Design and Verification of a Complete Application Specific IC

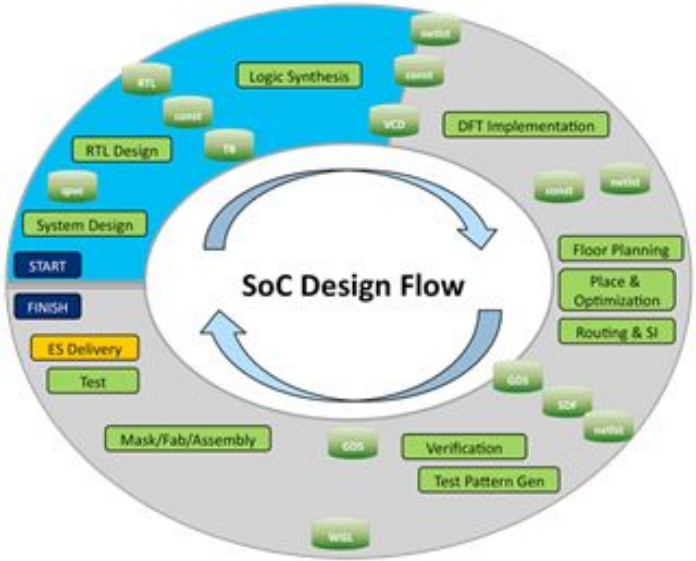
Zachary Nelson*, Kevin Cao, Whitley Forman, Dhruvit Naik and Julie Swift
* Team Leader
Advisors: Dr. Orlando Hernandez and Dr. Larry Pearlstein

December 2nd, 2015

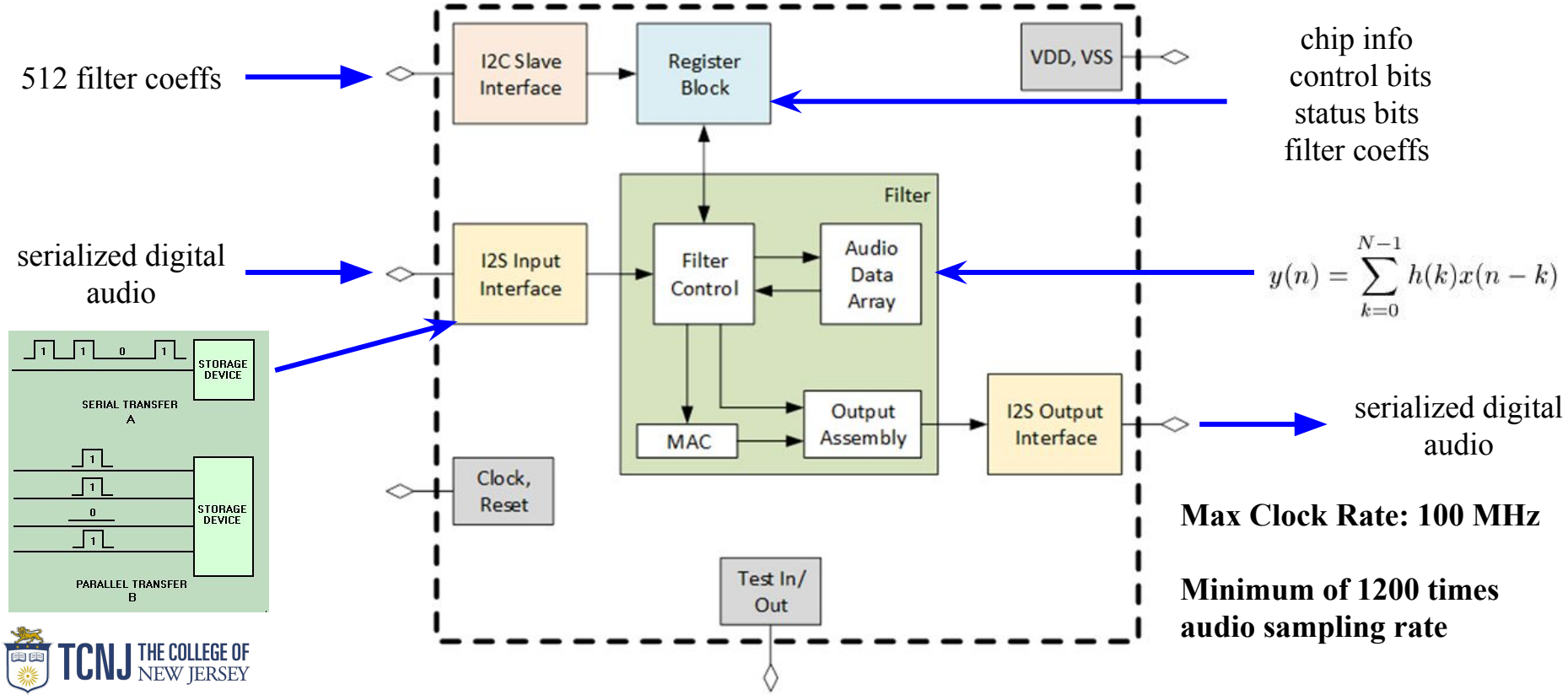
TCNJ THE COLLEGE OF NEW JERSEY

Introduction

Project Goal: Produce an application-specific integrated circuit (ASIC) that processes digital audio data by applying a 512-tap digital finite impulse response (FIR) filter to the input stream.



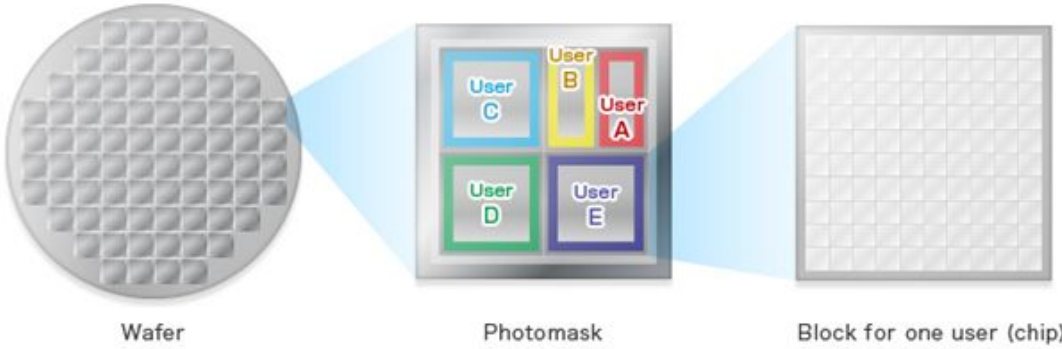
Block Diagram



TCNJ THE COLLEGE OF NEW JERSEY

MOSIS

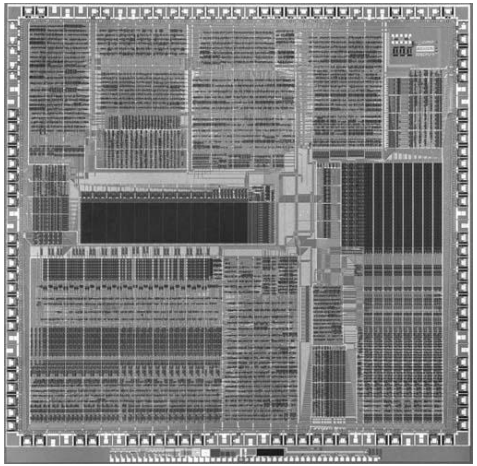
- Metal Oxide Semiconductor Implementation Service
- Known for multi-project wafers (MPW)
- 3 options for accredited programs
  - Commercial account, **Instructional account**, Research account
- GlobalFoundaries 180 nm CMOS (7HV) or (7RF) process
- Due Date: **March 7th, 2016**
- Caltech Intermediate Format (CIF) or Graphic Data System II (GSDII) Format



ASIC vs FPGA

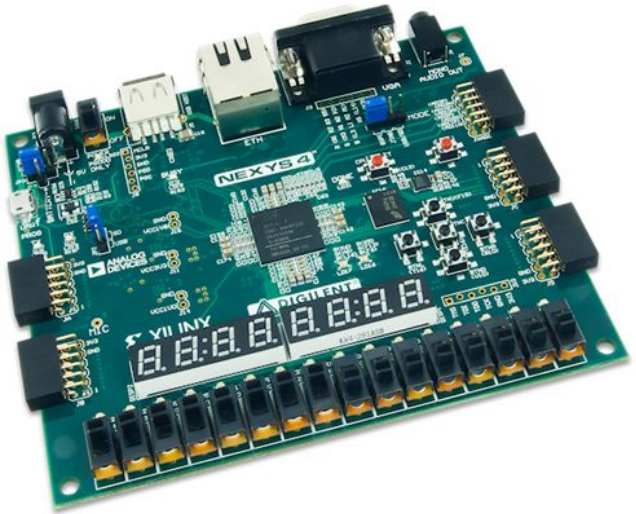
ASIC

- Fully custom design
- Lower unit cost for high quantities
- Smaller size



FPGA

- Reprogrammable
- No manufacturing
- Simple



EDA Tools

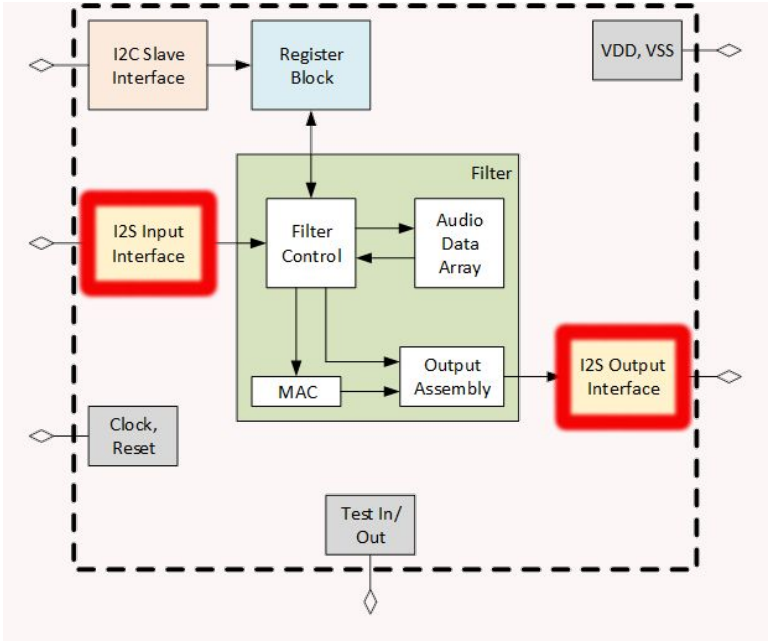
Why EDA Tools?

- Mentor Graphics Design Tool

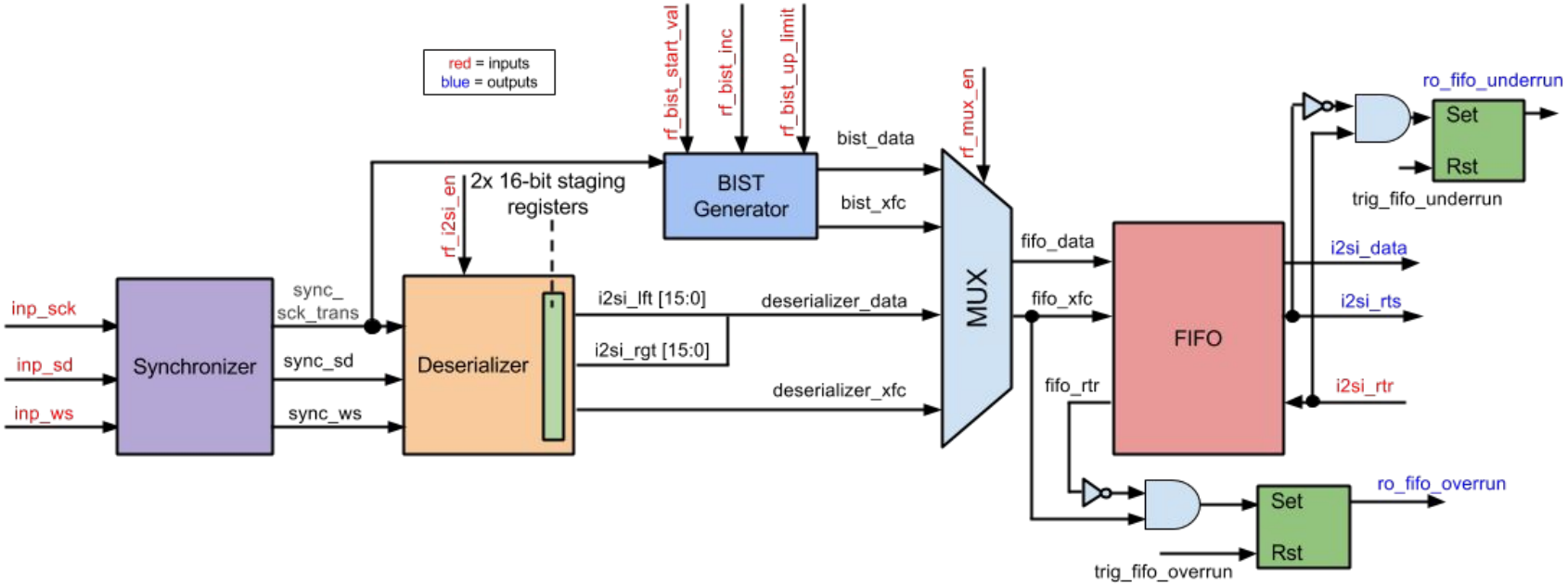
- Logic Synthesis
- Static Timing Analysis
- Design For Testing
- Gate Level Simulations
- Place and Route Gates

I2S

- Integrated Interchip Sound
- Electrical serial bus interface standard that is used for connecting digital audio devices together



I2S Input Interface



I2S Input Interface: Synchronizer

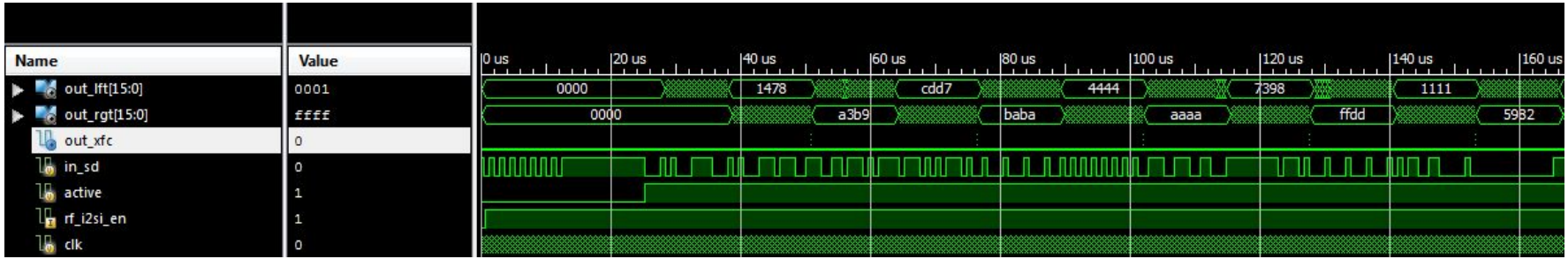
Signal Name	Direction	Bits	Comment
clk	in	1	Master Clock
rst_n	in	1	Reset
_sck	in	1	Digital Audio Bit Clock (max of 48kHz)
_ws	in	1	Word Select (Left/Right Audio Channel)
_sd	in	1	Digital Audio Serial Data
sck	out	1	Delayed and Synced Serial Clock
sck_transition	out	1	Serial Clock Level to Pulse Converter
ws	out	1	Delayed and Synced Word Select
sd	out	1	Delayed and Synced Serial Data

I2S Input Interface: Synchronizer

I2S Input Interface: Deserializer

Signal Name	Direction	Bits	Comment
clk	in	1	Master Clock
rst_n	in	1	Reset
sck_transition	in	1	Serial Clock Level to Pulse Converter
in_sd	in	1	Digital Audio Serial Data
in_ws	in	1	Word Select
rf_i2si_en	in	1	I2S Input is Enabled
out_lft	out	16	Left Audio Channel
out_rgt	out	16	Right Audio Channel
out_xfc	out	1	Read Data Transfer Complete

I2S Input Interface: Deserializer

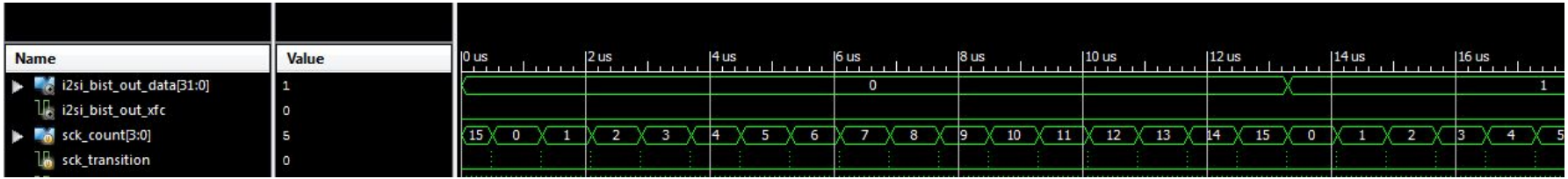


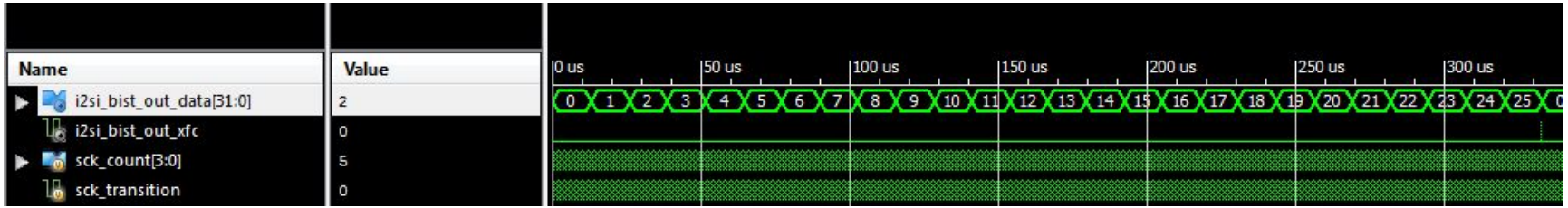
test_data [0] [0] = 16'hAAAA;
test_data [0] [1] = 16'hFFFF;
test_data [1] [0] = 16'h1478;
test_data [1] [1] = 16'hA3B9;
test_data [2] [0] = 16'hCDD7;
test_data [2] [1] = 16'hBABA;
test_data [3] [0] = 16'h4444;
test_data [3] [1] = 16'hAAAA;
test_data [4] [0] = 16'h7398;
test_data [4] [1] = 16'hFFDD;
test_data [5] [0] = 16'h1111;
test_data [5] [1] = 16'h5982;

I2S Input Interface: BIST Generator

Signal Name	Direction	Bits	Comment
clk	in	1	Master Clock
rst_n	in	1	Reset
sck_transition	in	1	Serial Clock Level to Pulse Converter
rf_bist_start_val	in	12	Start Value
rf_bist_inc	in	8	Increment
rf_bist_up_limit	in	12	Upper Limit
i2si_bist_out_xfc	out	1	Transfer Complete
i2si_bist_out_data	out	32	Output Data

I2S Input Interface: BIST Generator

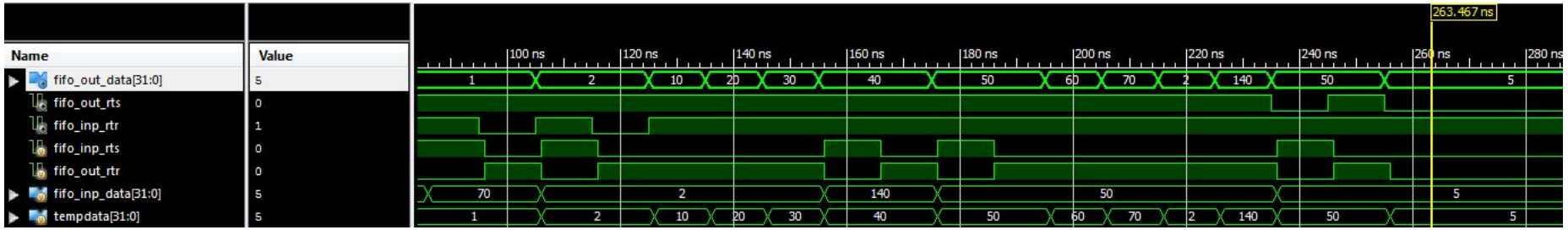




I2S Input Interface: FIFO

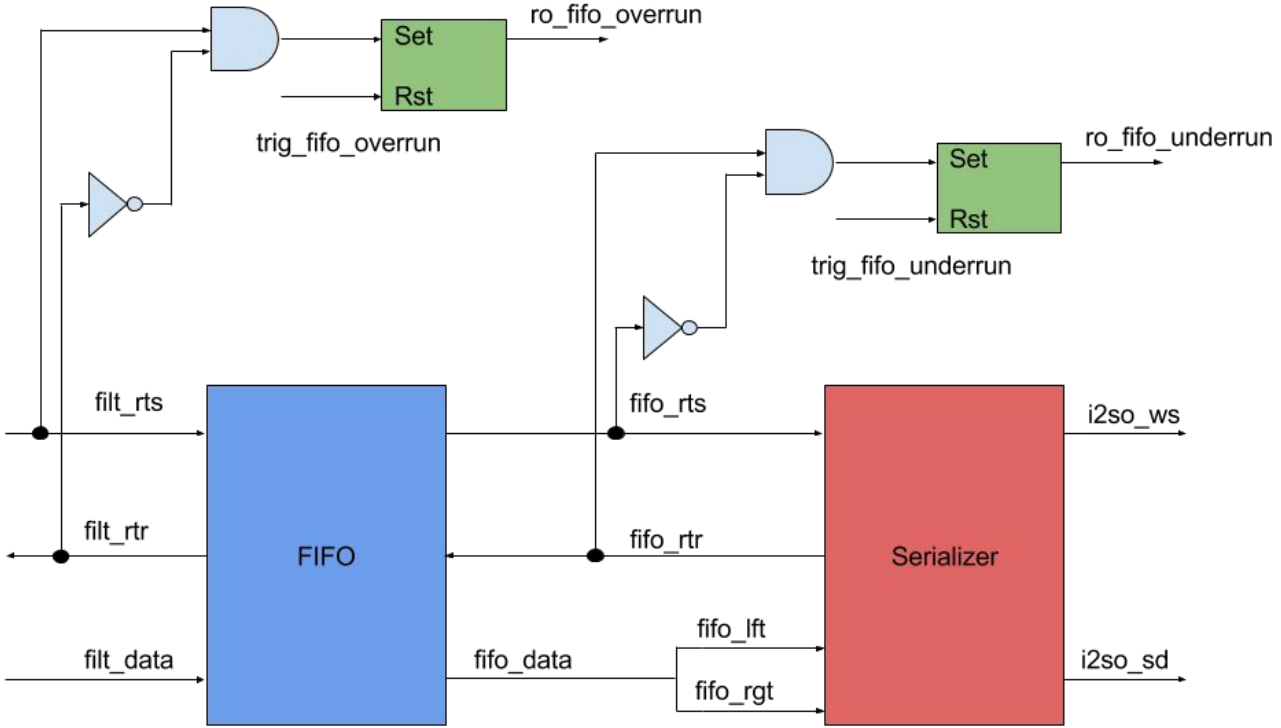
Signal Name	Direction	Bits	Comment
clk	in	1	Master Clock
rst_n	in	1	Reset
fifo_inp_data	in	32	Input Data
fifo_inp_rts	in	1	Write Client Asserts Ready to Send
fifo_inp_rtr	out	1	Output FIFO Asserts Read to Receive
fifo_out_data	out	32	Output Data
fifo_out_rts	out	1	Output FIFO Asserts Ready to Send
fifo_out_rtr	in	1	Read Client Asserts Read to Receive

I2S Input Interface: FIFO



Pushed 1
Pushed 2
----------Poped 1
Pushed 10
Pushed 20
Pushed 30
Pushed 40
Pushed 50
Pushed 60
Pushed 70
---Cannot push: Buffer Full---
---Cannot push: Buffer Full---
---Cannot push: Buffer Full---
---Cannot push: Buffer Full---
---Cannot push: Buffer Full---
---Cannot push: Buffer Full---
----------Poped 2
Pushed 2
----------Poped 10
----------Poped 20
----------Poped 30
----------Poped 40
Pushed 140
----------Poped 50
Pushed 50
----------Poped 60
----------Poped 70
----------Poped 2
----------Poped 140
----------Poped 50
---Cannot Pop: Buffer Empty---
---Cannot Pop: Buffer Empty---
---Cannot Pop: Buffer Empty---
---Cannot Pop: Buffer Empty---
---Cannot Pop: Buffer Empty---
---Cannot Pop: Buffer Empty---
Pushed 5
----------Poped 5

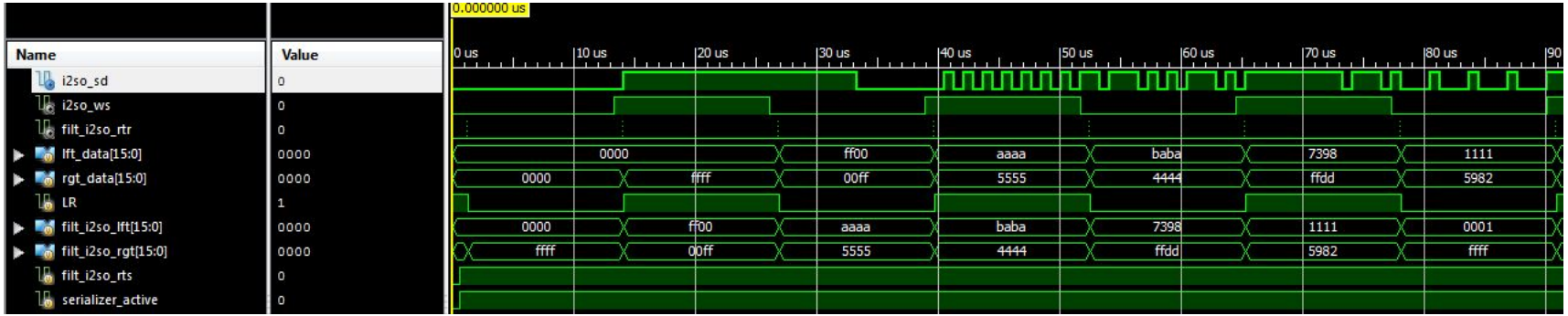
I2S Output Interface



I2S Output Interface: Serializer

Signal Name	Direction	Bits	Comment
clk	in	1	Master Clock
rst_n	in	1	Reset
sck_transition	in	1	Serial Clock Level to Pulse Converter
filt_i2so_lft	in	16	Left Audio Channel
filt_i2so_rgt	in	16	Right Audio Channel
filt_i2so_rts	in	1	Ready to send
filt_i2so_rtr	out	1	Ready to read
i2so_en	in	1	I2S Output is Enabled
i2so_sd	out	1	Digital Audio Serial Data
i2so_ws	out	1	Word Select

I2S Output Interface: Serializer



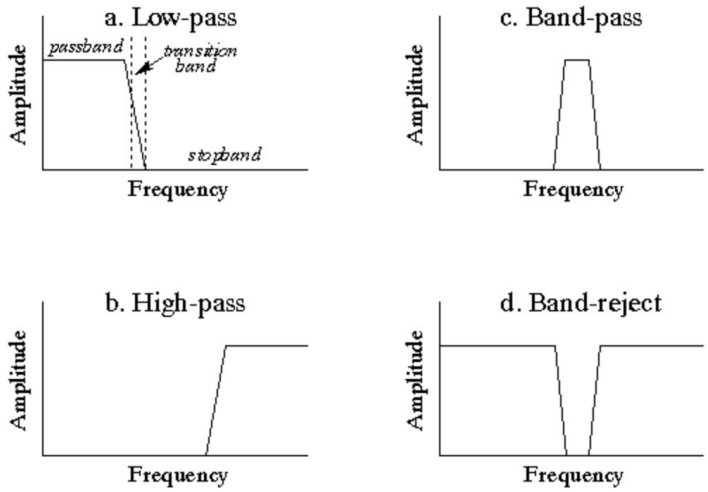
What is Digital Filtering?

Two Most General Purposes of Filters:

- Separation of Signals
- Restoration of Signals

The Four Most Common Frequency Responses

- Low-Pass
- High-Pass
- Band-Pass
- Band-Reject



Finite Impulse Response

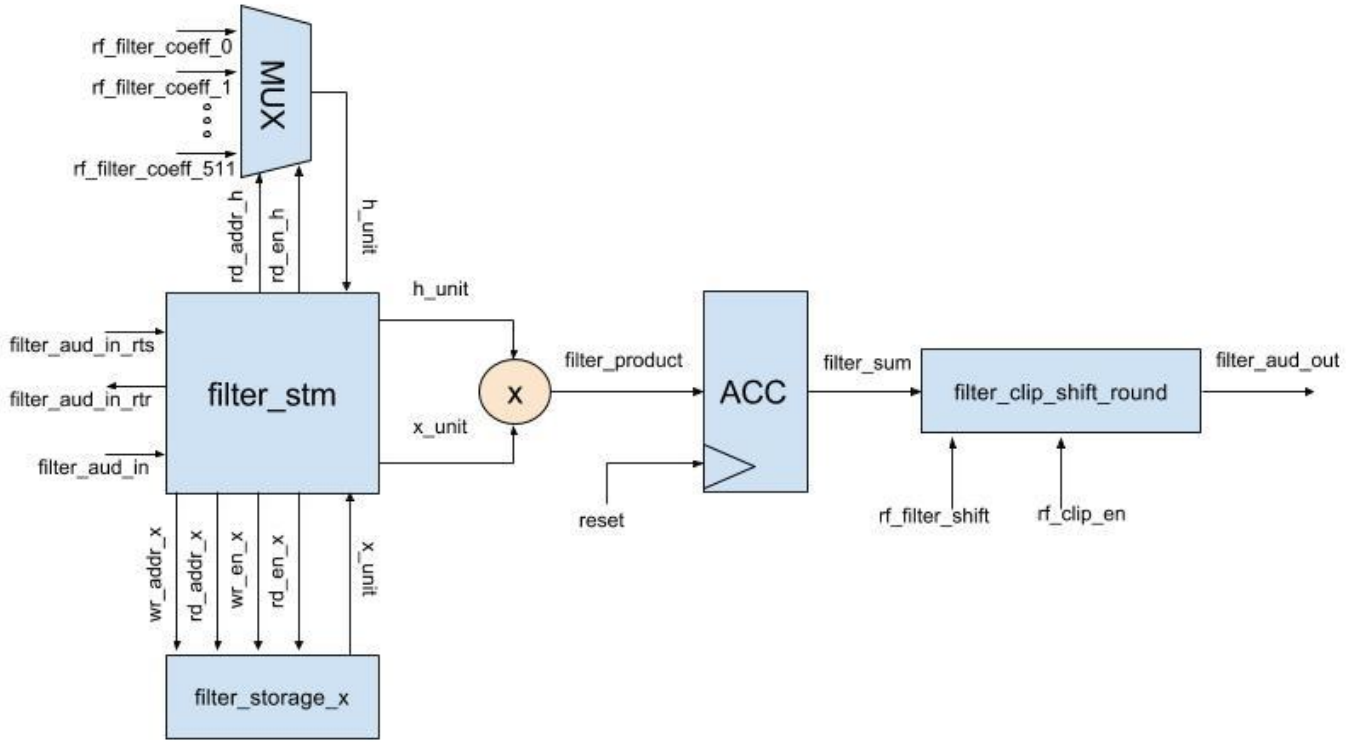
The most straightforward method to implement a Filter is convolution:

$$y[n] = \sum_{k=0}^{N-1} h[k]x[n-k]$$

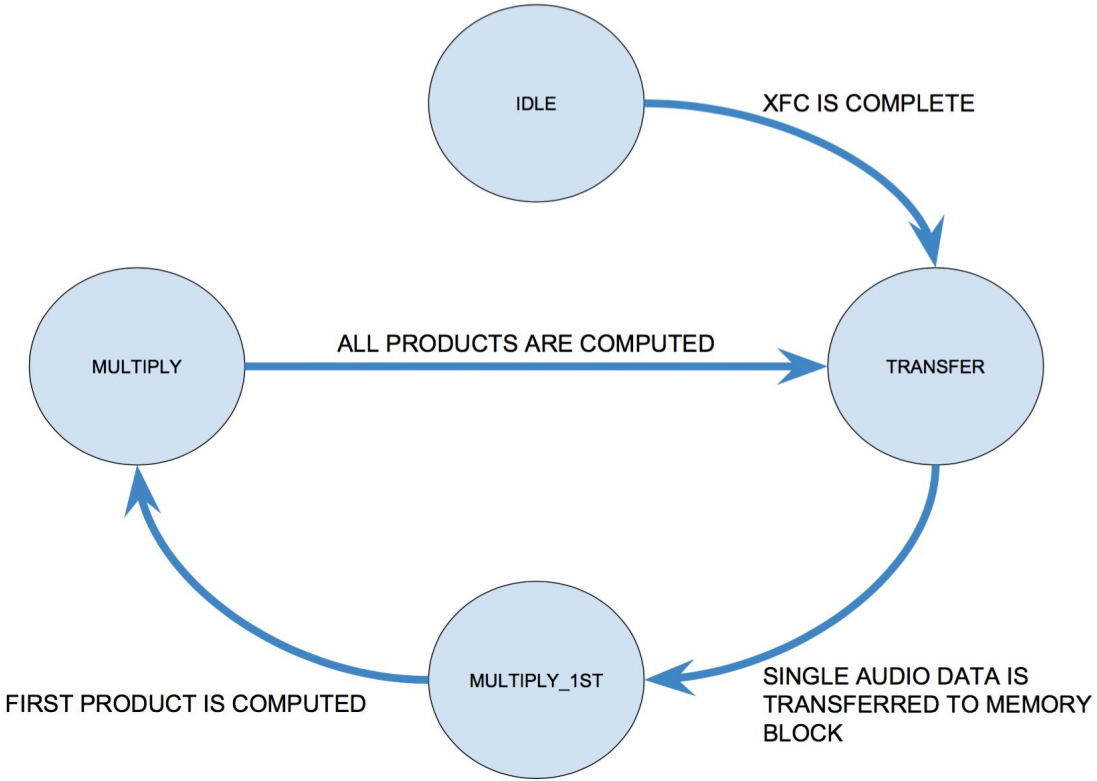
Requirements:

- 512-tap filter
- Programmable filter coefficients to achieve different filter types

Filter Block Diagram



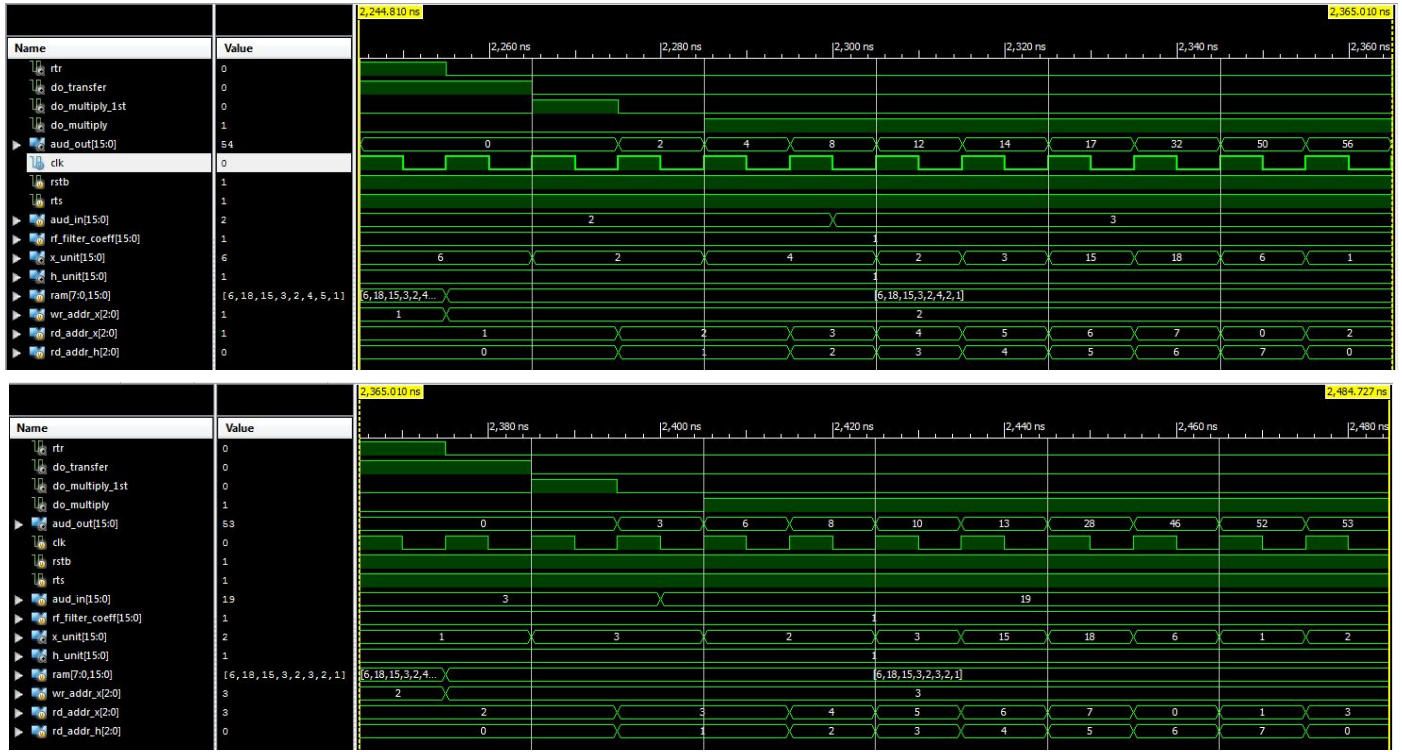
Filter Finite State Machine



Audio Data Array

Signal Name	Direction	Bits	Comment
clk	in	1	Master Clock
wren	in	1	1-wrdata will be written into wrptr
wrptr	in	8	Current Location of Write Pointer
wrdata	out	32	Data to be Written
rden	in	1	1-rddata will be read from rdptr
rdptr	in	8	Current Location of Read Pointer
rddata	out	32	Data Being Read

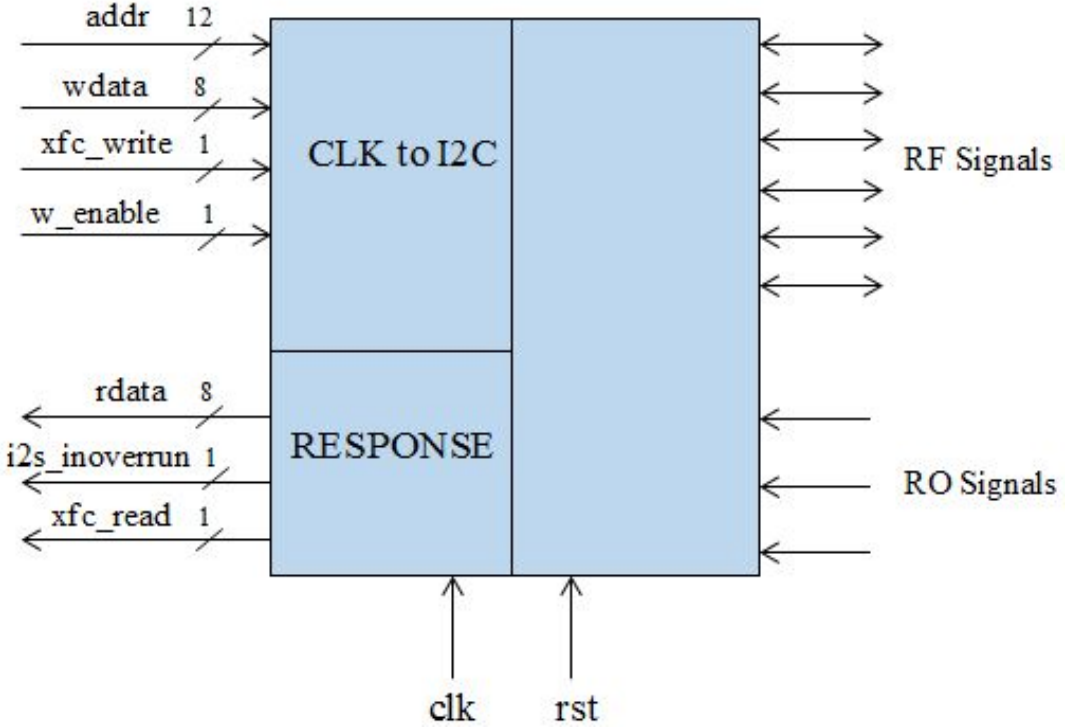
Example Outputs



Register Block Functionality

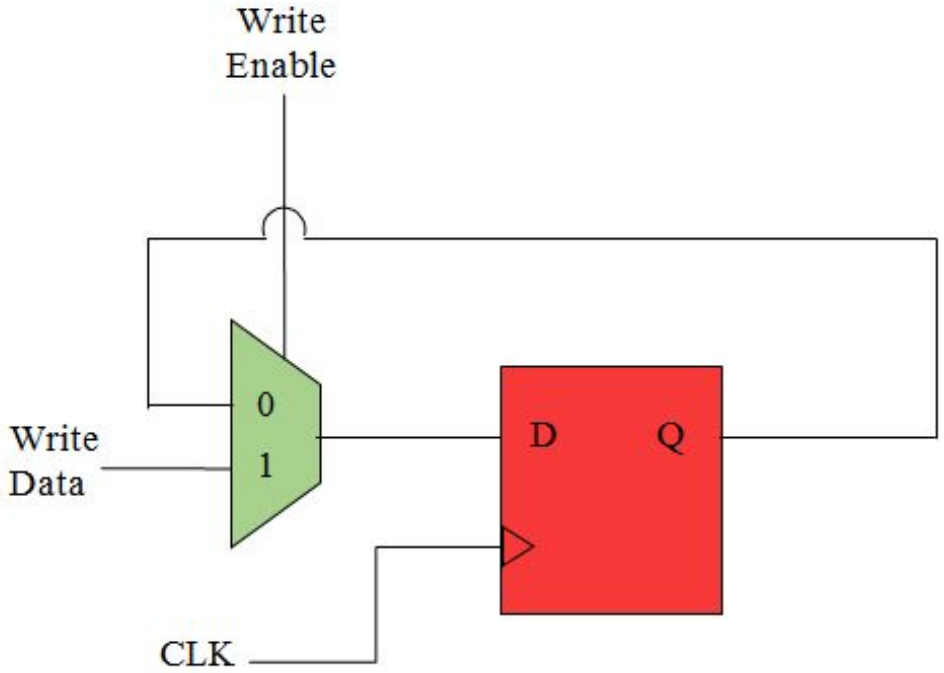
- Provides read/write access to control/status registers
- Source select bit (I2S vs. BIST)
- Filter bypass bit (pass through or filter input stream)
- 512 16-bit filter coefficients
- Overflow/saturation detector
- Audio FIFO overrun/underrun
- Clear bits to reset sticky (latching) detectors

Register Block



Register Block

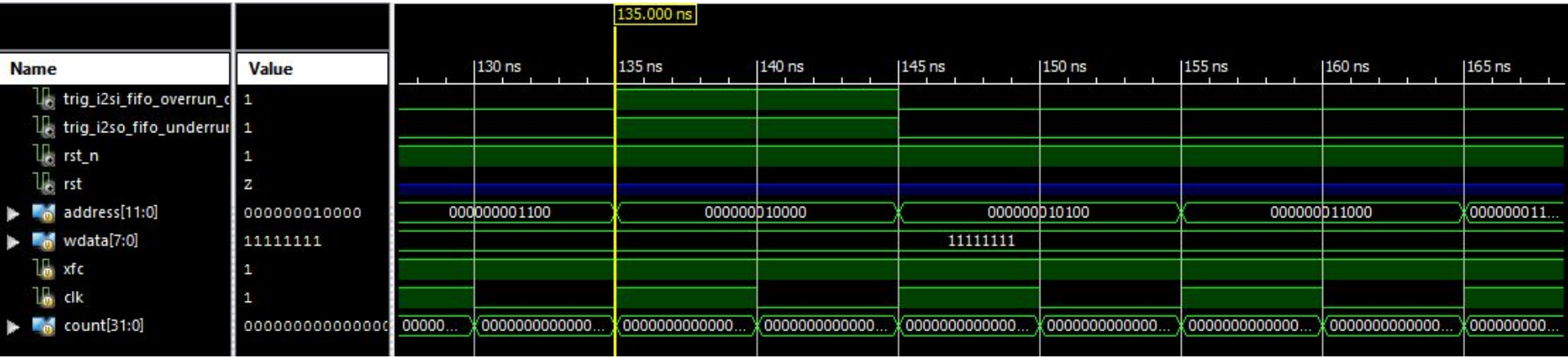
- Given address, write enable, and write data, the data demultiplexer writes to the correct register bit(s) based on the address.



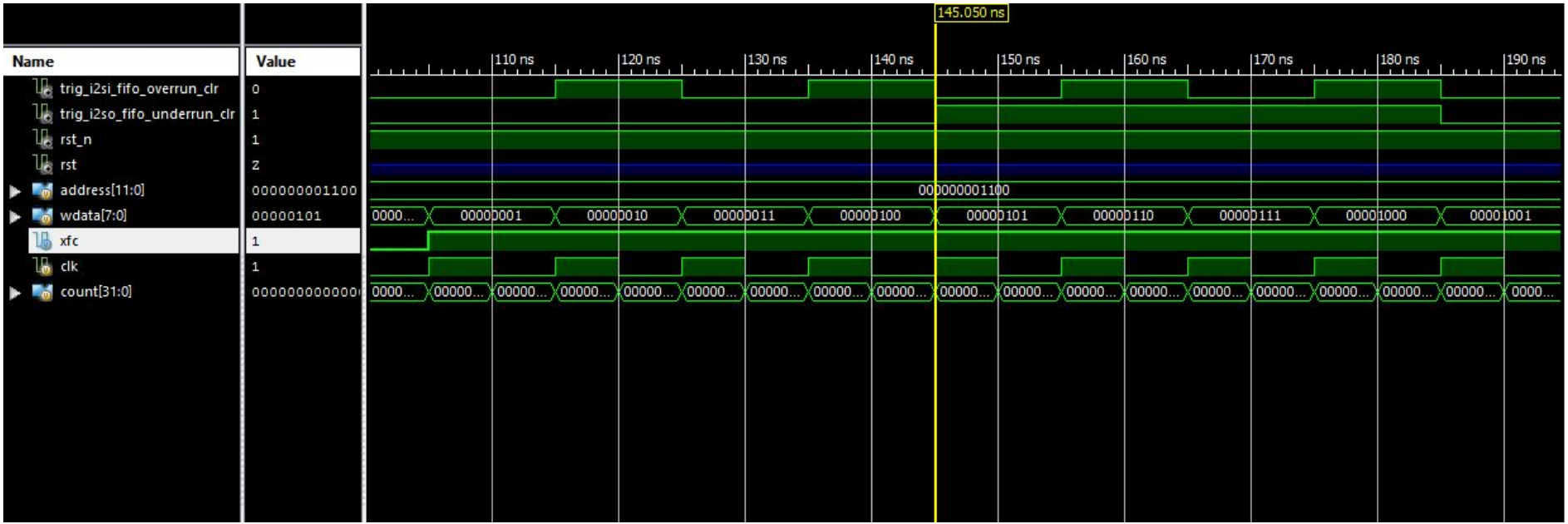
Address	Register Name	Field Name	Bits	Description	RO/WO/RW	Default Value
0x000	CHIP_INFO					
		ro_chip_id	7:0	Fixed Chip ID	RO	0x1234
		ro_revision_id	15:8	Fixed Revision ID	RO	0
0x004	CONTROL					
		rf_soft_reset	1	0- normal operation. 1- assert soft reset	RW	0
		rf_i2si_bist_en	1	0- audio sorce is i2si. 1- audio source is BIST	RW	0x1
		rf_filter_shift	4	number of bit postions to shift after filter accumulator	RW	0xF
		rf_filter_clip_en	1	1- performs clipping 0- no cliping	RW	0x1
	optional?	rf_i2si_dec_factor	4	sample and hold audio values	RW	0
	optional?	rf_i2so_dec_factor	4	sample and hold audio values	RW	0
0x008	I2S_CLOCK_CONTROL					
		rf_i2so_clk2sck_div	16	half of the clock frequency divided by this #	RW	0x40
0x00C	STATUS					
		trig_fifo_overrun	1	fifo overrun clear	WO	NA
		ro_fifo_overrun	1	input audio fifo overrun	RO	0
		trig_fifo_underrun	1	fifo underrun clear	WO	NA
		ro_fifo_underrun	1	output audio fifo underrun	RO	0
0x010	BIST					
		rf_i2si_bist_start_val	12	start value of sawtooth wave	RW	0x800
		rf_i2si_bist_incr	8	increment of sawtooth wave	RW	0x010
		rf_i2si_bist_upper_limit	12	upper limit of the sawtooth wave	RW	0x7FF
0x014	I2C_REG_INDIR_ADDR					
		rf_i2c_reg_indir_addr	11	address register used for indirect addressing via i2c	RW	0
0x018	I2C_REG_INDIR_DATA					
		ro_i2c_reg_indir_data	8	data access register	RO	NA
0x040	FILT_COEFFS_0_1					
		rf_filter_coeef0	15:0	Filter Coefficient 0	RW	0x0
		rf_filter_coeef1	31:16	Filter Coefficient 1	RW	0x0
0x044	FILT_COEFFS_2_3					
		rf_filter_coeef2	15:0	Filter Coefficient 2	RW	0x0
		rf_filter_coeef3	31:16	Filter Coefficient 3	RW	0x0
0x43C	FILT_COEFFS_510_511					
		rf_filter_coeef510	15:0	Filter Coefficient 510	RW	0x0
		rf_filter_coeef511	31:16	Filter Coefficient 511	RW	0x0

Register Trigger Generator

- trig_i2si_fifo_overrun_clr - dwata[0] & xfc = 1
  - the I2S FIFO is trying to be written and the FIFO is full
- trig_i2so_fifo_underrun_clr - dwata[2] & xfc = 1
  - the I2S FIFO is trying to be read and the FIFO is empty



Register Trigger Generator 1

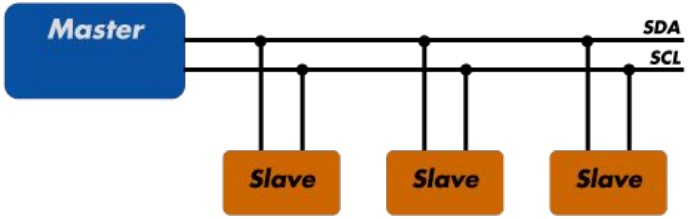


C Programming Involvement

- 512 two part (a and b) filter coefficients
- Initialized all coefficients to address 16’h000
- Set all coefficients to appropriate address
  - 16’h400 - 16’h7ff

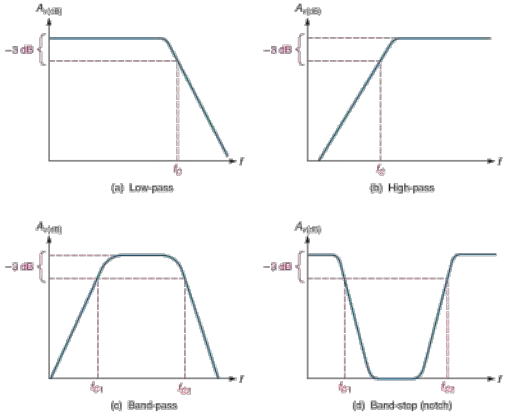
What is I2C?

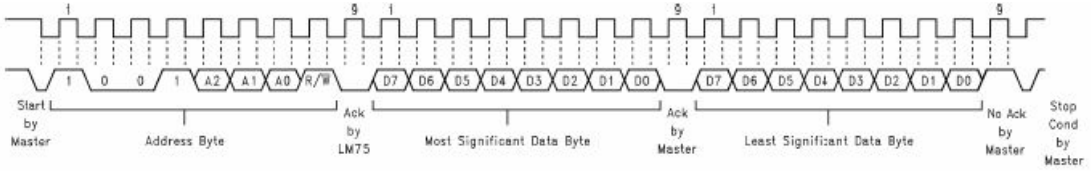
- I2C is a serial to parallel interface between devices developed in 1982
- Uses only 2 wires, serial clock and serial data
- Read and write capabilities between devices
- Up to 128 devices on a single bus
- 400 kbits/second up to 3.4 Mbits/second
- Master and multiple slave architecture
- Widespread use in current electronics designs
- Highly flexible data transfer capabilities



I2C Utilization for Our Project

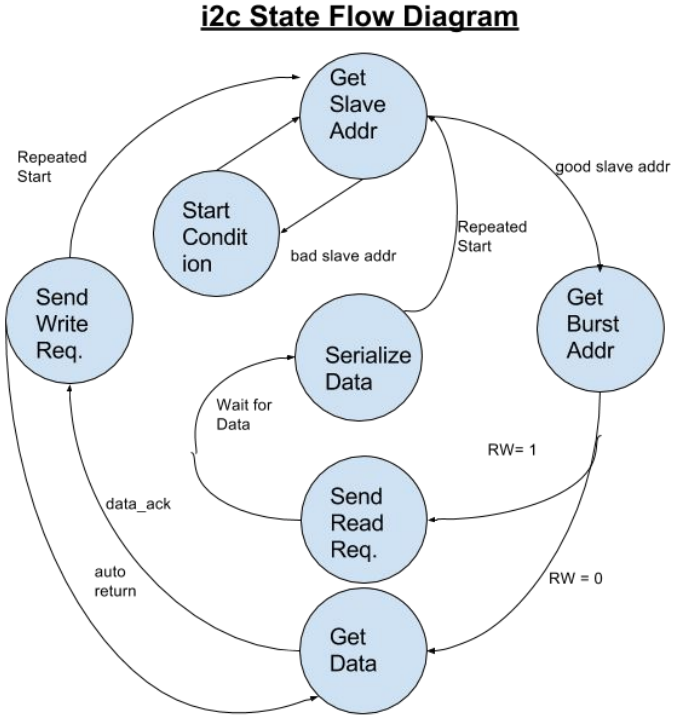
- Transfer 1 byte filter coefficients to to describe the filter type
- 1024 coefficients used by filter
  - Can transfer all within .023 Seconds
- Will read back coefficients to ensure correct data transfer
- Operate at 400 kbit/second frequency
- Written Using Verilog
- Custom Data Transfer Protocol
  - First 2 bytes are register address, rest are data
- Our project will use a PSoC 5LP to act as I2C Master





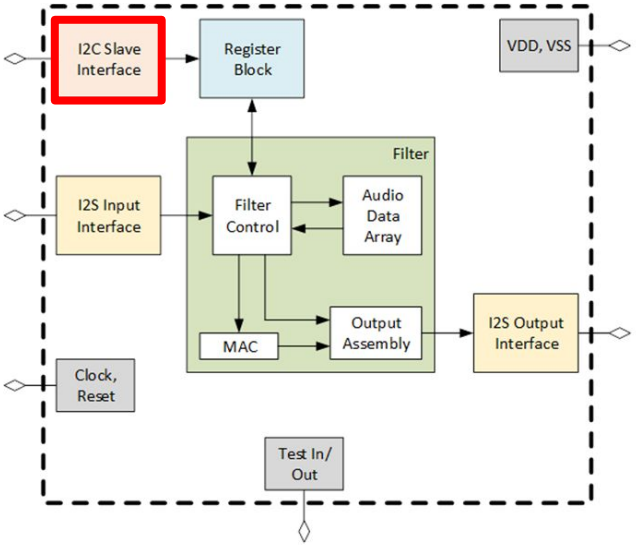
I2C State Machine

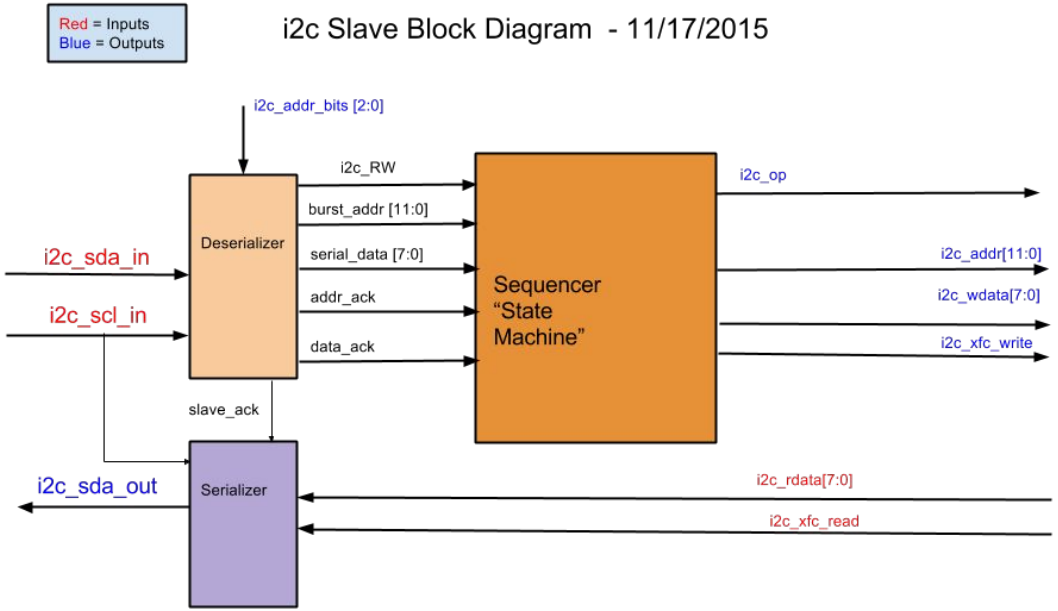
- Utilizes Write Data Stream
  - Minimizes write time
- Repeated Start for Read
  - Minimizes Read Time



*A return to the start state can happen during any part of the cycle if a stop or reset condition is seen

I2C Block Diagram





Budget

Software Expenses

Item	Cost
ISE Design Suite 14.7	$0.00
CORE 9 University	$0.00
Dropbox	$0.00
Git/GitHub Desktop	$0.00
Microsoft Project 2013	$0.00
Mentor Graphics	$0.00
Linux RedHat OS	$0.00

Hardware Expenses

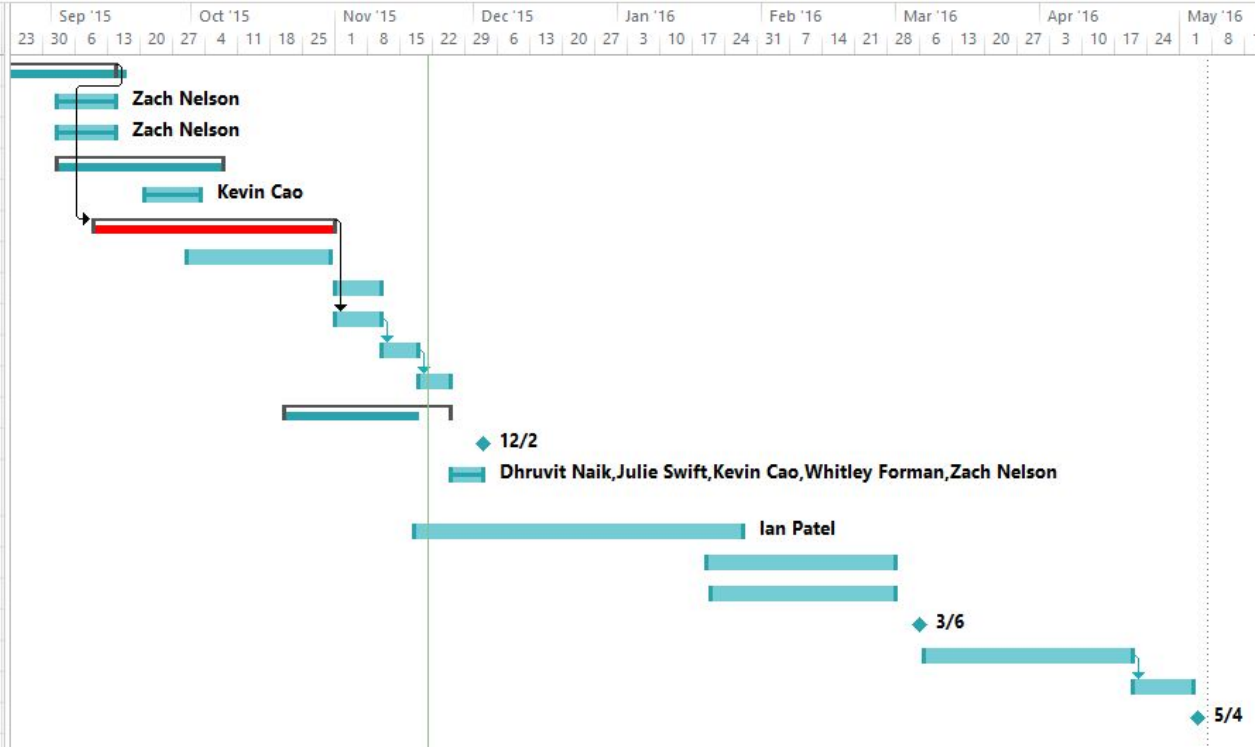
Item	Cost
Nexys 4 Artix-7 FPGA	$192.41
UDA 1380 Board	$20.00
Crystal Oscillators	$20.00
CY8CKIT-050 PSoC 5LP Development Kit	$0.00

Total Budget - Total Costs

$500.00 - $232.41 = $267.59

Schedule

		Task Name	Finish
1	✓	System Design	Mon 9/14/15
11	✓	Create a Detailed Schedule	Mon 9/14/15
12	✓	Install Microsoft Project	Mon 9/14/15
13		Install EDA Tools	Wed 10/7/15
20	✓	Create Web Page	Fri 10/2/15
21		RTL Design and Testing	Sat 10/31/15
53		Create Text Fixtures	Fri 10/30/15
54		Integration Testing	Tue 11/10/15
55		Logic Synthesis	Tue 11/10/15
56		Floor Planning	Wed 11/18/15
57		Place and Route Gates	Wed 11/25/15
58		Board Design	Wed 11/25/15
62		Have a Functional FPGA	Wed 12/2/15
63	✓	Write Final Report and Presentation	Wed 12/2/15
64		PSoC for I2S Stream	Wed 1/27/16
65		Verification	Mon 2/29/16
66		Prepare for Fabrication	Mon 2/29/16
67		MOSIS Fabrication Deadline	Sun 3/6/16
68		Additional FPGA Testing	Wed 4/20/16
69		Final Documentation	Tue 5/3/16
70		Final Presentation and Report II	Wed 5/4/16



TCNJ THE COLLEGE OF NEW JERSEY

Conclusion

- System design and documentation complete
- Challenging but also enjoyable
- Surprised at the amount of time it took to get comfortable with Verilog
  - Testing >> Writing Code
- EDA tools need to be installed
- Behind schedule
  - Utilize winter break to get back on schedule
  - Functional FPGA completed by the end of winter break
  - Rest of time can be used for fabrication submission date preparation

Any Questions?

Design and Verification of a Complete Application Specific IC

Zachary Nelson, Kevin Cao, Whitley Forman, Dhruvit Naik and Julie Swift
Advisors: Dr. Orlando Hernandez and Dr. Larry Pearlstein
December 2nd, 2015