Voice Synthesize IC

MC93CV402

User's Manual

Ver. 1.9

ABOV Semiconductor Co., Ltd.

Version 1.9
Published by FAE team

Additional information of this manual may be served by ABOV Semiconductor offices in Korea or Distributors.

ABOV Semiconductor reserves the right to make changes to any information here in at any time without notice.

The information, diagrams and other data in this manual are correct and reliable; however, ABOV Semiconductor is in no way responsible for any violations of patents or other rights of the third party generated by the use of this manual.

Table Of Contents

1. Overview

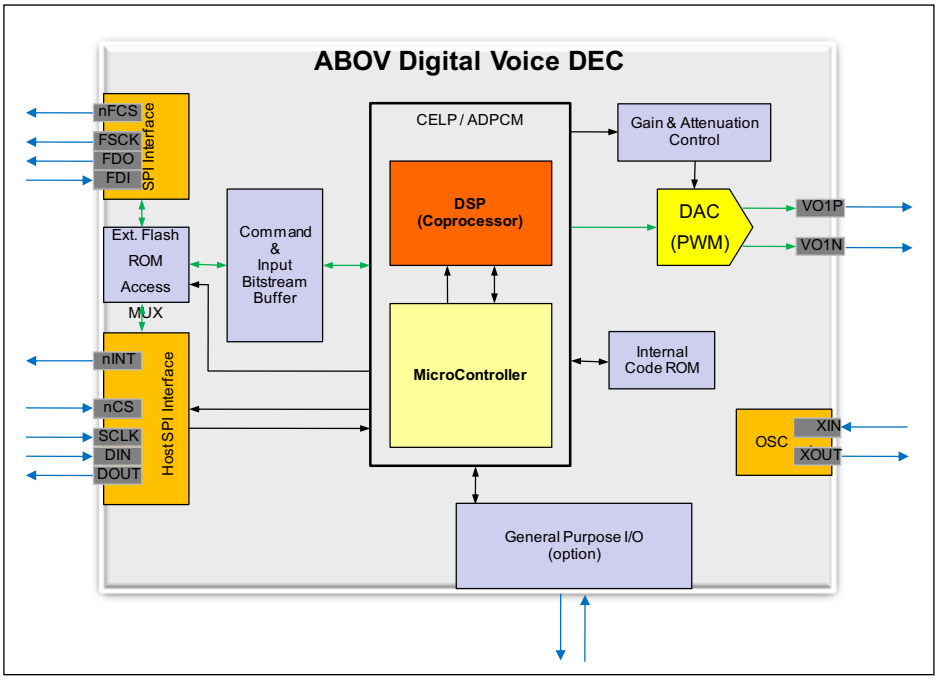


Figure 1-1 MC93CV402 Block Diagram

MC93CV402 is a voice-synthesize IC which has 14 bit digital amplifier inside. And it provides following two decoders.

- Narrow Band SPEEX
- 4-bit ADPCM

SPEEX is a CELP(Code Excited Linear Prediction) based voice en/decoder which is specialized for voice compression. And it provides variety of bit rates. MC93CV402 can provide 'Narrow Band' (8KHz and 12 KHz) sampling frequencies.

Besides, MC93CV402 provides 4 bit ADPCM. It is useful to play music or effect sounds.

Especially, it is possible to play both SPEEX and ADPCM sound at the same time. It is useful when playing voice with background music.

SPEEX and ADPCM sounds can be synthesized with different bit rates. And it is possible to control playing speed and the each volume separately.

VM(Voice Message) feature is provided. It means it is possible to make a message by synthesize some voices. It is useful to reduce the duplicated voice data.

MC93CV402 uses an external SPI Flash ROM for sound data. So the ROM size is selectable.

2. Getting Started

2.1 Hardware Configuration

Chapter 3 and 4 describes packages and pin assignments. And chapter 5 describes hardware configuration. Especially, "5 Hardware Configurations" is essential to make it work. So please check it before design your hardware.

2.2 Control

MC93CV402 is controlled by SPI communication. So please read the chapter "6 SPI Control Command" and understand how it works. And at chapter "7 Major Functions" you can see how to use major functions such as 'initialization', 'play', 'stop', 'volume control',… etc.

In order to test the SPI communication, there are some test commands. If you read '0x02' or '0x03' port by read command 0x49('I'), "B" or 0x03 is returned based on the port number. It is useful to test the SPI communication. Besides this test is working even without PLL circuit and Serial FLASH ROM. (Note that if the first byte of the ROM is 0x02, it is not working in normal mode. So if you are not sure about it, please remove the ROM when you test it.)

Please see the chapter "8 SPI test" for more information.

2.3 Sound Data

MC93CV402 uses a Serial FLASH by SPI communication for the sound data. It is possible to synthesize the sound data from 'WAV' files with 'WaveStudio'. Please see the manual of 'WaveStudio' in order to know more about it.

3. Package And Pin Assignment

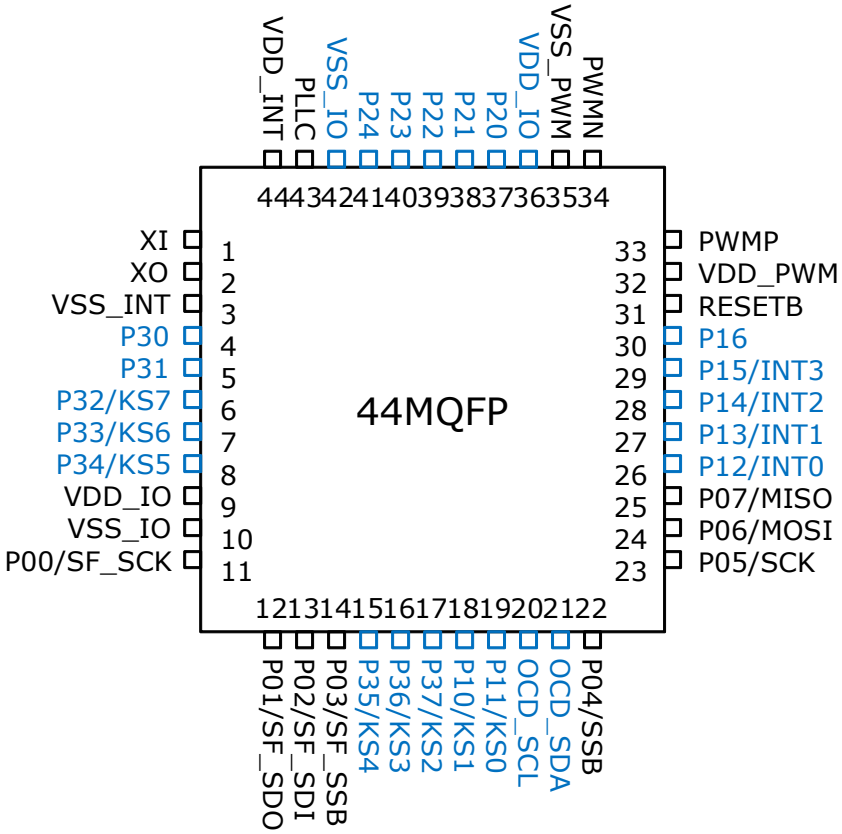


Figure 3-1 Pin map for the 44 pin package (44MQFP)

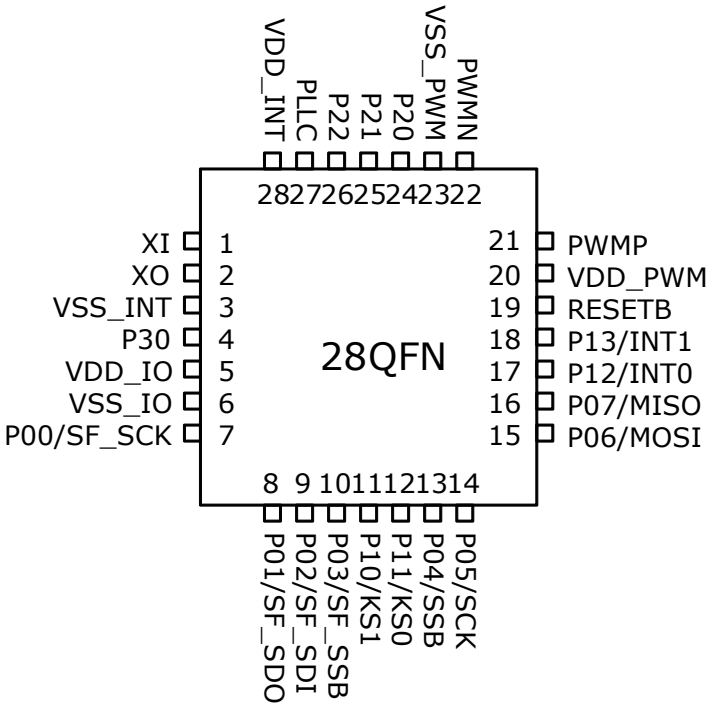


Figure 3-2 Pin map for the 28 pin package (28QFN)

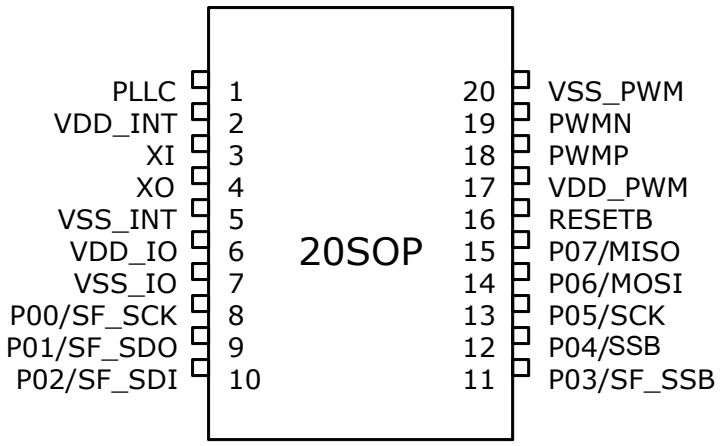


Figure 3-3 Pin map for the 20 pin package (20SOP)

Table 3-1 Pin Description

PIN	I/O	Function	@Reset	Shared With
RESETB	I	System Reset	-	-
XI	I	Resonator or Crystal	-	-
XO	O			
R00	I/O	▶ Port R0. ▶ Can be set in input or output mode in 1-bit units. ▶ Internal pull-up register can be used via software when this port is used as input or open drain port. ▶ Open Drain enable register can be used via software when this port is used as output port.	Input	Serial Flash SCK
R01				Serial Flash SDO
R02				Serial Flash SDI
R03				Serial Flash SSB
R04				SPI SSB
R05				SPI SCK
R06				SPI MOSI
R07			Output	SPI MISO
R10	I/O	▶ Port R1. ▶ Can be set in input or output mode in 1-bit units. ▶ Internal pull-up register can be used via software when this port is used as input or open drain port. ▶ Open Drain enable register can be used via software when this port is used as output port.	Input	Key Scan 6
R11				Key Scan 7
R12				External Interrupt 0
R13				External Interrupt 1
R14				External Interrupt 2
R15				External Interrupt 3
R16				
R20	I/O	▶ Port R2. ▶ Can be set in input or output mode in 1-bit units. ▶ Internal pull-up register can be used via software when this port is used as input or open drain port. ▶ Open Drain enable register can be used via software when this port is used as output port.	Input	
R21				
R22				
R23				
R24				
R30	I/O	▶ Port R3. ▶ Can be set in input or output mode in 1-bit units. ▶ Internal pull-up register can be used via software when this port is used as input or open drain port. ▶ Open Drain enable register can be used via software when this port is used as output port.	Input	
R31				
R32				Key Scan 0
R33				Key Scan 1
R34				Key Scan 2
R35				Key Scan 3
R36				Key Scan 4
R37				Key Scan 5
PWMP	O	PWM Driver	-	-
PWMN				
DSCL	I/O	On Chip Debugger Interface	-	-
DSDA				
PLLC	O	PLL Loop Filter	-	-
VDD_PWM	Power	Power for PWM output	-	Note: It is ok to connect those three power ports to one power source. Even so it is more safety then one power port.
VSS_PWM				
VDD_IO	Power	Power for I/O ports		
VSS_IO				
VDD_INT	Power	Power for Internal Operation		
VSS_INT				

4. Package Dimension

Note that all dimensions are in millimeter.

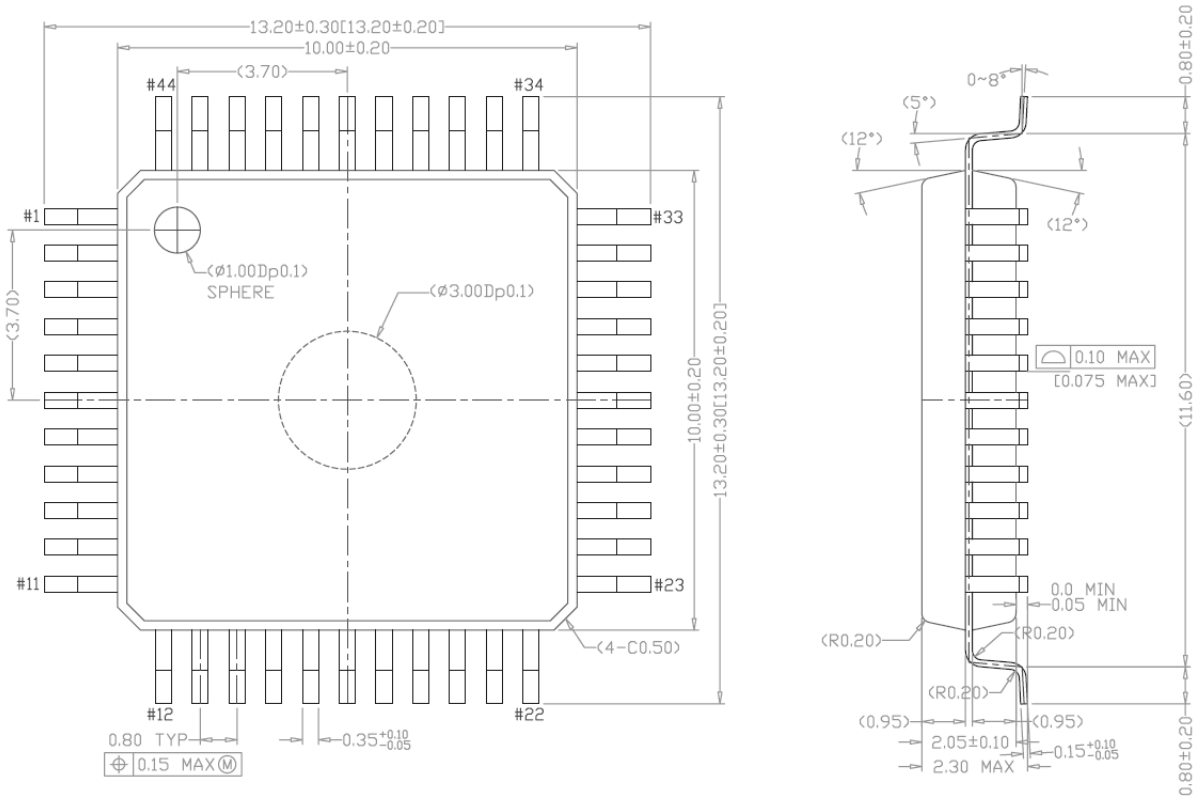
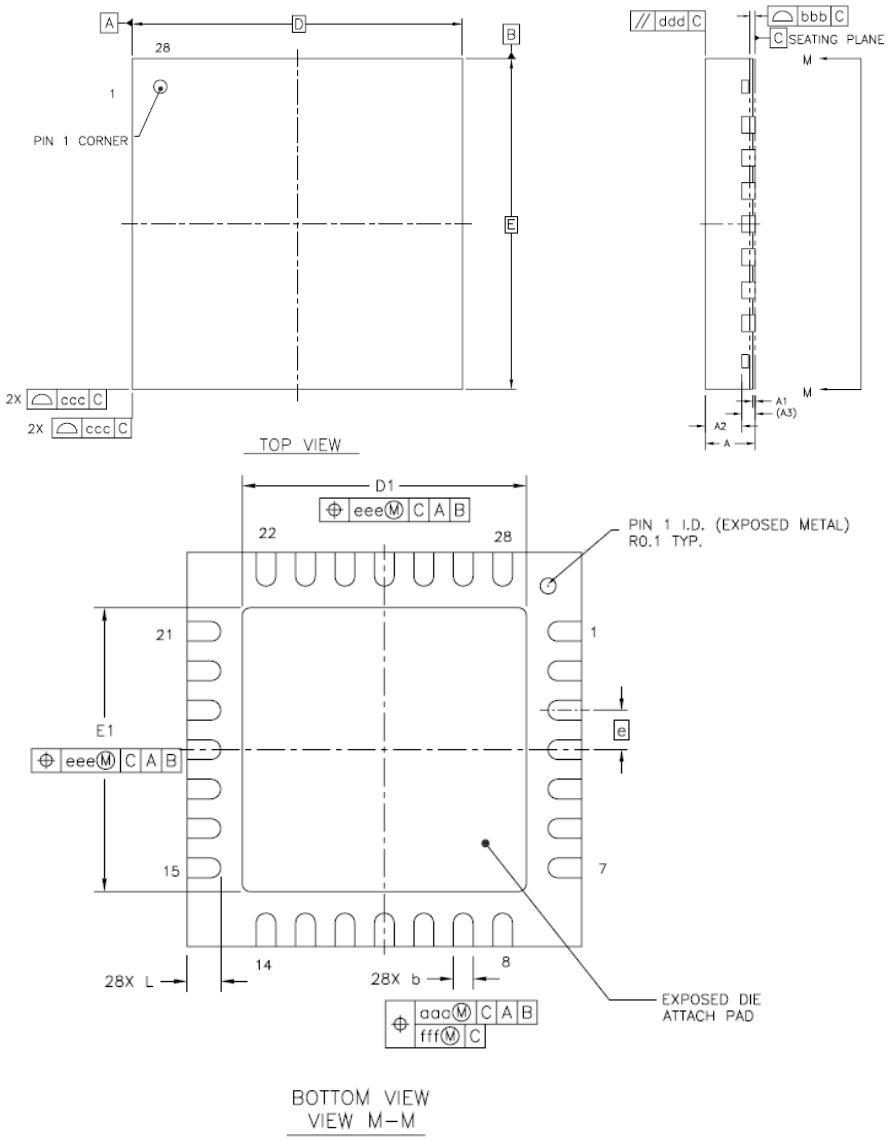


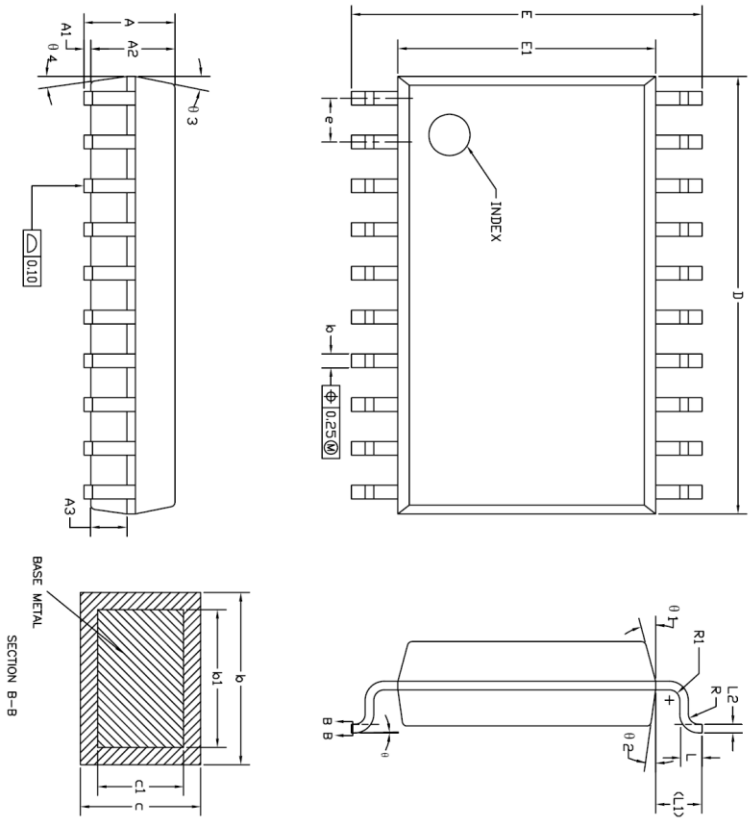
Figure 4-1 44MQFP package diagram



FOR CUSTOMER ONLY				
PACKAGE TYPE	QFN			
PIN COUNT	28			
DESCRIPTION	SYMBOL	MILLIMETER		
		MIN	NOM	MAX
TOTAL THICKNESS	A	0.7	0.75	0.8
STAND OFF	A1	0	0.035	0.05
MOLD THICKNESS	A2	---	0.55	0.57
MATERIAL THICKNESS	A3	---	0.203	---
PACKAGE SIZE	D	4.9	5	5.1
	E	4.9	5	5.1

EP SIZE	D1	3.5	3.6	3.7
	E1	3.5	3.6	3.7
LEAD LENGTH	L	0.3	0.4	0.5
LEAD PITCH	e	0.5		
LEAD WIDTH	b	0.2	0.25	0.3
LEAD POSITION OFFSET	aaa	0.10		
LEAD COPLANARITY	bbb	0.08		
PACKAGE EDGE PROFILE	ccc	0.10		
MOLD FLATNESS	ddd	0.10		
EP POSITION OFFSET	eee	0.10		
	fff	0.05		

Figure 4-2 28QFN package diagram



SYMBOL	MIN	NOM	MAX
A	2.35	2.52	2.65
A1	0.10	0.20	0.30
A2	2.05	2.35	2.55
A3	0.90	1.00	1.10
b	0.35	–	0.49
b1	0.35	0.40	0.45
c	0.23	–	0.32
c1	0.20	0.25	0.30
D	12.60	12.70	12.80
E	10.00	10.20	10.60
E1	7.40	7.50	7.60
e	1.27BSC		
L	0.50	0.80	1.27
L1	1.35REF		
L2	0.25BSC		
R	0.07	–	–
R1	0.07	–	–
θ	0°	–	8°
θ1	10°	12°	14°
θ2	6°	8°	10°
θ3	9°	11.5°	14°
θ4	6°	8°	10°

Figure 4-3 20SOP package diagram

5. Hardware Configuration

5.1 Interface with Microcontroller

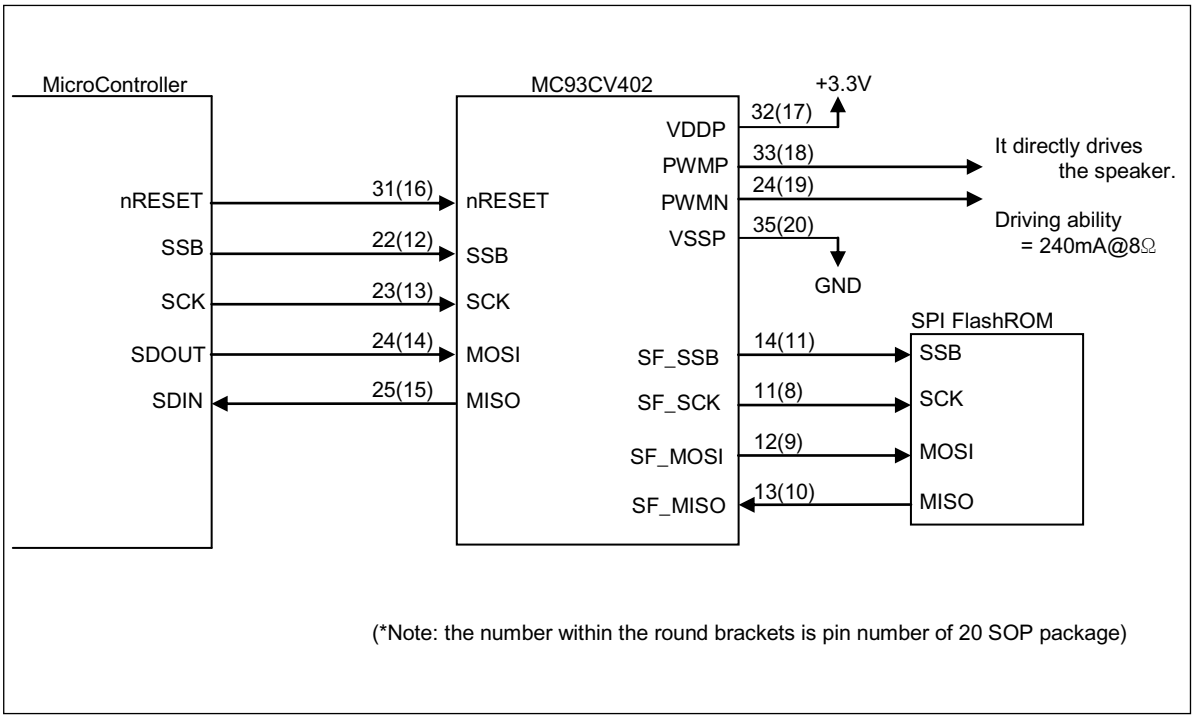


Figure 5-1 Example of interface with microcontroller

It is recommended to drive the reset signal directly by controller even it is possible to use an external reset circuit. It makes it possible that the controller can reset it in abnormal situation.

The operating voltage range is 3.0v ~ 3.6v and 3.3v is recommended. Over voltage may cause the damage on MC93CV402. So please be careful when designing the hardware.

VDD_INT, VDD_PWM and VDD_IO are separated for more safety. But it is ok to connect those power ports to one power souce.

5.2 Clock

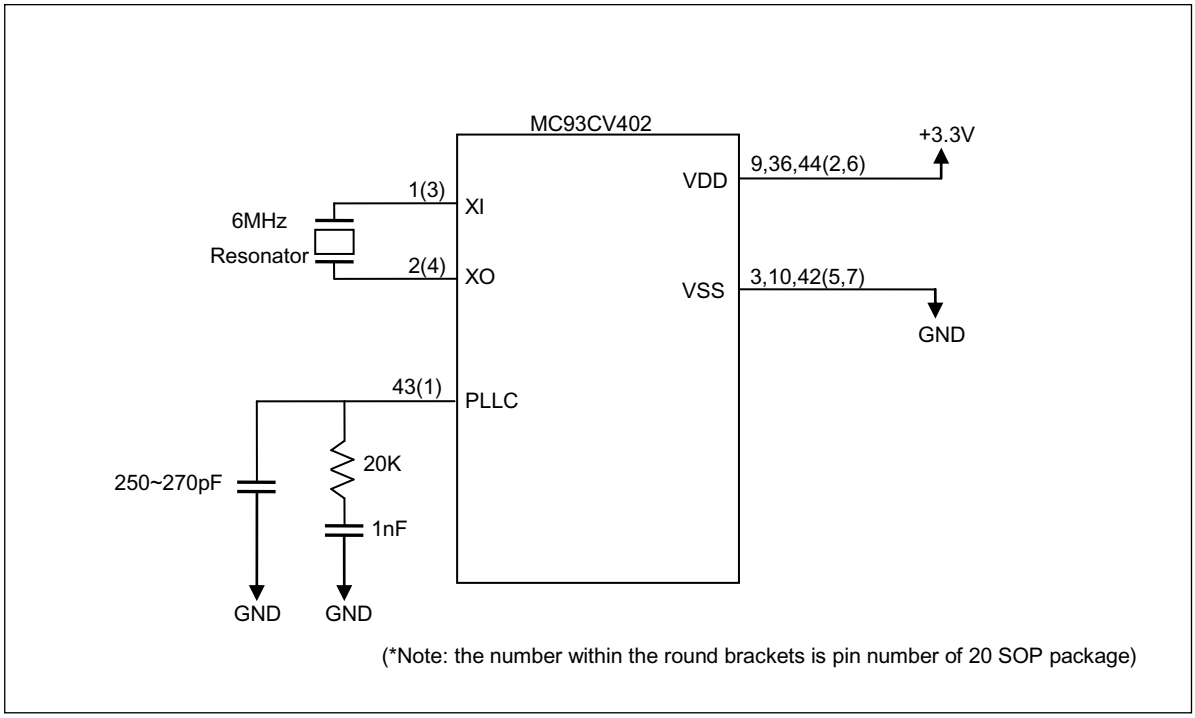


Figure 5-2 Clock circuit

The system clock must be 6MHz. Both crystal and resonator are available.

Please make the PLL circuit exactly same with above “Figure 5-3”. All resister and capacitor values in the figure are recommended. So please do not change it before you completely understand what you are doing.

6. SPI Control Command

MC93CV402 is controlled by SPI communication. And one command frame is consisted of three bytes.

6.1 SPI Timing

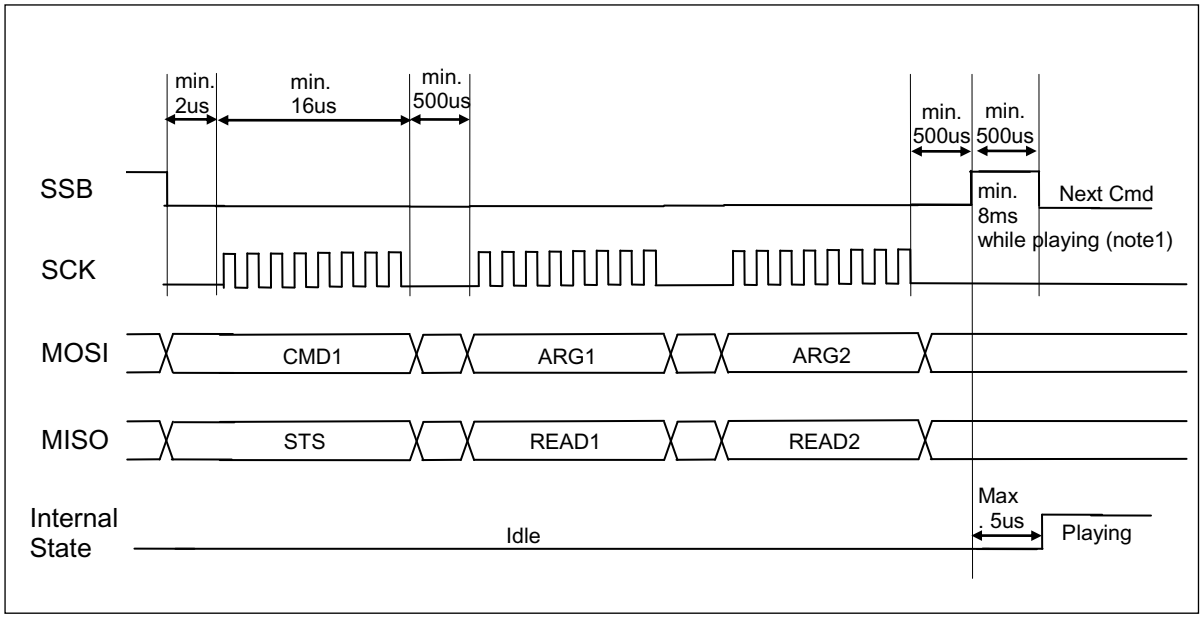


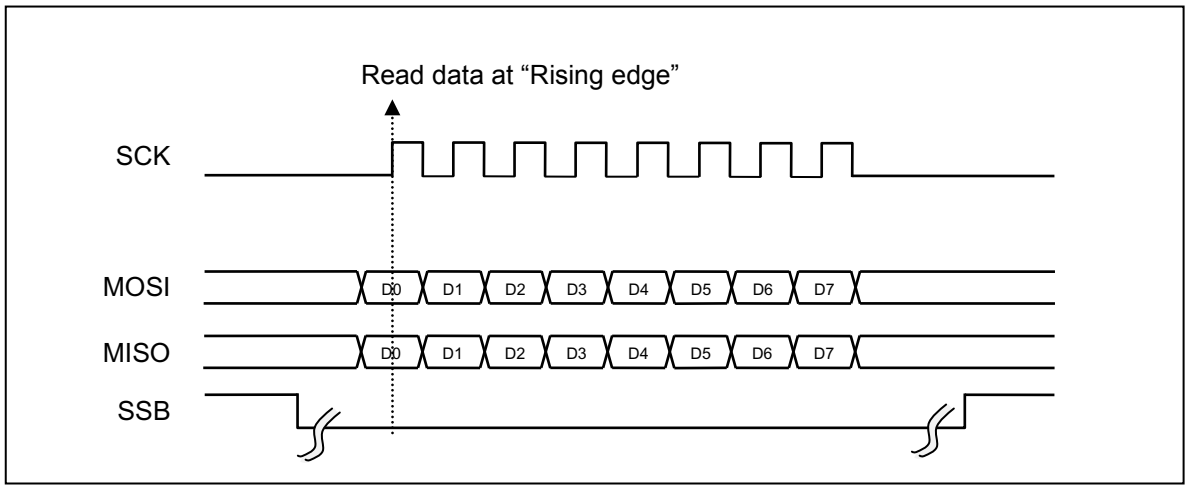
Figure 6-1 SPI command timing

- CMD1: command code
- ARG1,ARG2: arguments
- STS: status flag
- READ1,READ2: return data

Falling SSB is a start signal of one command frame. So SSB must falling at starting and rising at ending.

Over the 500us delay is required between each byte and each command frame.

Note 1) While playing time, the spi command processing can be delayed. So more than 8ms delay is required between frames.



6-2 SPI Byte Timing

In SPI communication, the data is valid when SCK is HIGH status. It means data line must be changed while SCK is LOW status. We call this property as a ‘Polarity’.

And idle status of SCK is LOW. Some device uses a opposite signal. We call it as a ‘Phase’.

Those ‘Polarity’ and ‘Phase’ must be same between the master and slave. Some MCUs are different and some MCUs can select it.

It is recommendable to make a SPI signal with general purpose I/O ports. Because, It is simple to implement.

6.2 Control Commands

All control commands are started with one of following command codes.

Table 6-1 Command codes

	CMD	HEX	Function	Description
Play	Q	0x51	Play with bank address	0x51,mm,nn: Play mmnn'th bank.
	P	0x50	Play with bank number	0x50,bank_number,mode (mode=0x01:single,mode=0x10:repeat)
	S	0x53	Stop playing	0x53,which,0x00 (which=1:speex channel , which=2:adpcm channel)
	X	0x58	Cancel all playing	0x58,0x00,0x00
Volume	M	0x4D	Speex volume control	0x4D,vol,0x00 (vol= 0x00 ~ 0x7F)
	N	0x4E	ADPCM volume control	0x4E,vol,0x00 (vol= 0x00 ~ 0x7F)
	V	0x56	Volume normalize value	0x56,vol,0x00 (shift value=0x00 ~ 0x0F)
Control	F	0x46	SPI Flash ROM bypass mode	0x46,0x00,0x00 Make a direct communication channel to Serial FLAHS. See "7.8 By Pass Mode"
	I	0x49	Read from port	0x49,port1,port2
	O	0x4F	Write to port	0x4F,port,data
	r	0x72	Read XDATA	0x72,mm,nn: read xdata at 0xmmnn (read from internal xdata area)
	c	0x63	Read code memory	0x63,mm,nn: read at 0xmmnn in code ROM
	i	0x69	Read RAM data	0x69,mm,nn: read at 0xmm in RAM area
	D	0x44	Set DAC sampling value	0x44,mm,xx : set DAC value with mm (ddac_con register)
	C	0x43	Set play profile	0x43,mm,nn: set play profile with mm nn is must be zero
	K	0x48	Set PLL	arg1=0: PLL OFF arg1=1: PLL ON arg1=2: PLL/2 arg1=0xFE: set PLLD and turn on PLL

- All numbers in above table are hexadecimal.
- Warning: PLL must be activated before send a play command

  (Send a 0x48-0x01-0x00 command)
- Warning: All reading data is returned while next command is sending.

6.3 Control Ports

With the control command 0x49('I) and 0x4F('O'), it is possible to read and write the MC93CV402's internal status. At this time we use a term 'Control Port' to refer the reading or writing target.

Table 6-2 Control ports

HEX	Name	Description	Note
0x00	PLAY_STS	Play status	Read only
0x01	'B'	'B'(0x42) return (for testing)	Read only
0x02	0x03	0x03 return (for testing)	Read only
0x80	R0DA	sfr R0DA read/writing	
0x86	SCCR	Clock Control Register	See 7.11 STOP Mode
0x87	PCON	Power Control Rigster	0x03 : STOP 0x01 : SLEEP
0x9D	R0IO	sfr R0IO read/writing	
0x94	R0PU	sfr R0PU read/writing	
0x85	R0OD	sfr R0OD read/writing	
0x84	R0DB	sfr R0DB read/writing	
0x88	R1DA	sfr R1DA read/writing	
0x89	R1IO	sfr R1IO read/writing	
0x8A	R1PU	sfr R1PU read/writing	
0x8B	R1OD	sfr R1OD read/writing	
0x8C	R1DB	sfr R1DB read/writing	
0x90	R2DA	sfr R2DA read/writing	
0x91	R2IO	sfr R2IO read/writing	
0x92	R2PU	sfr R2PU read/writing	
0x93	R2OD	sfr R2OD read/writing	
0x94	R2DB	sfr R2DB read/writing	
0x98	R3DA	sfr R3DA read/writing	
0x99	R3IO	sfr R3IO read/writing	
0x9A	R3PU	sfr R3PU read/writing	
0x9B	R3OD	sfr R3OD read/writing	
0x9C	R3DB	sfr R3DB read/writing	
0xD9	SF_SPEED	SPI FlashROM Speed read/writing	

Play status is consist of following bits.

PLAY_STS

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
-		ADPCM	SPEEX	-	-	-	START

ADPCM ADPCM play status

0 : ADPCM sound is not playing

1 : ADPCM sound is playing now

SPEEX SPEEX play status

0 : SPEEX voice is not playing

1 : SPEEX voice is playing now

-

-

START Sound start bit

0 : sound is playing or stop

1 : It is set when playing command is accepted.
And cleared when the sound play is started.

7. Major Functions

7.1 Initialization

Following initialization procedure is required to use MC93CV402.

Table 7-1 Initialization procedure

Order	CMD	Command Frame	Description
1	'D'	0x44-0x15-0x00	DDAC_CON=0x15
2	'K'	0x4B-0xFE-0xE1	PLLD=0xE1 and PLL Enable
3	'O'	0x4F-0x84-0xFF	Debounce Enable on SPI communication port.
3	'V'	0x56-0x06-0x00	Set „Normalize Shift" with 6
4	'M'	0x4D-0x7F-0x00	Set SPEEX volume with 0x7F
5	'N'	0x4E-0x7F-0x00	Set ADPCM volume with 0x7F
6	'O'	0x4F-0x08-0x02	sf_dma_speed=0x02
7	'O'	0x4F-0xD9-0x02	sf_cache_speed=0x02
8	'C'	0x43-0x05-0x00	profile: SPEEX=3, ADPCM=3

Note that it is required to reset MC93CV402. In order to do it, please keep nRESET signal in LOW status over 2us and set it HIGH status.

7.2 Play

(0x50-'P') command is used to play a sound.

Command frame : 0x50-[bank]-[type]

Bank is number of playing sound.

Type represents that it is repeat or not

- 0x01: play once
- 0x10: repeating

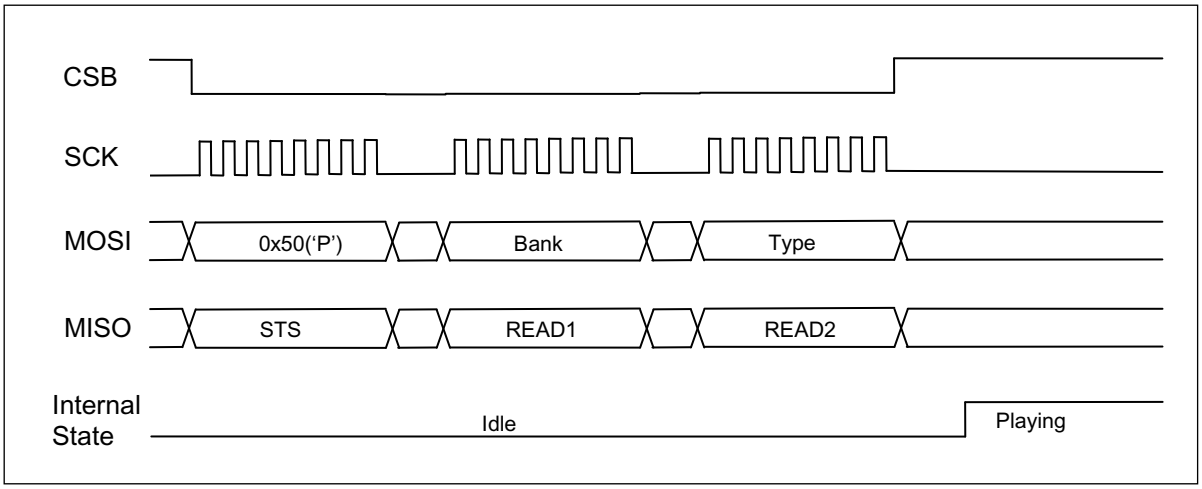


Figure 7-1 Play command timing

7.3 Stop Playing

(0x53-'S') command is used to stop playing.

Command frame : 0x53-[type]-[0x00]

Type is a decoding type of stopping sound.

- 0x01: SPEEX
- 0x10: ADPCM

Note that, it only stops the current playing sound even some sounds are played sequentially by VM(Voice Message). So if you want to stop all sounds at once, please use the 'Cancel all playing' command.

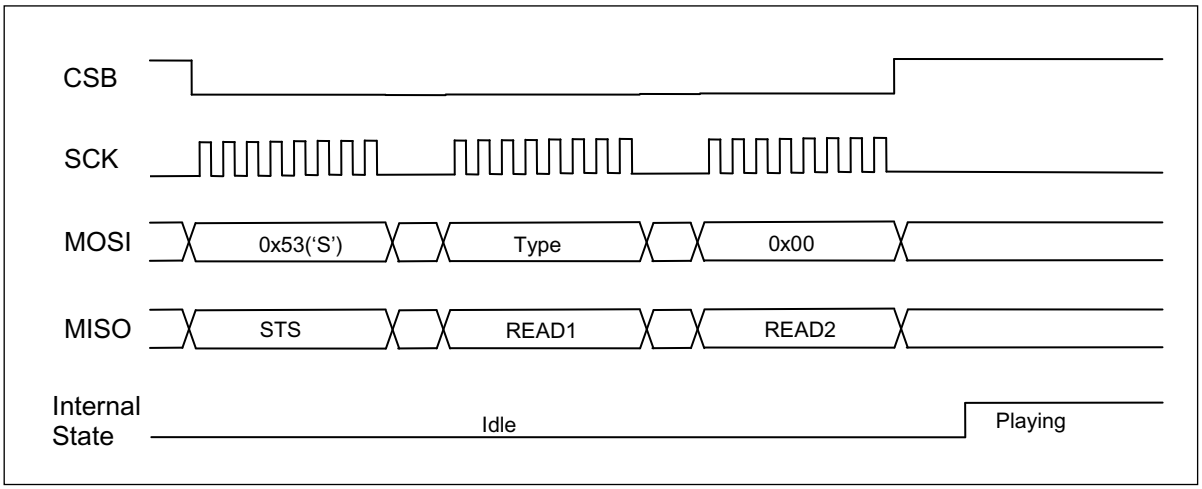


Figure 7-2 Stop command timing

7.4 Cancel All Playing

(0x58-'X') command is used to cancel all playing.

Command frame : 0x58-0x00-0x00

Note that, it takes a little time to cancel all playing. So user must take a delay time at least 8ms(Because spi command processing can be delayed in playing time) and make sure all playing is finished by described method in "7.5 Check Play Status".

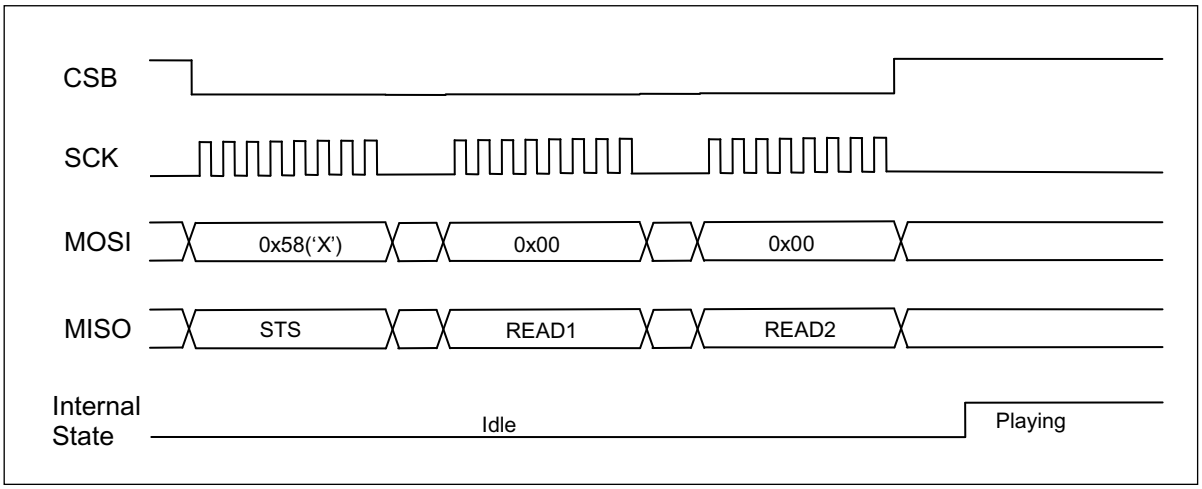


Figure 7-3 Cancel all playing command timing

7.5 Check Play Status

It is possible to check play status by reading one of control ports the PLAY_STS. But another method is required to check VM is terminated. Even the VM is not finished PLAY_STS returns zero(there is no playing sound) while some command like W(Wait) is executing.

So in order to check VM is finished the 0x69('i') command is required. When you read the ram data at 0x08, it returns internal VM status. When there are commands to process in current VM, it returns 0x80. And if there is nothing to process in current VM, it returns 0x00.

Note that it returns 0x00 while MC93CV402 process the last command in the VM. So if the last command is playing command, it is required to read the PLAY_STS to check VM is finished or not. If the last command is not a playing command, you must consider the last command's processing time. W(Wait) command only takes given waiting times. So it is easy to figure out how long it takes. And other commands will only takes few micro seconds. So it is not that important.

Normally W(Wait) command is not used as the last command of VM and the other commands processing time is ignorable. So in that case it is possible to check the VM is finished or not by following example code.

- send_byte() is a function that send/receive one byte by SPI.
- delay_500us() is a 500us delay function.
- V_LCS() makes Chip Select Signal to LOW(means VoiceIC activate).
- V_HCS() makes Chip select Signal to HIGH(means VoiceIC deactivate).

```
int vm_check()
{
    int r1,r2,r3;

    //PLAY_STS will be returned at next command.
    V_LCS();
    r1=send_byte('I');
    delay_500us();
    r1=send_byte(0x00);
    delay_500us();
    r1=send_byte(0x00);
    delay_500us();
    V_HCS();

    delay_500us();

    //VM status will be returned at third byte.
    V_LCS();
    r1=send_byte('i');
    delay_500us();
    r2=send_byte(0x08);
    delay_500us();
    r3=send_byte(0x00);
    delay_500us();
    V_HCS();

    return r2+r3;
 }
```

In above code, PLAY_STS value is stored at “r2” and VM status value is stored in “r3”. So if “r2+r3” value is not zero, it means the VM is not finished yet.

7.6 Volume Control

There are 128 volume levels in MC93CV402. The volume value is signed 1byte from -128 to 127. (So over the 0x7F values are assumed as a minus volume.)

See following diagram to know how it works.

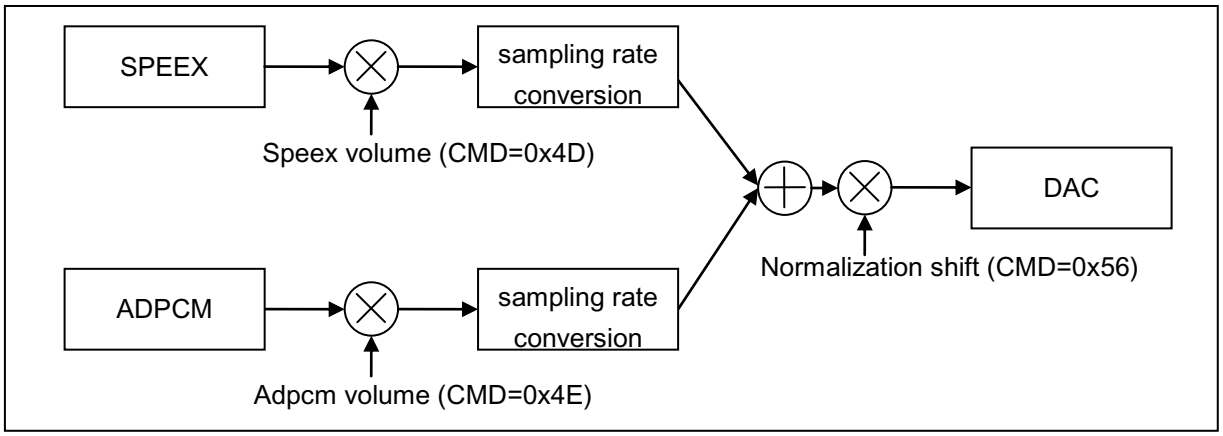


Figure 7-4 Flow diagram of playing sound

It is possible to control both SPEEX and ADPCM separately. And it also possible to control the synthesized sound by normalization shift value.

Following equation describes how normalization shift value changes the volume.

$$\text{Volume}_{\text{OUT}} = \text{Volume}_{\text{IN}} / 2^{\text{Normalization shift}}$$

dB representation of the volume

The reference level is the initial volume setting values ‘Normalized Shift: 6, Volume: 127’.

7-2 dB representation of the volume and setting value

dB	Normalized Shift Value	Volume Value		dB	Normalized Shift Value	Volume Value	
		HEX	DEC			HEX	DEC
12	4	7E	126	-11	6	24	36
11	4	70	112	-12	6	20	32
10	4	64	100	-13	6	1C	28
9	4	59	89	-14	6	19	25
8	4	4F	79	-15	6	17	23
7	4	47	71	-16	6	14	20
6	5	7E	126	-17	6	12	18
5	5	71	113	-18	7	10	16
4	5	64	100	-19	8	E	14
3	5	59	89	-20	9	D	13
2	5	50	80	-21	10	B	11
1	5	47	71	-22	11	A	10
0	**6**	**7F**	**127**	-23	12	9	9
-1	6	71	113	-24	13	8	8
-2	6	65	101	-25	14	7	7
-3	6	5A	90	-26 ~ -27	15	6	6
-4	6	50	80	-28 ~ -29	17	5	5
-5	6	47	71	-30 ~ -31	19	4	4
-6	6	40	64	-32 ~ -34	21	3	3
-7	6	39	57	-35 ~ -38	24	2	2
-8	6	33	51	-39 ~ -48	28	1	1
-9	6	2D	45	-49	38	0	0
-10	6	28	40				

Warning: Volume setting with over the initial values can course the sound noise.

7.7 Playing Speed

It is possible to change the play speed by changing the sampling rate. 0x44('D') command is used to change it by modifying the internal DDAC_CON register.

Following table shows the sampling rates based on ddac_con values.

Table 7-3 Sampling rate table based on DDAC_CON register

DDAC_CON	Sampling rate(@6.144MHz)	Sampling rate(@6MHz)	note
0x1D	8 KHz	7812.50 Hz	x16 oversampling
0x15	12 KHz	11718.75 Hz	x16 oversampling
0x19	16 KHz	15625.00 Hz	x16 oversampling
0x11	24 KHz	23437.50 Hz	x16 oversampling
0x1B	32 KHz	31250.00 Hz	x8 oversampling

So, sound data must be encoded with above sampling rate based on the system clock. Or the system clock must be provided based on the sampling frequency of sound data. Normally 6MHz resonator is recommended. So, it is recommend to generate the sound data for 6MHz.

7.8 Profile

Bit rate of SPEEX and ADPCM can be selectable from following combinations.

Table 7-4 Sound synthesize profile

Profile No.	SPEEX	ADPCM	rate
1	8K	8K	1:1
2	8K	12K	1:1.5
3	8K	16K	1:2
4	12K	8K	1.5:1
5	12K	12K	1:1
6	12K	16K	1.5:2 *Note
7	16K	16K	1:1 *Note

Note: Profile No. 6 and 7 can't provide playing both ADPCM and SPEEX at the same time. Only playing alone is possible with Profile No 6 and 7.

Up-sampling is possible. With this feature it is possible to increase the quality of low rate sound. Following table shows the possible up-sampling rates.

Table 7-5 Up-Sampling list

No.	Up-Sampling Rate	
0x00	1.0	-
0x10	2.0	8K→16K
0x20	1.5	8K→12K
0x30	1.25	12K→16K

0x43('C') is used to set both play profile and up-sampling rate. To do that profile number plus up-sampling number is used as a argument of 0x43('C') command.

For example, the profile number 0x01 means "SPEEX 8K + ADPCM 8K" combination. And Up-Sampling No 0x10 means double rate. So 0x11 is used with 0x43('C') command to set it like that.

Table 7-6 Play profile example

argument	combination	Play sampling rate	DDAC_CON value
0x01	SPEEX 8K + ADPCM 8K	8K	0x1D
0x11	SPEEX 8K + ADPCM 8K	16K (x2 playing)	0x19
0x32	SPEEX 8K + ADPCM 12K	16K (x1.25 playing)	0x19
0x21	SPEEX 8K + ADPCM 8K	12K (x1.5 playing)	0x15

7.9 Reading Port

Reading port is consist of two command frames. At first reading port is addressed by reading command frame. And at second command frame, the port value is returned.

The two argument of 0x49('I') command is a target port numbers. And two values are returned by second command. After then all commands return the two values of the target ports. And the default return value is play status value.

Following diagram shows the timing of reading port command.

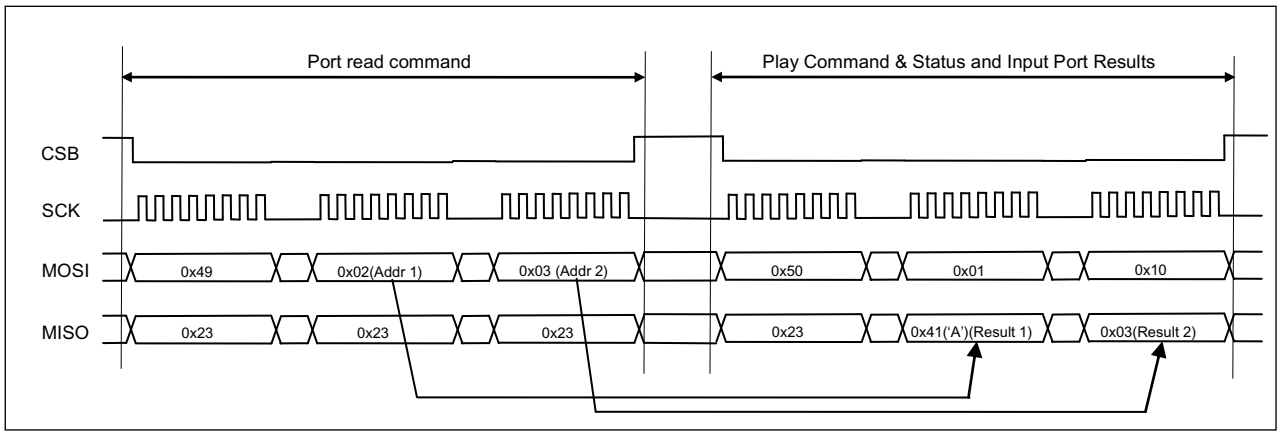


Figure 7-5 Read port timing diagram

7.10 By Pass Mode

By Pass Mode makes a direct channel between controller and Serial FLASH. The controller means the device which controls MC93CV402, and Serial FLASH refers the ROM which has the sound data. In this mode, the controller communicates with Serial FLASH like those two devices are connected directly.

In order to enter the mode, 0x46('F') is used. And the arguments are not important. For example, when "0x46 - 0x00 – 0x00" command is accepted, MC93CV402 enters into By Pass Mode.

In order to break out from the By Pass Mode, sending "0xFF-0xFF" command is used. 0xFF is not used as a Serial FLASH command but can be a data. So remember that it must be translated right after the 'Chip Selection Signal' is enabled. (Right after the 'Chip Selection Signal' is always a command.)

7.11 STOP Mode

It is possible to reduce the current consumption by falling STOP mode when sound is not played. Please see the following procedures.

Entering STOP mode

Here is a entering STOP mode procedure. Note that 'voice_send_cmd(…) is a command sending function.

```
voice_send_cmd( 0x4F, 0x86, 0x0E ); // Changing clock from PLL to 6MHz system clock.

voice_send_cmd( 0x4F, 0x86, 0x00 ); // PLL disable

voice_send_cmd( 0x4F, 0x87, 0x03 ); // Entering STOP mode
```

When the PCON(0x87) register is set by 0x03, it falls in STOP mode. So to do that port writing command 0x4F(O') is used.

Important thing is PLL must be disabled. Of course the clock source must be changed to 6Mhz system clock before PLL is disabled. To do than SCCR(0x86) register is used with the port writing command.

Wake up

It can be waked up from STOP mode by RESET signal or SPI communication. When one byte is received by SPI, it wakes up. In this case only thing to do is PLL initialization. But when it is waked up by RESET signal, whole initialization procedure is required.

It is recommended to wake up by RESET signal to prevent unpredicted mistake. Besides, in most cases all you have to is recall the initialization routine again what is called at the beginning.

```
voice_hw_reset();        //Generates the RESET signal
ms_delay(15);            //15msec delay for System stabilization
voice_init();            //Call the VoiceIC initialization routine
```

It is possible to wake up by SPI communication like following codes.

```
voice_send_cmd(0x00,0x00,0x00);
ms_delay(15);
voice_send_cmd('K',0xFE,0xA3);
ms_delay(2);
```

Warning

1) 15msec delay is required for system stabilization after the dummy data.
2) The dummy data can be only one byte. But it must be finished by SSB signal rising.(to start the next command with SSB falling)

7.12 Deep Power Down Mode

Even MC93CV402 falling in the STOP mode, Serial FLASH ROM still in standby mode. So it is possible to reduce the power consumption by making the Serial FLASH ROM into Deep Power Down Mode.

Following table shows the test results of three samples.

Table 7-7 Test Results of Power Consumption

MC93CV402	MX25L8005 One of 8 M bit Serial FLASH	#1	#2	#3
STOP Mode	Standby Mode	2.54	2.41	2.75
STOP Mode	Deep Power Down Mode	0.55	0.42	0.46

(unit: uA)

Note : This is a test results in the laboratory(Not a specification). So please use this data as a reference.

Entering STOP Mode with Deep Power Down Mode

Actually MC93CV402 can't make the Serial FLASH into Deep Power Down Mode. Instead, user has to do it in 'By Pass' mode(See '7.10 By Pass Mode' for more information). For example 'MX25L8005' is going to Deep Power Down Mode when it receive 0xB9' one byte command. So following example code is working with MX25L8005.(It is working with most SPI type Serial FLASH ROM).

```
voice_send_cmd( 0x46, 0, 0 );        //Entering By Pass Mode
SCK = 1;                             //Default status for spi of Serial FLASH
SO   = 1;
CSS = 0;
flash_send_a_byte( 0xB9 );           //Make the Serial FLASH ROM into Deep Power Down Mode
CSS = 1;

CSS = 0;
voice_send_a_byte( 0xFF );           //Escape from By Pass Mode
voice_send_a_byte( 0xFF );
CSS = 1;

SCK = 0;
SO   = 0;                            //Default status for spi of MC93CV402
//Following codes are same with normal entering STOP mode procedure.
voice_send_cmd( 0x4F, 0x86, 0x0E ); // Changing clock from PLL to 6MHz system clock.
voice_send_cmd( 0x4F, 0x86, 0x00 ); // PLL disable
voice_send_cmd( 0x4F, 0x87, 0x03 ); // Entering STOP mode
```

```
// voice_send_cmd(…); is a function what sends the chip selection signal and 3 byte SPI data.
// CSS is a chip selection signal port, SCK is a clock port and SO is a MOSI port.
// flash_send_a_byte(…); is a function what sends a one byte SPI data for Serial FLASH ROM.
// voice_send_a_byte(…); is a function what sends a one byte SPI data for MC93CV402.
// Note that, pole and phase properties are different between MC93CV402 SPI and MX25L8005 SPI.
// So the sending functions are different for each device.
```

Wake Up

After wake-up from STOP mode, user has to wake-up the Serial FLASH from Deep Power Down Mode. For example 'MX25L8005' is waked-up from Deep Power Down Mode when it receive 0xAB' one byte command. So following example code is working with MX25L8005.(It is working with most SPI type Serial FLASH ROM).

```
//After waked-up from STOP mode.
voice_send_cmd( 0x46, 0, 0 );        //Entering By Pass Mode

CSS = LOW;
flash_send_a_byte( 0xAB );           //Wake up from Deep Power Down Mode
CSS = HIGH;

CSS = LOW;
voice_send_a_byte( 0xFF );           //Escape from By Pass Mode
voice_send_a_byte( 0xFF );
CSS = HIGH;
```

```
// voice_send_cmd(…); is a function what sends the chip selection signal and 3 byte SPI data.
// CSS is a chip selection signal port. And It assumes that 'LOW' and 'HIGH' are defined properly.
// flash_send_a_byte(…); is a function what sends a one byte SPI data for Serial FLASH ROM.
// voice_send_a_byte(…); is a function what sends a one byte SPI data for MC93CV402.
// Note that, pole and phase properties are different between MC93CV402 SPI and MX25L8005 SPI.
// So the sending functions are different for each device.
```

Note

All ports must be in stable mode. For example SCK, CSS and MOSI ports are input ports. So if those are not hold with LOW or HIGH signal, those can be floating. In that case current leakage is occurred by those ports. So user must make it sure that there is no floating port.

8. SPI Test

It is possible to test the SPI communication, with the port reading function. The SPI communication is working even the PLL is disabled and the Serial FLASH ROM is not existed. So it is possible to figure out whether MC93CV402 is working(SPI, 6MHz clock) or not.

The reading function is described at chapter 7.7 Reading Port. The port 0x01 and 0x02 are testing ports. So those ports return 0x42 and 0x03 each.

For example, when 0x49-0x01-0x00 command is accepted twice, 0x42 is returned at the second byte of second command.

Anything can be the third byte(and it is not important). And the return values of first command are meaningless.

8.1 Communication Signal Example

Following figures are the waveform of real communication signal. The upside is captured signal, and the downside is enlarged view. And all the names of each signal are clarified at the left side of each signal.

“Figure 8-1 Signal_example 49_01_A” is an enlarged view of the first byte of the first 0x49-0x01-0x00 command. Data has a meaning when the clock line(green) is HIGH, MSB first. So if you read the data line(blue), you can see it is 0x49 and 0x01.

“Figure 8-2 Signal_example 49_01_B” is an enlarged view of the first byte of the second 0x49-0x01-0x00 command. If you read that, you can see the MC93CV402 returns 0x42 as a second byte.

Next Figures are captured when the 0x49-0x02-0x00 command is transmitted twice. If you read it in same way, you can see it returns 0x03.

Figure 8-1 Signal_example 49_01_A

Figure 8-2 Signal_example 49_01_B

Figure 8-3 Signal_example 49_02_A

Figure 8-4 Signal_example 49_02_B

9. VM(Voice Message)

MC93CV402 has VM(Voice Message) feature inside. It is synthesizing function to make a sound from peace of sounds. It is possible to define a sound source as a VM instead of sound data. At this point, the VM data is consists of play commands sequentially.

For example it is possible to define the bank 0x07 as a meaning of play 0x08 and 0x09 continuously. After then, It is played just like that when the play-0x07-command is accepted. At this case we call the data in the bank 0x07 as a VM.

So it is more efficient than storing the synthesized data of 0x08 and 0x09.

Playing VM is exactly same with playing sound data. Please see the 'WaveStudio' manual to know how to define a VM. 'WaveStudio' is a GUI tool for build the ROM file of sound data.

9.1 'W' Command

In fact the VM message is consist of the SPI commands. So VM works like the SPI commands in the bank are executed continuously. But there is one more command what is used only in VM.

'W' command is only used in VM. This two bytes command means 'wait'. So it can make a delay between two sound playing. More precisely say, 'W-0xNN' means wait until sound playing is finished and make a delay for NN times.

So if 'W'-0x00 is required when continuously play the sound. But when playing an ADPCM sound and a SPEEX sound at the same time, there must be no 'W' command between two play commands.

For example, let's assume that 0x07 is an ADPCM sound and 0x08 is a SPEEX sound.

When the VM is defined as "playing 0x07 and 'W'-0x00 and play 0x08", the 0x08 sound is played after 0x07.

But when the VM is defined as "playing 0x07 and playing 0x08", the 0x07 and 0x08 are played at the same time.

In case of there is no 'W' command between two SPEEX or two ADPCM, the first play command is ignored by next play command.

NN time in above description is 'NN' * 'basic period' time and the 'basic period' time is generated by internal Timer0.

And the equation of 'basic period' is,

$$'\text{basic period}' = \frac{1}{6\text{MHz}} * 2^{\text{T0MR[7:4]}+9} \text{ sec}$$

And the default T0MR[7:4] value is '8' so, default 'basic period' time is,

$$'\text{default basic period}' = \frac{1}{6\text{MHz}} * 2^{8+9} \cong 21msec$$

It is possible to change the 'basic period' by change the T0MR value with 0x4F('O') writing port command. For example 0x4F-0x96-0x71 command is accepted, 7~4 bits of T0MR(@0x96) value is changed into '7'. So the 'basic period' is now,

$$'\text{changed basic period}' = \frac{1}{6\text{MHz}} * 2^{8+9} \cong 10.9msec$$

At here, 0x4F is a command which changes the specific port, and above command means write the value 0x71 at the address 0x96(T0MR).

The default value of T0MR is 0x81. So in above example, we change the ‘basic period’ by changing the higher nibble of T0MR from ‘8’ to ‘7’.

10. SPI FlashROM File Architecture

There are three major areas in SPI FLASH ROM. From the beginning, 0x8000 bytes of area is used for program area. This area is used to run a user's custom code.

Sound data is stored from the 0x8000. And this area is started with index area.

Note that, the 0'th in sound table is not used.

Table 10-1 ROM file architecture

Address	Description	Note
0x00_0000 0x00_7FFF	Program area (When the data of 0x00_0000 is 0x02, custom code is executed.)	
0x00_8000 0x00_87FF	Sound data index table (256-index(bank) * 8-bytes each)	
0x00_8800	Sound or VM data	

Table 10-2 Index table architecture

Position	Description	note
0x00	Start address of data L (Low byte)	
0x01	Start address of data M (Middle byte)	
0x02	Start address of data H (High byte)	
0x03	Not used.	
0x04	Data size L (Low byte)	
0x05	Data size M (Middle byte)	
0x06	Data size H (High byte)	
0x07	Data type	0x00 : not defined 0x01 : SPEEX data 0x02 : ADPCM data 0x40 : VM(Voice Message)

11. Electrical Characteristics

11.1 Absolute Maximum Ratings

Table 11-1 Absolute Maximum Ratings

Parameter	Specifications
Supply Voltage	2.7V ~ 3.6V
Input Voltage	-0.3V ~ VDD+0.3V
Operating Temperature	-40℃ ~ 85℃

11.2 DC Characteristics

Table 11-2 DC Characteristics

Parameter	Symbol	MIN	TYP	MAX	UNIT	Condition
Supply Voltage	VDD_INT VDD_IO VDD_PWM	2.7	3.3	3.6	V	
Operating Current	I_{OP}			30	mA	Except VDD_PWM
PWM output Current	I_{PWM}	150	190	-	mA	2.7v VDD_PWM
		160	200	-		3.0v VDD_PWM
		180	220	-		3.3v VDD_PWM
		190	240	-		3.6v VDD_PWM
Sleep Current	I_{SLEEP}			2	mA	
Stop Current	I_{STOP}			10	uA	
Input Voltage	V_{IH}	0.8 VDD			V	
	V_{IL}			0.2 VDD		
Input Current	I_{IH}			1	uA	
	I_{IL}	-1				
Pull-up Resistance	R_{RESETB}	200K		400K	Ω	
	R_{PORT}	40K		70K		
PWM Output Current	I_{PWM}	200	240	280	mA	Load = 8 Ω

11.3 AC Characteristics

Table 11-3 Fundamental AC Characteristics

Parameter	Symbol	PIN	MIN	TYP	MAX	UNIT
Operating Frequency	fMCP	XI		6		MHz
Oscillation Stabilization Time	tMST1	XI, XO		6		ms
External Clock “H” or “L” Pulse Width	tCPW	XI	90			ns
External Clock Transition Time	tRCP, tFCP	XI			10	ns
RESETB Input Pulse “L” Width	tRST	RESETB	2			us
Reset to System Ready Time	tSYS_READY	SPI_SSB(P0[4])	15			ms

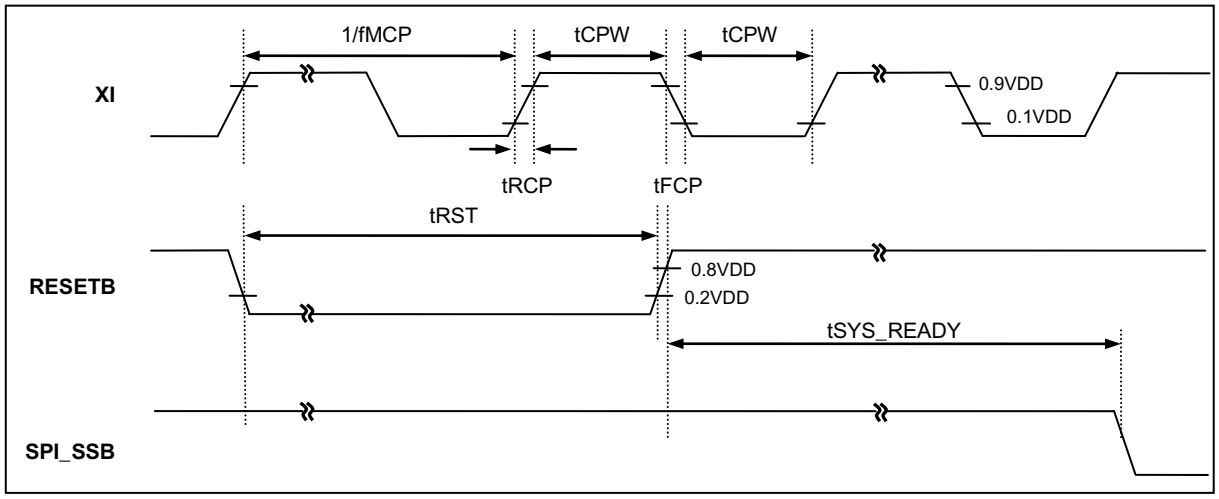


Figure 11-1 Fundamental timing diagram

Table 11-4 SPI AC Characteristics

Parameter	Symbol	PIN	MIN	TYP	MAX	UNIT
SSB Blank Time	tSSB	SPI_SSB(P0[4])	1			us
Clock Pulse Period	tSCK	SPI_SCK(P0[5])	2			us
Clock Pulse “H” or “L” Pulse Width	tSCKH, tSCKL	SPI_SCK(P0[5])	1			ns
Clock Pulse Transition Time	tFSCK, tRSCK	SPI_SCK(P0[5])			30	ns
Input Setup Time	tDIS	SPI_MOSI(P0[6])	0.5			us
Input Hold Time	tDIH	SPI_MOSI(P0[6])	0.5			us
Output Delay Time	tDS	SPI_MISO(P0[7])			100	ns
First Output Clock Delay Time	tFOD	SPI_MISO(P0[7])			100	ns

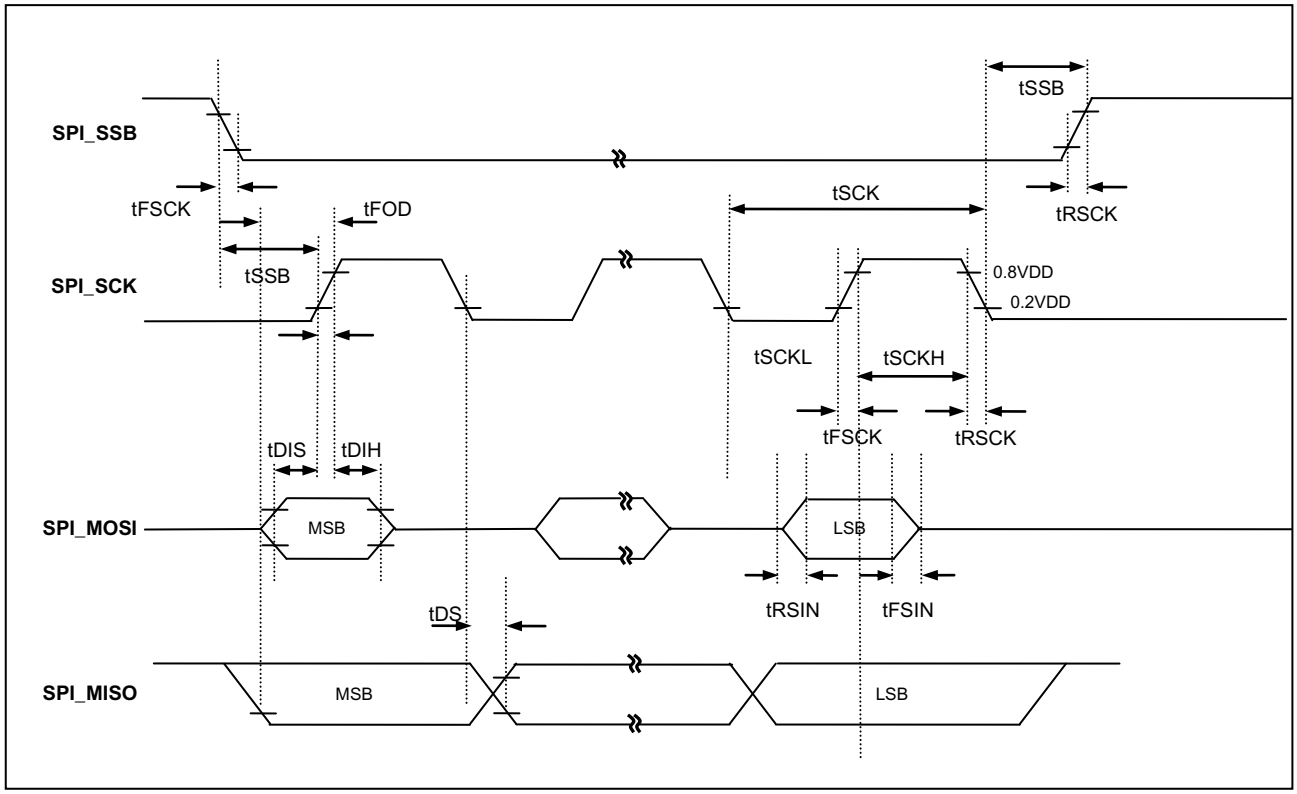


Figure 11-2 SPI timing diagram

REVISION HISTORY (I)

Version 1.9	Apr.2, 2013	28 QFN Package Dimension is updated.
Version 1.8	Feb.27, 2013	Correct the reset status of the 'SPI MISO' from 'Input' to 'Output' in 'Package And Pin Assignment'.
Version 1.7	Jan.23, 2013	28 QFN Package Dimension is changed.
Version 1.61	Oct.28, 2011	Table 7-1 Initialization procedure is corrected.
Version 1.6	May.17, 2010	VDD Operating range is changed and some descriptions are added. I_{PWM} is added to "DC Characteristic".
Version 1.5	May.03, 2010	Initial value of PLLD is changed from 0xA3 to 0xE1. Operating Current is changed from 20mA to 30mA. "dB representation of the volume" is added.
Version 1.47	Mar.23, 2010	"7.7 Play Speed" is updated. The title "7.8 Bit Rate" is changed into "7.8 Profile".
Version 1.46	Mar.18, 2010	"6.1 SPI Timing " is updated. "7.4 Cancel All Playing" is updated. "7.12 Deep Power Down Mode" is updated.
Version 1.45	Mar.08, 2010	"7.3 Initialization" is updated. "Table 7-3 Sound synthesize profile" is updated.
Version 1.44	Fab.18, 2010	"11.3 AC characteristics" is updated. "7.11 STOP Mode" is updated. "7.12 Deep Power Down Mode" is added.
Version 1.43	Fab.04, 2010	"11.3 AC characteristics" is updated. "7.10 By Pass Mode" is updated.
Version 1.41	Fab.03, 2010	"11.3 AC characteristics" is updated.
Version 1.4	Jan.26, 2010	"7.5 Check Play Status" is added. "7.1 Initialization" is optimized. "6.2 Control commands" is updated. "6.1 SPI timing" is update.
Version 1.33	Jan. 6, 2010	"6.2 Control commands" is updated.
Version 1.32	Jan. 6, 2010	"7.4 Cancel All Playing" is updated.
Version 1.31	Dec.23, 2009	"7.3 Stop Playing" is updated. "7.4 Cancel All Playing" is added.
Version 1.3	Dec.21, 2009	"6.1 SPI timing" is updated. "6.2 Control commands" is updated. "Table 6.2 Control ports" is updated. "Table 10 2 Index table architecture" is updated. "Table 11-3 Fundamental AC Characteristics" is updated. Translated in English.

Version 1.22		“Table 6-1 Command codes” is updated.
Version 1.21	Dec. 7, 2009	“7.1 Initialization” is updated. “Figure 7-11-3 SPI command timing” is updated.
Version 1.2	Dec. 7, 2009	“6.2 Control Commands” is updated.
Version 1.1	Dec. 2, 2009	“6.3 Control Ports” is updated. “7.4 Volume Control” is updated. Switching the order of chapter 7.5 and 7.6. “Table 7-5 Profile example” is updated. “7.9 STOP mode” is added.
Version 1.06	Nove. 20, 2009	“3. Package And Pin Assignment” is updated. “7.1 Initialization” is updated. “7.8 By Pass Mode” is added.
Version 1.05	Nove. 02, 2009	Control Port table is corrected.
Version 1.04	Octo. 29, 2009	Revised.
Version 1.032	Sept. 30, 2009	Revised/Sampling rate.
Version 1.02	July 17, 2009	Revised/Initialization process.
Version 1.01	July 17, 2009	Revised.
Version 1.0	June 30, 2009	Created.