Application Note: AN01011

DSP performance on XS1-L devices

This application note describes some of the DSP features the XMOS L-family of devices provide and how they could be used in an example application. The second part of this document describes in more detail how the L1 USB Audio reference design can be modified to include Biqaud Filters to provide equalizer functionality.

1 Introduction

The XMOS XS1-L architecture has a number of features that make DSP operations predictable and straightforward. The architecture supports multi-core configurations with the following performance available from a single xCORE Tile:

- 500M instructions per second
- 8 logical cores
- 64kB of unified memory with single cycle access
- Single cycle channel communication between cores
- Single cycle 32 * 32 into full 64 bit precision MACC instruction
- Single cycle instruction latency
- Single cycle CRC32 instruction

For more details on the device architecture see the xCORE Architecture Introduction[1].

1.1 Typical application scenario

A typical application run on an xCORE device will use several cores and communicate data between the cores via channels. The core diagram in Figure 1 can be used to represent this - e.g. for the L1 USB audio design:

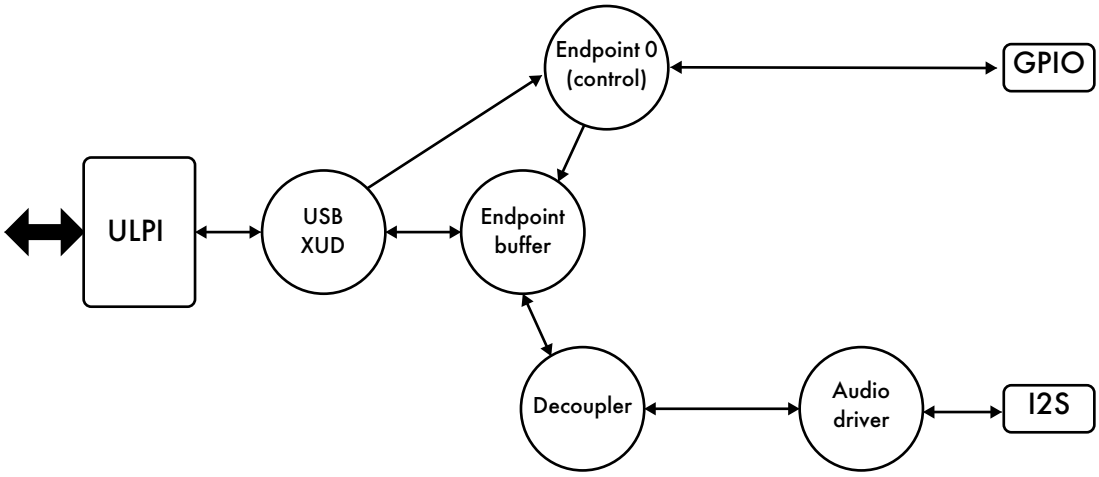

Figure 1: L1 USB Audio Software Core Diagram

This shows five cores along with the channel communication between them. A core will typically accept a data stream, apply some processing, then pass it to either the next core or to some I/O pins via the ports. It should be easy to anticipate how a DSP core could be added to this arrangement by splicing a new core in between the decouple and audio cores - see Figure 2.

The addition of an extra core will slightly reduce the available MIPS to the other cores, so it is important to make sure the application still meets any timing requirements. XMOS offer a solution to this which will be discussed in the next section.

[1] http://www.xmos.com/published/xcore-architecture

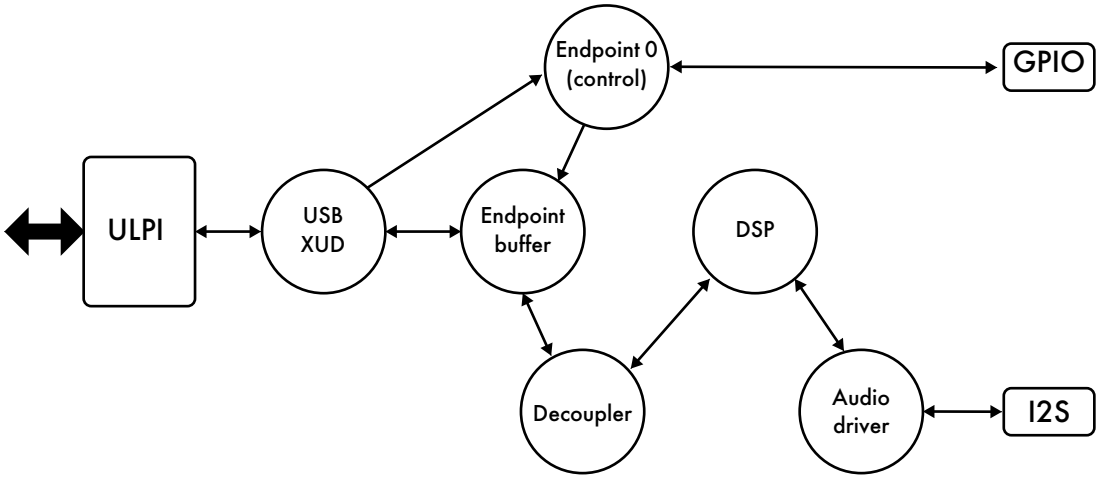

Figure 2: L1 USB Audio Software Core Diagram with DSP core

1.2 Performance and buffering

The XMOS architecture is completely deterministic and the toolchain provides a static timing tool (XTA) which will allow you to guarantee meeting timing, however it's useful to work out what sort of performance might be available and how you can best use it.

Take an audio stream as an example. This might be a stream at 48kHz containing 2 channels of audio in 32 bit precision. This tells us we need to process 96000 words per second, or we have 10.4us between sample times. If a core is running at 62.5MIPS (for example a 500 MHz part with 8 cores), this gives us 650 core cycles between samples[2].

If the required DSP needs more cycles, then the scalability of the xCORE architecture means you can simply add another core doing some of the processing in each. This can even be done by scaling into a larger device - e.g. from an XS1-L8 to an XS1-L16.

Cores often send data in bursts with some local buffering. It may therefore be necessary to store data for a few samples and process the data in between these bursts. A local buffer also allows data from multiple samples to be used if the DSP function requires that.

There is another potential benefit from local buffering of data. Some parts of the DSP function may only operate on every 10th sample (e.g. changing the coefficients slowly to avoid artefacts in the sound track). By buffering 10 samples of data then batch processing them, the additional calculation time can be averaged across all 10 samples, rather than having to allow for the worst case and potentially have some idle time during the faster operations.

Batch processing samples would also rely on cores either side of the DSP being able to buffer a similar amount of data to avoid under/overflow situations and the associated logic to cope with these.

The lack of cache and single cycle memory access in the xCORE architecture mean that this can be done with very low performance overheads and without risking timing corner cases.

[2] A core cycle represents the time it takes one core to execute one instruction - excluding divide operations.

2 Typical performance figures:

The following performance figures are approximate benchmarks for the performance achievable on an XS1-L8 device running eight cores at 400MHz - i.e. 50MIPS per core.

- 16 million MACs / second[3]
- 20 biquad EQ filters, 48kHz[4]
- 100 tap FIR filter, 150kHz[5]
- 256 point fast FFT, 48kHz (8 bits)[6]
- AES decoder at 9 Mbits/sec[7]
- JPEG encoder 1.7 million samples / second (using 3 cores)[8]

These figures are for a single core. Performance at 500MHz (i.e. 62.5MIPS/core) will scale linearly.

As shown in the performance metrics, many of these algorithms are available in open source libraries via the XCore community site on GitHub (https://github.com/xcore).

[3] Architectural performance limit
[4] Based on https://github.com/xcore/sc_dsp_filters
[5] Based on https://github.com/xcore/sc_dsp_filters
[6] Based on https://github.com/xcore/sc_dsp_transforms
[7] Based on https://github.com/xcore/sc_crypto
[8] Based on https://github.com/xcore/sc_dsp_transforms