

Altera Innovate Nordic Competition

Analog Modeling Synthesizer *FPGA SOPC implementation*

Arnaud Taffanel, Peyman Pouyan

Teacher: Professor Lambert Spaanenburg

Lund Institute of Technology



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan

Motivation

- To produce an analog style sound and to simulate the analog user experience by permitting to change any parameter of the sound generation path in real-time.



Outline

- Introduction
 - Music synthesizers
 - History of Analog Synthesizer
- System Implementation
 - NIOS II system
 - Sound stream
- Software Implementation
 - Project Mapping
 - μ c/os II
- Conclusion & results

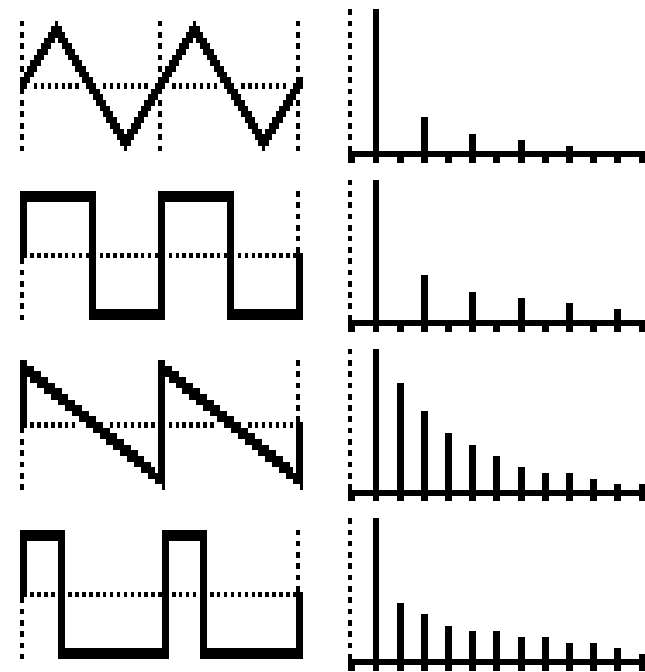


Music Synthesizers



Analog synthesizer=Subtractive Synthesizer

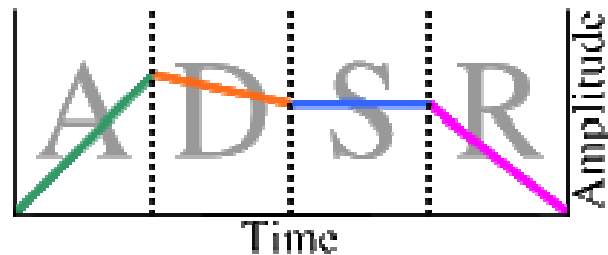
- Subtractive synthesis starts with a rich harmonic waveform (such as square or sawtooth wave) and filters out unwanted spectral components.



Subtractive Synthesis

- **Basic Subtractive Synthesizer parts:**

- Oscillators
- Filters
- Voltage Controlled Amplifiers or VCAs.
- Envelop Generator such as an **ADSR**.



- Lfos, Low Frequency Oscillators.



Modular Analog Synthesizer

- Highly configurable
 - Completely manual interconnection
 - One interconnection configuration is called a path
- Heavy
- Very expensive



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



Next generation Analog Synthesizer

- Partially pre-cabled
 - Electronic elements have a fixed place in the circuit.
 - Less heavy and expensive



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



Digital analog modeling Synthesizer

- Appear in the 90s
- Very Light
- Cheaper



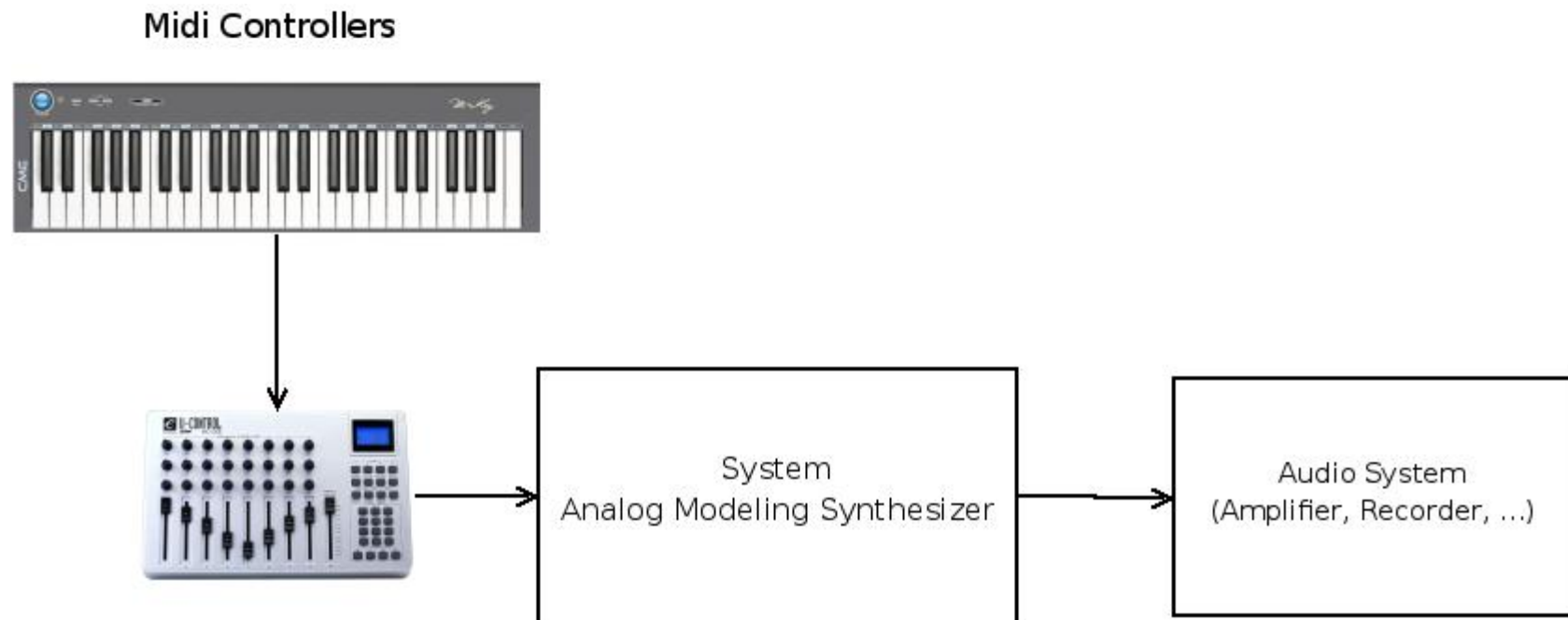
Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



Our Project

- **Main target:**

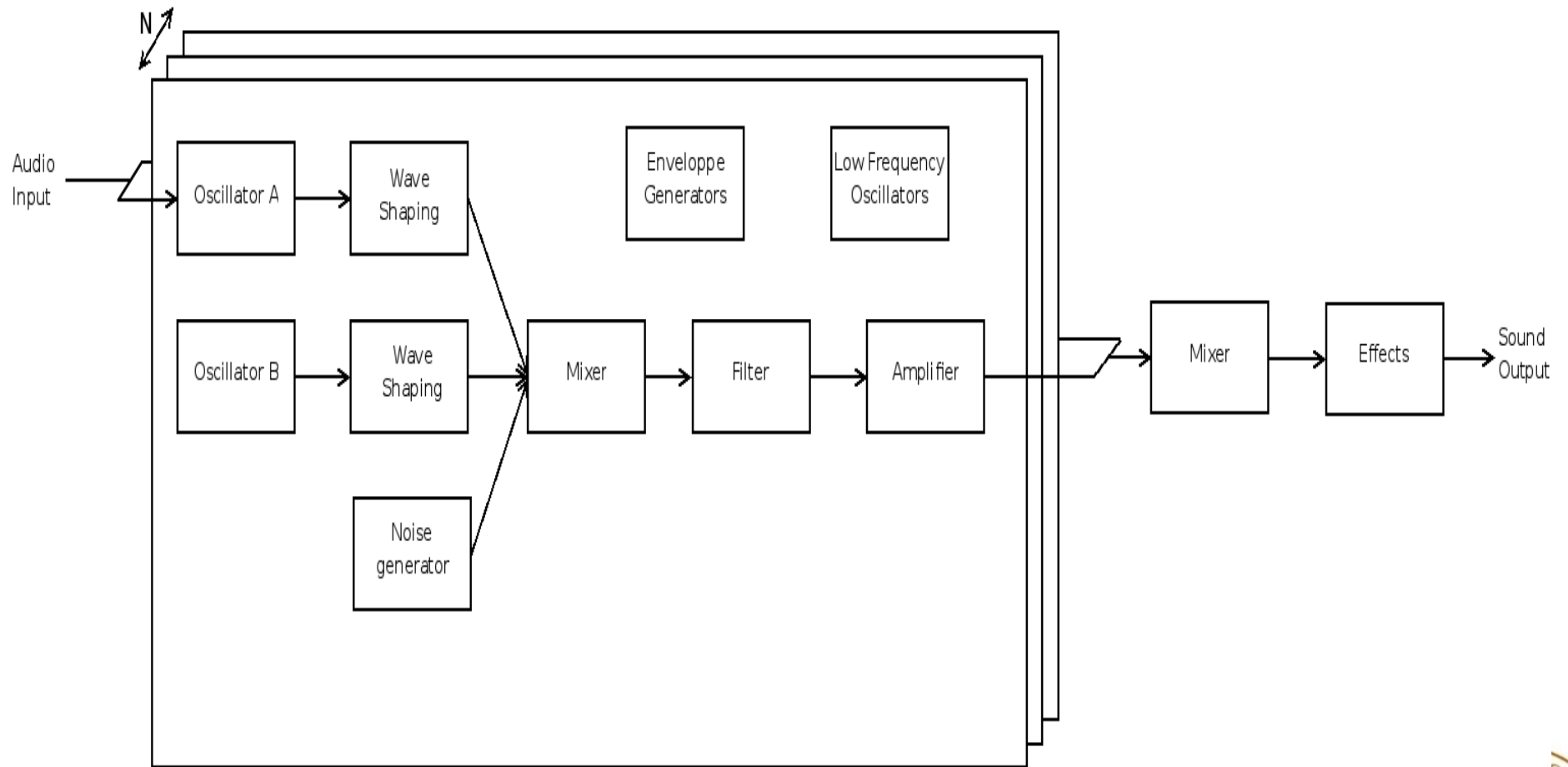
- Affordable and high performance analog modeling synthesizer.



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



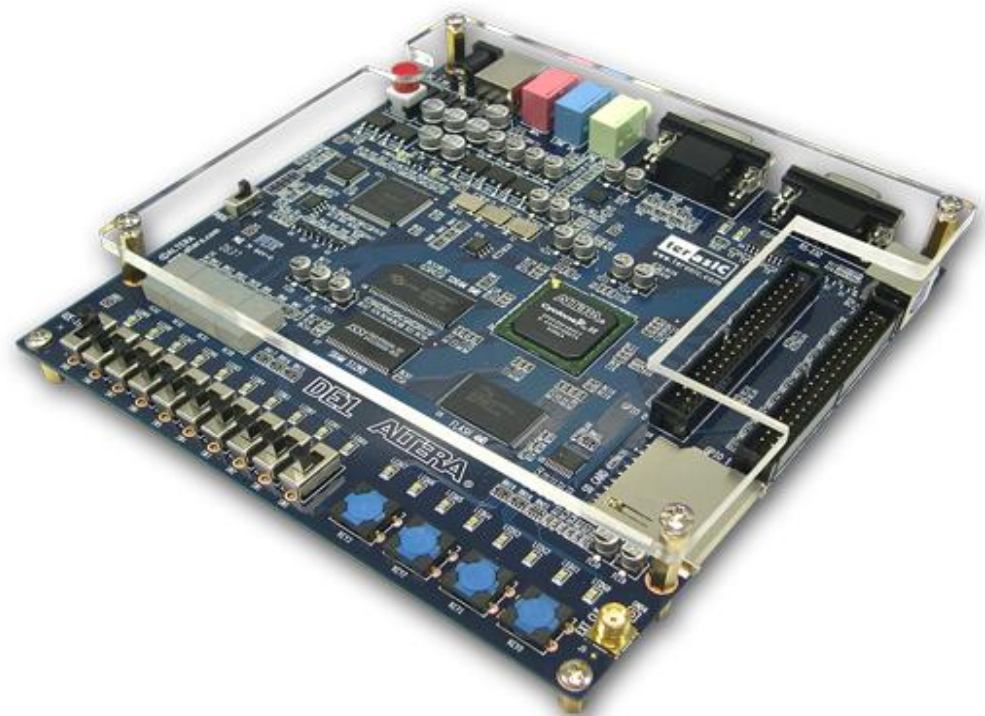
User view on schematic



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



System Implementation

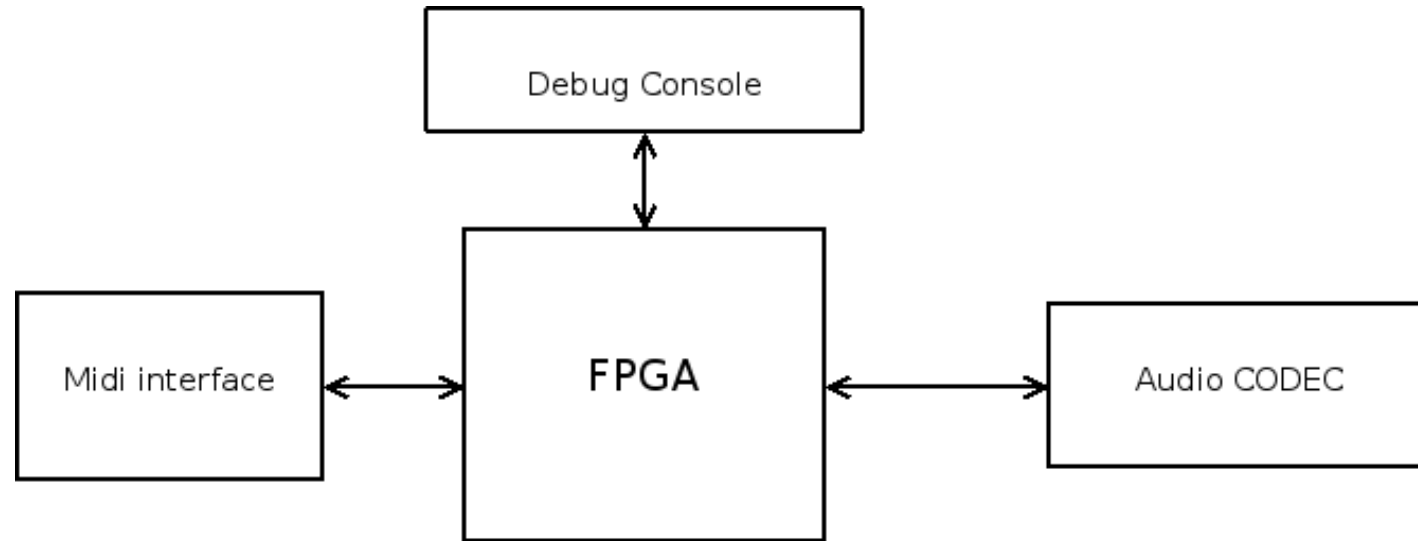


- Introduction
- System Implementation
 - NIOS II system
 - Sound stream
- Software Implementation
 - Project Mapping
 - μ c/os II
- Conclusion & results

Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



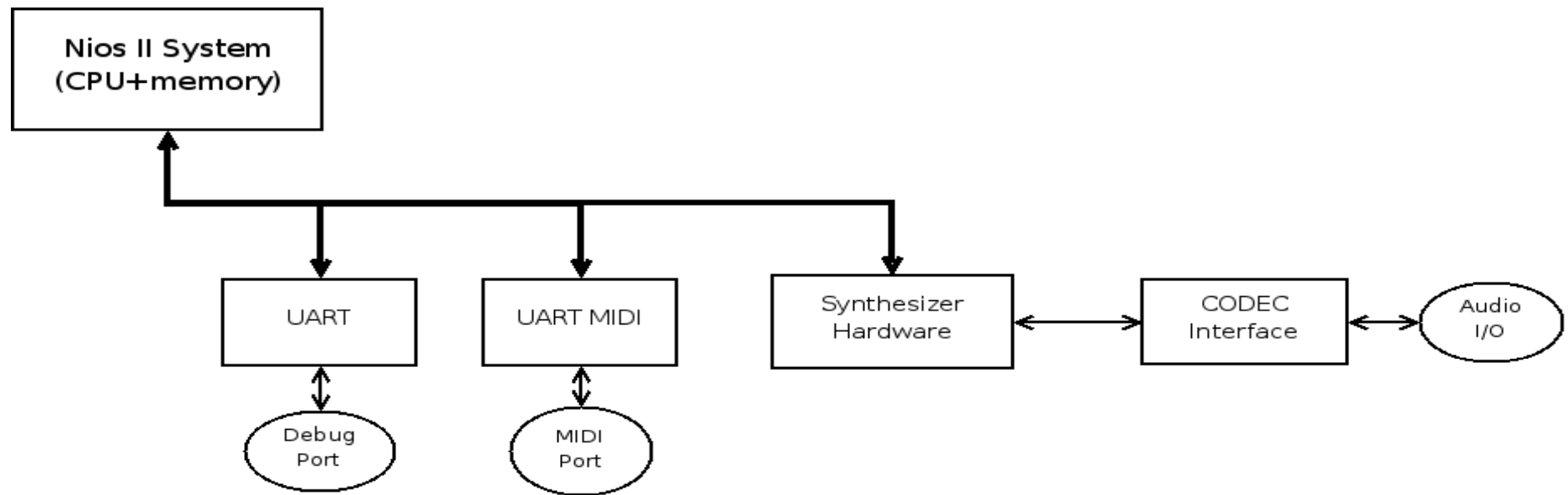
Board level Architecture



- The MIDI interface implemented with serial port.
- The CODEC interfaces is on the DE1 board.



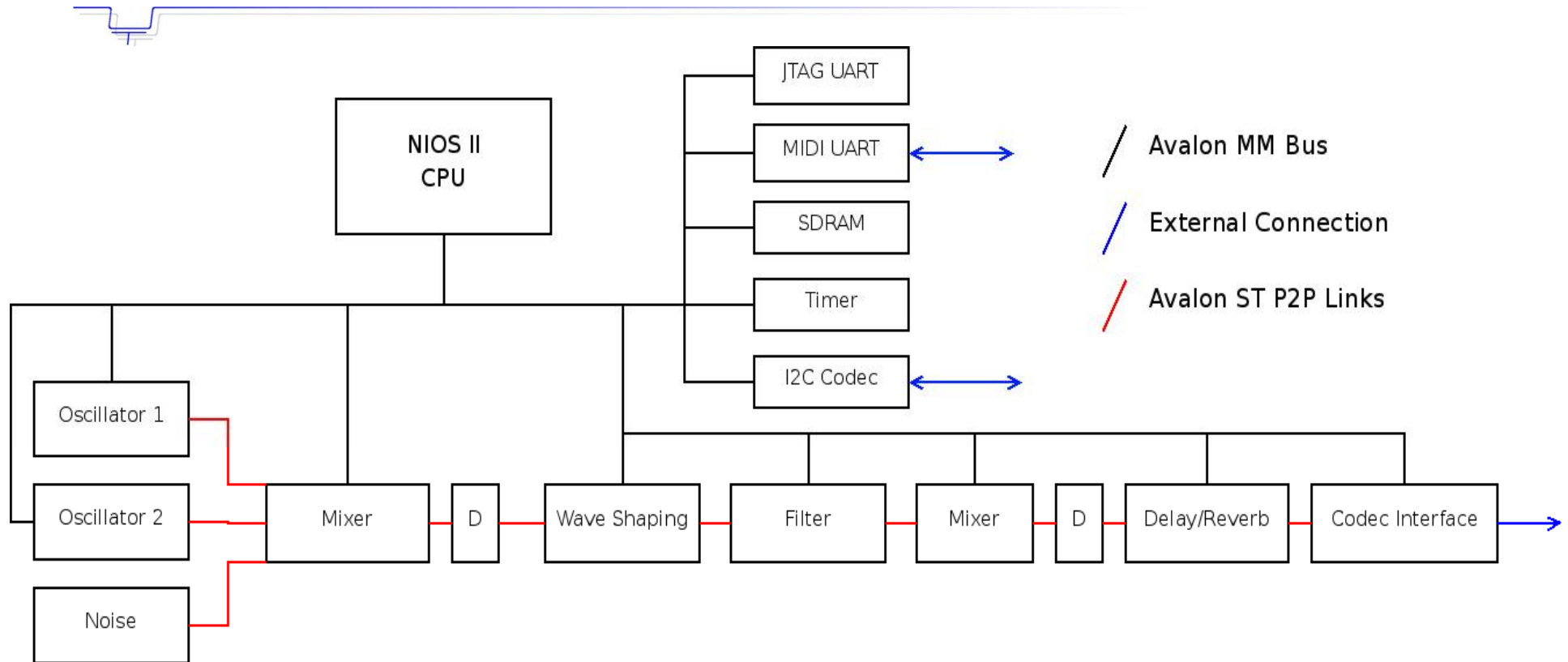
Inside the FPGA



- The program inside the CPU responds to the MIDI command.



Actual system implementation(1)



- Sound stream element configuration mapped to the NIOS II Memory
- Everything is wired in SOPC Builder

Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



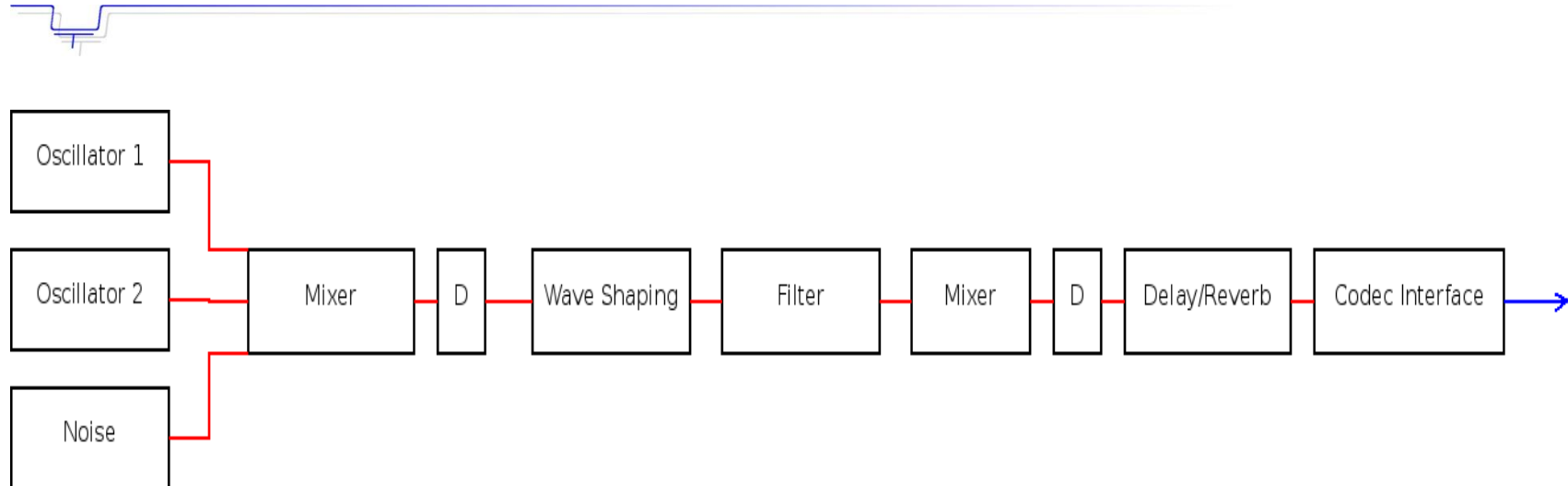
Actual system implementation(2)

Use	Connections	Module Name	Description	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		cpu	Nios II Processor				
		instruction_master	Avalon Memory Mapped Master	clk			
		data_master	Avalon Memory Mapped Master				
		jtag_debug_module	Avalon Memory Mapped Slave				
<input checked="" type="checkbox"/>		jtag_uart	JTAG UART				
		avalon_jtag_slave	Avalon Memory Mapped Slave	clk	0x01003490	0x01003497	0
<input checked="" type="checkbox"/>		jtag_control	JTAG UART				
		avalon_jtag_slave	Avalon Memory Mapped Slave	clk	0x010034a8	0x010034af	1
<input checked="" type="checkbox"/>		onchip_mem	On-Chip Memory (RAM or ROM)				
		s1	Avalon Memory Mapped Slave	clk	0x01001000	0x01001fff	
<input checked="" type="checkbox"/>		sdram	SDRAM Controller				
		s1	Avalon Memory Mapped Slave	clk	0x00800000	0x00ffffff	
<input checked="" type="checkbox"/>		timer	Interval Timer				
		s1	Avalon Memory Mapped Slave	clk	0x01003400	0x0100341f	2
<input checked="" type="checkbox"/>		disp	PIO (Parallel I/O)				
		s1	Avalon Memory Mapped Slave	clk	0x01003480	0x0100348f	
<input checked="" type="checkbox"/>		uart	UART (RS-232 Serial Port)				
		s1	Avalon Memory Mapped Slave	clk	0x01003420	0x0100343f	3
<input checked="" type="checkbox"/>		codec	PIO (Parallel I/O)				
		s1	Avalon Memory Mapped Slave	clk	0x01003460	0x0100346f	
<input checked="" type="checkbox"/>		audio	Audio codec interface				
		s1	Avalon Memory Mapped Slave	clk	0x01003000	0x010033ff	
		sink1	Avalon Streaming Sink				
<input checked="" type="checkbox"/>		mono2stereo_inst	Mono to Stereo ST adapter				
		source1	Avalon Streaming Source	clk			
		sink1	Avalon Streaming Sink				
<input checked="" type="checkbox"/>		wshape	waveshaping				
		s1	Avalon Memory Mapped Slave	clk	0x00000000	0x000007ff	
		source1	Avalon Streaming Source				
		sink1	Avalon Streaming Sink				
<input checked="" type="checkbox"/>		mixer	Oscillator mixer				
		s1	Avalon Memory Mapped Slave	clk	0x01003470	0x0100347f	
		source1	Avalon Streaming Source				
		osc0	Avalon Streaming Sink				
		osc1	Avalon Streaming Sink				
		noise	Avalon Streaming Sink				
<input checked="" type="checkbox"/>		saw0	Sawtooth generator				
		source1	Avalon Streaming Source	clk			
		s1	Avalon Memory Mapped Slave		0x01003498	0x0100349f	
<input checked="" type="checkbox"/>		saw1	Sawtooth generator				
		source1	Avalon Streaming Source	clk			
		s1	Avalon Memory Mapped Slave		0x010034a0	0x010034a7	
<input checked="" type="checkbox"/>		noise	Noise generator				
		source1	Avalon Streaming Source	clk			

Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



Sound Stream



- Based on Avalon ST Bus
- Clocked by the CODEC
- Back Pressure Avalon ST capability
 - The samples are 'pull' through the stream by the CODEC
- 'D' elements to implement a pipeline



Software Implementation

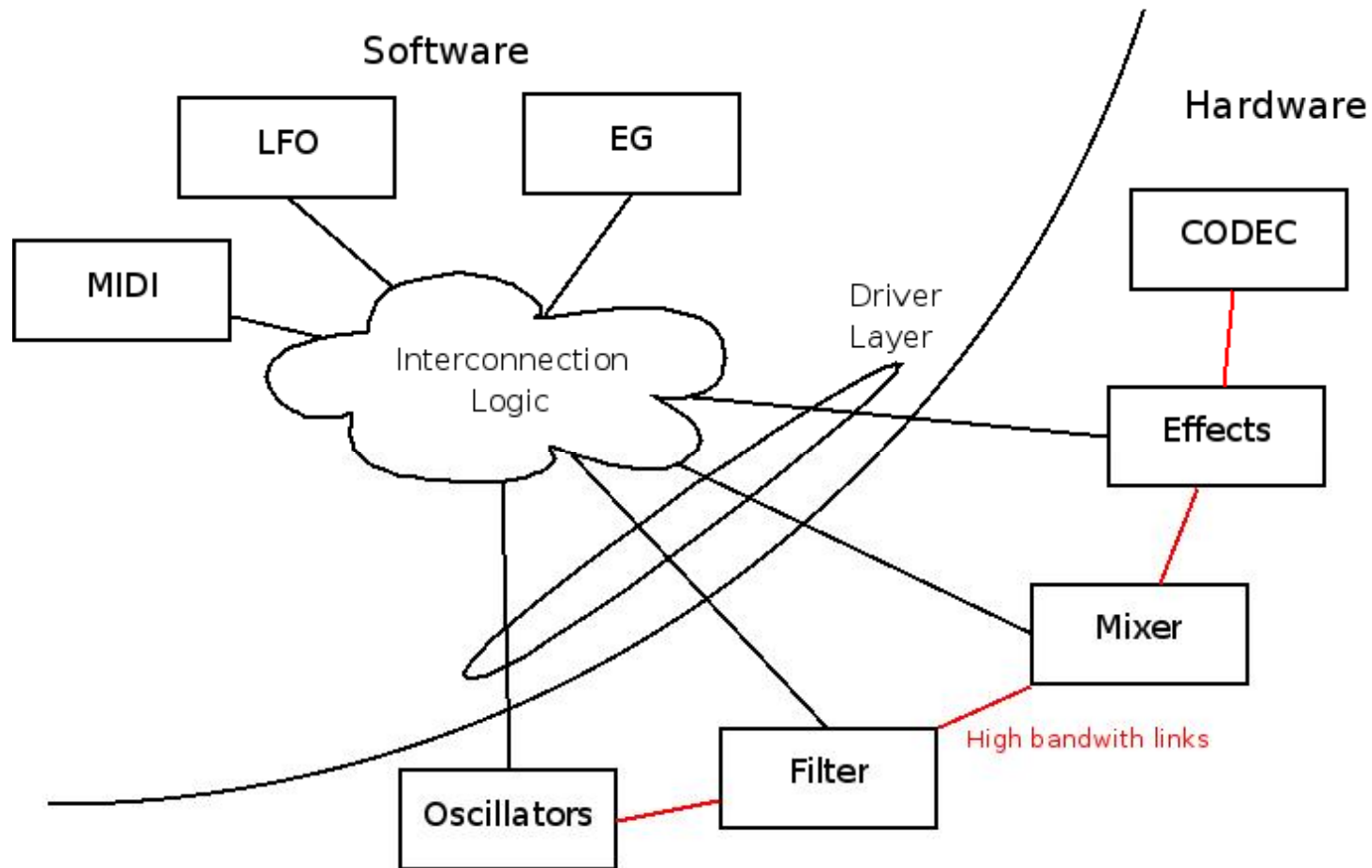
- Introduction
- System Implementation
- Software Implementation
 - Project Mapping
 - μ c/os II
- Conclusion & results



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



Project Mapping



Analog Modeling Synthesizer: Arnaud Taffanel, Peyman Pouyan



Project Mapping(continued)...



- Rules for Mapping:
 - . All the blocks which are in the sound flow will be implemented in hardware.
 - . All the slow or computational blocks will be implemented in software.
- The interconnection between all the hardware blocks is simplified by the usage of the Avalon bus.
- All the design is clocked by the same 50MHz clock which is also routed by SOPC.



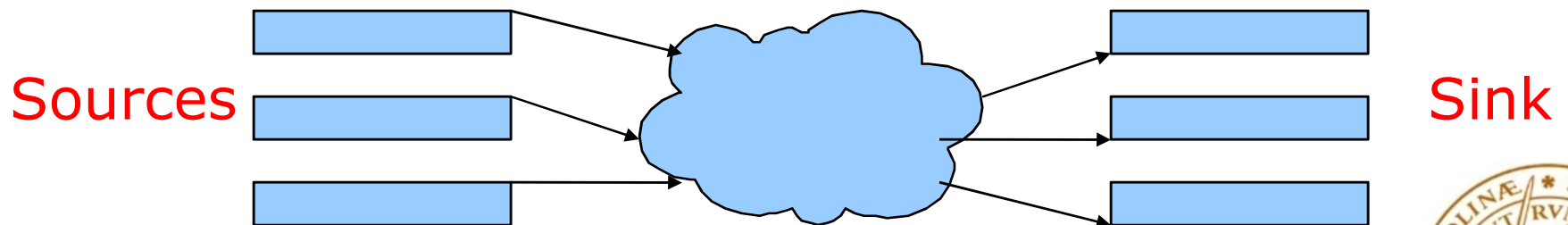
μC/OS II

- Simple and efficient RTOS
- Integrated to NIOS II IDE
- Mainly 3 tasks implemented
 - . **Midi Task** : receive and execute the MIDI commands
 - . **EG Task** : envelope generator refresh
 - . **LFO Task** : Low frequency oscillator refresh



Software organisation

- Operating Tasks
- Interconnection matrix system
 - Almost everything can be interconnected dynamically
 - Define 2 connectors
 - Sink to receive data (i.e. Oscillators, Filter)
 - Source to emit data (i.e. LFO/EG/MIDI)
 - Automatic refresh
- Control system to modify non-dynamic data



Conclusion(1)

- FPGA itself was pretty adapted for the signal processing as:
 - It contains a lot of internal RAM .
 - It has a lot of multipliers which permit to create many high performance design blocks.
 - It has a Parallel Architecture which can help us to achieve a better throughput.
- FPGAs are cheaper than DSPs .
- The Avalon bus system is very efficient and simple to implement
 - Mostly thanks to SOPC Builder



Performance (2)

- Smooth configuration
 - Easy to accomplish with an FPGA
- Polyphony
 - Pipelined architecture of FPGA
 - 1000 cycles/sample available
 - Actual Implementation should achieve a polyphony of at least 100.
- Minimal response time
 - Sample processing Vs. block processing



References

- www.altera.com
- www.wikipedia.org
- www.dsppmusic.com
- www.micrium.com

