Stanford University
Department of Electrical Engineering

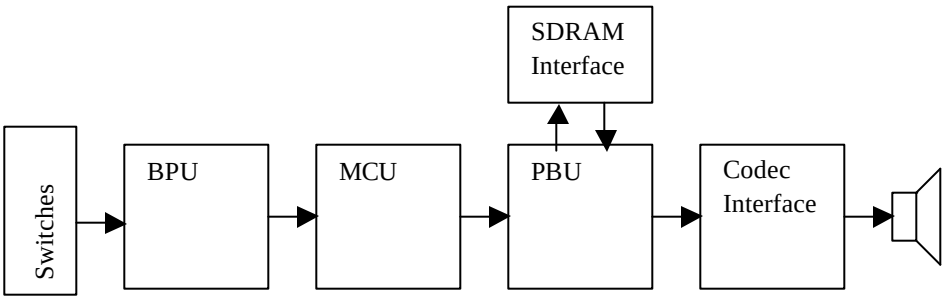
EE 121
Winter 2001-2002

Lab 6
Digital Four-Track Sound Player

Prelab Due Date: Tuesday, February 19, 2002 at 9:30am

1 Overview

Design and implement a Digital Four-track SDRAM player with the XESS Xilinx Spartan-II board in the lab. A suggested decomposition is shown below.



Button Processor Unit (BPU)

The BPU accepts inputs from four switches representing play forward, play reverse, stop and track. These four switches can produce six different events; these events and the implications are:

- Stop pressed: Stop playing and reset to the beginning of track zero.
- Track pressed: Jump to the beginning of the next track and play.
- Play (fwd or rev) held < ~0.5s: Play in the specified direction at normal speed.
- Play (fwd or rev) held > ~0.5s: Play in the specified direction at twice normal speed.

A change in speed should continue playing from the same point that the button press is received, i.e. no skipping/jumping within the track.

Master Control Unit (MCU)

The MCU interprets the output of the BPU and maintains state information about the Digital Four-Track. In the MCU, the state information is decoded and passed on to the playback unit. The state of the MCU represents the current mode of the Digital four-

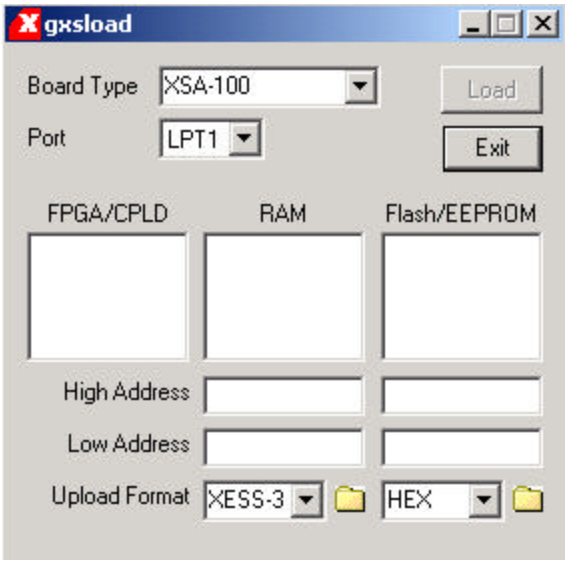
track (e.g stopped, playing reverse double-speed, etc.). You should resolve the possibility of multiple simultaneous events by assigning a priority to each event.

Playback Unit (PBU)

The PBU accepts state information and generates addresses for the SDRAM. The two highest addresses of the SDRAM are track numbers, and the low sixteen addresses are sample numbers. The sample rate is approximately 8kHz. When the end of a track is reached, the track repeats.

SDRAM

The Digital Four-track player plays back sampled sounds stored in the XESS board SDRAM by cycling through the SDRAM addresses. You will be provided with files containing sound data. The sound files can be downloaded to the SDRAM by dragging the sound file to the RAM area of XSLOAD when downloading the FPGA design.



The SDRAM acts as a lookup table. You provide it a 23-bit address and raise the "read" signal (note that we don't use all 23 bits of addressing—set the others to zero). The data is available to read when the "done" signal is high. This may not happen for 10's of 25MHz clock cycles. Your design needs to handle the variability in when data is available for reading. The delay has to do with the refreshing of dynamic RAM and will be discussed later in the term.

As with Lab 5, a starter SDRAM project will be provided with more info.

CODEC

To produce the sound, the digital data stored in the SDRAM must be converted to an analog signal. The Coder/Decoder (CODEC) on the XESS board will serve this function.

See the course web-page for a sample codec application and documentation. You should take the codec schematic and use it in your design to output sound to the speakers.

Laboratory Requirements

NOTE: This lab will take SUBSTANTIALLY more time than the previous labs, *SO START EARLY!!*

For this laboratory assignment, partners should work together.

There is one core state machine that controls the game state. For this state machine, use the synchronous sequential state machine design process discussed in class. You can use either binary or one-hot encoding.

A top-level schematic will be provided on the course web page that includes signal names and pin assignments. Please make sure that all signals defined in this file are still in your design.

NOTE: For all your logic blocks, you can use any of the elements in the Foundation library (including CoreGen) but ***you must not gate the clock***!!! This means that all state machines must be fully synchronous.

What to turn in for this week's Prelab:

1. Complete Foundation Project with schematics of all logic blocks in your design (submitted electronically)
2. Test script demonstrating that your design works (submitted electronically)
3. A report file that specifies resources used in the FPGA for the design—ie, the number of CLBs/IOBs/etc used. Look under the "reports" tab in Xilinx foundation. Summarize and discuss the results in your README in one paragraph (submitted electronically)
4. Extract timing information for the design as demonstrated in lecture and submit the file. The "Analyze against Auto Generated Design constraints…" button in the Xilinx Timing Analyzer tool will provide the values needed. Summarize and discuss the results in your README in one paragraph (submitted electronically)
5. Look at your design in the Xilinx FPGA Editor—be impressed it did all that wiring and not you! ☺ (JPEG screen dump submitted electronically)
6. Readme file (submitted electronically).

Lab Exercise

During lab time, you will get a chance to compile your design, download it onto your Xilinx chip, and test it with real signals.

Submission

Submit this lab using the same instructions as for all previous labs.

Remember, you must submit your project electronically, whether or not it works correctly, before 9:30 AM on the due date or you will receive no credit.