# Digital Systems Design 2014/2015

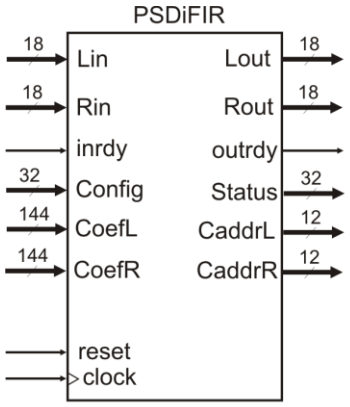
3rd laboratory assignment - PRELIMINARY
V0.2 - 1 Dec 2014

## 1 - Objective

Design and implement a digital signal processing IP core for to compute in real-time long FIR filters with up to 16384 coefficients.

## 2 - Introduction

In this work you will implement a custom DSP system to compute in real-time long FIR filters with up to 16384 coefficients. The input signal is a stream of high-quality digital audio, provided as two 18-bit samples sampled at 48 kHz. This signal must be processed by the digital filter and sent back to the inputs of the audio codec, with the same format and sampling frequency. The DSP block must implement two independent FIR filters for each channel or a single filter applied to the mono audio signal calculated as the average of the left and right channels. The filter coefficients are read from a RAM memory that provide eight 36-bit coefficients in each read cycle (4 for each channel). Figure 1 shows the top level interface of the IP core and the corresponding Verilog code. Table 1 lists the specification of the input/output ports.




```
module PSDiFIR (
       clock,
       reset,
       Lin,
       Rin,
       inrdy,
       Lout,
       Rout,
       outrdy,
       Config,
       Status,
       CaddrL,
       CaddrR,
       CoefL,
       CoefR    );

input          clock,
               reset;
input  [17:0] Lin, Rin;
input          inrdy;
output [17:0] Lout, Rout;
output         outrdy;
output [11:0] CaddrL, CaddrR;
input [143:0] CoefL, CoefR;
input  [31:0] Config;
output [31:0] Status;
```

**Figure 1** – Top level interface of PSDiFIR IP core.

**Table 1** – Input/output ports of PSDiFIR IP core.

| Name | Direction | Function |
|---|---|---|
| clock | Input | Master clock signal. The module should operate synchronously with the rising edge of this clock signal. Minimum clock frequency should be 100 MHz |
| reset | Input | Master reset, active high and synchronous with the active edge of the master clock. |
| Lin[17:0]<br>Rin[17:0] | Input | Input data buses for left and right channels; input data is signed, represented in two's complement and enabled by signal "inrdy" |
| inrdy | Input | Clock enable to synchronize data in the Lin/Rin inputs. This signal is set to 1 during one clock cycle to indicate |

| | | |
|---|---|---|
| | | that new data samples have been placed in the Lin/Rin inputs. This is synchronized by the codec interface with the global sampling frequency (48 kHz) |
| Lout[17:0]<br>Rout[17:0] | Output | Output buses for left and right channels (signed, two's complement). Data should be placed in these outputs at the global sampling frequency (48 kHz) and enabled by signal "outrdy" |
| outrdy | Output | Clock enable signal to synchronize data in the Lout/Rout outputs. This should last for exactly one clock signal to enable the capture of the output data by the audio codec controller |
| CaddrL[11:0]<br>CaddrR[11:0] | Output | Address buses to read the two external memories with the filter's coefficients (two independent addresses are provided for the coefficients of the left and right channels). |
| CoefL[143:0]<br>CoefR[143:0] | Input | Filter coefficients. A read operation from each memory provides four 36-bit coefficients (signed, two's complement). |
| Config[31:0] | Input | General purpose configuration input. This input will be driven by the external controller to set the different operation modes. |
| Status[31:0] | Output | General purpose status output. This output is read by the external controller |

The processor will receive two data samples from a audio codec for the left and right channels, represented in 18 bit, two's complement and with a sampling frequency equal to 48 kHz. For each pair of audio samples received, the processor should calculate one or two FIR filters, whose coefficients are read from external memories. To provide sufficient memory access bandwidth, each coefficient memory is organized as 4096 words of 144 bits, thus allowing to access two sets of four 36-bit coefficients in each clock cycle.

**3 – Numeric representation**

Each FIR filter can have up to 16384 coefficients. The coefficients are represented in two's complement, normalized to the range $[-1,+1-2^{-35}[$, with 35 fractional bits, which corresponds to the interval $[-2^{35} / 2^{35}, +2^{35}-1 / 2^{35}]$. These limits can be interpreted as the 36-bit two's complement range $[-2^{35} , +2^{35}-1 ]$ shifted 35 bits to the right.

The input and output audio samples are represented as 18-bit, two's complement integers. The dynamic range of the output signal will be dictated by the magnitude of the impulse response of the filter. The design should guarantee that overflow does not occur in any point of the internal computing process, although the filter coefficients should guarantee the final output matches the allowable 18-bit dynamic range. Rounding may be used at any point of the process.

**4 – FIR calculation**

The block to develop should compute, for each incoming sample, the following equation that represents the convolution between the input signal $x$ and the FIR impulse response $b$ of length $N$:

$$y_n = \sum_{k=0}^{N-1} b_k \cdot x_{n-k}$$

Where $x_{n-k}$ are the N previous input samples, $b_k$ are the filter coefficients and $y_n$ is the output sample to calculate. The maximum FIR length is 16384. Shorter filters can be implemented just by filling the non-existent coefficients with zeros.

## 5 - Reference design

Figure 2 shows a simplified block diagram of the reference design to integrate the PSDiFIR block.

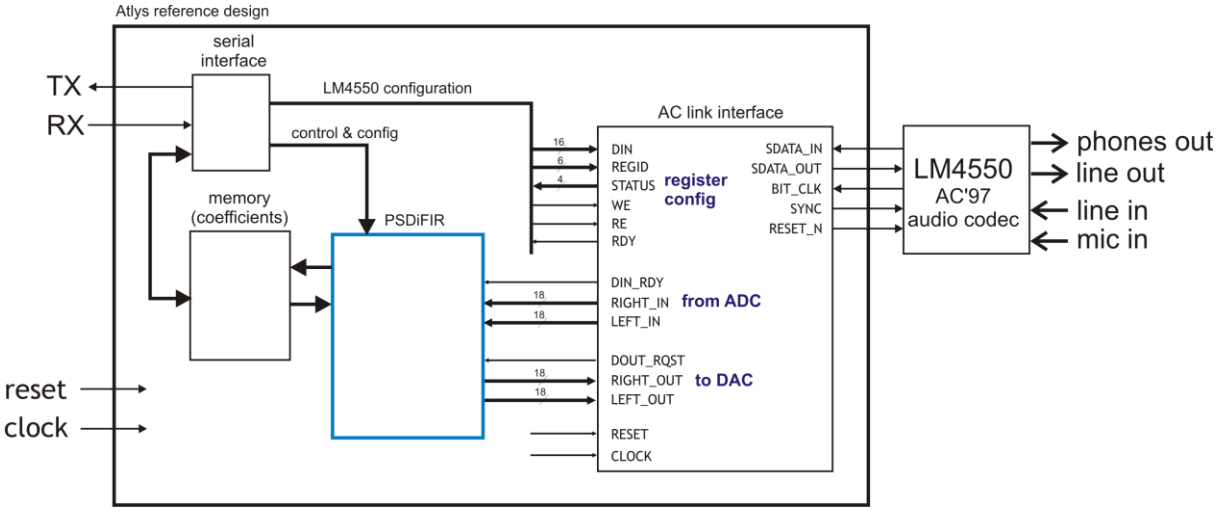



**Figure 2** – Reference design for integration of the PSDiFIR IP core.

### Serial interface

The reference design includes a serial interface to configure the LM4550 audio codec, load the filter coefficients to the RAM_coefs (RAM_coefs.v) memory and also to access a general purpose configuration and status port. This is implemented with a UART block (module uart.v, operating at 921600 baud) and a variant of the serial to parallel interface already used in the previous project (module ioports_psd14.v). In the reference design only the input/output ports listed in tables 2 and 3 are used.

**Table 2** – Output ports used by the reference design.

| | |
|---|---|
| P2out, P3out | used by the LM4550 control interface |
| P5out | Digital gain applied to the digital audio, when SW5 is ON |
| P6out | Defines the frequency of the sawtooth test signal. The value N loaded into this port sets the frequency to ( 100 000 / $2^{18}$ x N ) kHz |
| PFout | This output port has an automatic return to zero. Bits 0 and 1 are the write enable and read enable of the LM4550 interface, respectively. The other bits are not used in this design. |

**Table 3** – Input ports used by the reference design.

| | |
|---|---|
| P0in | Reads the status of the slide switches (bits 7 to 0) and push buttons (bits 16 to 20); the other bits are assigned to zero. |
| P1in | Reads the status of the LM4550 controller |
| P2in | Reads the RDY (ready) bit of the LM4550 controller |
| P7in | Outputs a dummy function with the output data at the application read port of the coefficient memory; this is necessary to synthesize the memory as a true dual-port memory and map it to RAM blocks instead of LUTs; this can be used to read a status word from the application circuit. |

**Coefficients memory**
The memory holding the FIR coefficients is implemented as a dual-port RAM in module **RAM_coefs.v**. One memory access port is managed by the serial interface to allow writing and reading the coefficients by an external computer and the second access port is a read-only port to be used by the PSDiFIR block.

The interface ports available are the two 12-bit address buses **addrL[11:0]** and **addrR[11:0]** and the two 144-bit data buses **coefL[143:0]** and **coefR[143:0]**. In the reference project these buses are connected to dummy signals to avoid the logic trimming by the logic optimization processes executed during RTL synthesis.

**AC link interface**
The interface with the LM4550 audio codec is performed by the module **LM4550_controller.v**, which is controlled by the input/output ports of the block **ioports** referred above. This should not be modified to maintain the compatibility with the software interface running in the PC.

**5.1 - Implementing and testing the reference design**
To implement the reference design, create a new project in ISE with the following settings:

| Property Name | Value |
|---|---|
| Top-Level Source Type | HDL |
| | |
| **Evaluation Development Board** | None Specified |
| Product Category | All |
| Family | Spartan6 |
| Device | XC6SLX45 |
| Package | CSG324 |
| Speed | -3 |
| | |
| Synthesis Tool | XST (VHDL/Verilog) |
| Simulator | Modelsim-SE Verilog |
| Preferred Language | Verilog |
| Property Specification in Project File | Store all values |
| Manual Compile Order | ☐ |
| VHDL Source Analysis Standard | VHDL-93 |
| | |
| Enable Message Filtering | ☐ |

Add to the project the following files (for now ignore the other Verilog files):

**./src/verilog-rtl/s6base_top.v**
**./src/verilog-rtl/uart.v**
**./src/verilog-rtl/ioports_psd14.v**
**./src/verilog-rtl/LM4550_controler.v**
**./src/verilog-rtl/RAM_coefs.v**
**./src/data/s6base.ucf**

Implement the design by selecting the toplevel file **s6base_top.v** and executing the process "*Generate programming file*". After terminating successfully, program the Atlys board with the bitstream **s6base_top.bit** using the Adept application (do not program the FPGA with the process "*Configure target device*" in ISE). Connect a set of headphones or loudspeakers in the HP OUT (black) or LINE OUT plug (green), a microphone in the MIC plug (pink) and an audio source in the LINE IN plug (blue).

Run the **LM4550_V2013.exe** application in the PC (check the assignment of the serial port mapped to the Digilent USB driver and if necessary modify the file **config.dat** to correct it). With the slide bars and check boxes you can control the LM4550 codec in a similar way a PC does with the audio interface. Figure 3 shows the control interface of the audio codec. The hexadecimal numbers between brackets represent the address of the audio codec register that is programmed when a control is actuated. To help understand the operation of the audio codec, figure 4 shows a block diagram of the audio codec, with the analog and digital paths relevant for this application highlighted. For example, changing the line-out volume (or master volume) is done by writing register 02h.

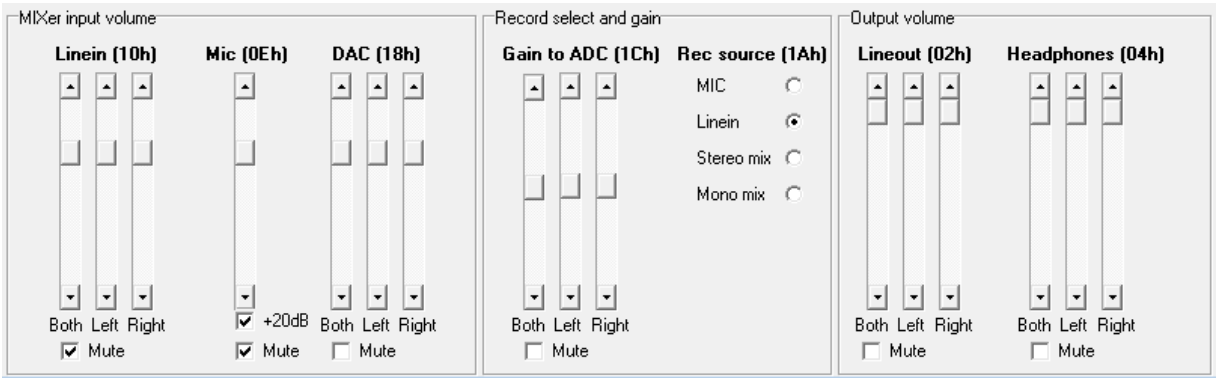



**Figure 3** – User interface to control the audio codec

First verify that all the slide switches are in the OFF (down) position. Then, un-mute the DAC input, the ADC input and the Line out or headphones out, depending on the audio output you are using. In the "Record select and gain" section, select "Line in" as the source for the ADC. You should listen now the audio signal injected in the LINE IN input. This is sampled by the ADC, sent to the digital system as a 48 kHz data stream (left and right channels) and fed back to the DAC after passing by a simple processing circuit (see below). By sliding the volume controls for the ADC, DAC and LINE OUT/HP OUT you can control the volume at various points in the internal LM4550 analogue path.

The 8 most significant bits of the absolute value of the average of the signal in the left and right channels are connected to the 8 LEDs above the slide switches (see lines 248-251 of s6base.v).

Setting the switches SW7 and SW6 to ON will send to the DAC the sawtooth test signal (see lines 243-244 of s6base.v). The frequency of this signal is controlled by the value written into port P6out. Setting the SW5 switch to the ON position will multiply the input digital signal by a 8-bit gain written into port P5out.

Setting the switch SW6 to ON will pass the input digital signals through a digital gain set as an integer loaded into the 8 LSbits of port P5out.

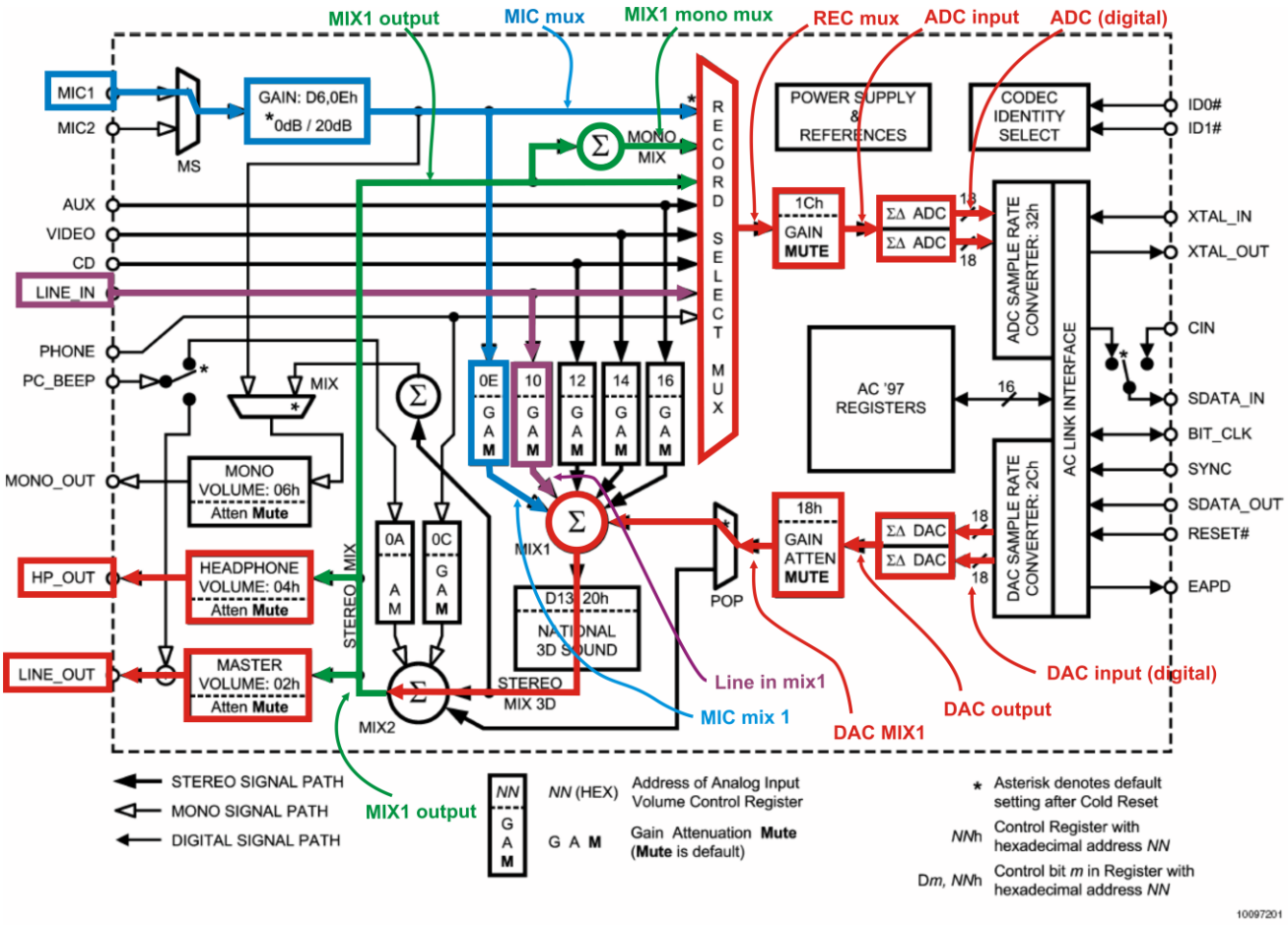



**Figure 4** – Block diagram of the LM4550 audio codec and the digital/analog paths used in the Atlys board. The 8-bit hexadecimal values indicated near each box are the address of the register that programs the corresponding function.

## 6 – Implementation considerations

The DSP block must be able to perform the calculation of the long FIR filter within the sampling period of the audio signal (20.833 μs), which requires the execution of 16384 MAC (multiply-accumulate) operations. With the 100 MHz minimum clock frequency only 2083 clock cycles are available between consecutive audio samples. Assuming that one MAC operation can be completed in a single clock cycle, this means that 8 MAC units have to operate in parallel to meet the real-time constraint. This will require reading from some memory 8 coefficients and 8 data samples in each clock cycle, representing a total memory bandwidth of 43.2 Gb/s.

The module **RAM_CB_16k.v** (included in **./src/verilog-rtl**) implements a dual-port circular buffer, with a read access port providing 4 data samples per clock cycle. This block includes an internal address calculator to implement the circular access behaviour. A complete data sheet will be available soon (the Verilog file includes a short description of the block).