Processor PIC18F46K22 Applied as DTMF Generator

Radek NOVAK

Department of Telecommunications, Faculty of Electrical Engineering and Computer Science, VSB–Technical University of Ostrava, 17. listopadu 15/2172, 708 33 Ostrava, Czech Republic

radek.novak@vsb.cz

DOI: 10.15598/aeee.v13i4.1471

Abstract. *Submitted article describes an application of microcontroller PIC18F46K22 for generating DTMF (Dual Tone Multi-Frequency) signal. Used microcontroller is optimal for this application - it has timers with compare facilities, sufficient number of PWMs, powerful instruction set (extended - in comparison to classic PIC16xxx microcontrollers), etc. problematic of article relates to more branches - electronics, filters, signal generators, microcontrollers, telecommunications, programming. An important purpose was to create an experimental platform based on powerful microcontroller enabling implementation additional functions. There are together three different solutions of DTMF generating are described in article, they are compared as to the demands on external hardware and machine time of microcontroller. The first solution is based on sinus table, truncated onto 32 values. This solution uses 8-bit PORTD for value intended to external D/A converter. The second one solution exploits compare facilities of PIC18F46K22, only two pins of microcontroller are needed for described application. And the third solution is similar to the second, difference consists in exploiting of PWM microcontroller hardware. Presented solutions were created in context of works on dotation program "Support of Science and Research in Moravia-Silesia Region".*

Keywords

Digital to analog converter, DTMF decoder MT8870DE, DTMF generator, low-pass filter, microcontroller, operating amplifier, PIC18F46K22, PLL, Pulse Width Modulation, resistance ladder R-2R.

1. Introduction

A technical task generating of DTMF signal was surely realized on many families of microcontrollers [8], in the family PIC18 it is innovative. This article presents three solutions how to solve DTMF generator with the PIC18F46K22 microcontroller:

- Generating DTMF signal with using sinus table.
- Generating two discrete signals using two 16-bit timers having compared facilities.
- Generating two discrete signals using two PWMs.

Generally known assignment of frequencies to a keyboard is in Fig. 1. DTMF signal is generated only

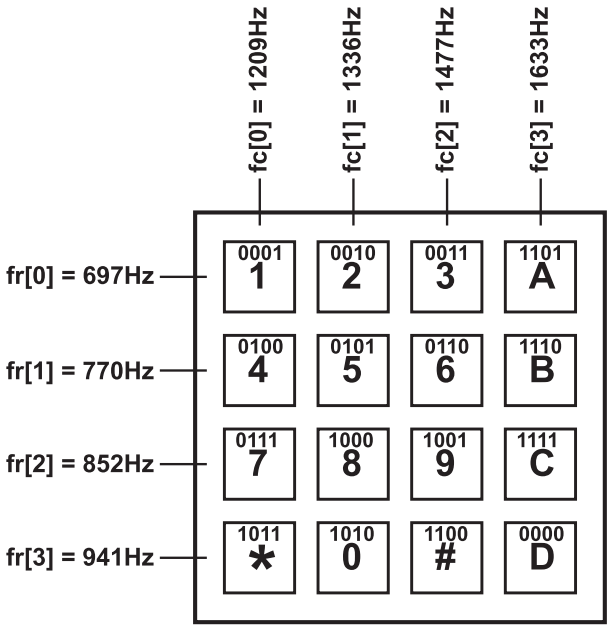

Fig. 1: Keyboard, row and column frequencies.

if some key is pressed, and this signal is summation of appropriate row frequency and column frequency [6], [7]. So for example if key "7" is pressed then sinus 852 Hz is summed with sinus 1209 Hz, their sum is DTMF signal.

There are also adequate 4-bits combinations on every key in Fig. 1. For example button "7" has adequate combination 0111. These 4-bit combinations are created by a DTMF decoder, if it is connected to DTMF signal, and if DTMF signal has regular shape. Concretely DTMF decoder of type MT8870DE (Zarlink Semiconductor) was used. Specialized DTMF generators (e.g. MV5089, Zarlink) are produced, but main motivation for mentioned three microcontroller solutions was to explore the PIC18F46K22 facilities for creating DTMF signal and its rest compute power for intended more functions.

2. Generating DTMF Signal Using Sinus Table

This solution uses crystal 18.432 MHz and internal PLL, so final internal frequency is
$4 \cdot 18.432 = 73.728$ MHz, and machine cycle T_{mc} is 54.3 ns [1]. All 32 values of sinus function are contained in field "s" (values: 7F... middle, FFh... high peak, 00h... down peak), see Fig. 2.

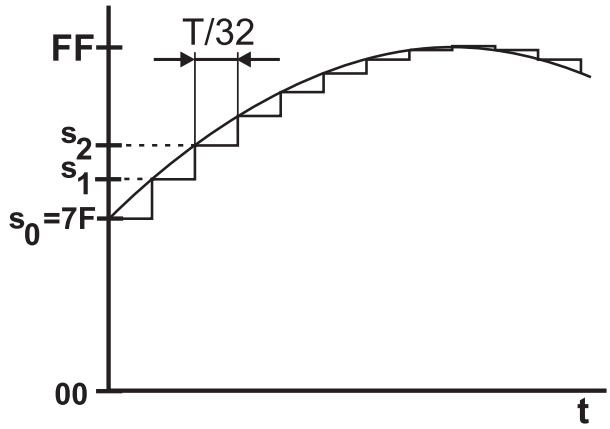

Fig. 2: Approximation of sinus function by 32 values.

The period of sinus function is divided into 32 elements and appropriate time interval is measured by appropriate timer. Let us take for example pressing of key "7". Row frequency is 852 Hz [7], period is $T_r = 1/852 = 1174$ µs. Then element $T_r/32 = 36.7$ µs. For this element $T_r/32$ measuring is used Timer1 (16 bit length), it must overflow every 36.7 µs and its appropriate pre-set is:

$$p_r = 2^{16} - \frac{\frac{T_r}{32}}{T_{mc}} = 64859.9 = FD5Chex. \qquad (1)$$

Column frequency for key "7" is $f_c = 1209$ Hz [7], for measuring appropriate $T_c/32$ is used Timer3, its pre-set proceeds 65059.6 = FE24hex. Now sinus table is present in RAM only once – as field s[]. So for work with field s[] must be used two indexes, for row frequency it is index "ri", for column frequency index "ci". If some time element $T_r/32$ or $T_c/32$ has elapsed, then interrupt service routine is invoked and new actual value of sum row sinus and column sinus is calculated, see Fig. 3. time moment t_m. Because both sinus signals are realized in 8 bits (range 00hex to FFhex), and used PORTD is also of 8 bit length and so it can overflow. Maximal value of sum is FF + FF = 1FE (510dec). For this reason result of sum is contained in 16-bit variable "*vuint*", that is rounded and then divided by 2. So in final LSB contains value of sum (i.e. DTMF signal), MSB has value 0. This low byte is sent through PORTD to external D/A converter, resistance ladder R-2R is used [3]. Proper part of interrupt service routine follows:

Algorithm 1

Require: rem "rs", "cs" were actualized in main(), for example. They have actual values rs = s[9] and cs = s[29].

```
1: vuint = rs + cs;
   //adding row_sinus + column_sinus;
2: ++vuint; //rounding
3: vuint »= 1; //dividing by 2
4: PORTD = vuint; //writing to PORTD
```

Designer working with this system has chance for experiments how to filter signal, if filter is realized by Sallen–Key low pass filter. For this low pass filter valid formulas [3]:

$$R_{1,2} = \frac{a_1 C_2 \pm \sqrt{a_1^2 C_2^2 - 4 b_1 C_1 C_2}}{4\pi f_c C_1 C_2}. \qquad (2)$$

For considered filter parameters a_1, b_1 have values [3]:

$$a_1 = \sqrt{2} = 1.414; \; b_1 = 1. \qquad (3)$$

Capacitors C_1, C_2 are coupled with parameters a_1, b_1 by condition :

$$\frac{C_2}{C_1} > \frac{4 b_1}{a_1^2} \text{ i.e. in our case } \frac{C_2}{C_1} > 2. \qquad (4)$$

There are voted these values in described application: $C_1 = 1$ nF, $C_2 = 10$ nF, $f_c = 2000$ Hz. Now we can calculate resistors R_1, R_2 using formula Eq. (2), it leads to values $R_1 = 106.6$ kΩ, $R_2 = 5.9$ kΩ. There is all the electrical scheme of this solution in Fig. 4.

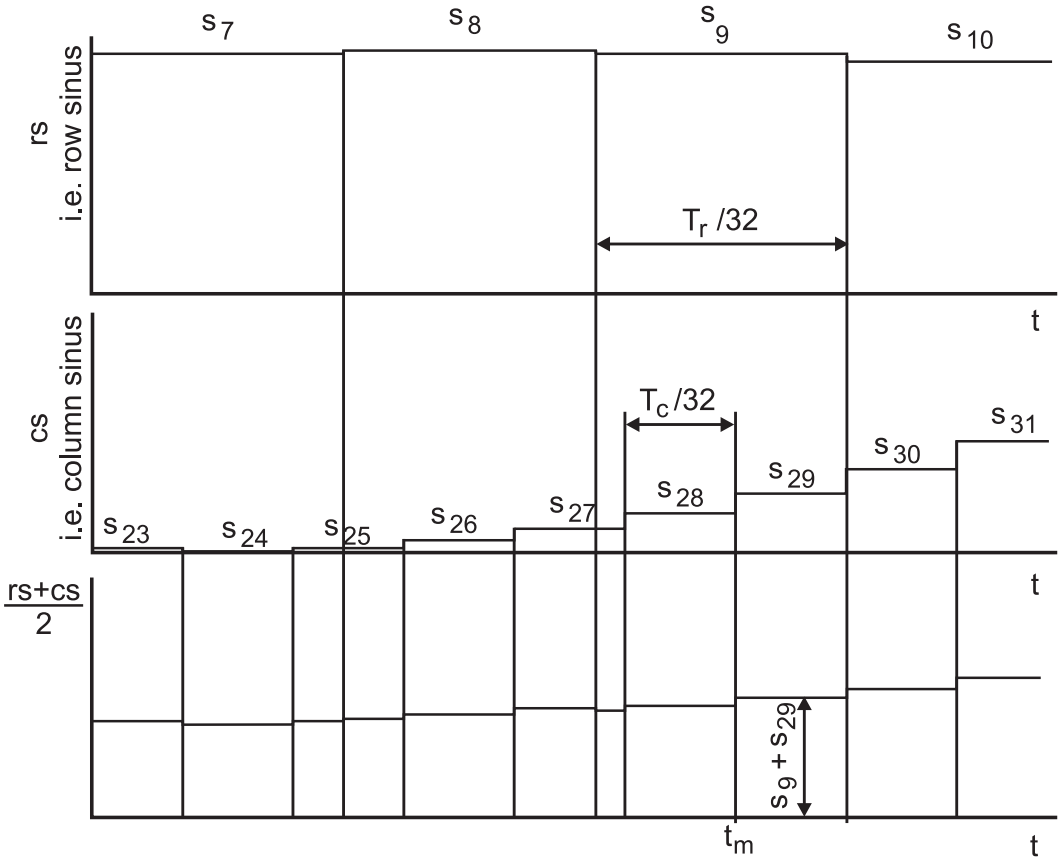

Fig. 3: Adding of row and column sinus.

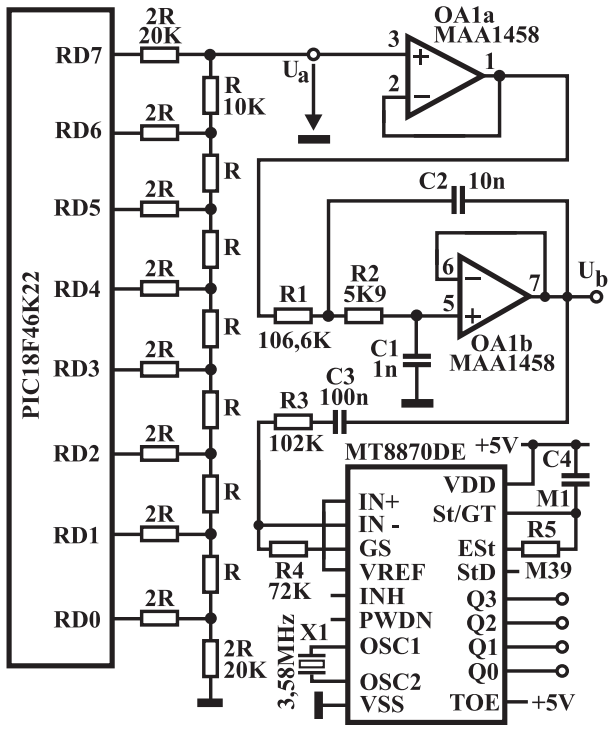

Fig. 4: Electrical scheme of solution using sinus table.

3. Generating DTMF Signal Using Compare Facilities of Timers

Idea of this solution is consequential. Not sinus, but only rectangle signal having proper period will be generated in digital output pin RC2 for row signal [6]. Analogously on other output pin RB5 will be generated rectangle signal for column signal. An operating amplifier adds these two signals, and this signal is connected to input of low pass filter. There is used identical low pass filter that was described in chapter 2. here. Counters/Timers CT1 and CT3 (both 16-bit) contained in PIC18F46K22 have compare facilities [1]. It means, that at moment of identity with predefined level an interrupt can be invoked and some output pin is set or reset (if it was configured). CT1 affects output pin RC2, CT3 affects pin RB5. Unfortunately toggle output pin facility is not implemented in PIC18F46K22, only setting or resetting. Control bit determining this facility is b_0 in SFR CCP1CON for CT1 (and b_0 in SFR CCP3CON for CT3). Value $b_0 = 0$ means setting output pin, value 1 means resetting. So if just identity of timer with some level is detected, then output pin is actualized and interrupt service routine is invoked. In this routine is just the

right moment for inverting control bit b_0 – so in future identity an opposite value of output pin will be created. There is also actualized (incremented) level in interrupt service routine. It is clear, that time difference between edges must have $T/2$ value. Situation is shown in Fig. 5. Timer CT1 (ev. CT3) is incremented about 1 periodically in every machine cycle. The machine cycle in PIC18F46K22 is equal to four periods of oscillator. The oscillator works with frequency FOSC, in described application it is 18432 kHz. So all these facts lead to values of increments for levels [5]:

Algorithm 2 # define FOSC 18432000

```
unsigned int increments_CT1[]=
{ (0.5·1.0/697.0)/(4.0/FOSC),
  (0.5·1.0/770.0)/(4.0/FOSC),
  (0.5·1.0/852.0)/(4.0/FOSC),
  (0.5·1.0/941.0)/(4.0/FOSC) }
```

Shown values serves for CT1, it is generating digital row signal U_r. Analogical values serves for CT3, that is generating digital column signal U_c. Let us explain matter on CT1 and signal U_r. Decisive is moment, when CT1 reaches of compare level (compare level is present in compare registers CCPR1H, CCPR1L). At this moment output pin RC2 automatically takes value determined by control bit (bit b_0 of register CCP1CON). Also an interrupt service routine is invoked automatically, there is its body:

Algorithm 3 Interrupt service routine.

```
if(CCP1IF&&CCP1IE)
{CCP1IF=0; //clearing of interrupt flag bit.
//actualization of compare level_ CT1.
level_CT1 ± increment_CT1;
//copy of 16-bit var. "level_ CT1" to compare registers.
CCPR1H = ((unsigned char*)(&level_CT1))[1];
CCPR1L = ((unsigned char*)(&level_CT1))[0];
//inverting of control bit, for opposite value in future.
CCP1CON = 0b00000001;}
```

Compare level is actualized (incremented) about increment_CT1. It is clear, that "increment_CT1" has some value from field "increments_CT1", mentioned above. Then the control bit must be inverted, for reaching an opposite value of output pin RC2 in future moment of equality.

Operating amplifier OA1a is in known summing connection [4], its input signals are U_r, U_c and –15 V.

Summator's output U_s is determined by next formula [4]:

$$U_s = -0.5U_r - 0.5U_c + 2.5. \quad (5)$$

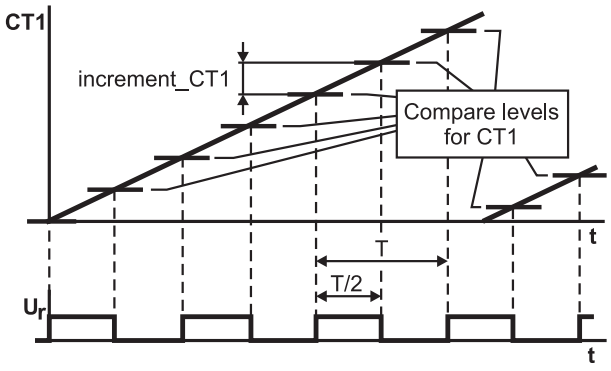

Fig. 5: Using timer CT_1 having compare facilities.

Proper signals are drawn in next figure. Filter produces output signal U_f (i.e. DTMF signal) that is connected to DTMF decoder MT8870DE. If DTMF signal U_f is generated properly, then the MT8870DE creates 4-bit information in pins Q3–Q0, that is adequate to pressed button on keyboard, concrete values see Fig. 1. To pins Q3–Q0 can be connected LEDs for visual control. Described solutions using compare facilities needs much less machine time than the first solution.

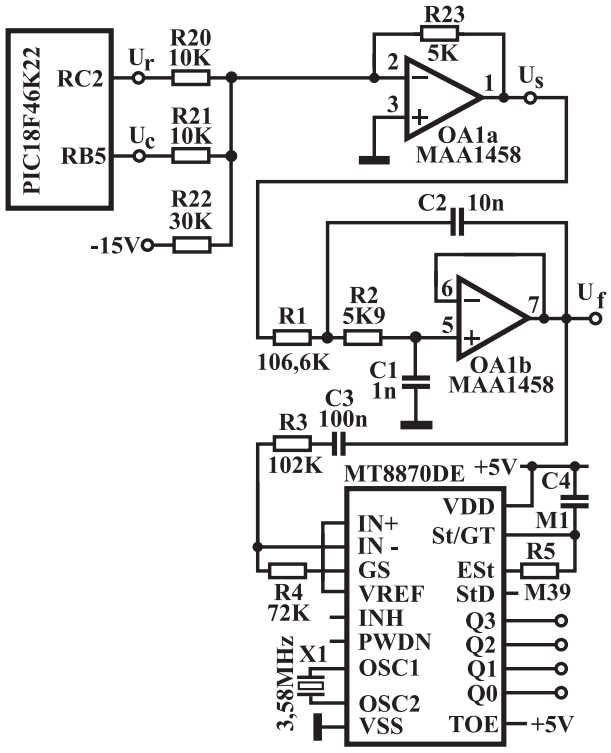

Fig. 6: Electrical scheme for solution using compare facilities of CT1, CT3, ev. using PWMs.

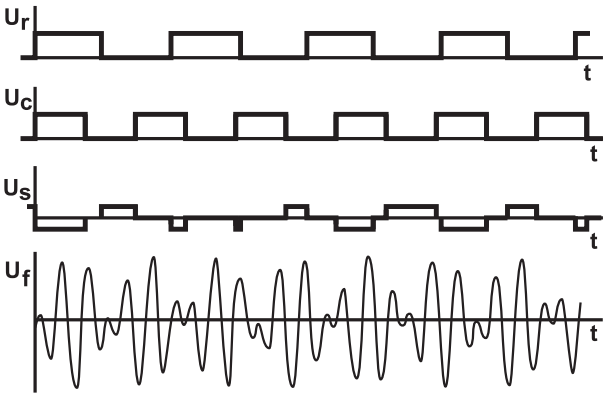

Fig. 7: Electrical signals of circuit drawn in Fig. 6.

4. Generating DTMF Signal Using PWM Outputs

This solution needs practically no machine time for generating DTMF signal, generating is automatic. External hardware is total the same, shown in Fig. 6, also electrical signals (Fig. 7) are identical. The difference consists inside the microcontroller – its PWM hardware is used [1], [2], [6]. The PIC18F46K22 has 5 PWMs, there are used PWM1 (output pin RC2) and PWM3 (output pin RB5) in my application. Period of PWM is specified in [1] by formula:

$$Period = (PR2 + 1) \cdot 4T_{OSC} \cdot PV, \tag{6}$$

where $PR2$ is period register 2, T_{OSC} is period of oscillator (i.e. $1/f_{OSC}$) and PV is dividing factor of prescaler. From formula Eq. (6) we can determine formula for period register $PR2$:

$$PR2 = \frac{Period}{4T_{OSC} \cdot PV} - 1. \tag{7}$$

Duty (filling by H) is determined by value of register CCPR1L. Let us demonstrate it on example row sinus signal. The PWM1 uses a Timer2. A part of program definitions PR2_xxxHz dealing with values of periods (values for period register) and duty 50 % of PWM1:

Algorithm 4

```
1: FOSC 18432000.0
2: PV 16.0, Timer2_Prescale_Value.
3: PR2_697Hz (1.0/697.0)/(PV·(4.0/FOSC))-1.0;
4: PR2_770Hz (1.0/770.0)/(PV·(4.0/FOSC))-1.0;
5: PR2_852Hz (1.0/852.0)/(PV·(4.0/FOSC))-1.0;
6: PR2_941Hz (1.0/941.0)/(PV·(4.0/FOSC))-1.0;
7: duty_50 percent 0x80; value for SFR CCPR1L
```

rem – shortened, in real C-language program these lines are by definitions. Analogical values are used for column signal (PWM3 using Timer4). In this third variation on theme "generator of DTMF signal" processor has plenty of time for other tasks. As to the DTMF processor must only read keys of keyboard, and (if some key is pressed) to rewrite proper period registers. From this moment signals U_r, U_c are generated total automatically.

5. Conclusion

There are described three solutions how to generate DTMF signal in this article. The first presented solution consists in sinus function contained in memory as table of 32 values. External D/A converter was realized by resistance ladder R-2R [4]. Following circuit is low-pass filter [3]. The second and third solutions are based on fact, that generating only rectangle signal is enough for filter producing only first harmonic as output signal. The second solution is based on classic timers having compare facilities. And the third solution exploits PWM hardware of PIC18F46K22 for generating rectangle signals, this solution needs practically no machine time of microprocessor. Time consumption of these three methods in machine cycles: $1^{st} \ldots 43, 2^{nd} \ldots 14, 3^{rd} \ldots 0$. So in presented article is shown, that normal (not DSP) microcontroller can be applied for DTMF generating without degrading its compute performance (the third variant). The three described solutions can serve to developers in their work on similar technical tasks. I consider appropriate programs in C-language (for generating DTMF signal by mentioned three methods) to be my original contribution to related technical branches. It is offering a consequent task-to recognize two basic frequencies in DTMF signal, but it is matter of signal processing, programming of DSP etc.

Acknowledgment

The research described in this article could be carried out thanks to the active support of the Ministry of Education, Youth and Sports of the Czech Republic through grant project no. CZ.1.07/2.3.00/20.0217 within the frame of the operation programme Education for competitiveness financed by the European Structural Funds and from the state budget of the Czech Republic.

References

[1] MICROCHIP. PIC18(L)F2X/4XK22 Data Sheet. *Microchip Technology Inc.* [online]. 2012.

Available at: http://ww1.microchip.com/downloads/en/DeviceDoc/41412F.pdf.

[2] NOVAK, R. and K. WITAS. Jitter elimination at optical control of servomotors. *Advances in Electrical and Electronic Engineering*. 2014, vol. 12, no. 2, pp. 117–122. ISSN 1336-1376. DOI: 10.15598/aeee.v12i2.990.

[3] GAJDOSIK, L. *Analog filter design*. 1st ed. Praha: BEN, 2013. ISBN 978-80-7300-468-2.

[4] VRBA, K. *Technika analogovych obvodu a systemu*. 3rd ed. Brno: VUTIUM, 1989. ISBN 80-214-1060-4.

[5] SALOUN, P. *Jazyk C*. 1st ed. Praha: Neokortex, 2003. ISBN 80-863-3008-7.

[6] BOULET, B. *Fundamentals of Signals & Systems*. 1st ed. Boston: Charles river media, 2005. ISBN 1-58450-381-5.

[7] *British Standards Institute Staff Specification for Dual-Tone Multi-Frequency (DTMF) Signalling Protocol for Social Alarm Systems*. BSI Standards. 2009. ISBN 978-0-580-55079-9.

[8] Generating DTMF Tones Using Z8 Encore! XP F64xx Series MCUs. *MCU Application Note* [online]. Available at: http:/www.zilog.com/docs/z8encore/appnotes/an0248.pdf.

About Authors

Radek NOVAK was born in Most, Czech Republic. He received his M.Sc. from VSB–Technical University of Ostrava in 1984. His research interests include single-chip microcontrollers, electronics.