VS1063A UART STREAMING

"VLSI Solution Audio Decoder/Encoder"

Project Code: VS1063
Project Name: Support

Revision History

Rev.	Date	Author	Description
1.2	2019-07-16	POj	VS1103 receiver with IMA ADPCM.
1.1	2019-07-08	POj	More information.
1.01	2015-11-11	POj	Added demo explanation.
1.00	2015-09-08	POj	Initial version.

Contents

List of Figures

1 Description

vs1063a provides encoding with output through UART, but it does not provide decoding through UART by default. This pair of applications implements a door phone type of system where one vs1063a encodes a MP3 stream, transferred through UART, and another vs1063a or vs1053b (or several) receive the UART data and decode it into audio.

Alternatively you can change the encoding to mono IMA ADPCM and receive the resulting stream with vs1103b.

- UartSender - Starts the vs1063a encoder in UART output mode with the desired parameters. Reads GPIO6 on powerup and reset to choose between mono microphone and stereo line inputs.
- vs1063a-uartin - Implements audio decoding through UART. For mp3 supports samplerate tuning according to data received.
- vs1053b-uartin - Implements audio decoding through UART. For mp3 supports samplerate tuning according to data received.
- vs1103b-uartin - Implements mono IMA ADPCM decoding through UART. Detects the nominal encoding rate from the data received, supports 44100 Hz, 32000 Hz, 24000 Hz, 16000 Hz, 12000 Hz, and 8000 Hz with samplerate finetuning.

Uses 460800bps UART baudrate - thus supports mp3 bitrates upto 320kbit/sec.

The VSIDE solution creates the SPI boot images: `UartSender.spi`, `vs1063a-uartin.spi`, `vs1053b-uartin.spi`, `vs1103b-uartin.spi`.

If you change the baudrate from the source code, remember to change it from both the sender and the applicable receiver. When you calculate new uart divider values, note that vs1103b uses $4.0\times$ clock while the other chips use $4.5\times$ clock.

NOTE: the RX pin should be forced high until the code has been loaded from SPI memory, otherwise when encoded data is received during SPI boot, a reception of byte 0xef from UART would cause the chip to jump into monitor and stops the SPI boot.

When `vs1063a-uartin`, `vs1053b-uartin`, and `vs1103b-uartin` start execution, they switch GPIO3 to output mode and drives it high, so GPIO3 can be used to gate the operation of the RX pin.

1.1 UartSender.spi

`UartSender` starts the vs1063a encoder in UART output mode with 460800bps UART speed.

If GPIO6 is high at startup (in the VS10xx Prototyping Board GPIO6 connected to DREQ), `UartSender` starts in stereo line-in mode with a fixed 1.0 gain.

If GPIO6 is low at startup, `UartSender` starts in mono microphone mode with microphone amplifier active, with AGC and max 4.0x gain.

1.2 vs1063a-uartin.spi, vs1053b-uartin.spi

`vs1063a-uartin` and `vs1053b-uartin` receive stream data from UART at 460800bps, and route the received data to the stream buffer (input FIFO) to be decoded.

Adjusts rate by filtered audio buffer and stream buffer fill state at each mp3 frame boundary. This is why mp3 is assumed and why rate adjustment does not work for other audio formats. A different algorithm is needed for non-mp3 formats. Also currently assumes the samplerate is lower than 48 kHz so that samplerate finetuning can be performed. Samplerate tuning with 48 kHz samplerate needs either a higher crystal than 12.288MHz or the use of a resampler.

Currently the SPI boot image disables the RX interrupt when starting to load the program, which makes it a bit more robust than before even without forcing RX high (or low).

vs1063a and vs1053b use $4.5\times$ clock. UART bitrate is set by the UART_SPEED define in `vs1063a-uartin.s` and `vs1053b-uartin.s`.

At the moment all vs1063a encoding formats are supported by `vs1063a-uartin` (and most by `vs1053b-uartin` as long as the header is received, i.e. the sender is started after the receiver.

Define `STREAM_ADPCM` to synchronize to a mono IMA ADPCM stream (with 44100 Hz rate).

1.3 vs1103b-uartin.spi

`vs1103b-uartin` receives stream data from UART at 460800bps, routes the received data to the stream buffer (input FIFO) to be decoded.

When `vs1103b-uartin` starts execution, it switches GPIO3 to output mode and drives it high, so GPIO3 can be used to gate the operation of the RX pin.

The format has to be mono IMA ADPCM with nominal rate of 44100 Hz, 32000 Hz, 24000 Hz, 16000 Hz, 12000 Hz, or 8000 Hz. `vs1103b-uartin` detects the rate automatically from the delay between two IMA ADPCM frames. Samplerate finetuning is performed to adapt to the clock difference between the encoder and playback and keep the decoding latency constant.

`vs1103b-uartin` uses $4.0\times$ clock, which you should remember if you change the UART bitrate.

2 Can I Use Something Else Than MP3?

The MP3 format has some features that make it perfectly suitable for streaming: each audio frame has a header fully defining the decoding parameters and the audio data is (mostly) self-contained.

1. The receiver can be turned on at any time and it can synchronize to the stream instantly.
2. The receiver can connect to a different stream that has different encoding parameters without issue.
3. The transmitter can decide the encoding parameters depending on the application, even in a case-by-case basis.
4. The transmitter can be turned on before the receiver.
5. The decoding is robust against lost or inserted bytes.

If you want to stream a format like WAV, which contains the parameters only in the file header, you need to either:

1. Start the receiver first, then the transmitter. The WAV header is then received and decoding can start without issues.
   This method works by just changing the format on the transmitter. However, some formats like PCM WAV cannot detect missing or inserted bytes.
2. Use a fixed format agreed by the transmitter and receiver.
   The `uartin` program needs to self-generate the required header to start decoding before starting to receive the data. The transmitter do not need to create and send the header, only the data. However, some formats like PCM WAV (other than 8-bit mono) cannot start receiving from an arbitrary point in the data.

The `UartSender` program can be configured to send IMA ADPCM, and the `uartin` programs can be configured to receive IMA ADPCM (vs1103b-uartin receives only mono IMA ADPCM).

3 Additional Information

If you find a bug, a curious feature, or are unsure how to use VS1063, let us know.

VSDSP Forum is at **http://www.vsdsp-forum.com/** .

Support e-mail address is **support@vlsi.fi** .

4 Uartlink Demo

The uartlink demo consists of two VS1063 Protyping Boards.

One board is the encoder (jumper block in GPIO6), the other board is the receiver.

Insert batteries and connect the TX pin of the encoder to the RX pin of the receiver. Also connect GND pins together. If you are not sure which is the encoder and which is the receiver, connect also RX to TX.

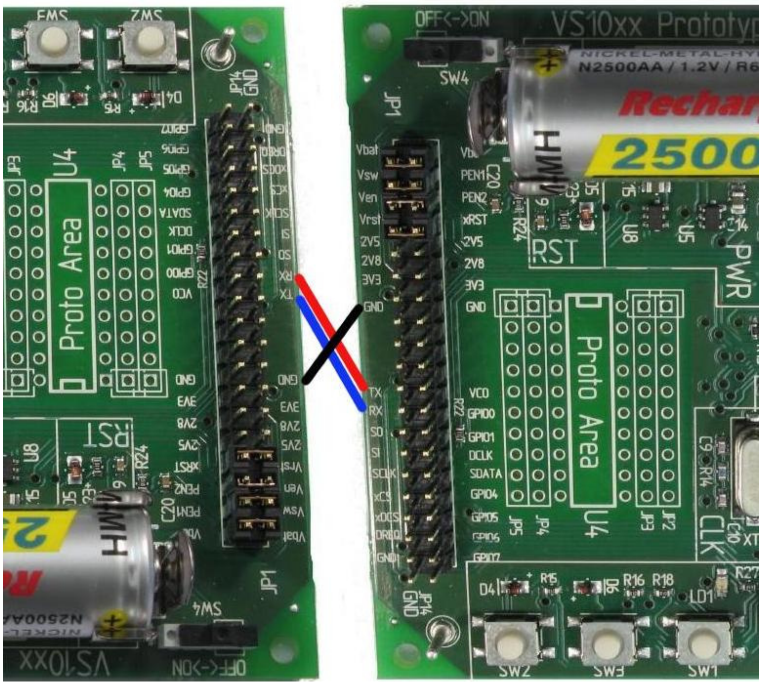

Figure 1: Connect the boards

Turn on the receiver first, then the encoder. After a short while the sound from the encoder will be played back in the decoder.

If you don't hear anything, check the uart connection between the boards (GND, TX→RX, RX→TX). You can also try powering down the boards and turning them on again.

The encoding parameters for the demo are:

- Samplerate 44100Hz
- GPIO6 low (no jumper): Mono microphone, AGC, 128kbit/sec MP3
- GPIO6 high (jumper from GPIO0 to DREQ): Stereo line input, fixed gain, 256kbit/sec MP3
- UART bitrate 460800bps