SYN6288 Chinese Speech Synthesis Module

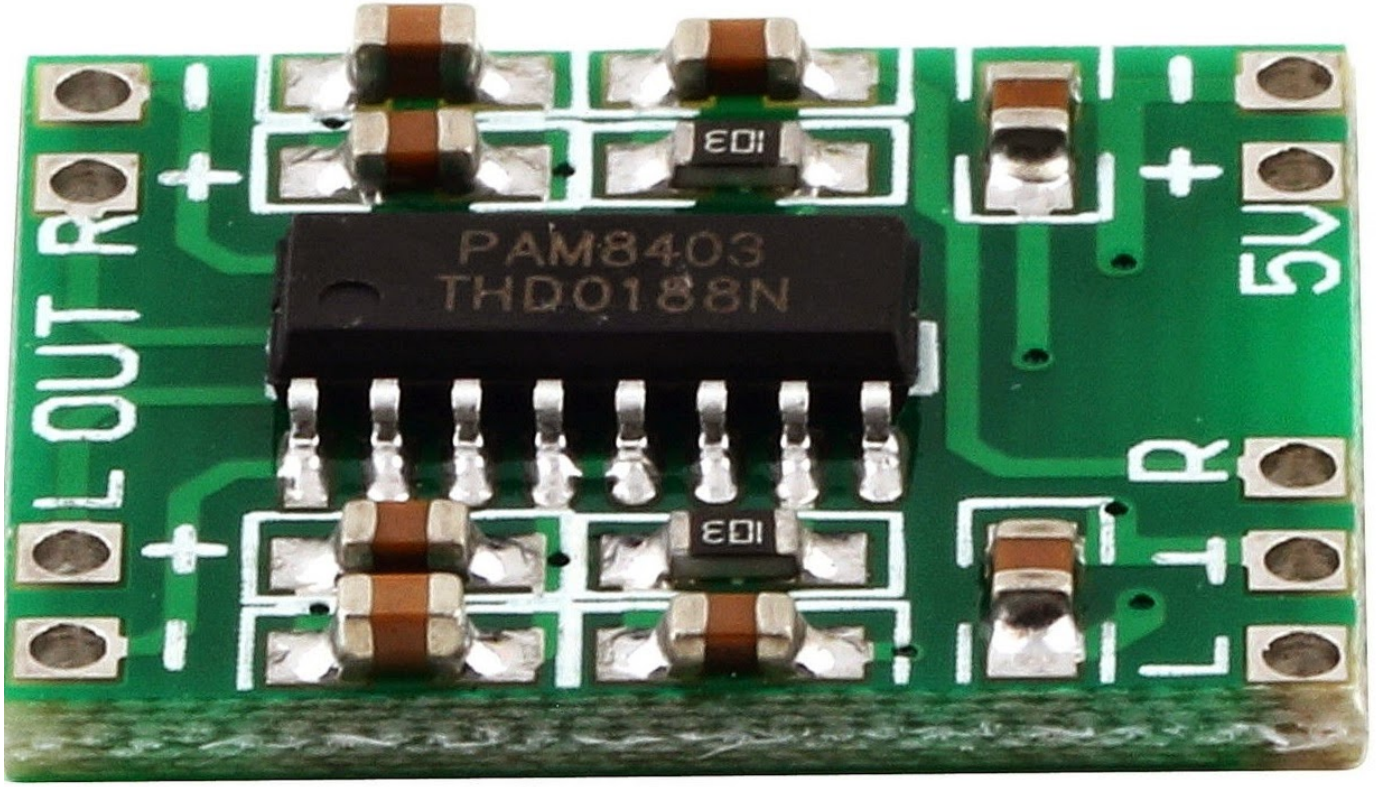
July 18, 2014 ToughDev 4 Comments

0.00 avg. rating (**0**% score) - **0** votes

When searching eBay for a text to speech IC equivalent to the TTS256, I came across the SYN6288, a cheap speech synthesis module made by a Chinese company called Beijing Yutone World Technology specializing in embedded voice solutions and decided to give it a try. Although the IC only comes in SSOP28 10.2mm*5.3mm package, the eBay item which I purchased provided a nice breakout board having 2.54mm pin pitch for easy prototyping:

The module operates on 5V, receives commands via a 9600bps, 8 data bit, 1 stop bit, no parity UART connection and provides mono audio output on the BP0 and BN0 pins. There is also a BUSY output pin which turns high when the module is still processing the commands sent to it via UART. A minor inconvenience is that, due to the SYN6288 breakout board physical dimensions, it is impossible to plug it into a single breadboard for testing – you will need to use two.

Because the audio output levels are low, you will need either a crystal earpiece to listen to it or a suitable audio amplifier such as the LM386 to play it on an 8-ohm speaker. In my case, I use the PAM8403, another cheap but great audio amplifier purchased from eBay:



Reading the datasheet

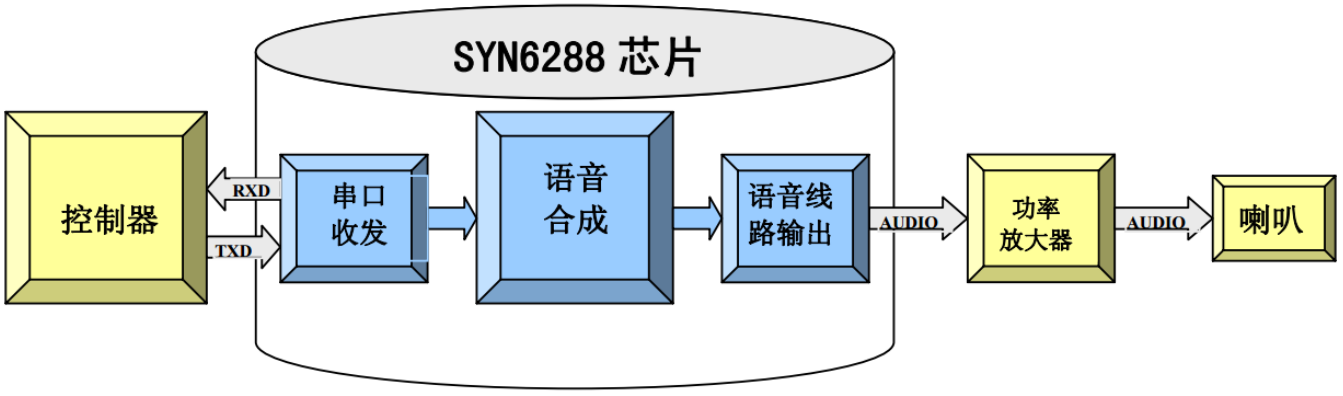
Because the module specifically targets the Chinese market, the datasheetunderstandably comes in Chinese only with no official English version. Luckily I managed to use this free tool to translate the Chinese PDF datasheet into English, while preserving the file format and layout.

Knowing that the translation tool works based on the Google Translate API, I decided to cross-compare the original Chinese version and the English translation to verify the translation quality. Below is the SYN6288 block diagram in Chinese:

1.5 系统构成框图

最小系统包括：控制器模块、SYN6288语音合成芯片、功放模块和喇叭。

主控制器和SYN6288语音合成芯片之间通过UART 接口连接，控制器可通过通讯接口向SYN6288语音合成芯片发送控制命令和文本，SYN6288语音合成芯片把接收到的文本合成为语音信号输出，输出的信号经功率放大器进行放大后连接到喇叭进行播放。



This is the translated block diagram:

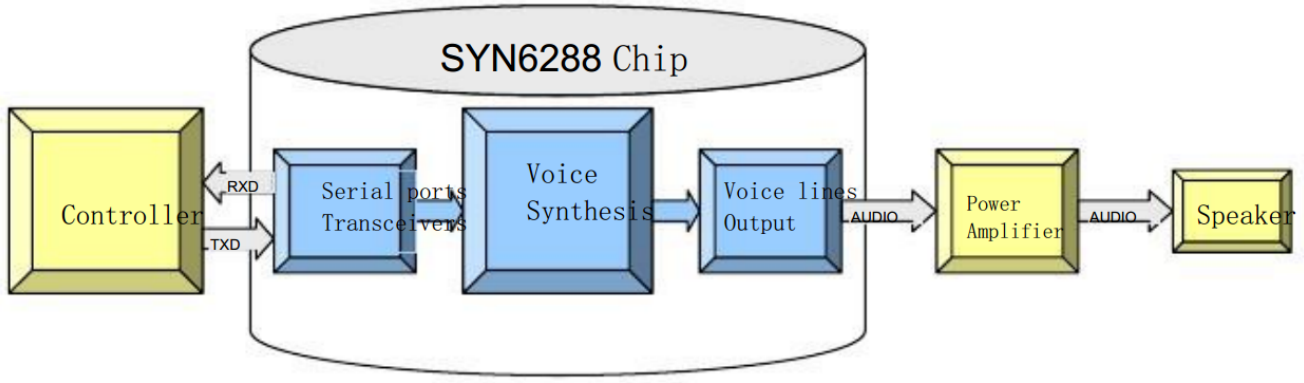
1.5 Constitute a block diagram of the system

Minimum system comprising: a controller module, SYN6288 speech synthesis chip, power amplifier module and speakers.

Between the main controller and SYN6288 through speech synthesis chip UART Interface, the controller via the communication interface to speech synthesis core SYN6288

Transmitting control commands and text sheet, SYN6288 speech synthesis chip to the received signal output text speech synthesis, the output signal is amplified by the power

After the device is connected to the amplifying horn for playback.



Although the translation is still understandable, multiple text blocks seem to overlap each other. This is because the PDF file format allocates each line or block of text at a fixed location and size. When the translated text is longer than the original text, the text block will wrap to the next line, but without knowing the position of the next text block, causing the alignment issue. Also, as the tool feeds the texts to be translated into the Google Translate API block-by-block, not knowing that many of them are parts of sentences which in turn form paragraphs, the translation quality will be reduced significantly as Google Translate is known to work better when a full sentence is provided.

Getting it to work

I found the sample code for this module here, provided by CooCox, another Chinese company. Although it compiles fine under MPLAB C30, I spent a few frustrating hours figuring why there is no audio output from the SYN6288 using the code. It turns out that the provided sample code does not send the completed frame header and the function SYN_SentHeader() needs to be modified to fix this:

```
void SYN_SentHeader(uint16_t DataAreaLength) {
    FrameHeader[1] = (DataAreaLength & 0xFF00) >> 8;
    FrameHeader[2] = (DataAreaLength & 0xFF);
    SendUART2(FrameHeader[0]);
    SendUART2(FrameHeader[1]);
    SendUART2(FrameHeader[2]);

    // Added by MD - missing from original code!
    SendUART2(FrameHeader[3]);
    SendUART2(FrameHeader[4]);
}
```

The module supports 4 different types of Chinese encoding, GB2312, GBP, Big5 or Unicode. Interestingly, this module also supports playing 5 different types of background music together with the speech. The encoding currently in use and the type of background music will need to be specified prior to sending the text to be spoken. The code also needs to monitor the SYN_BUSY output pin and only sends the text when the module is no longer busy processing the input. This is shown in the following code:

```
void SYN_Say(const uint8_t * Text) {
    uint16_t i;
    uint16_t Length;

    for(Length = 0;;Length++){
        if(Text[Length] == '')
        break;
    }

    FrameHeader[3] = 0x01;
    FrameHeader[4] = BackGroundMusic | TextCodec;
    SYN_SentHeader(Length + 3);
    for(i = 0;i < Length;i++){
        SendUART2(Text[i]);
    }
    SendUART2(CheckXOR(Text,Length));

    // wait for processing, otherwise the SYN_BUSY won't turn on immediately and
    // the following wait is useless.
    delay_ms(20);

    // wait until speaking is completed before exiting the function.
    while (SYN_BUSY);

    // wait a bit more before exiting, otherwise the chip is still busy
    // and may ignore the next command set
```

```
        delay_ms(20);
    }
```

With this code, the module is able to speak some Chinese characters. The following is a voice recording of the SYN6288 trying to say 恭喜，万事如意 (**Gōngxǐ, wànshì rúyì**, *Congratulations and good luck*) – you can hear the background music if you turn the volume loud enough:

The above recording is done by amplifying the SYN6288 output using PAM8403, feeding the audio into an 8-ohm speaker, and recording the sound using an iPhone. Although we can still tell that the module tries to speak the Chinese text syllable by syllable, the audio quality is quite good and much more natural than the mechanical voice of the TTS256.

The recording when done by feeding the output audio directly into the computer's line-in input will have slightly better quality. The following is the sample audio provided by the manufacturer:

The module also supports some predefined tones to be played together with the Chinese text. This is done by prefixing the text with a predefined string (e.g msga, msgb, msgc, etc.) specifying the tone to be played. The following will show the SYN6288 playing "ding-dong" and counting numbers in Chinese:

Finally the module supports spelling the English alphabet, from A to Z:

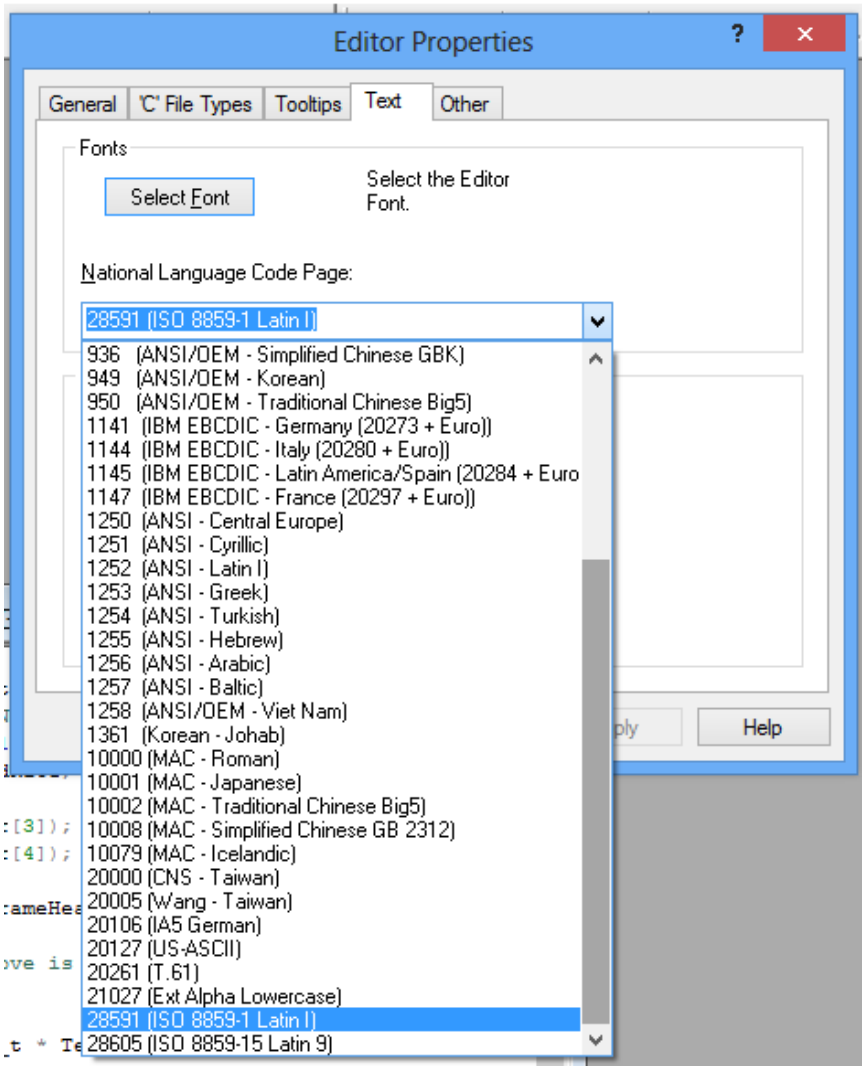
It does not support English words. If you try to send English words, it will try to spell them character-by-character. The following shows what the module will say when we send the text "This is an English sentence":

My last test on this module is to view the audio output on an oscilloscope and see the waveforms. The purpose is to see whether the module is using PWM (Pulse Width Modulation) with a simple RC filter, or a more complicated mechanism for audio output. The following video shows the output waveform during playback:

The waveform looks quite smooth, hence I concluded that it does not use PWM, but more likely a simple DAC (Digital to Analog Converter) in order to achieve the desired audio quality.

MPLAB 8.92 and Chinese characters

When testing the SYN6288, I encountered some problems using MPLAB to edit text files with Chinese characters. First, the code page for the current file needs to be changed by right clicking the editor for the file, selecting **Properties** and opening the **Text** tab:



There is no Unicode in the selection list – only GBK/GB2312 for simplified Chinese and Big5 for Traditional Chinese. However, when I select any of the Chinese encodings, the MPLAB editor starts to show erratic behaviour – texts are misplaced and the cursor behaves weirdly:

```
// chinese characters declared as array, last character is for string ending
// codes taken from datasheet page 14
// 宇音天下 (Yu yin tianxia)
unsigned char voice[9] = {0xD3, 0xEE, 0xD2, 0xF4, 0xCC, 0xEC, 0xCF, 0xC2, 0x00};
SYN_Say(voice);

// chinese character as BIG5 strings
SYN_SetTextCodec(SYNTextCodecBIG5);

// Gongxi fac蚯, w跣sh?r□? zhu li m酈g en (Good fortune, good luck)
// the chip supports special tone to play before each announcement (soundxxx, msgxxx, ringxxx). Refers to Datasheet page 22.
SYN_Say((unsigned char *)"sounda 恭喜,万事如意,主里蒙恩"); e, good

// numbers from 0 to 10
SYN_Say((unsigned char *)"soundk 零一二三四五六七八九十");  e, go

// english characters
SYN_Say((unsigned char *)"msga A B C D E F G H I");
SYN_Say((unsigned char *)"J K L M N O P Q R"); // ringb, ringk: long music
SYN_Say((unsigned char *)"S T U V W X Y Z");
SYN_Say((unsigned char *)"This is an English sentence");

SendUARTStr("\nSYN6288 Demo completed...");
```

I decided to use the default encoding (ISO 8859-1 Latin I). In this mode, Chinese text will display wrongly once pasted in the editor but the editor will operate normally without any other issues:

```
// Gongxi facái, wànshì rúyì, zhu li méng en (Good fortune, good luck)
// the chip supports special tone to play before each announcement (soundxxx, msgxxx, ringxxx). Refers to Datasheet page 22.
SYN_Say((unsigned char *)"sounda ®¥³ß;AÉE¨E¦p·N;A¥D¨½»X®¦");

// numbers from 0 to 10
SYN_Say((unsigned char *)"soundk ³s¤@¤G¤T¥¦¤-¤»¤C¤K¤E¤Q");
```

To get the Chinese text display properly without any other issues, we will need to use MPLAB X, which is Netbeans-based and supports Unicode natively.

The verdict

My conclusion after testing is that the SYN6288 is a great simple speech synthesis module for the Chinese language. Although there will always be words which are not pronounced correctly due to the complexity of Chinese (the datasheet, on page 21, indicates that the chip can pronounce around 98% of Chinese characters accurately), I feel that the SYN6288 will work well for small Chinese embedded voice applications. It is unfortunate that there is no equivalent module for the English language. The TTS256 with its horrible mechanical voice is long dead, and some companies are now trying to make huge profits by making clones of the TTS256 and selling them at high price, with little or no improvement at all in the speech quality.

Downloads

Original Chinese datasheet
Translated English datasheet
SYNC6288 C30 library

To use the library, you need to first set the configuration (background music, encoding) for the SYN6288 module:

```
SYN_SetBackGroundMusic(SYNBackGroundMusic1);
SYN_SetTextCodec(SYNTextCodecGBK);
```

After that, use SYN_Say to send a text string to the module:

```
SYN_Say((unsigned char *)"恭喜，万事如意");
SYN_Say((unsigned char *)"msga A B C D E F G H I");
```

Of course prior to calling the SYN6288 functions, you will need to configure the UART module properly to send commands. The code provided above also contains a simple UART library to facilitate this task.

See also

LD3320 Chinese Speech Recognition and MP3 Player Module
English text to speech on a PIC microcontroller