Direc
tory

First, the hardware description of the chip

1 , power supply of chips

The optimal operating voltage of the chip is 4.2V . So if the user is using 5V Power supply, it is recommended to cascade a diode

2, the chip led[Power on state]

工作状态	下载模式	播放语音	暂停	睡眠
状态	快闪	慢闪	常亮	灭

备注：正常工作状态指示灯

3, the chip led[working status]

工作状态	一对一可打断	抬起停止	一对一不可打断	标准MP3功能
状态	常亮2S	快闪2S[100ms取反]	中慢闪2S[200ms取反]	慢闪2S[500ms取反]

备注：此时指示灯，只在上电初始化的时候，指示2秒

4 , audio output instructions

1, SPK1 And SPK2 Received the horn two Rui, can be positive and negative, note: only allowed to receive 2W The following speakers

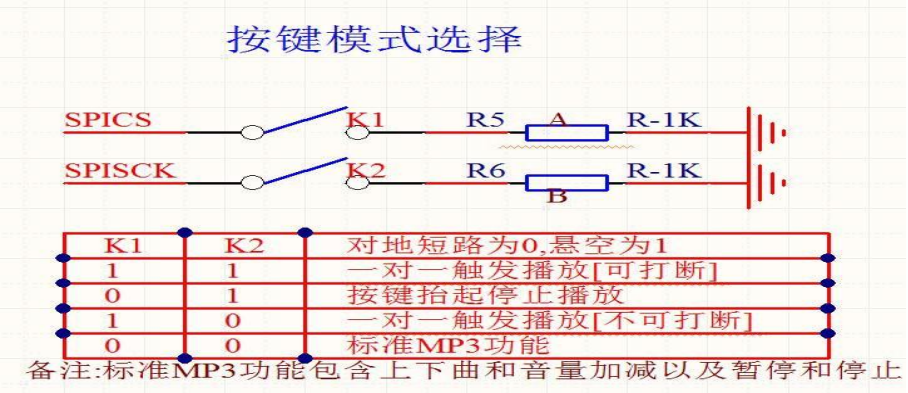
2 , power amplifier or headphones, connect directly DAC--R And DAC--L Both ends, note: Common (Power)

5 , chip debugging instructions

(1), Our chip is by default plugged in Usb Line, you go to download mode. When the user has finished updating the speech, triggers any Io

, you can exit download mode and return to normal working condition.

(2) , debugging chips first from easy to difficult, first simple to complex, the key to ensure normal after the adjustment of the serial port, serial debugging hand Debugging normal, and then the single chip, TF card does not sound, first through the USB plug computer, see if you can read the TF card letter



Serial debugging assistant for testing	commands to send [with Check]	commands to send [without checksum]	Note

[next song]	7E FF 06 01 00 00 00 FE FA Ef	7E FF (EF)	
[Previous song]	7E FF 06 02 00 00 00 FE F9 Ef	7E FF/EF	
[Specify Tracks]	7E FF 06 03 00 00 01 FE F7 Ef	7E FF (EF)	Specify first play
	7E FF 06 03 00 00 02 FE F6 Ef	7E FF (EF)	Specify second head
	7E FF 0A FE EE Ef	7E FF 0A EF	Specify subsection 10 First
Volume Plus	7E FF 06 04 00 00 00 FE F7 Ef	7E FF/EF	
Volume reduction	7E FF 06 05 00 00 00 FE F6 Ef	7E FF/EF	
[Specify Volume]	7E FF 1E FE D7 Ef	7E FF 1E EF	Specifies that the volume is 30 Level
[Specify EQ]	7E FF 06 07 00 00 01 FE F3 Ef	7E FF/EF	Keep
[loop play track]	7E FF 06 08 00 00 01 FE F2 Ef	7E FF EF	Loop play the first song
	7E FF 06 08 00 00 02 FE F1 Ef	7E FF/EF	Loop second Song
	7E FF 0A FE E9 Ef	7E FF 0A EF	Loop to play the tenth song
	7E FF 06 08 00 01 01 FE F1 Ef	7E FF-EF	Loop broadcast Put of FLASH FOLDER1 The first song
	7E FF 06 08 00 02 01 FE F0 Ef	7E FF EF	Loop broadcast Put of FLASH FOLDER2 The first song
[Specify Playback device]	7E FF 06 09 00 00 01 FE F1 Ef	7E FF EF	Specify playback Udisk
	7E FF 06 09 00 00 02 FE F0 Ef	7E FF (EF)	Specify playback Tf
	7E FF 06 09 00 00 04 FE EE Ef	7E FF (EF)	Specify playback FLASH
[Enter sleep mode]	7E FF 0A 00 00 00 FE F1 Ef	7E FF more 0A EF	
[wake Up Sleep]	7E FF 0B 00 00 00 FE F0 Ef	7E FF more 0B EF	
[chip reset]	7E FF 0C 00 00 00 FE EF Ef	7E FF more 0C EF	
[play]	7E FF 0D 00 00 00 FE EE Ef	7E FF more 0D EF	
[pause]	7E FF 0E 00 00 00 FE ED Ef	7E FF more 0E EF	
[Specify folder file name]	7E FF 0F 00 01 01 FE EA Ef	7E FF (0F) EF	specified as "a" the folder, track for "001"

	7E FF 0F 00 01 02 FE E9 Ef	7E FF (0F) EF	specified as "a" the folder, track for "002"

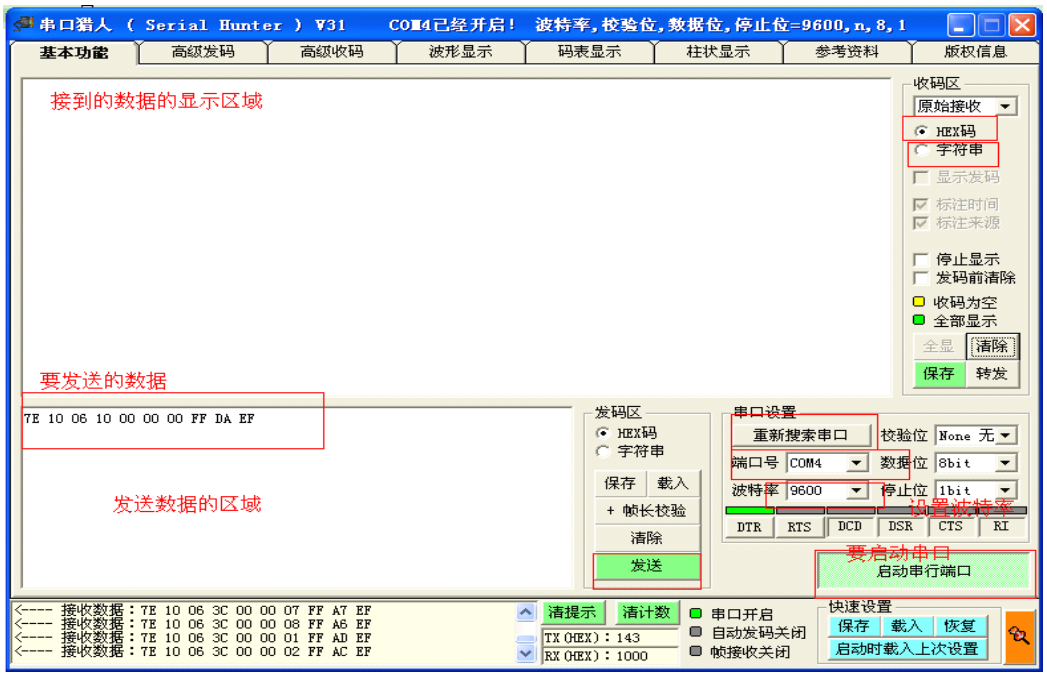
Support 1000 First	7E FF modified FF FD D8 Ef	7E FF EF	specified as "a" the folder, track for "0255"
	7E FF Modified CF FE 01 Ef	7E FF + CF EF	specified as "a" the folder, track for "1999"
	7E FF 01 C0 FE 26 Ef	7E FF C0 EF	specified as "a" the folder, track for "0001"
	7E FF C0 FF FD 28 Ef	7E FF C0 FF EF	specified as "a" the folder, track for "0255"
	7E FF C7 CF FD 51 Ef	7E FF C7 CF EF	specified as "a" the folder, track for "1999"
Stop playing	7E FF 06 16 00 00 00 FE E5 Ef	7E FF/EF	Stop software decoding
Specify a folder loop to play	7E FF 06 17 00 02 00 FE E2 Ef	7E FF EF	Specified 02 Folder Loop Playback
	7E FF 06 17 00 01 00 FE E3 Ef	7E FF ' EF	Specified 01 Folder Loop Playback
Random playback	7E FF E3 EF	7E FF/EF	Random playback
Single Loop playback	7E FF E2 EF	7E FF/EF	Single Loop playback Open
	7E FF-FE E1 EF	7E FF ' EF	Single Loop playback off
Dac The settings	7E FF 1 FE E1 EF	7E FF 1 EF	Open Dac
	7E FF 1 a 00 00 01 FE E0 Ef	7E FF 1 EF	Off Dac
Group Playback	7E FF 09 21 00 05 01 02 03 04 FE C8 Ef	7E FF 09 21 00 05 01 02 03 04 Ef	Group Playback 5，1，2，3，4
Group Playback	7E FF 0C 21 00 05 01 02 03 04 06 07 08 FE B0 Ef	7E FF 0C 21 00 05 01 02 03 04 EF	Group Playback 5，1，2，3，4，6，7，8
Play with Volume	7E FF 01 1E FE BA Ef	7E FF 1E EF	30 Level volume play the

			first 1 Song
	7E FF 01 0F FE C9 Ef	7E FF 0F EF	15 Level volume play the first 1 Song
	7E FF 02 0F FE C8 Ef	7E FF 0F EF	15 Level volume play the first 2 Song

Query Current state	7E FF-FE B9 EF	7E FF/EF	
[Query Volume]	7E FF 06 43 00 00 00 FE B8 Ef	7E FF EF	
[Query current EQ]	7E FF-FE B7 EF	7E FF EF	
U Total file Number of disk	7E FF 06 47 00 00 00 FE B4 Ef	7E FF EF	The total number of files for the current device
Tf Total number of documents	7E FF B3 EF	7E FF EF	
FLASH Total number of documents	7E FF-FE B2 EF	7E FF EF	
U Disk Current Track	7E FF 4 B 00 00 00 FE B0 Ef	7E FF 4 EF	Currently playing tracks
Tf Current track	7E FF 4C 00 00 00 FE AF Ef	7E FF more 4C EF	
FLASH Current folder track Pointer	7E FF 4D 00 00 00 FE AE Ef	7E FF more 4D EF	
Query the total number of folder tracks	7E FF 4E 00 00 01 FE AC Ef	7E FF (4E) EF	Query 01 folder or FOLDER1 The total number of tracks
Query TF or U Disk Total File Number of Clips	7E FF 4F 00 00 00 FE AC Ef	7E FF more 4F EF	only supports Tf Card and U Plate
The current folder pointer [FLASH]	7E FF 06 61 00 00 00 FE 9A Ef	7E FF EF	querying the currently playing folder [Support Hold FLASH]

Third, test methods

1 , the operation of serial port software



(1), first install the information in the " Serial Hunter " Software, open the software, first of all to search the serial port, find the specified port, specify "baud rate", our module default baud rate is 9600 , the last is "Start serial port", so the software is configured well. Here are two concepts that need to be clear. The first is " HEX Code ", we default is this, this is used to display data. So it has to be set here. The second is "string", which is used to display the printed characters, which we can't use here.

(3) , software configuration Ok After that, copy the required instructions to the sending area. Please refer to the module's data manual for specific instructions

(4) , if the module's data manual does not have the test instructions, please calculate yourself, especially to note that the checksum of these two bytes how to calculate the wrong words , the module is not accepting instructions.

How to calculate the checksum code:

```
/*****************************************************************************************************
  - 功能描述：  求和校验
  - 隶属模块：
  - 参数说明：
  - 返回说明：
  - 注：
                和校验的思路如下
                发送的指令，去掉起始和结束。将中间的6个字节进行累加，最后取反码
                接收端就将接收到的一帧数据，去掉起始和结束。将中间的数据累加，再加上接收到的校验
                字节。刚好为0.这样就代表接收到的数据完全正确。
*****************************************************************************************************/
void DoSum( INT8U *Str, INT8U len)
{
    INT16U xorsum = 0;
    INT8U i;

    for(i=0; i<len; i++)
    {
        xorsum = xorsum + Str[i];
    }
    xorsum      = 0 -xorsum;
    *(Str+i)    = (INT8U)(xorsum >>8);
    *(Str+i+1)  = (INT8U)(xorsum & 0x00ff);
}
```

0 = 24 + X class than 0000 0000（0）=0010 0100（ ）+1101 1011（db+1）

Appendix I: **Spi-flash** Comparison of capacity and audio length

Schedule 1-1YX5300-24SS FLASH Volume vs. audio time length swap table:(unit: S)

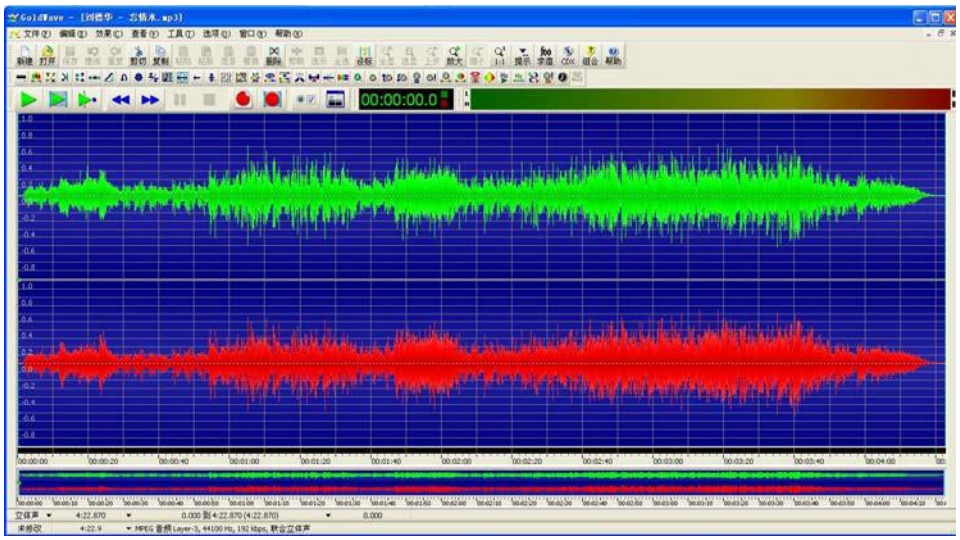
Capacity / Code rate	4MBits	8MBits	16MBit	32MBit	64MBit	
16Kbps	252	505	1011	2022	4045	
24Kbps	163	327	654	1309	2618	
32Kbps	113	226	453	906	1812	
64Kbps	59	119	239	477	955	
96Kbps	41	81	162	325	651	
128Kbps	31	61	123	246	493	
160Kbps	24	49	97	194	389	
192Kbps	20	40	81	161	323	
256Kbps	15	30	60	120	241	
320Kbps	11	23	47	95	191	

Note: MP3 The file size is determined by the code rate, regardless of the sample rate. Voice Broadcast recommended use 16Kbps ~ 64Kbps , music playback recommended use 32Kbps ~ 96Kbps .

Third, the conversion mode of the code rate

1, contention on Spiflash the small capacity, stability of the characteristics, we developed a YX5300-24SS. Directly through the phone

of the microUSB line to update the voice, but for common MP3 documents, most of them are 4M byte or so, using Spiflash, the space seems very laborious. But we generally



do not need a very high sampling rate as a voice broadcast and cue occasion.

from the bottom left corner of the image above, we can see that "the world's first.MP3"sample rate of up to 44100HZ. Bit rate is

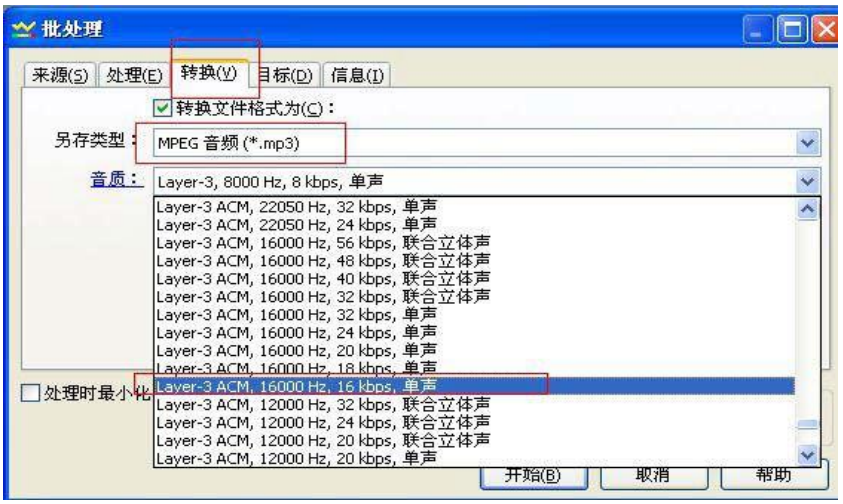
256KBS. This parameter shows that the current song quality is quite good, so it takes up 4.5M of space.

2, but actually we do not need such a high sound quality, then we can compress. are as follows:

Use the Goldwave "This software.

Click batch processing,

Add files



Select "Convert" to set the sample rate to 16000KHZ, the bit rate is 16KBS.

Then specify the path to store the file after the conversion

after compression, 4.5M The file into 507K Out. That's the step.

Note:

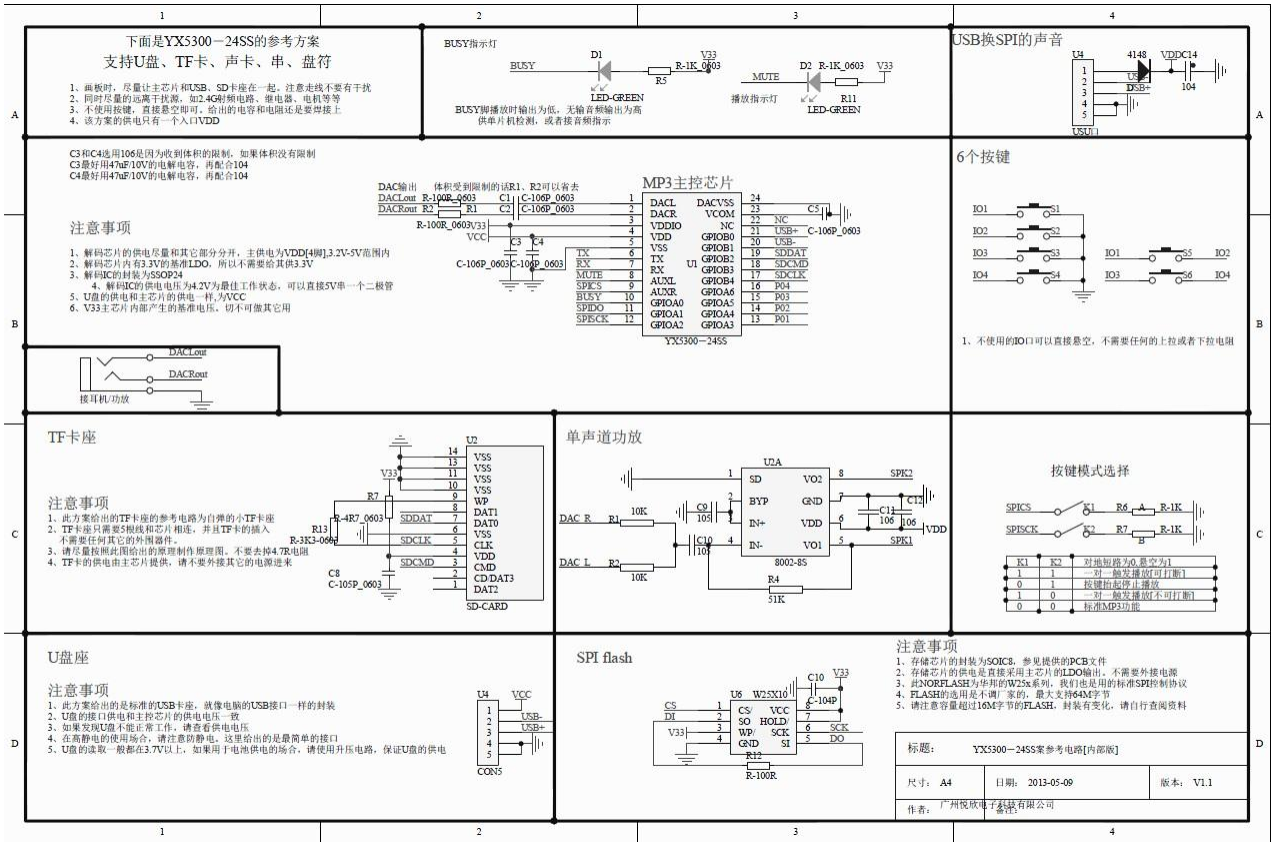
1, if it is wav file, you can also use this software to convert MP3

2 , after the conversion of the effect, the user can directly on the computer to try to listen to the effect of the computer above the effect of playing, and we chip play the effect is consistent

3 , if you think the sound quality is not good, you can increase the sampling rate and bit rate of the two parameters. You can try it yourself .

4 , you can use this software if you need to change the volume of the sound source, as well as the clipping sound source .

Four, the chip schematic diagram



Four, Program Examples

Program Example: Serial port specified playback

```
/*********************************************************************************
 - Realization function: Realize the chip on the power to specify the first and second play, the
   basic program for users to test
 - Day      Period: 2013-05-06
 - Operating Environment:                STC   crystal oscillator:     11.0592M baud rate : 9600
 - Preparation  Note: In Pu-Zhong Technology's  51  Debug on Development Board    OK---stc89c516rd+
1  , the test program must be a module or a chip program in the device online, such as  U  disk,
TF  Card, FLASH
*********************************************************************************/
#include "REG52.h"

#define                         Comm_baud_rate       9600//   serial port baud rate
#define                       Osc_freq       11059200// Operation
Crystal Oscillator: 11.05926MHZ static int8u send_buf[10] = {0};

void Delay_ms (int32u z)
{
     Int32u x=0, y=0;
     for (x=110; x>0; x--)
     for (y=z; y>0;y--);
}

/*********************************************************************************
 - Function Description: Serial port   1  Class
 - Note       : Set to   9600  Baud rate
/void Serial_init (void)
{
     Tmod =                          0x20;//   Set up   T1  For the baud rate generator
     Scon =                         0x50;//0101,0000 8   bit data bits, No parity
     PCON =                         0x00
     th1=256-(OSC_FREQ/COMM_BAUD_RATE/32/12);   set to   9600  Baud rate
     tl1=256-(OSC_FREQ/COMM_BAUD_RATE/32/12);
     TR1       = 1;                //     Timer   1
                                   Open it
     REN       = 1;                //     Serial
                                   port   1  Receive
                                   to enable
     Es        = 1;                //     Serial
                                   port   1
                                   Interrupt Enable
                                   to
}
void Uart_putbyte (int8u ch)
{
               sbuf= ch;
     while (! TI) {;}
     TI = 0;
}

/*********************************************************************************
 - function Description: The serial port sends out the command [ including controls and Queries ]
```

```
 - Parameter description:   CMD:   to indicate control instructions, consult the instruction
   list, as well as the related instructions for the query.
               Feedback: Whether you need to answer [0: no answer required, 1: answer required ]
               Data:   parameters to transfer
**************************************************************************************/
```

```
void Sendcmd (int8u len)
{
    int8u i = 0;
    Uart_putbyte (0x7E); Start
    For  (i=0 i<len; i++)// Data
    {
        Uart_putbyte (Send_buf[i]);
    }
    Uart_putbyte (0xEF);//    End
}

/*********************************************************************************
 - Function Description: Sum checksum
 - And check the following ideas:
    Send the instructions to remove the start and end. Will the middle of the   6   Byte to
add, and finally the inverse code. The receiving side will receive a frame of data, remove
the start and end. The intermediate data is added together with the received checksum byte.
Just for   0.   This means that the data received is completely correct.
/void Dosum (int8u *str, int8u len)
{
    int16u xorsum = 0; int8u i;
    for (i=0; i<len; i++)
    {
                    xorsum= xorsum + str[i];
    }
                  Xorsum= 0-xorsum;
               * (str+i) = (int8u) (Xorsum >>8);
     * (str+i+1) = (int8u) (Xorsum & 0X00FF);
}

void Uart_sendcmd (int8u CMD, int8u feedback, int16u dat)
{
    Send_buf[0] =          0xff;//
    reserved byte send_buf[1] =
                           0x06;//
    length
    SEND_BUF[2] =            cmd;//
    Control instruction send_buf[3] =
    feedback;// need feedback
    SEND_BUF[4] = (int8u) (DAT >> 8);//datah
    send_buf[5] =        (int8u)     (DAT);
                        Datal        dosum
    (&send_buf[0],6);            Checksum
                        Sendcmd     (8);//
    Send this frame data
}

void Main ()
{
    serial_init ()//   initialization settings for serial registers
    uart_sendcmd (0x03, 0, 0x01); play the first
    song Delay_ms (1000);   Delay probably   6S
    uart_sendcmd (0x03, 0, 0x02); Play second song
    Delay_ms (1000);   Delay probably   6S
    uart_sendcmd (0x03, 0, 0x04); play the fourth
```

```
    song While (1);
}
```