us each = 1536 us
          Delay_ms(500);
     }
     return;
}
```

ГЕНЕРИРАНЕ НА ШИРОЧИННОИМПУЛСНО МОДУЛИРАН СИГНАЛ ЧРЕЗ ПРЕКЪСВАНЕ ОТ ТАЙМЕР НА МИКРОКОНТРОЛЕР PIC18F2550

К. Методиев

Резюме

В настоящия доклад се разглежда подход за генериране на широчинноимпулсно модулиран (ШИМ) сигнал посредством микроконтролер PIC18F2550. Предложената техника използва т. нар. вътрешно прекъсване от таймер. Това решение може да се използва в автоматизиран процес, като изпълнително звено, например за задвижване на сервомотор. Въпреки че контролерът разполага с модул за генериране на ШИМ сигнал, този модул е зависим от честотата на кристалния осцилатор. В повечето случаи е трудно да се генерира ШИМ сигнал на ниски честоти. Показано е, че предложеното решение е гъвкаво и подходящо за задвижване на сервомотори, широко използвани от ентусиасти в радиоуправляеми модели.

Специално внимание е отделено на особеностите на програмата за микроконтролера. Допълнително е направена компютърна симулация. Използван е софтуер MikroC Pro for PIC и Proteus VMS. Показано е, че предложеното решение функционира със задоволителна точност. Кодът на програмата също така е публикуван в настоящата статия.