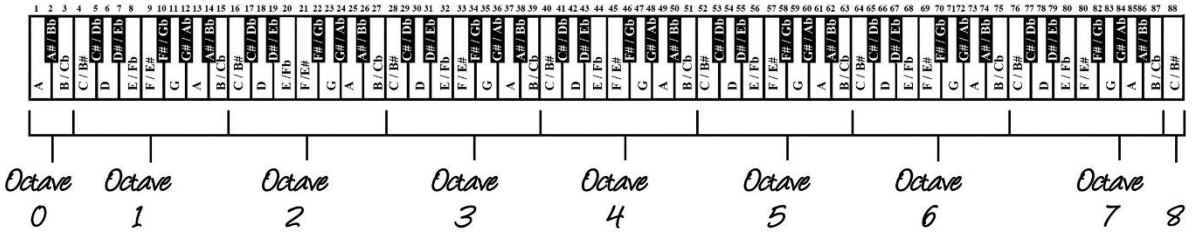
نواختن نُت‌های موسیقی با استفاده از میکروکنترلر

هدف از این تمرین راه‌اندازی واحد Timer و واحد DAC میکروکنترلر و استفاده از آن‌ها برای نواختن نُت‌های موسیقی است.

برای نواختن یک قطعه موسیقی باید به ترتیب، نُت‌های مربوط به آن را اجرا کرد. هر نُت، فرکانس معین مخصوص به خود را دارد. در بین ابزارهای موسیقی، بعضی از انواع پیانوها دارای بیشترین بازه‌ی فرکانس‌های تولیدی هستند. در شکل زیر یک پیانوی ۸۸-نُت نشان داده شده است:



در جدول زیر فرکانس مربوط به هر نُت در پیانوی فوق محاسبه شده است:

Note										A0	A#0	B0
Freq.										27.500	29.135	30.868
Note	C1	C#1	D1	D#1	E1	F1	F#1	G1	G#1	A1	A#1	B1
Freq.	32.703	34.648	36.708	38.891	41.203	43.654	46.249	48.999	51.913	55.000	58.270	61.735
Note	C2	C#2	D2	D#2	E2	F2	F#2	G2	G#2	A2	A#2	B2
Freq.	65.406	69.296	73.416	77.782	82.407	87.307	92.499	97.999	103.83	110.00	116.54	123.47
Note	C3	C#3	D3	D#3	E3	F3	F#3	G3	G#3	A3	A#3	B3
Freq.	130.81	138.59	146.83	155.56	164.81	174.61	185.00	196.00	207.65	220.00	233.08	246.94
Note	C4	C#4	D4	D#4	E4	F4	F#4	G4	G#4	A4	A#4	B4
Freq.	261.63	277.18	293.67	311.13	329.63	349.23	369.99	392.00	415.30	440.00	466.16	493.88
Note	C5	C#5	D5	D#5	E5	F5	F#5	G5	G#5	A5	A#5	B5
Freq.	523.25	554.37	587.33	622.25	659.26	698.46	739.99	783.99	830.61	880.00	932.33	987.77
Note	C6	C#6	D6	D#6	E6	F6	F#6	G6	G#6	A6	A#6	B6
Freq.	1046.5	1108.7	1174.7	1244.5	1318.5	1396.9	1480.0	1568.0	1661.2	1760.0	1864.7	1975.5
Note	C7	C#7	D7	D#7	E7	F7	F#7	G7	G#7	A7	A#7	B7
Freq.	2093.0	2217.5	2349.3	2489.0	2637.0	2793.0	2960.0	3136.0	3322.4	3520.0	3729.3	3951.1
Note	C8											
Freq.	4186.0											

جدول فوق برای راحتی کار و حذف محاسبات آورده شده است. به طور کلی فرکانس‌ها به این طریق محاسبه می‌شوند که ابتدا یک نُت به عنوان نُت مرجع انتخاب می‌شود (معمولا A4 با فرکانس 440Hz)، سپس فرکانس بقیه‌ی نُت‌ها از رابطه‌ی زیر به دست می‌آید:

$$f_n = 2^{\frac{n}{12}} \times f_{ref}$$

که در آن n فاصله‌ی نُت مورد نظر از نُت مرجع است.

همان‌طور که اشاره شد، هر قطعه موسیقی توسط ترتیب اجرای نُت‌ها که روی sheet music نوشته شده است، نواخته می‌شود. یک قطعه‌ی معروف، ساده و کوتاه موسیقی، "Twinkle, Twinkle, Little Star" نام دارد که sheet music آن به شکل زیر است:

و معادل آن به صورت زیر می‌باشد:

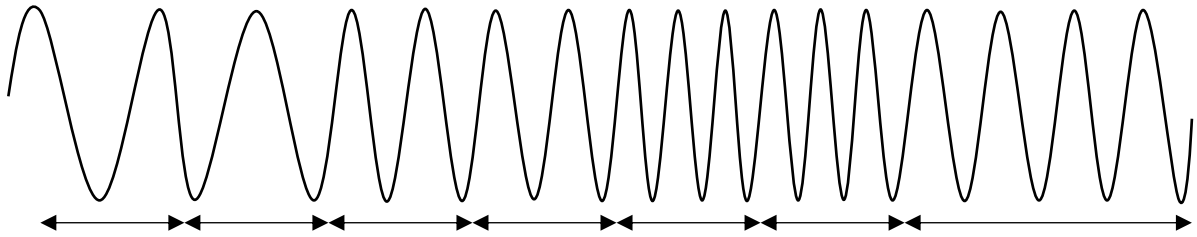
```
C4  C4  G4  G4  A4  A4  G4  ; twinkle twinkle little star
F4  F4  E4  E4  D4  D4  C4  ; how I wonder what you are
G4  G4  F4  F4  E4  E4  D4  ; up above the world so high
G4  G4  F4  F4  E4  E4  D4  ; like a diamond in the sky
C4  C4  G4  G4  A4  A4  G4  ; twinkle twinkle little star
F4  F4  E4  E4  D4  D4  C4  ; how I wonder what you are
```

که در آن نُت آخر در هر سطر از نوع half note بوده (مدت زمان اجرای آن‌ها نصف whole note است) و بقیه‌ی نُت‌ها همگی quarter note هستند (مدت زمان اجرای آن‌ها ربع whole note است).

پارامتر دیگری به نام BPM (beats per minute) هم هنگام نواختن باید در نظر گرفته شود. این پارامتر سرعت اجرا را مشخص می‌کند. برای این قطعه BPM=120 مقدار مناسبی است. پس کوتاه‌ترین زمان اجرای یک نُت در این قطعه برابر $\frac{60}{120} = 0.5\ s$ خواهد بود.

برای نواختن موسیقی فوق با استفاده از میکروکنترلر، باید به جای هر نُت فرکانس مربوط به آن را جایگذاری کرده و زمان اجرا را با توجه به نوع نُت و مقدار BPM تنظیم کنیم.

مثلا برای نواختن سطر اول (C4 C4 G4 G4 A4 A4 G4)، شکل موج سینوسی زیر باید تولید شود:



note:	C4	C4	G4	G4	A4	A4	G4
duration:	0.5s	0.5s	0.5s	0.5s	0.5s	0.5s	1s
frequency:	261.63Hz	261.63Hz	392Hz	392Hz	440Hz	440Hz	392Hz

برای تولید شکل موج سینوسی از واحد DAC و برای تنظیم فرکانس‌ها و مدت‌زمان هر نُت، از Timer میکروکنترلر استفاده خواهیم کرد.

فرض کنید می‌خواهیم یک موج سینوسی تولید کنیم. دامنه‌ی این شکل موج بین 0 و Vcc خواهد بود. دقت مبدل دیجیتال به آنالوگ (DAC) را هم برابر n بیت در نظر بگیرید. یعنی ورودی DAC عددی بین 0 تا $2^n - 1$ و خروجی آن بین 0 تا Vcc خواهد بود. اگر ورودی DAC را با $y_{Digital}$ و خروجی آن را با y_{Analog} نشان دهیم، می‌دانیم رابطه‌ی آن‌ها به صورت زیر است:

$$y_{Analog} = \frac{y_{Digital}}{2^n - 1} \times Vcc$$

برای ساختن موج سینوسی فرض می‌کنیم که از موج سینوسی نمونه‌برداری کرده‌ایم و می‌خواهیم از روی این نمونه‌ها، موج سینوسی را بسازیم. برای این کار یک look-up table به شکل زیر باید نوشته شود:

$$y_{SineDigital}(i) = \left(\sin(\frac{2\pi i}{m}) + 1\right) \times 2^{n-1} \quad , \quad i = 0, 1, \dots, m-1$$

m تعداد نمونه‌هایی است که یک سیکل از موج سینوسی قرار است توسط آن‌ها ساخته شود. انتخاب صحیح مقدار m هم مهم است. هر چقدر m بزرگ‌تر باشد، سیگنال ساخته شده به سینوسی نزدیک‌تر خواهد بود و در نتیجه تأثیر هارمونیک‌ها کمتر شده و در نهایت صدای صاف‌تری تولید خواهد شد. اما مقدار m را هم تا یک حدی می‌توان افزایش داد.

* برای سهولت در نوشتن look-up table ، از نرم‌افزارهایی مثل Matlab یا از زبان‌های برنامه‌نویسی مختلفی می‌توانید استفاده کرده و آرایه‌ی حاصل را در کد نوشته شده کپی کنید.

سپس با استفاده از یک Timer و تنظیم زمان انتقال اعداد از look-up table فوق به DAC، می‌توان موج سینوسی با فرکانس دلخواه را ساخت.

حال فرض کنید می‌خواهیم شکل موج سینوسی نشان داده شده‌ی فوق را بسازیم. ابتدا از نُت C4 با فرکانس $f_{C4} = 262Hz$ شروع می‌کنیم. پس باید موج سینوسی به مدت نیم ثانیه با فرکانس ۲۶۲ هرتز تولید کنیم، سپس موجی با فرکانس ۲۶۲ هرتز به مدت نیم ثانیه، سپس موجی با فرکانس ۳۹۲ هرتز به مدت نیم ثانیه و به همین ترتیب.

برای تنظیم مدت زمان اجرای هر نُت نیز از تایمر دیگری می‌توانید استفاده کنید.

برای تبدیل سیگنال الکتریکی تولید شده در خروجی DAC، به سیگنال صوتی می‌توان از اسپیکرهای آهنربایی استفاده کرد. یک نمونه اسپیکر 8 اهم و 1 وات در زیر نشان داده شده است:

این اسپکیر به خروجی DAC وصل خواهد شد. البته با توجه به جریان‌کشی بالای اسپیکر، در صورت نیاز از یک بافر بین خروجی DAC میکروکنترلر و اسپیکر استفاده کنید.

فرمت کلی کد نوشته شده به صورت زیر خواهد بود:

```
/* corresponding digital values of sine wave with m = 10 and n = 12 */
int SineWave[10] = {2048, 3251, 3995, 3996, 3253, 2051, 847, 101, 98, 839};

/* frequency of notes used in "Twinke, Twinkle, Little Star" */
int A4 =440, C4 = 262, D4 = 294, E4 = 330, F4 = 349, G4 = 277;

void play_note(int note_freq, int note_duration){
  /* generate a sine waveform with frequency and duration of a given note
     by setting Timer1 Compare value according to frequency and
     Timer2 Compare value according to duration */


}

int main(void)
{
  /* clock configuration*/



  /* Timer Initialization */



  /* DAC Initialization */


```

```

  int BpM = 120;  // beats per minute
  int Q_Duration = 60000/BpM; // duration of each quarter note in milliseconds
  int H_Duration = 2*60000/BpM; // duration of each half note in milliseconds

  while (1)
  {

    play_note (C4, Q_Duration); //play C4 note for "Q_Duration" milliseconds
    play_note (C4, Q_Duration); //play C4 note for "Q_Duration" milliseconds
    play_note (G4, Q_Duration); //play G4 note for "Q_Duration" milliseconds
    play_note (G4, Q_Duration); //play G4 note for "Q_Duration" milliseconds
    play_note (A4, Q_Duration); //play A4 note for "Q_Duration" milliseconds
    play_note (A4, Q_Duration); //play A4 note for "Q_Duration" milliseconds
    play_note (G4, H_Duration); //play G4 note for "H_Duration" milliseconds
    play_note (F4, Q_Duration); //play F4 note for "Q_Duration" milliseconds
    /*                .
                      .
                      .                          */
    play_note (C4, H_Duration); //play G4 note for "H_Duration" milliseconds

  }

}

```

موارد تشویقی:

- انتخاب یک موسیقی دلخواه دیگر و پخش آن

- نمایش متن موسیقی توسط LCD همزمان با پخش آن

- می‌توانید یک ردیف LED به تعداد نُت‌های موجود را، با توجه به نُتی که در حال حاضر اجرا می‌شود روشن و خاموش کنید. مثلا برای این قطعه که از ۶ نُت استفاده شده است، ۶ عدد LED را به هر یک از نُت‌ها اختصاص دهید و هنگام اجرای یک نُت LED مربوط به آن روشن باشد (LED ها را از چپ به راست به ترتیب فرکانس (از فرکانس پایین به بالا) مرتب کنید)

فایل‌های تحویلی:

۱. پروژه‌ی نرم‌افزار Keil uVision

۲. گزارش‌کار در فرمت PDF

۳. (در صورت کار عملی) فیلم کوتاهی از پروژه‌ی در حال اجرا

سوالات:

۱. همان‌طور که گفته شد، با انتخاب m بزرگ‌تر می‌توان موج سینوسی دقیق‌تری تولید کرد. مشکلات و عوامل محدود کننده برای انتخاب m های بزرگ را بیان کنید.

۲. مقادیر انتخابی یا محاسبه‌شده‌ی پارامترهایی مثل m، فرکانس تایمرها، مقادیر رجیسترهای تایمرها و نحوه‌ی انتخاب یا محاسبه‌ی آن‌ها را در گزارش کار ذکر کنید.

«موفق باشید»