ECE532: Digital Hardware

Karaoke Machine

Jen Pollock

Frances Lau

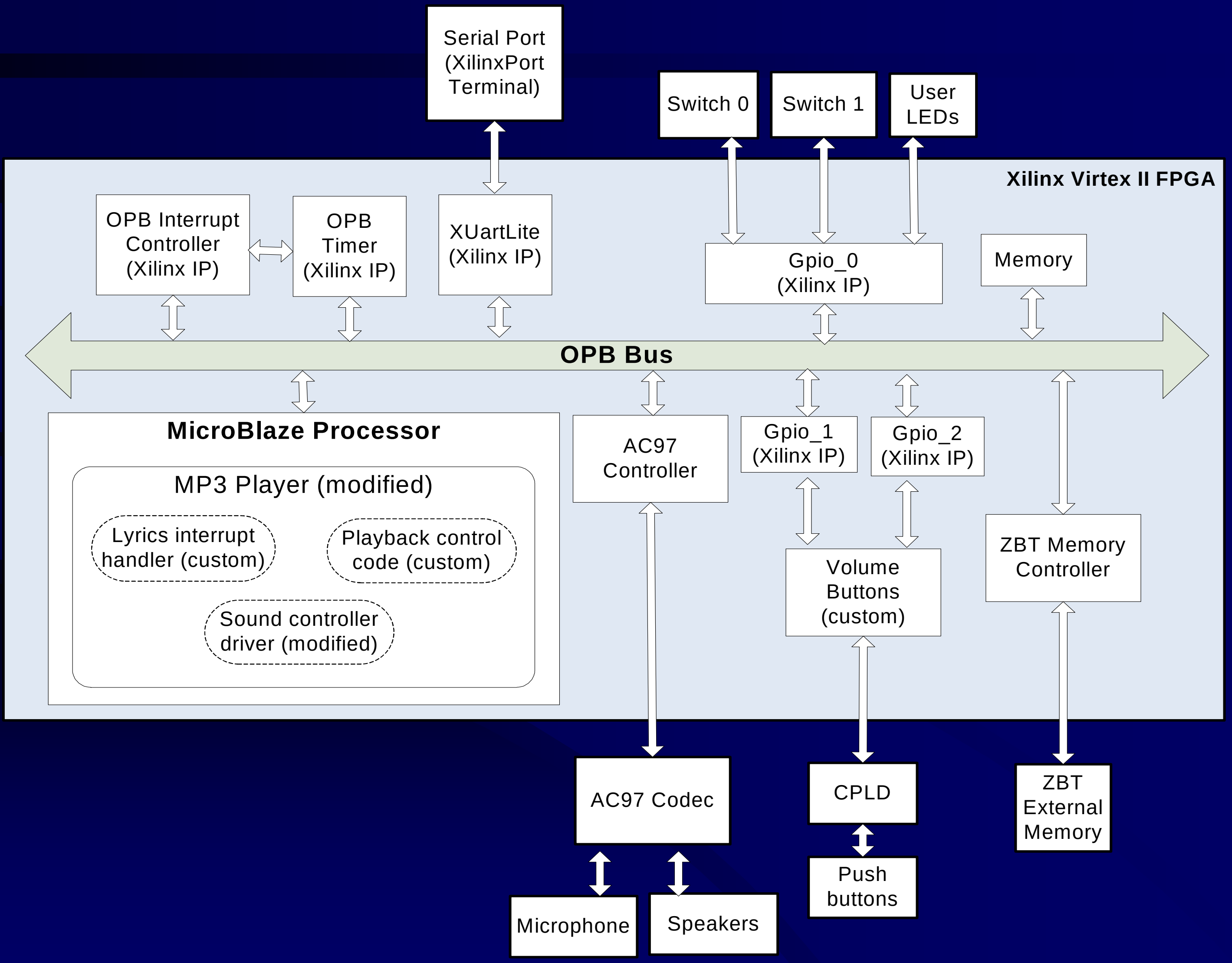
Initial Project Objective

- To create a karaoke machine that will:
  - Play a pre-recorded sound file mixed with microphone input
  - Display lyrics that are synchronized with the music
  - Have buttons that control the playback of the song and the volume of the song and the singer
  - Read pre-recorded audio and lyrics files from compact flash

Final Project Results

- Created a karaoke machine that:
  - Plays a pre-recorded sound file mixed with microphone input
  - Displays lyrics that are synchronized with the music
  - Has switches that control the playback of the song (pause and restart)
  - Has buttons that control the volume
  - Reads pre-recorded audio and lyrics files from memory (downloaded using XMD)

System



Serial Port (XilinxPort Terminal)
Switch 0
Switch 1
User LEDs
Xilinx Virtex II FPGA
OPB Interrupt Controller (Xilinx IP)
OPB Timer (Xilinx IP)
XUartLite (Xilinx IP)
Gpio_0 (Xilinx IP)
Memory
OPB Bus
MicroBlaze Processor
MP3 Player (modified)
Lyrics interrupt handler (custom)
Playback control code (custom)
Sound controller driver (modified)
AC97 Controller
Gpio_1 (Xilinx IP)
Gpio_2 (Xilinx IP)
Volume Buttons (custom)
ZBT Memory Controller
AC97 Codec
CPLD
ZBT External Memory
Push buttons
Microphone
Speakers

System Components

- Existing components:
  - Basic structure: Microblaze processor, OPB bus
  - AC97 codec and controller
  - Gpio
  - Interrupt controller
  - Timer
  - UARTlite

System Components

- Modified components:
  - MP3 player
    - Added code to display lyrics
    - Added code to read status of push buttons and control volume
    - Added code to pause and restart the song
  - Sound controller driver
    - Changed initialization function
      - Unmute microphone
      - Reduce PCM volume (song)
- Custom components:
  - Volume buttons hardware block

Challenges

- CPLD was unable to support desired push button function
  - Used push button Normal mode to set volume
  - Used switches instead of push buttons to control playback
- User Input Switch 0 used for reset, and we needed two switches
  - Connected the CPLD reset signal to the system reset

Challenges

- Poor documentation and lack of availability of formatting tools for compact flash
  - Compact flash goal was dropped

Challenges

- The MP3 Player was not available initially
  - For Milestone 1, we created our own system, building on top of the AC97 Controller project from last year.
  - This system does the following:
    - **Records sound:**
      - Records sound from the microphone and saves this into memory (custom C program)
      - Saves the data from the memory into a data file (TCL script)
    - **Plays recorded sound:**
      - Downloads the data file into memory (XMD)
      - Plays the sound (custom C program)
  - Later, the MP3 Player was available, so we decided to use that instead

Design Process

- Built the system incrementally, starting with working components
- Tested each feature separately before integration
- Set well defined boundaries between modules
- Worked within the structure of the available mp3 player

What we learned

- How to resolve problems due to interactions between different IP blocks
  - Clock frequencies
- The importance of simulations
  - Found errors in our Volume Buttons custom hardware module
- The importance of optimizing code for slower processors