

VS1003B 16/32-BUTTON PLAYER

VSMPPG “VLSI Solution Audio Decoder”

Project Code:

Project Name: VSMPPG

Revision History			
Rev.	Date	Author	Description
1.05	2009-03-02	PO	SD-card mini player sources added. Shuffle added.
1.04	2008-03-20	PO	Changed the polynomial fit for 32-button detection. Source code packet added, but no instructions yet.
1.03	2008-01-25	PO	Improved responsiveness of the buttons.
1.02	2008-01-10	PO	Increased MMC/SD SEND OP COND timeout.
1.01	2007-11-02	PO	32-button version and saving of song added.
1.00	2007-10-26	PO	First version with 16 buttons.

1 VS1003B 16/32-Button Player

All information in this document is provided as-is without warranty. Features are subject to change without notice.

The SPI bootloader that is available in VS10XX chips can be used to add new features to the system. Patch codes and new codecs can be automatically loaded from SPI EEPROM at startup. One interesting application is a single-chip standalone player.

The standalone player application uses MMC/SD directly connected to VS1003 using the same GPIO pins that are used to download the player software from the boot EEPROM.

The instruction RAM of 1280 words (5 kilobytes) is used for MMC communication routines, read-only handling of the FAT and FAT32 filesystems and a 16- or 32-button user interface.

- **No microcontroller is required**, boots from SPI EEPROM (25LC640).
- Low-power operation
- Uses MMC/SD/SD-HC for storage. Hot-removal and insertion of card is supported.
- Supports FAT and FAT32 filesystems, **including subdirectories** (upto 16 levels). FAT12 is partially supported: subdirectories or fragmented files are not allowed.
- Automatically starts playing from the first file after power-on.
- Power-on defaults are configurable.
- Transfer speed 4.8 Mbit/s (3.5×12.288 MHz clock).
- High transfer speed supports even 48 kHz 16-bit stereo WAV files.
- Watchdog prevents lockup situations in MMC communication.
- 16/32-Button interface allows pause/play, shuffle play and loudness toggle, song selection, and volume control.
- Optional LED for user interface feedback
- Saves the last played song to EEPROM. The playback will start from this song after next power-on.

Source code is now available to enable and disable different features, including SAVE_POSITION, LOOP_FILES (play each file until a new file selected), PAUSE_BEFORE_PLAY (goes to pause mode before each file, press play to start playback), and customizing the actions for user interface buttons.

2 SPI EEPROM and MMC/SD

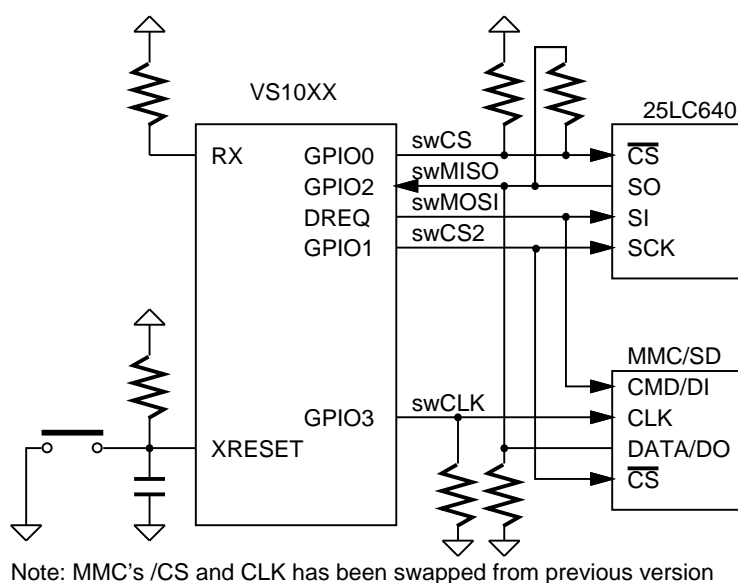


Figure 2.1: SPI-Boot and MMC connection

The standalone player software is loaded from SPI EEPROM at power-up or reset when GPIO0 is tied high with a pull-up resistor. The memory has to be an SPI Bus Serial EEPROM with 16-bit addresses. The player code currently requires almost 5 kB, thus at least 8 kB SPI EEPROM is recommended.

SPI boot and MMC/SD usage redefines the following pins:

Pin	SPI Boot	Other
GPIO0	swCS (EEPROM XCS)	100 kΩ pull-up resistor
GPIO1	swCS2 (MMC XCS)	Also used as SPI clock during boot
DREQ	swMOSI	
GPIO2	swMISO	100 kΩ between xSPI & swMISO, 680 kΩ to GND
GPIO3	swCLK (MMC CLK)	Data clock for MMC, 10 MΩ to GND

Pull-down resistors on GPIO2 and GPIO3 keep the MMC CLK and DATA in valid states on powerup.

Defective or partially defective MMC cards can drive the CMD (DI) pin until they get the first clock. This interferes with the SPI boot if MMC's drive capability is higher than VS10xx's. So, **if you have powerup problems when MMC is inserted, you**

need something like a **330 Ω resistor** between **swMOSI (DREQ)** and MMC's **CMD/DI pin**. Normally this resistor is not required.

To prevent the MMC/SD from interfering with the SPI EEPROM boot, MMC's chip select and clock inputs are swapped compared to the SPI EEPROM. This way MMC does not get clocked during the SPI boot and the system should work with all MMC's. Because the swap only occurred on the MMC pins, the SPI EEPROM connection is unchanged!

3 16-Button User Interface

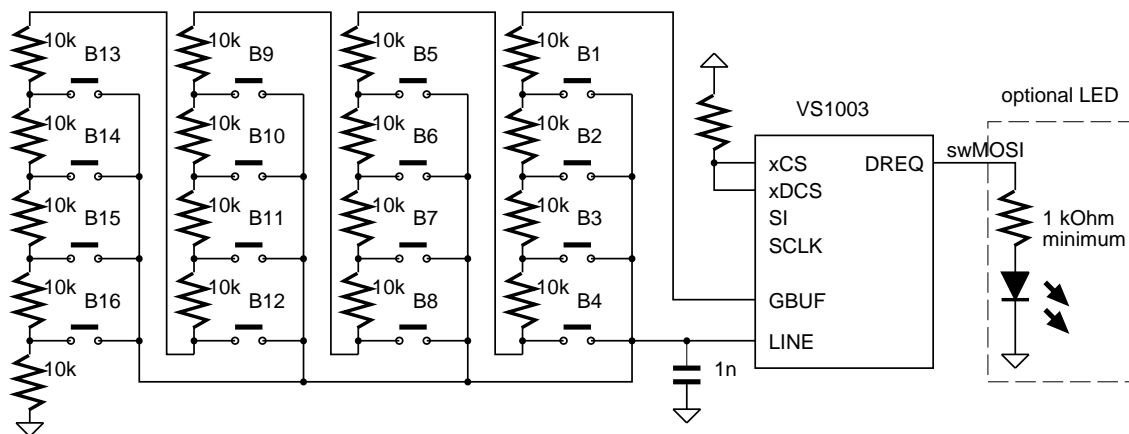


Figure 3.1: 16-button interface connection

A 16-button interface is implemented with 16 buttons and a 17 resistors. A 1 nF capacitor is used for noise filtering and to bias the LINE input. There are no other connections than shown here in the LINE input pin. Supply voltage for the resistor ladder is taken from GBUF to remove chip-to-chip gain variation in AD conversion. The GBUF pin has additional connections, as it is also used as the common voltage for the headphones.

Only one button is detected simultaneously. If two adjacent buttons are pressed, normally one of them is detected. The user interface provides the most needed controls.

Button	Short Keypress	Long Keypress
B1	Volume down	Volume down
B2	Volume up	Volume up
B3	Pause/Play	Play mode: Toggle loudness Pause mode: Toggle shuffle play
B4	Previous song	
B5	Next song	
B6	Song #1	
B7	Song #2	
..	..	
B16	Song #11	

The number of buttons can also be 32 (continue the button chain by adding 16 buttons and 10 kΩ resistors for each). Remember to use the right EEPROM image (player1003-32but.bin or player1003nw-32but.bin).

A LED indicates system activity. In play mode a long blink of the LED indicates loudness ON, in pause mode a long blink indicates shuffle play ON. Otherwise the LED shows MMC activity. In pause mode the LED lights up dimly.

3.1 Boot Images

The SPI EEPROM boot images can be found from the `code/` subdirectory. **Note that this application is highly chip-specific. It only works on the exact firmware versions mentioned.**

You can also select a version that does not play WMA files. If you use that version in your product, a WMA license should not be required.

There are also versions that support 32 buttons, and a version for the SD-card mini player. See SD-card mini player pages for schematics.

Chip	File	Features
VS1003B	player1003but.bin	16-button interface, watchdog
VS1003B	player1003nwbut.bin	16-button interface, watchdog, No WMA
VS1003B	player1003-32but.bin	32-button interface, watchdog
VS1003B	player1003nw-32but.bin	32-button interface, watchdog, No WMA
VS1003B	player1003-3but.bin	SD-card mini player

3.2 Power-on Defaults

Default values are loaded from SPI EEPROM at power-on reset. Before the MMC/SD card is first accessed after power-on, approximately 22 ms delay is executed. The startup delay time can be changed from the boot image. The middle bytes in the string 0x00 0x12 0x34 0x0e contain the default value 0x1234 (22 ms). This value can be changed between 0x0000 (0 ms) and 0x3fff (80 ms). Do not change the 0x00 and 0x0e bytes.

The input clock is assumed to be 12.288 MHz. If you want to use a different crystal, the SCLCLOCKF value can be found from byte offsets 10 and 11 in the boot image. The default value is 0xa000 (3.5×12.288 MHz) for VS1003B. You can reduce the power consumption a bit by using 0x8800 (3.0×12.288 MHz).

Volume (SCLVOL) default value is in byte offsets 26 and 27. Loudness default is in byte offsets 32 and 33 (treble and bass controls, respectively). The bass control value should be odd to make the loudness indicator LED blink work. SCLBASS default value is in byte offsets 8 and 9.

If you want the loudness ON by default, replace bytes 8 and 9 in the image with the same values you use as the loudness default in offsets 32 and 33.

Offset	Register	Default	Meaning
8, 9	SCLBASS	0x0000	Bass enhancer control at power-up
10, 11	SCLCLOCKF	0x9800	Clock control (for VS1003B/33C 0xa000)
26, 27	SCLVOL	0x2020	Power-up volume, left and right channel
28, 29	SCLAICTRL0	0	Song number to play at power-up
32, 33	SCLAICTRL2	0x33d9	Treble and bass control for loudness
34, 35	SCLAICTRL3	0	Play mode & Miscellaneous configuration

4 Recorder and SCI Control

Because the buttons are connected to the LINE input, and VS1003b only has one AD channel, standalone recorder is not possible.

If you want SCI control, just use the SCI-controlled player from the general standalone player application.

5 Example Implementation

The 16-button player was implemented by using the VS10xx prototyping board. The jumpers connecting the 3-button interface were removed, and an expansion card containing the 16 buttons was attached.

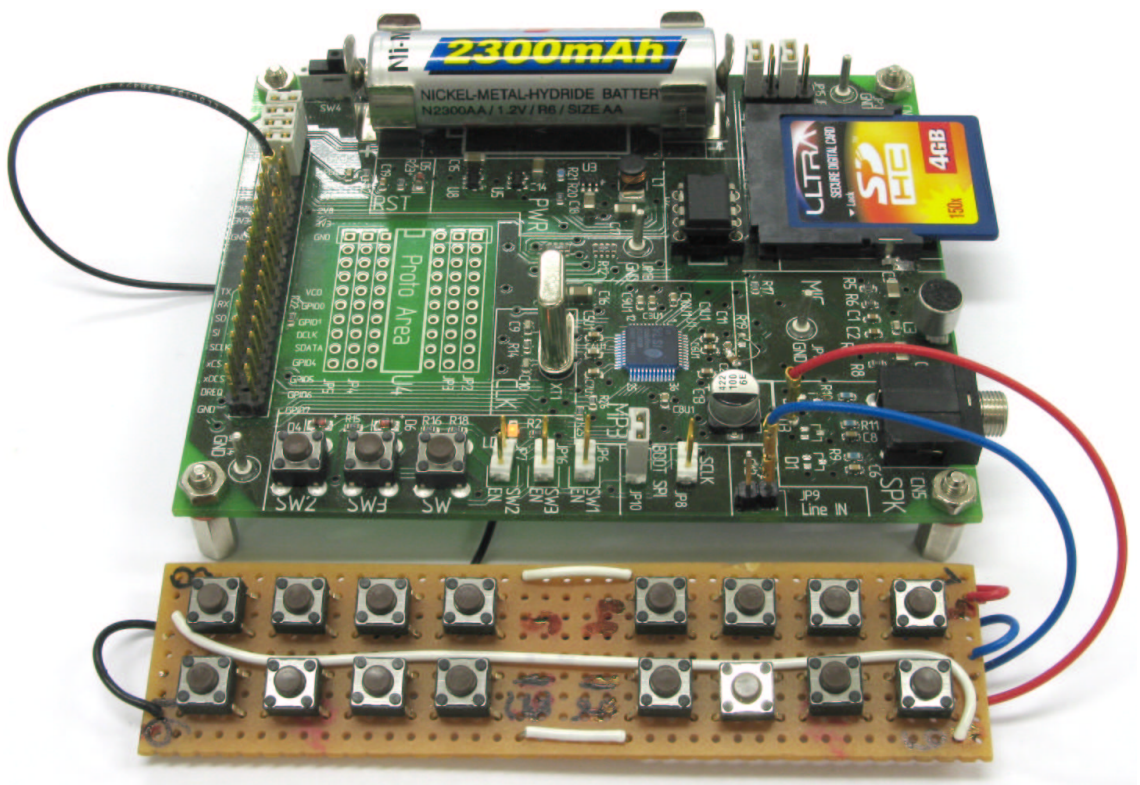
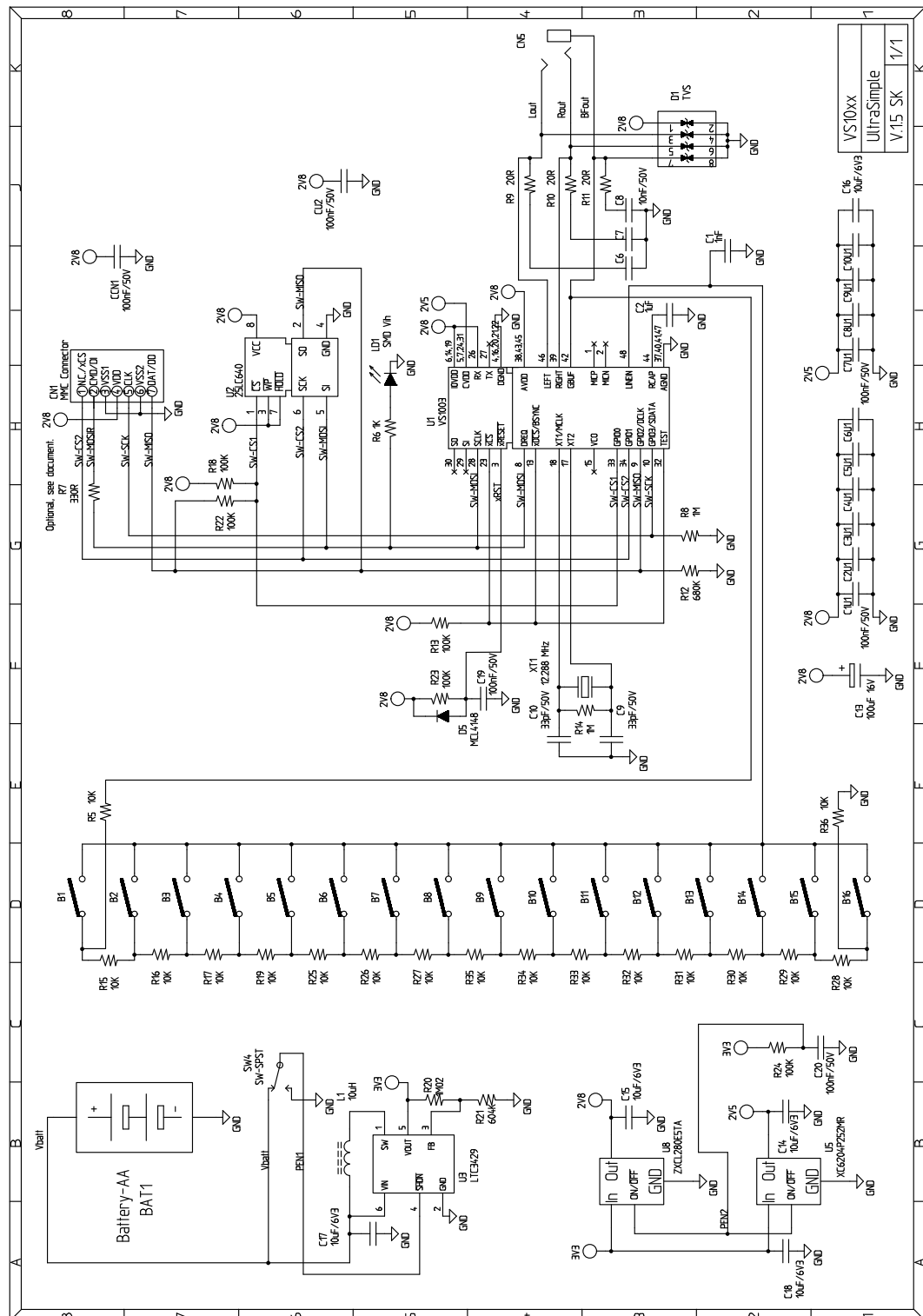


Figure 5.1: 16-Button Player in Prototyping Board

The following example schematics contains a simple implementation for VS1003B. Power generation and player logic are separated. **Note: the schematics is a stripped-down version of the Prototyping Board. The three-button interface is disabled and the 16-button interface is added. Use the attached schematics as a basis for your own designs and refer to the Prototyping Board schematics when you work with the Prototyping Board.**



Note: Optional resistor R7 fixes problems with some MMC's (chapter2).

6 Document Version Changes

6.1 Version 1.05, 2009-03-02

- Software for SD-card mini player included in the sources. **Note that the SD-card mini player has different schematics, which is not reproduced here.**
- Random play changed to shuffle play. In shuffle mode all files are played once in random order. When all files have been played a new order is selected.

6.2 Version 1.04, 2008-03-20

- Changed the polynomial fit for 32-button detection.
- Source code packet added to code/sources.zip. Does not yet contain any compile instructions.

6.3 Version 1.03, 2008-01-25

- Improved responsiveness to button presses.

6.4 Version 1.02, 2008-01-10

- Increased the MMC_SEND_OP_COND / SD_SEND_OP_COND timeout value, because one microSD needed it.

6.5 Version 1.01, 2007-11-02

- Added 32-button version.
- Added saving of last played song.

6.6 Version 1.00, 2007-10-26

- First version with 16 buttons.

7 Playing Order

The playing order of files is not the same order as how they appear in Windows' file browser. The file browser sorts the entries by name and puts directories before files. It can also sort the entries by type, size or date. The standalone player does not have the resources to do that. Instead, the player handles the files and directories in the order they appear in the card's filesystem structures.

If the filename suffix does not match any of the valid ones for the specific chip, the file is ignored.

Normally the order of files and directories in a FAT filesystem is the order they were created. If files are deleted and new files added, this is no longer true. Also, if you copy multiple files at once, the order of those files can be anything. So, if you want a specific play order: 1) only copy files into an empty card, 2) copy files one at a time in the order you like them played.

There are also programs like LFNSORT that can reorder FAT16/FAT32 entries by different criteria. See "<http://www8.pair.com/dmurdoch/programs/lfnsort.htm>" .

The following picture shows the order in which the player processes files. First DIR1 and then DIR2 has been created into an empty card, then **third.jpg** is copied, DIR3 is created and the rest of the files have been copied. **song.mid** was copied before **start.wav**, and **example.mp3** was copied before **song.mp3** because they appear in their directories first.

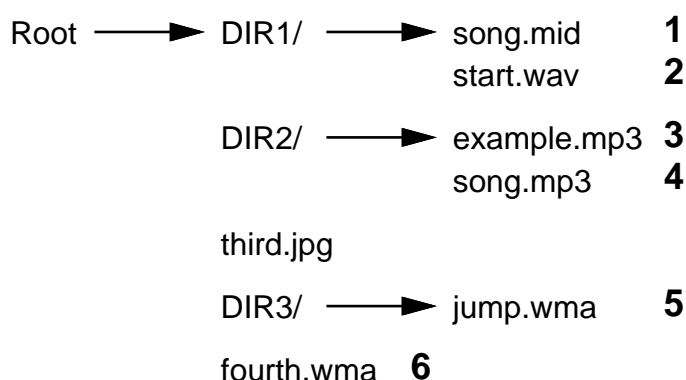


Figure 7.1: Play Order with subdirectories

Because DIR1 appears first, all files in it are processed first, in the order they are located inside DIR1, then files in DIR2. Because **third.jpg** appears in the root directory before DIR3, it is next but ignored because the suffix does not match a supported file type, then files in DIR3, and finally the last root directory file **fourth.wma**.

If DIR2 is now moved inside DIR3, the playing order changes as follows.

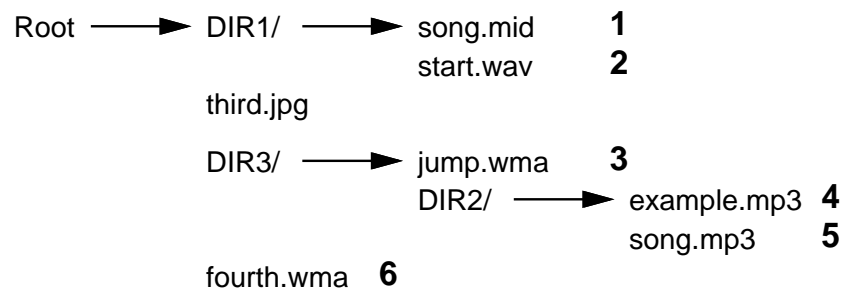


Figure 7.2: Play Order with nested subdirectories