ECE 378 Design Project: Audio Looper

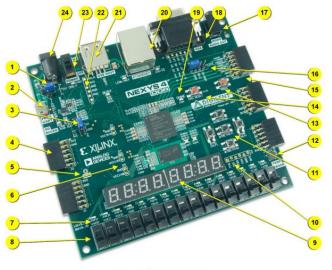
Prof. Daniel Llamocca William Blackburn, Syed Ahmed, Geoffrey Williston

Project Description

- The ideal goal was to create an audio looper and recorder
- Utilized switches, memory interface, clocks, counters, mux, combinatorial and sequential circuits, FSM, audio I/O
- Memory access

Why choose this project?

- Wanted to better understand ADC
- This project has real world applications



Project Specifics

- Half leds lit up = recording
- Full leds = playback
- Switches 12-15 used to select which track to record to
 - No track selected = no recording
- Switch 0 displays track on hex7seg display
- Switch 1 records the track
- Switch 2 plays back the selected track
 - If no tracks selected or contains no data = no output

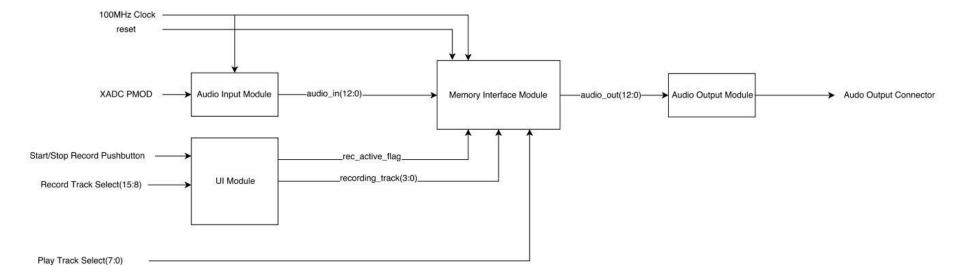
Project Specifics

- A toggle switch used to start and stop recording. If selected track contains previous recording data, it's overwritten.
- Another toggle switch will be used to cancel active recording. If selected track contains data from a previous recording, it's kept.
- Audio output jack used for audio playback

Project Plan

- Develop a project outline
- Block diagram top file
- Split VHDL into distinct modules
 - Memory
 - o UI
 - Audio In
 - Audio Out
- Flowchart the UI and memory modules
- Implement in VHDL and debug

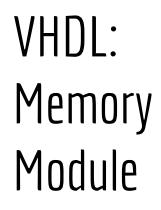
VHDL: Top File

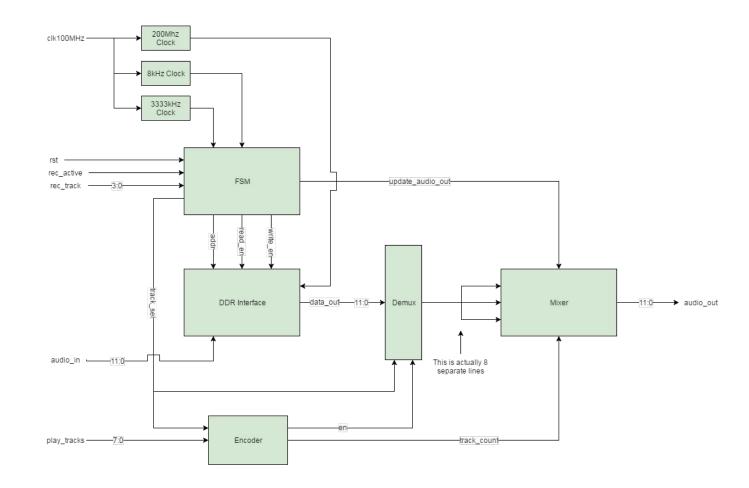


VHDL: UI Module

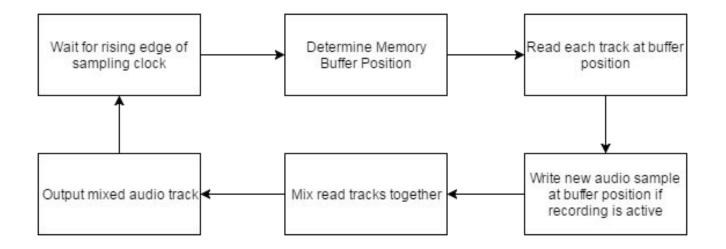
- Looked at tracks to be recorded
- If record button was pressed and if only one track was selected, then it sent a recording active flag for sampling time and 3 bits specifying the track

```
Transitions: process (resetn, clk)
        begin
                if resetn = '0' then
V <= S1;
                elsif (clock'event and clock = '1') then
                                case y is
                                         when S1 => if (isvalid = '1' and pressrec = '1') then y <= S2; else y <= S1; end if;
                                        when S2 => if (Vt <= 99999999) then y <= S3; else y <= S2; end if;
                                        when S3 => y <= S1;
                                end case:
                end if;
        end process;
Outputs: process (y, rec)
        begin
        active <= '0'; trackout <= "000"; encount <= '0';
                case y is
                        when S1 => encount <= '0';
                        when S2 =>
                                 active <= '1';
                                with rec select
                                trackout <= "111" when "10000000",
                                                         "110" when "01000000",
                                                         "101" when "00100000",
                                                         "100" when "00010000",
                                                         "011" when "00001000".
                                                         "010" when "00000100".
                                                         "001" when "00000010",
                                                         "000" when "00000001",
                                                         "000" when others;
                                encount <= '1';
                        when S3 => active <= '0';
                end case:
        end process;
end Behavioral:
```





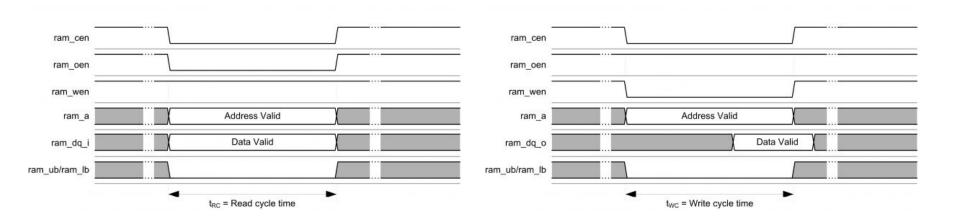
VHDL: Memory Module



VHDL: Memory Module: Buffer

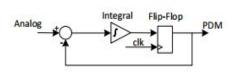
- Nexys 4 has 128 MiB DDR2 memory (2²⁴, 64-bit addresses, 16-bit accessible per address)
- 8 tracks of 16-bit audio samples, 2,097,152 samples max per track
- 128 kbps sampling rate, 8kHz decimation clock, ~4.3 minutes of audio max per track
- Total memory had to be split up into 8 tracks, 480,000 samples/track was chosen (60 seconds)
- Buffer index incremented on each 8kHz decimation clock then summed with track index multiplied by buffer size
 - Example: Writing to track3, at buffer index 5824, address = 0x160FC0 = 5824 + 480000 x 3

VHDL: Memory Module: RAM Timing



VHDL: Audio Input Module

- Convert from PDM (pulse density modulation) to PCM (pulse code modulation)
- PCM = series of stair steps
- PDM = logic high is up, logic low is down (relative description)
- Wanted to use algorithm to convert
- Send the PCM signal to memory
- Send out that PCM signal to audio out and then convert to PDM while sending the signal out to sound generation (headphones or speaker)



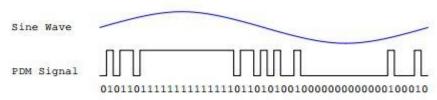


Figure 26. Simple delta-sigma modulator circuit.

Figure 25. PDM representation of a sine wave.

VHDL: Audio Output Module

- Audio output was the inverse of the audio input
- Wanted to to convert from PDM signal to PCM signal
- Then insert sound generation device to audio output jack (speakers/headphones)

VHDL: Audio Input Module

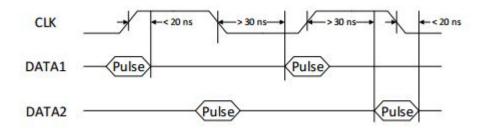
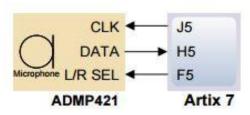


Figure 27. PDM Timing Diagram.



CLK: Clock Input to Microphone DATA: Data Output Signal L/R SEL: Left/Right Channel Select

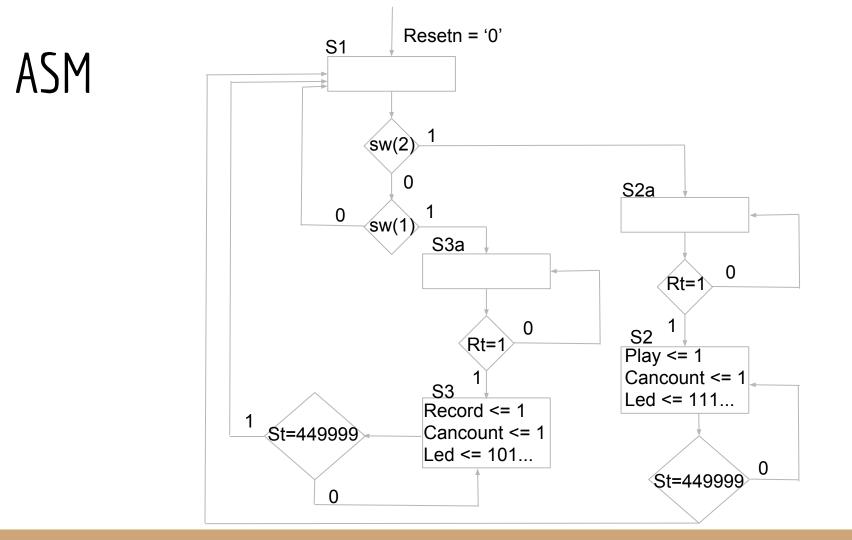
Figure 24. Microphone block diagram.

Value 0	0 ns	100 ns					
			200 ns	300 ns	400 ns	500 ns	600 ns
00000				00000)
0							
0							
0				$\infty \infty $		$\infty \infty $	
0	a de calencia de la construcción d Construcción de la construcción de l						
0							
0							
000000000000000000	000000	00000000	X	0	000000000000000000000000000000000000000		(00000000000000000000000000000000000000
0		0	X		1		(10)
00000000000000000				000000000000000000000000000000000000000			
00000000000000000				000000000000000000000000000000000000000)
0							
s1				s1			
0							
0				0			
0							
υ							
00000000000000000				00000000000000000			
1					10.1		
0							
0							
10000 ps				10000 ps)
10000 ps				10000 ps			
	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0	0	0 0	0 0	0 0	0 0

		0.000 n	S																									
		0.00		IEOO	-		1 000 pc		11 500 -		17	000 55		2 500 pg		12 000	-	12	E00 pc		14 000 -			1 E00 pc		IE 000 pg		IE 500 -
Name	Value	0 ns		500		1 1	1,000 ns		1,500 r			,000 ns		2,500 ns		3,000			,500 ns		4,000 n		-	4,500 ns		5,000 ns		5,500 r
U clk	0																											
🕨 式 sw[4:0]	00000									00000														10000				
🎼 cpu_resetn	0																											
🎧 m_data	0																		1									
Ug qt	0																											
🕼 canread	0																											
🕼 clk_mictime	0																											
🔓 m_sig0	0							-																				
🕨 式 r0sig[15:0]	000000000000000000000000000000000000000	0000	000000		.)(00000	000000.	.)00000	000000)	000000	00000>	0000	0000000	000000	00001)	00000000	010)	0000000	0101	. 0000	0001010	000000	010101	. 000	00101010	0000	010101	000101	01010
Vie rt	0	0	X		X	10	X	11	10)0)		101	X	10	111		100	0	X	1001	1	010	X	1011	X	100	11	.01
🕨 📑 rasig[15:0]	000000000000000000													0000	000000000	000												
🕨 📑 r1sig[15:0]	00000000000000000													0000	000000000	000												
🕼 m_sig1	0																											
Ца у	s1									s1									X					s2a				
Up cancount	0																											
1 st	0														0													
1 ston	0																											
🗓 aud_pwm	υ																											
Ied[15:0]	000000000000000000000000000000000000000										=			0000	000000000	000					-							
aud_sd	1												2	-														
Te m_cik	0										-											1						
🔓 m_irsei	0																											
Clk_period	10000 ps	_													10000 ps													
m_clk_period	10000 ps	\geq													10000 ps													
is m_ck_pendu	10000 ps														10000 pS													

Name Value	5,500 ns	6,000 ns	6,500 ns	7,000 ns	7,500 ns	8,000 ns	8,500 ns	9,000 ns	9,500 ns
Le cik 0									
▶ 🛃 sw[4:0] 10000					10000				
🔓 cpu_resetn 🛛 1									
🔓 m_data 🛛 0									
1101 1101									
Canread 0									
Clk_mictime 1									
🕼 m_sig0 0									
▶ 🎼 r0sig[15:0] 1010101	011101010 0001010101 00101010	0101010101010	(10101010101)0	1010101010	01010101	1010	1)(01010101010	/10101010101/0	1010101011
Ue rt 1000	110 / 1110	X 1111		1 χ	10 / 1	100	101	χ 110 χ	111 X1
	010101010 0000	00000000000	X			101010101	0101010	řem na klasti k stalova stalo	
▶ 🎼 r1sig[15:0] 0000000	00000000			0000	000000000000				
🔓 m_sig1 0									
Üα y s2		s2a	Х			s2			
🕼 cancount 1									
🗓 st 🛛 0					0				
Un ston 0									
🔓 aud_pwm 🛛 o									
▶ 🍓 led[15:0] 1111111	11111111 0000	00000000000	X			111111111	1111111		
🔓 aud_sd 1									
🔓 m_clk 1									
🕼 m_lrsel 🛛 0									
Clk_period 10000 p	s				10000 ps				
🔓 m_clk_period 10000 p	s				10000 ps				

Name	Value	5 us		6	us		7 us	a na		8 us		9 us	i na sa	10 us		ال ب	L1 us		12 us		13	us		14 us	
🗓 clk	1																								
🕨 式 sw[4:0]	10000												100	00											
🖓 cpu_resetn	1																								
🖓 m_data	Ó																1					i i			
Ug qt	1																								
Canlisten	Ó															:				1		***			
👍 clk_mictime	1																								
🗓 m_sig0	0																						1		-
▶ 📑 r0sig[15:0]	1010101010101010	0000	0001)	00101	0101)	(10101)	01010	1010	0101	0 \1010	1	. 10101.	.)(01010)	1010	01010	1010	. 01010	10101	0101	10111.	.)(01110.		11010.	. 1010	0101
∐a rt	0	1100	1101	1110	1111	0	1	10	χ 1	1 10	0 101	110	111	1000	1001	1010	1011	1100	1101	1110	X 1111	X 0	X 1	10	11
🕨 式 rasig[15:0]	00000000000000000		00000	00000000000	0000	$ \rightarrow $								1010101	010101010)							χ 111	01010101	01010
🕨 式 r1sig[15:0]	000000000000000000000000000000000000000										000	0000000	00000										10101.	. 0101	1010
Te m_sig1	0																								
Ца у	s2a			s2a												s2									
1 cancount	0																								
1 st	0									0								X				1			
16 ston	0																								
🗓 aud_pwm	0																								
🕨 式 led[15:0]	00000000000000000		00000	0000000	0000										1111	111111	11111								
🗓 aud_sd	1													_			-	-							
Un m_clk	1																								
Un m_Irsel	0																								
👍 clk_period	10000 ps												1000	0 ps											
🕼 m_clk_period	10000 ps												1000	0 ps											



Problems Faced

- Onboard vs external mic
- Onboard mic implementation
- Audio input/output and memory clock synchronization and resulting noise
- Mixing multiple tracks
- Thought PCM ⇔ PDM was easy
- Generating testbench for VHDL relying on physical HW

What we would have done with more time

- Reduce noise resulting from high mic sample frequency versus available memory
- Decide on external vs internal mic sooner in design process
- Research how to layer more effectively