Laboratory 4

(Due date: **005**: March 20th, **006**: March 21st)

Objectives

- ✓ Implement a Digital System: Control Unit and Datapath.
- ✓ Learn about reading PDM-coded input audio and playback PDM-coded output audio.
- ✓ Learn interfacing with MEMS microphones (that generate PDM signals) and using BlockRAMs in FPGAs.

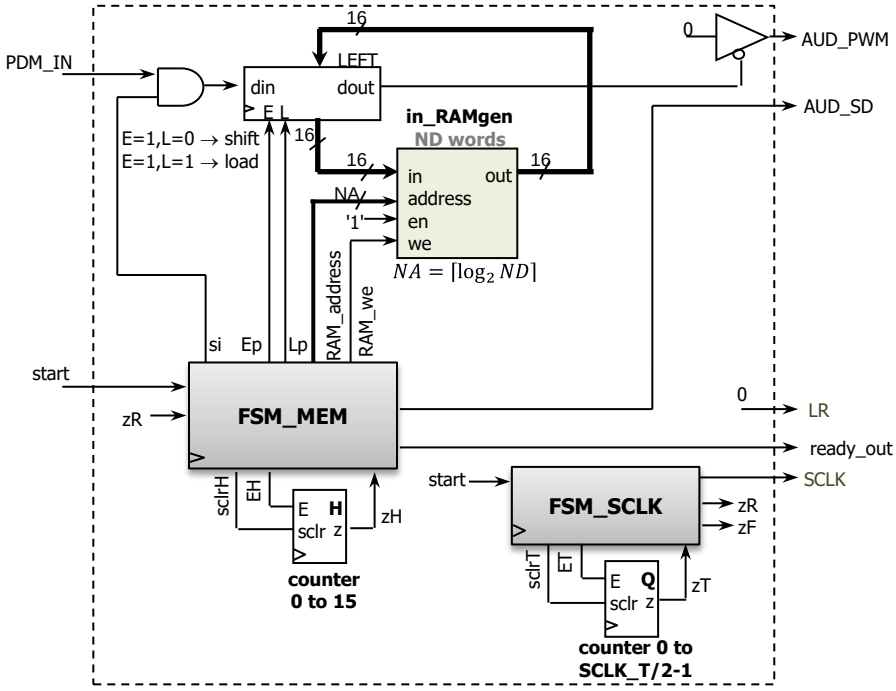
VHDL Coding

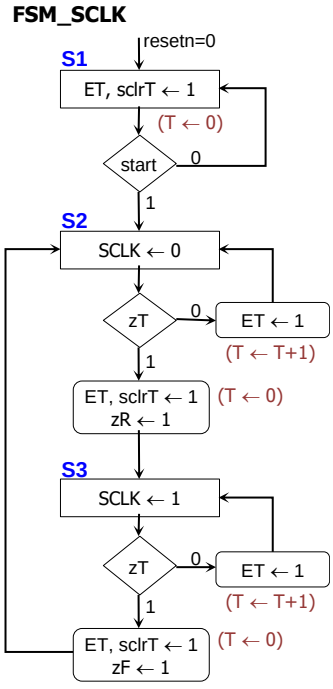
- ✓ Refer to the Tutorial: VHDL for FPGAs for a tutorial and a list of examples.

First Activity: Audio Retrieval and Playback: Design and Simulation (60/100)

Design Problem

- **MICROPHONE AND MONO AUDIO OUTPUT**: Implement the following circuit that reads data from the ADMP421 MEMS microphone, stores data in memory and plays data back on a mono audio output.





- **MEMS MICROPHONE (ADMP421):**
  - ✓ SCLK: 1 – 3 MHz. We use 1.2 MHz.
  - ✓ PDM_IN: PDM signal generated by the ADMP421.
  - ✓ LR: Left right control for stereo mode. We use LR = 0 (Data is captured on CLK rising edge).

- **MONO AUDIO OUTPUT:**
  - ✓ The on-board audio jack of the board (Nexys-A7-50T/A7-100T, Nexys 4-DDR) does not support stereo output. Thus, we only retrieve a mono audio input from the ADMP42 microphone (e.g. L/R = 0). Keep this in mind when testing (as only one speaker or earphone side will work).
  - ✓ The on-board audio jack is driven by a low-pass filter with 12 KHz cut-off frequency. We need to feed a PDM signal (AUD_PWM: open-drain output) into the low-pass filter. Note that AUD_PWM is the output of a tri-state buffer:
    - If we want to transmit a '0', AUD_PWM = '0'.
    - If we want to transmit a '1', AUD_PWM = 'Z'.

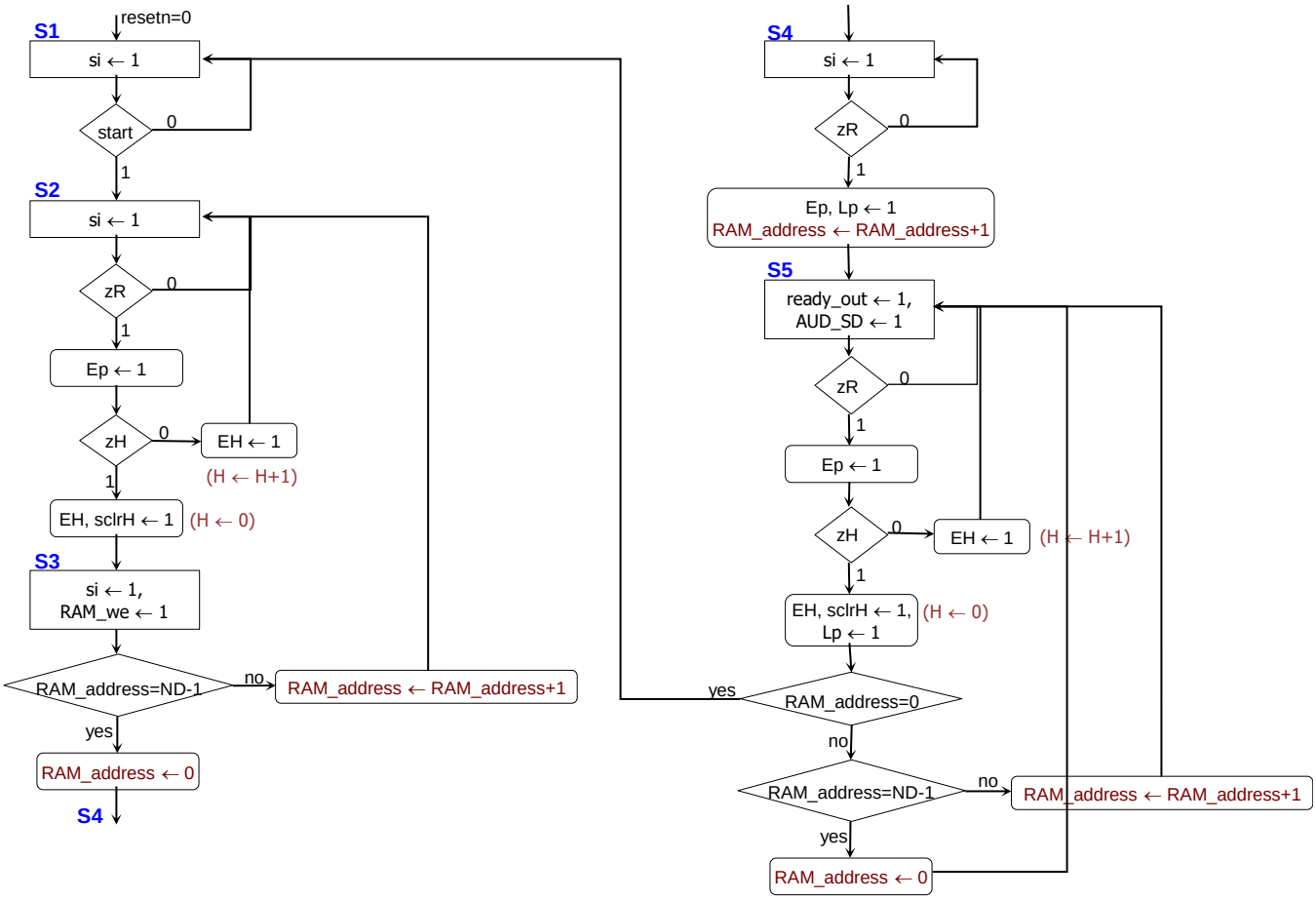
- **FSM_SCLK:**
  - ✓ This circuit generates a free running clock (SCLK). It also generates a pulse on zR when a rising edge on SCLK occurs, and a pulse on zF when a falling edge on SCLK occurs. We want to use a frequency of 1.2 MHz. This requires SCLK_T=84.

Instructor: Daniel Llamocca
TAs: Alex Fillmore, Luke Nuculaj

- To set up `SCLK_T=84`, you need to use the VHDL code `fsm_sclk.vhd` (along with `my_genpulse_sclr.vhd`) and set up its parameter `COUNT_SCKLHP = SCLK_T/2=42`.

- **MEMORY:** Implemented with the on-chip memory (BlockRAMs) inside the Artix-7 FPGA.
  - ✓ Data requested or to be written is available on the next clock cycle. Use the given VHDL code `in_RAMgen.vhd`. This is a 2D memory of `nrows×ncols`. Word length: 16 bits. We can use it as a 1D memory by making `ncols=1`. Use the following parameters `nrows=`(this depends on the board), `ncols=1, INIT_VALUES="NO", FILE_IMG ="myinival.txt"`.
    - For Nexys A7-100T/Nexys-4 DDR, use: `nrows=512*512`
    - For Nexys A7-50T, use `nrows=512*256` (there is less memory in the FPGA inside this board)

- **FSM_MEM:** This is the main controller. This FSM embeds the counter `RAM_address`. Make sure to use the correct ND:
  - ✓ $ND = 512 \times 512 = 2^{18}$ for Nexys A7-100T/Nexys-4 DDR. Sequence duration: $2^{18} \times 16 \times 84 \times 10ns = 3.52$ seconds.
  - ✓ $ND = 512 \times 256 = 2^{17}$ for Nexys A7-50T.



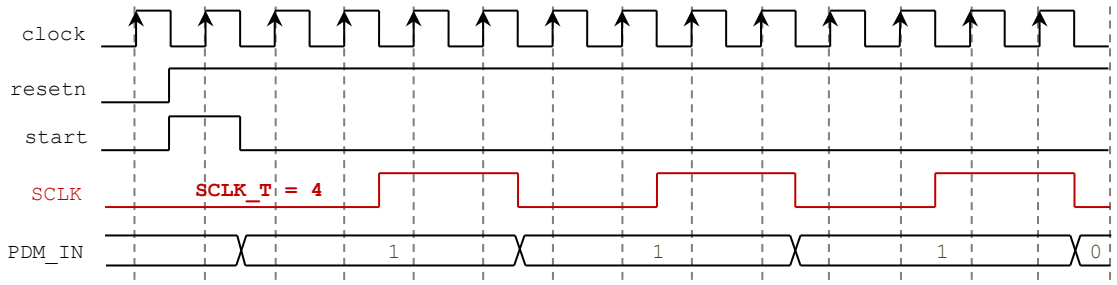
PROCEDURE

- **Vivado: Complete the following steps:**
  - ✓ Create a new Vivado Project. Select the corresponding Artix-7 FPGA device (e.g.: the XC7A100T-1CSG324 FPGA device for the Nexys A7-100T board).

  - ✓ Write the VHDL for the given circuit. Synthesize your circuit. (Run Synthesis).
    - Use the Structural description: Create (or re-use) a separate `.vhd` file for i) parallel access shift register, ii) inRAM_gen, iii) FSM_MEM, iv) FSM_SCLK, and v) top file (where you will interconnect them all).
      - Use the proper parameters and I/O connections. Note that FSM_SCLK is made from two `.vhd` files.

  - ✓ Write the VHDL testbench (generate a 100 MHz input clock for your simulations).
    - To avoid long simulation times, only for simulation purposes use these parameters in the following blocks:
      - Memory: `nrows=4x4, ncols=1`.
      - FSM_SCLK: `COUNT_SCLKHP = SCLK_T/2 = 2`.

- PDM_IN: Generate the following serial input stream (4x4 16-bit words), where each bit is to be captured at the rising edge of SCLK: 1110 1100 1110 1011 0100 0111 0001 0000 1010 1100 1110 1101 1110 1010 0101 0001 and then just 1's. These bits are stored in memory (inRAM_gen) as 16-bit values: `ECEB 4710 ACED EA51 FFFF … FFFF`.
  - The timing diagram depicts how to set up the input signals.



✓ Perform Behavioral Simulation (Run Simulation → Run Behavioral Simulation). **Demonstrate this to your TA.**
- To help debug your circuit, add internal signals to the waveform (e.g.: FSM_MEM/state, RAM_address, SCLK, inRAM_gen/in, inRAM_gen/out, etc).
- Run the simulation for 21 us (remember to use `nrows=4x4`, `ncols=1`, `COUNT_SCLKHP = SCLK_T/2 = 2`).
  - Verify that the hexadecimal values (see testbench directions) appear on inRAM_gen/out (memory output) during state S5 in FSM_MEM.
  - Verify that the serial stream (see testbench directions) appears on dout (parallel access left shift register output)

SECOND ACTIVITY: TESTING (40/100)

- **Vivado: complete the following steps:**

✓ I/O Assignment: Create the XDC file associated with your board.
- Suggestion (Nexys A7-50T/A7-100T, Nexys 4/DDR):

Board pin names	CLK100MHZ	CPU_RESET	BTNC	M_DATA	M_CLK	M_LRSEL	AUD_PWM	AUD_SD	LED0
Signal names in code	*clock*	*resetn*	start	PDM_IN	SCLK	LR	AUD_PWM	AUD_SD	ready_out

✓ Implement your design (Run Implementation).
- Ensure that the correct parameters values are set up in the Memory (`nrows`, `ncols`) and in FSM_CLK (`COUNT_SCLKHP`). Do not use the same parameter values you used for simulation.

✓ Generate and download the bitstream on the FPGA. Test the circuit. **Demonstrate this to your TA**.
- Press start and record an audio sequence for 3.52 seconds (this is for the Nexys A7-100T/Nexys-4 DDR). After that, the audio sequence is played back (use a headphone/speaker).

SUBMISSION

- Submit to Moodle (an assignment will be created):
  - ✓ This lab sheet (as a .pdf) completed (if applicable) and signed off by the TA (or instructor).
    - Note*: The lab assignment has two activities. You get full points of the 1st activity if you demo it by the due date. You can demo the 2nd activity by the due date or late (here, we apply a penalty towards the points of the 2nd activity).*
  - ✓ (As a .zip file) All the generated files: VHDL code, VHDL testbench, and XDC file. **DO NOT submit the whole Vivado Project.**
    - Your .zip file should only include one folder. Do not include subdirectories.
    - It is strongly recommended that all your design files, testbench, and constraints file be located in a single directory. This will allow for a smooth experience with Vivado.
    - You should only submit your source files AFTER you have demoed your work. Submission of work files without demoing will be assigned NO CREDIT.

TA signature: _____ **Date:** _____