Shoot Bubble Video Game

Team Name: Shoot Bubble Video Game

Team Members: Jun Dai jd2968

Shuo-Shu Tsai st2786

Weichia Chen wgc2111

Michael Shone yh2567

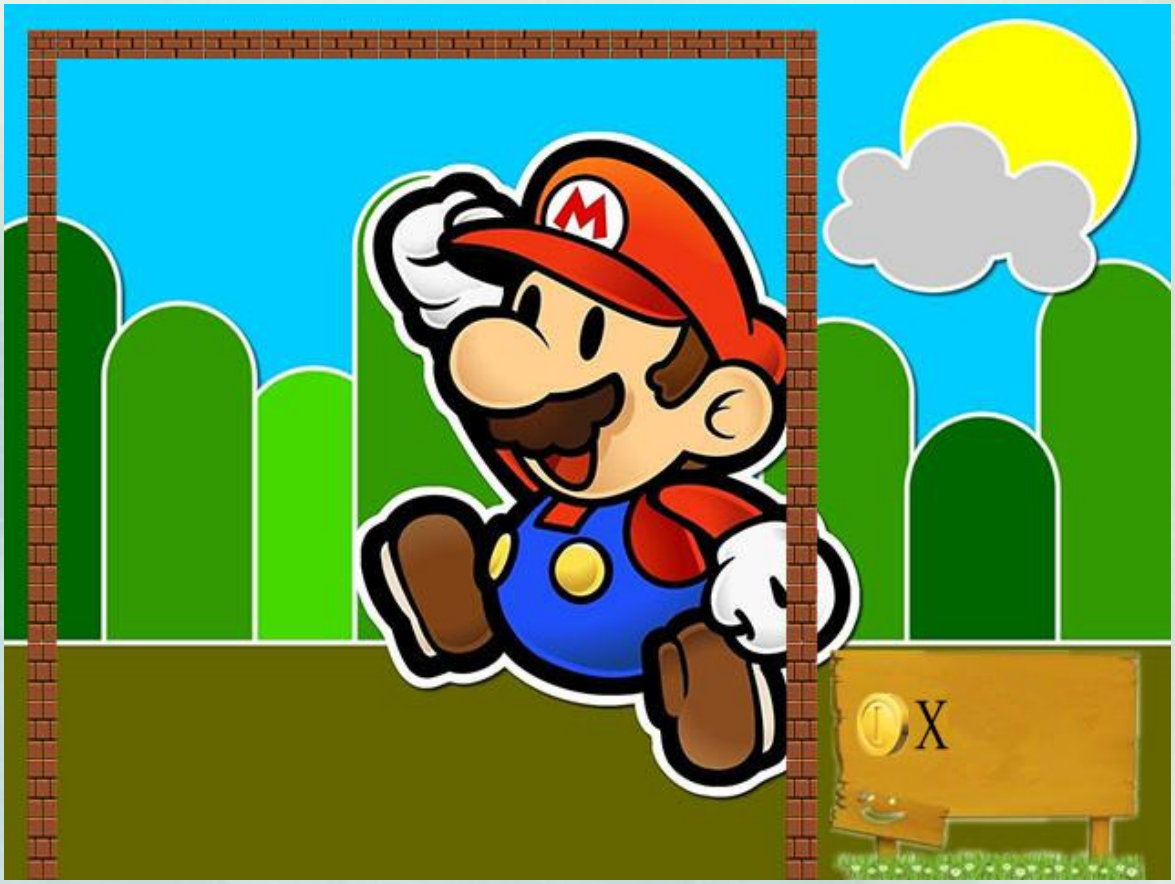
Professor： Stephen A. Edwards

Content

- 1. Overview & Objectives
- 2. Architecture & Timing Design
- 3. Experience and Issues in Implementation
- 4. Summary & Lessons Learned

Overview & Objectives

- Implementing an “Shoot Bubble” game that communicated between a computer and an Altera DE2 board



Overview & Objectives

- The game split in two mode: one player mode or two player mode

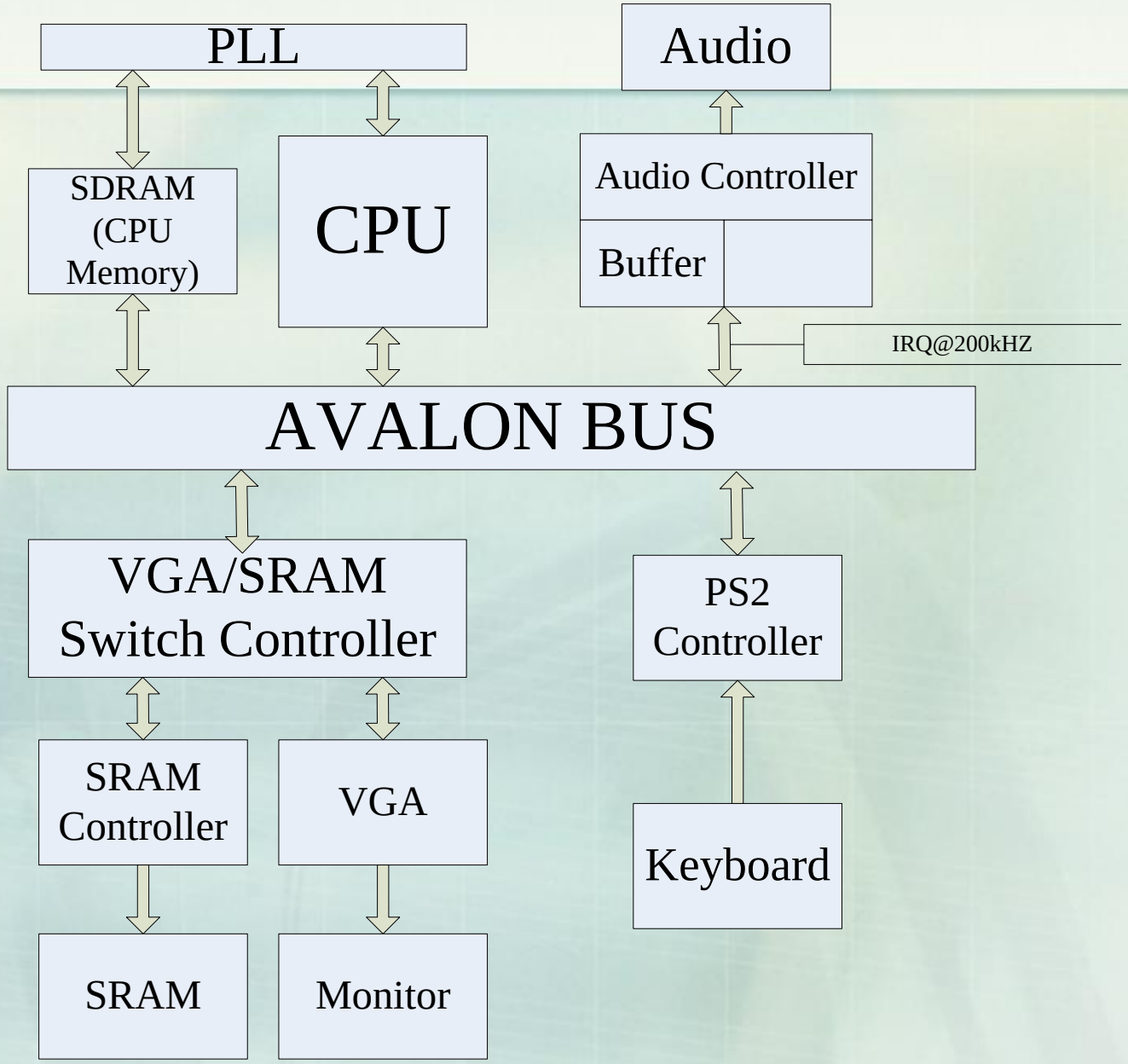
Overview & Objectives

- Start: Bubbles on the top of the screen will be downward
- End: Bubbles touch the ground, game over
- Game Play: Using Keyboard to blast 3 or more consecutive bubbles with the same color lined up. Meanwhile, bubbles floating beside will also be blasted
- The more you pop, the higher your score is. High score will be recorded

Architecture & Timing Design

- Major components: SRAM, SDRAM, JTAG/UART, PS2 keyboard, VGA, and Audio

Architecture & Timing Design

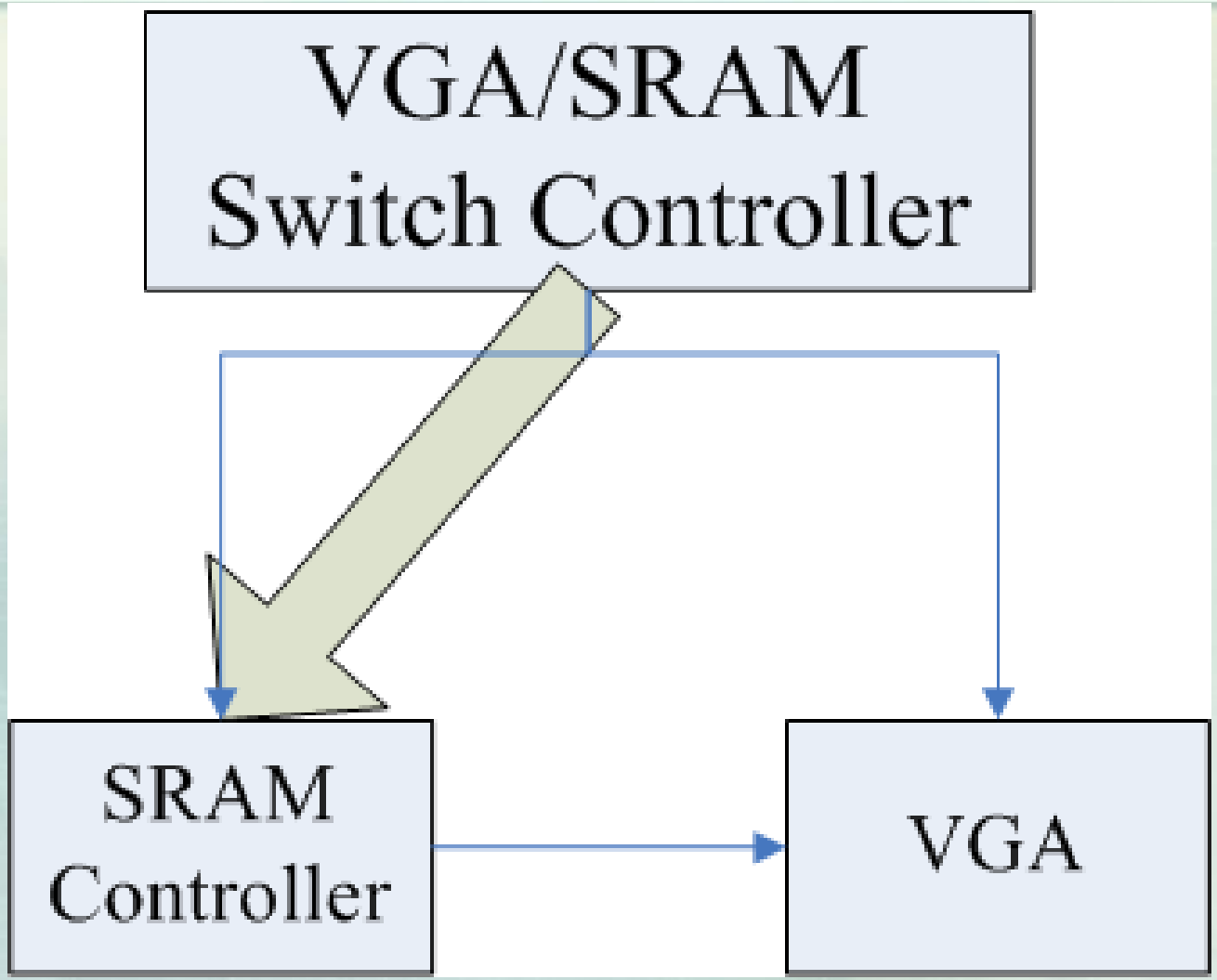


Architecture & Timing Design
VGA

- We have a VGA/SRAM switch controller, which can decide the mode:
- Mode 1: Write the data to SRAM
- Mode 2: VGA read the data from SRAM

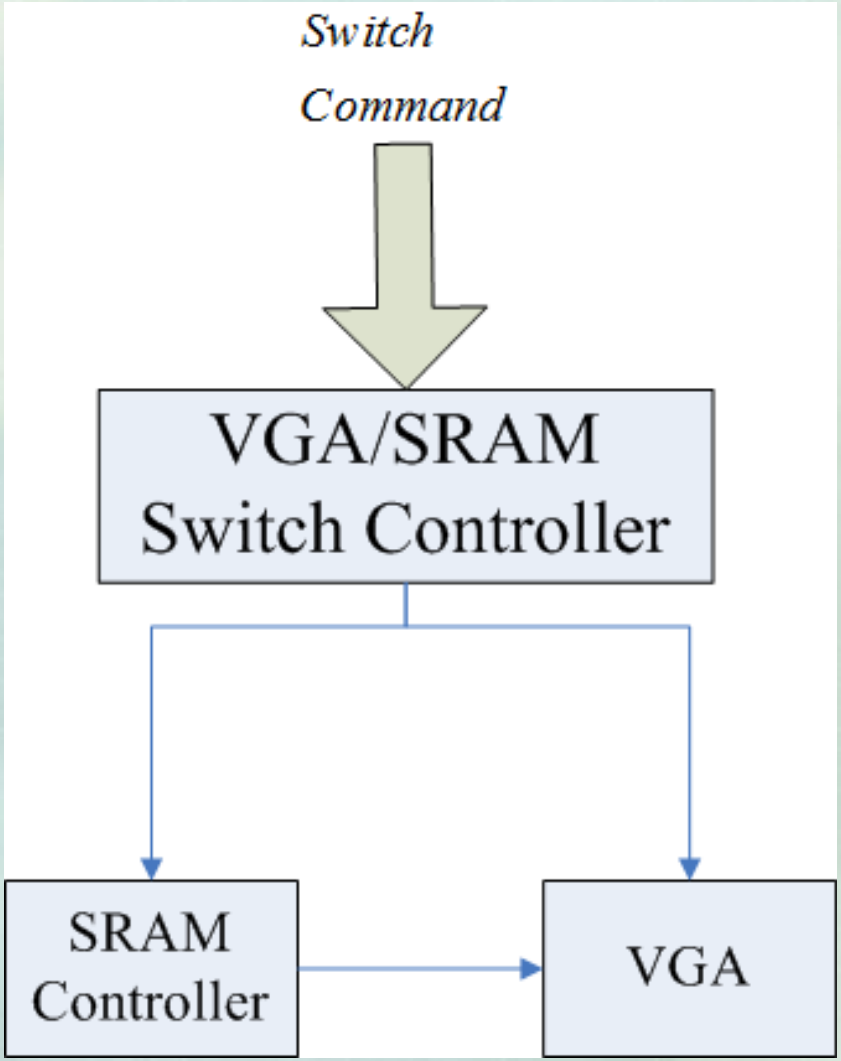
Architecture & Timing Design

VGA



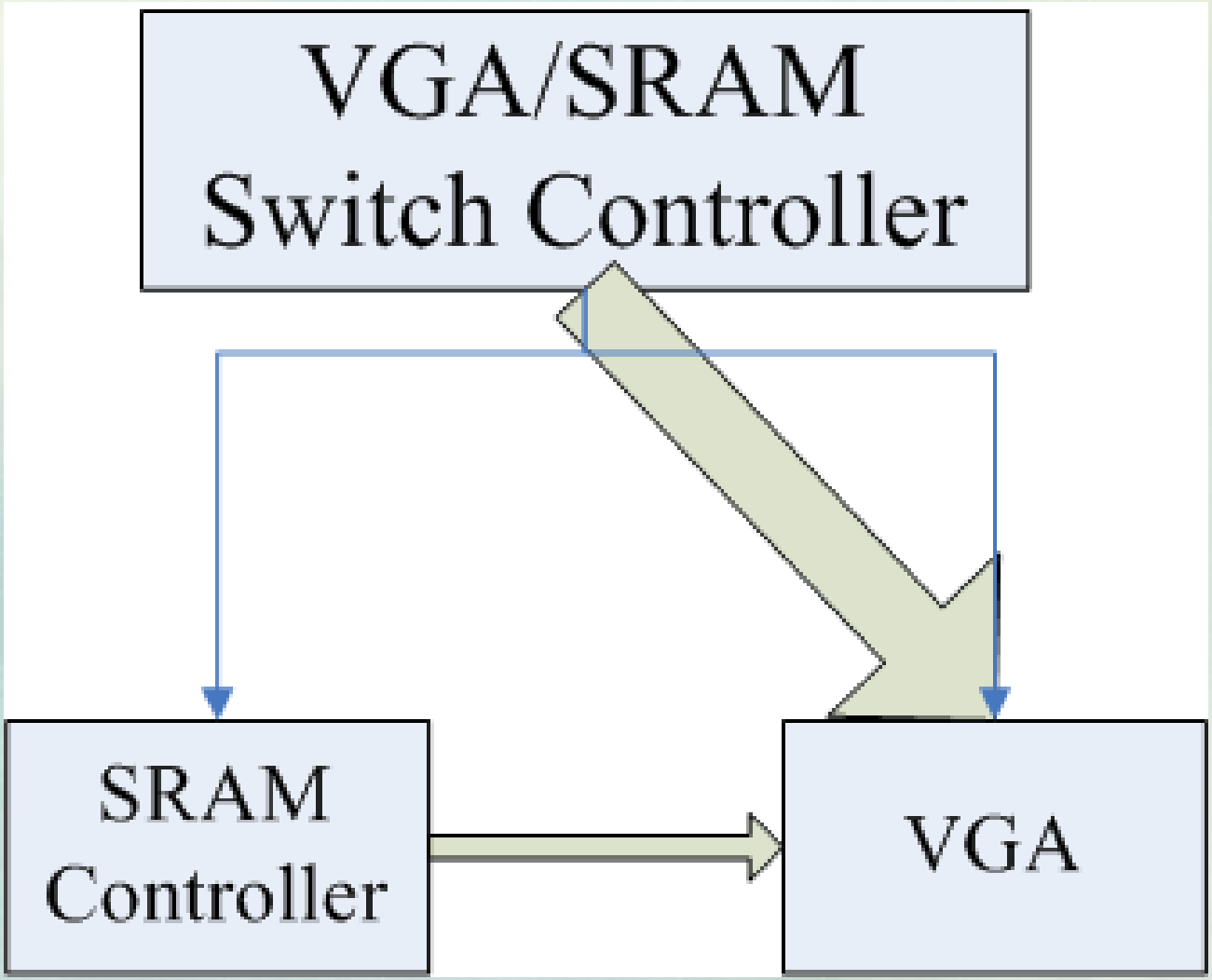
Architecture & Timing Design

VGA



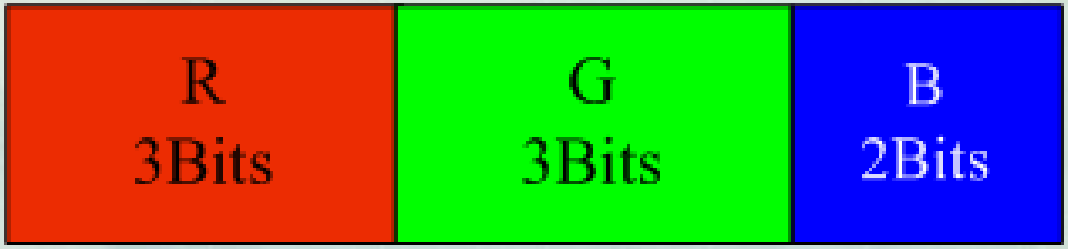
Architecture & Timing Design

VGA



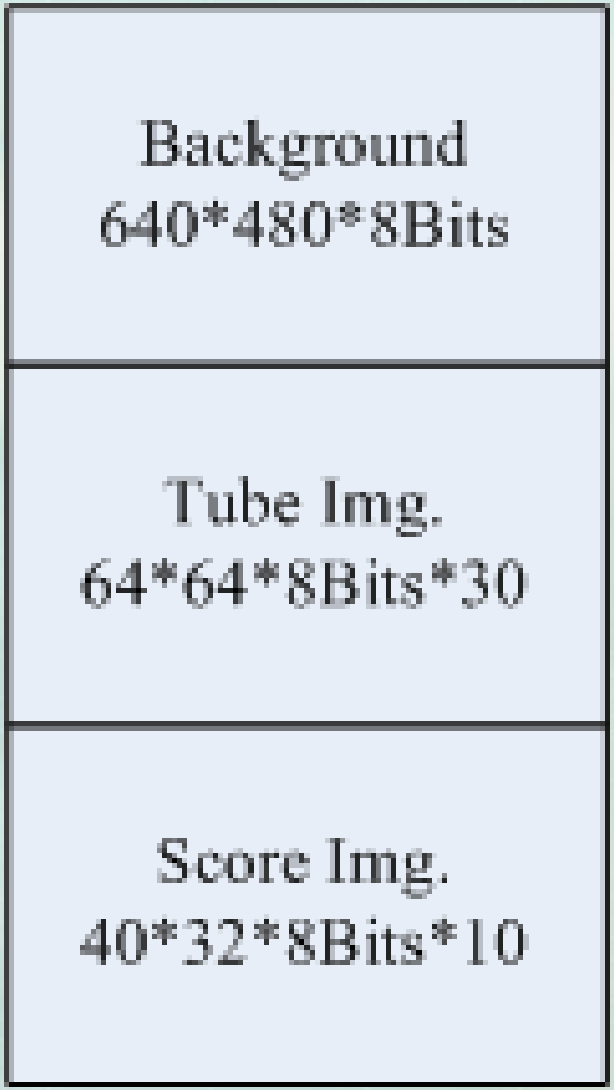
Architecture & Timing Design

VGA

- We use the MATLAB to read the picture to get RGB values.
- RGB format :
- 

Architecture & Timing Design

SRAM



Architecture & Timing Design

SDRAM

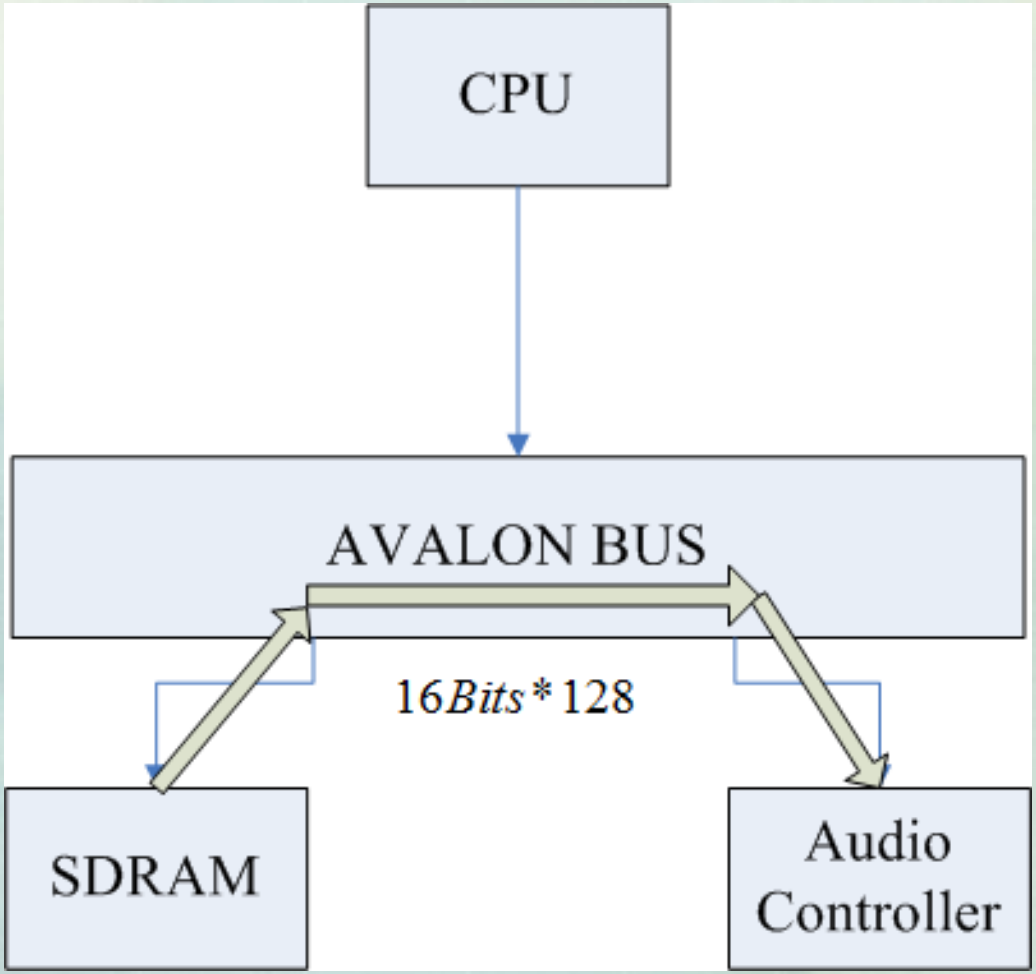
- We use SDRAM as the memory of CPU. We use the pll to generate a clock which is 3ns slower than the original 50MHZ clock.

Architecture & Timing Design
Audio

- We have a buffer in the audio controller which can store 128*16bits data
- Whenever finished playing the data in the buffer, the music controller will send the interrupt signal to the CPU and it will fill up the buffer with new data

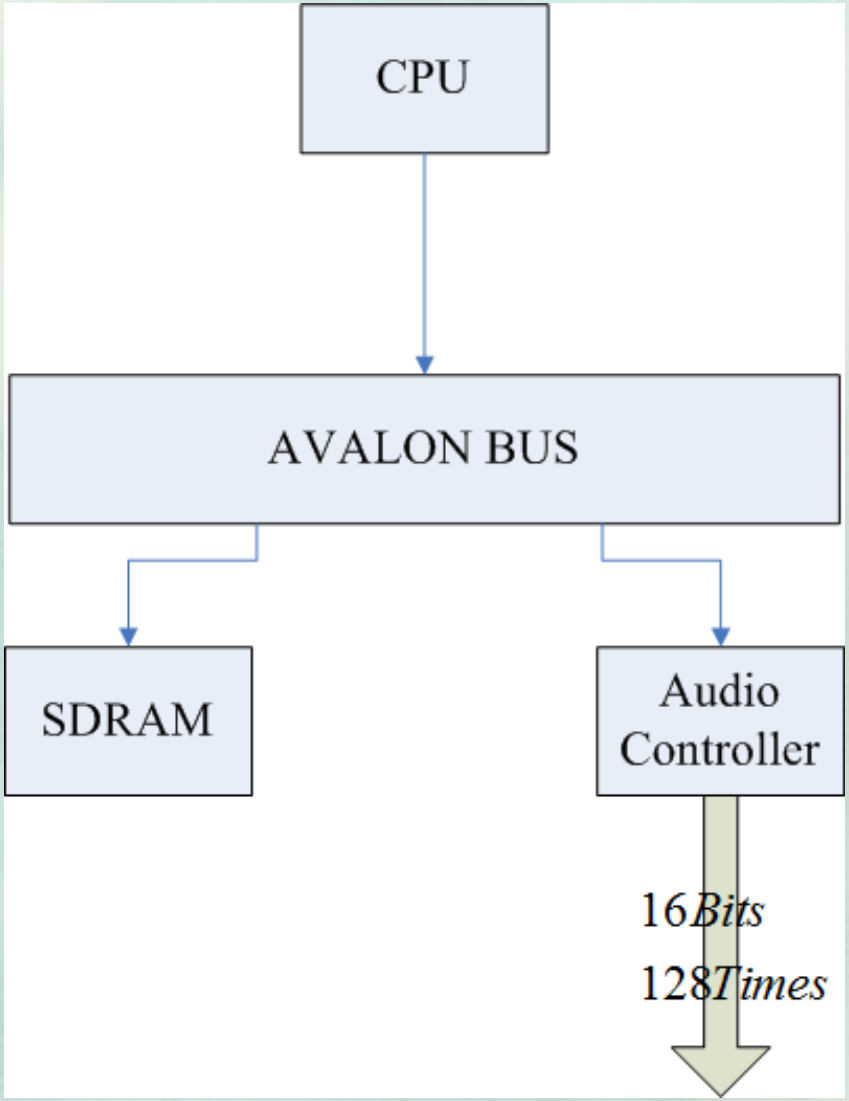
Architecture & Timing Design

Audio



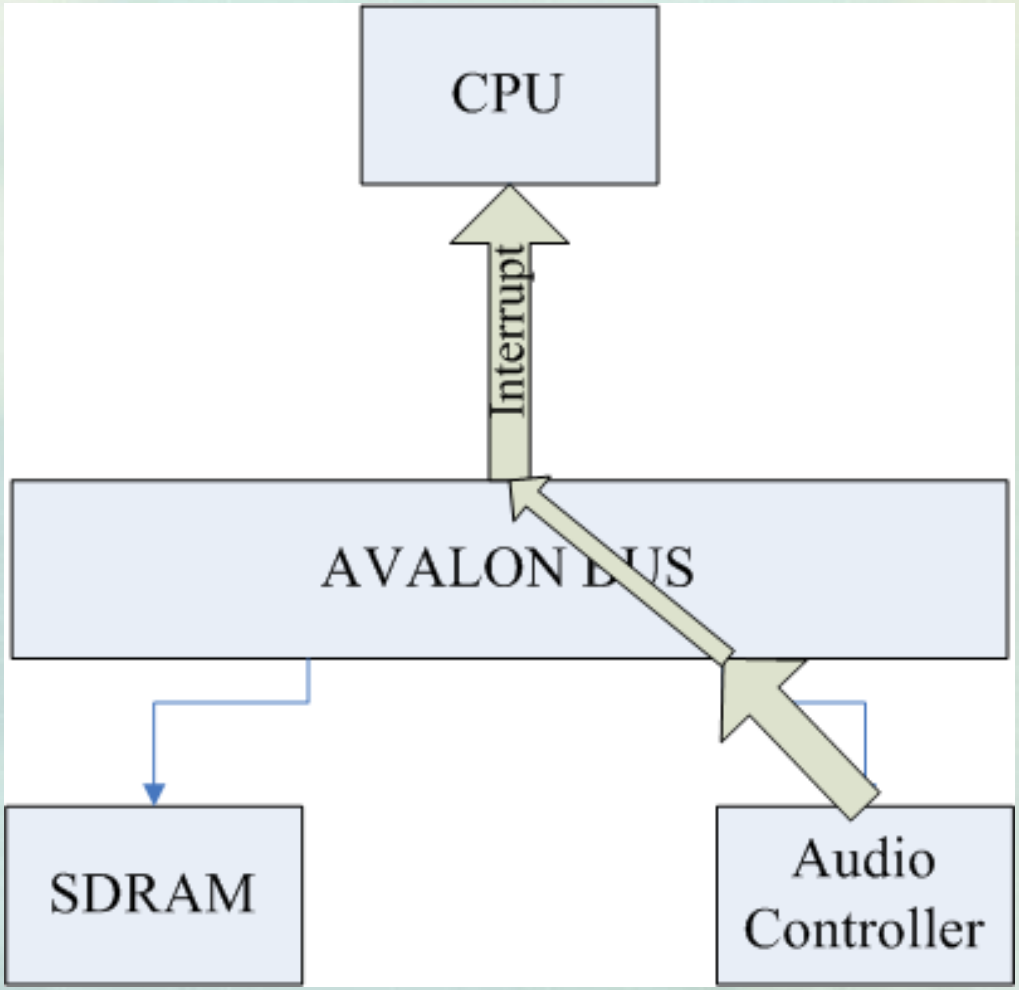
Architecture & Timing Design

Audio



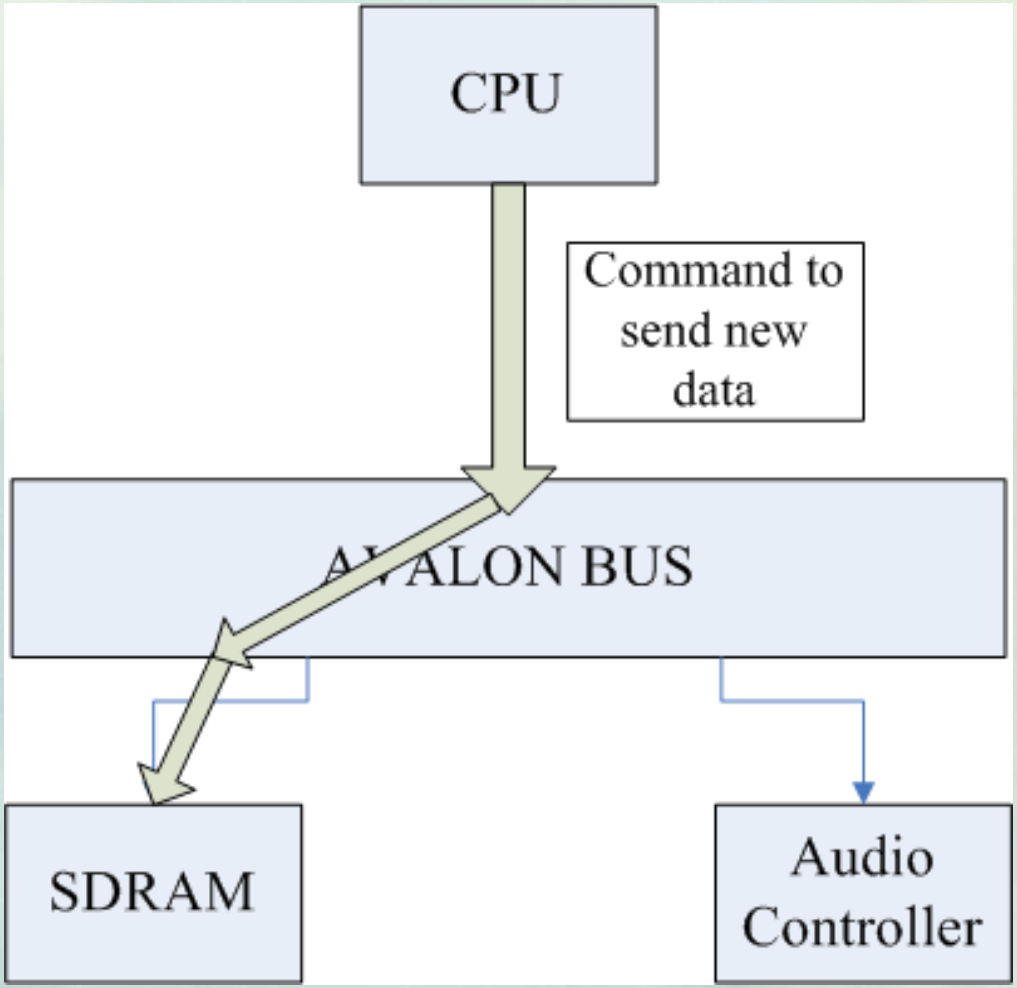
Architecture & Timing Design

Audio



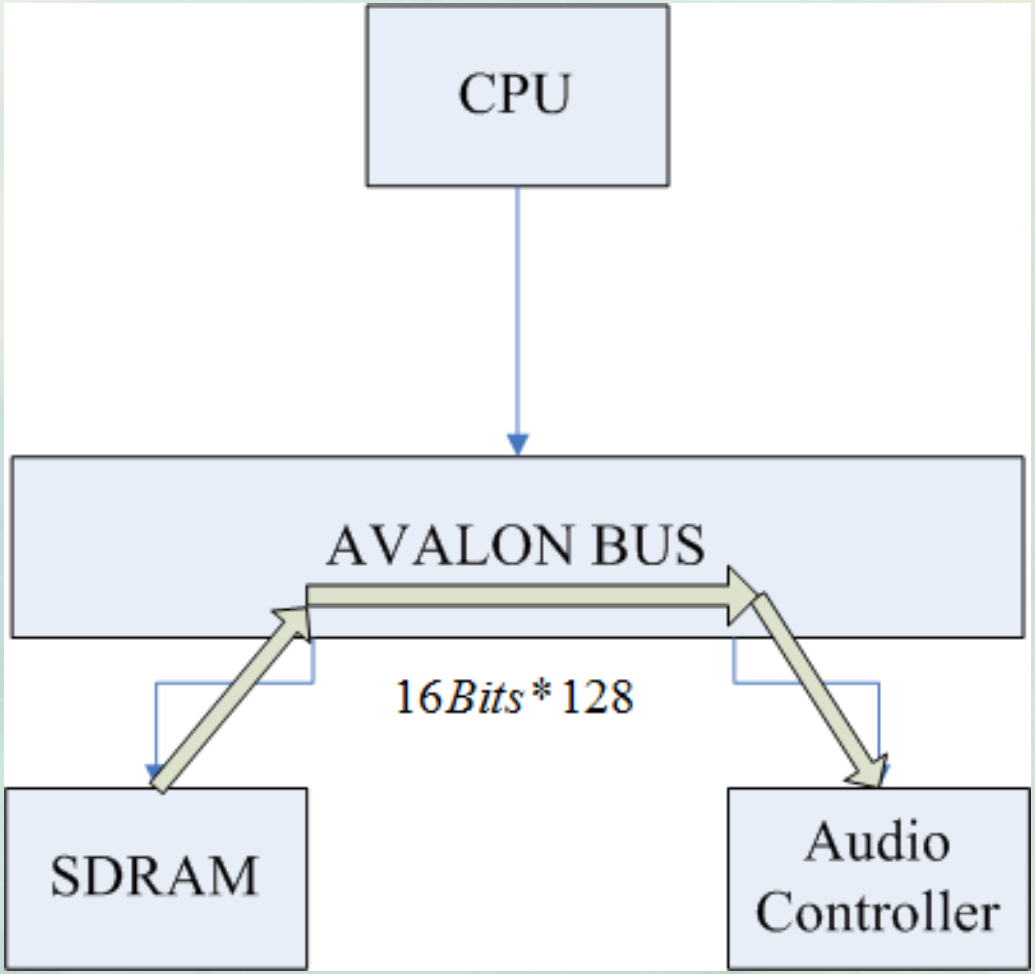
Architecture & Timing Design

Audio



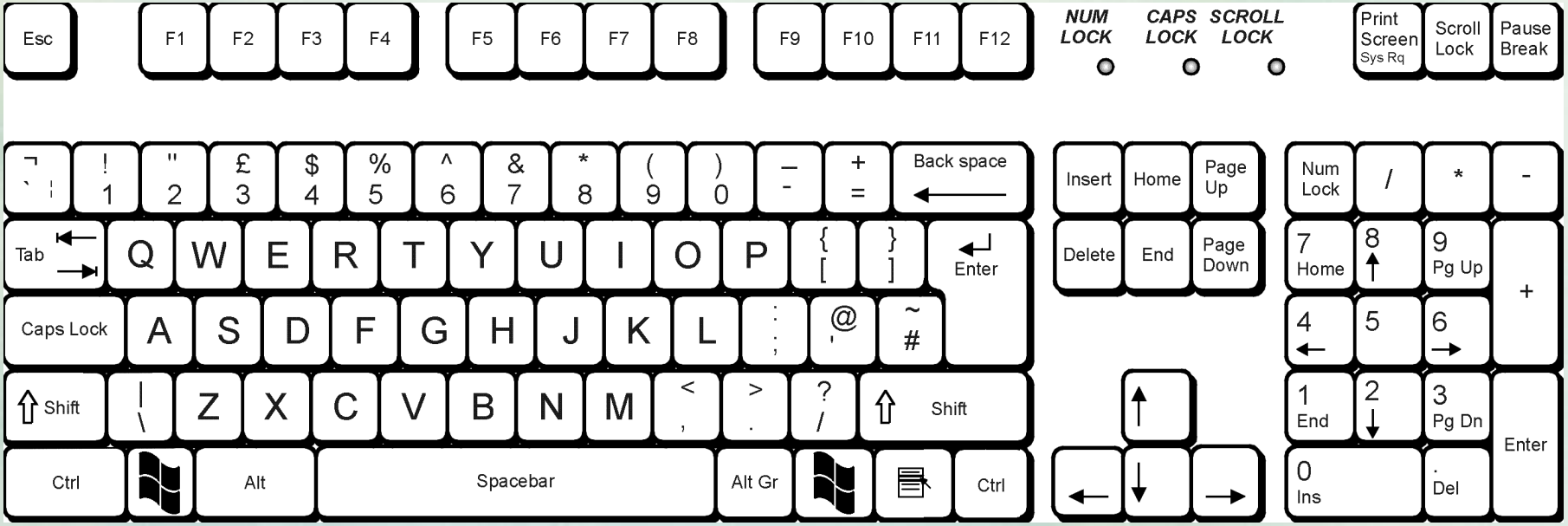
Architecture & Timing Design

Audio



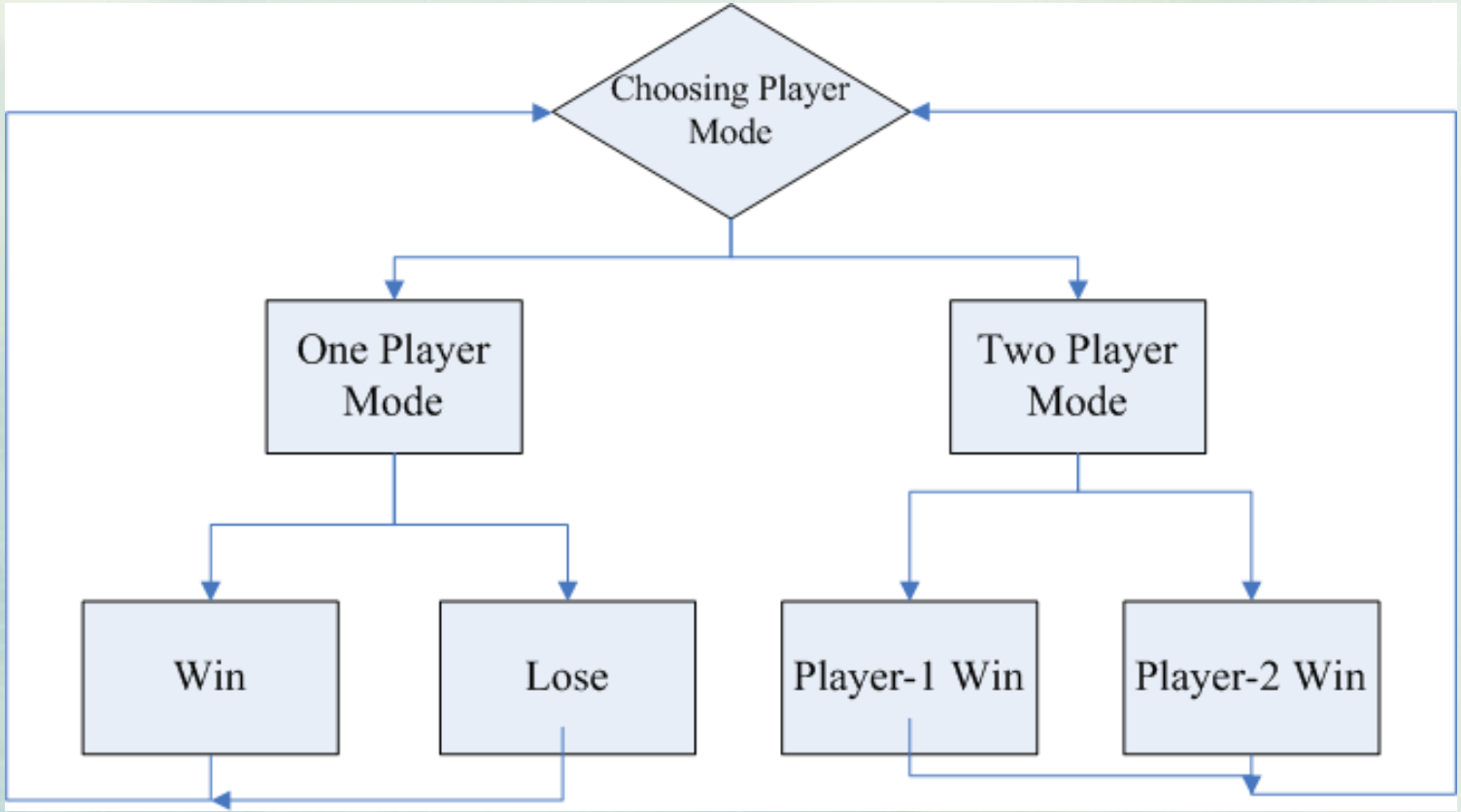
Architecture & Timing Design

PS2/Keyboard



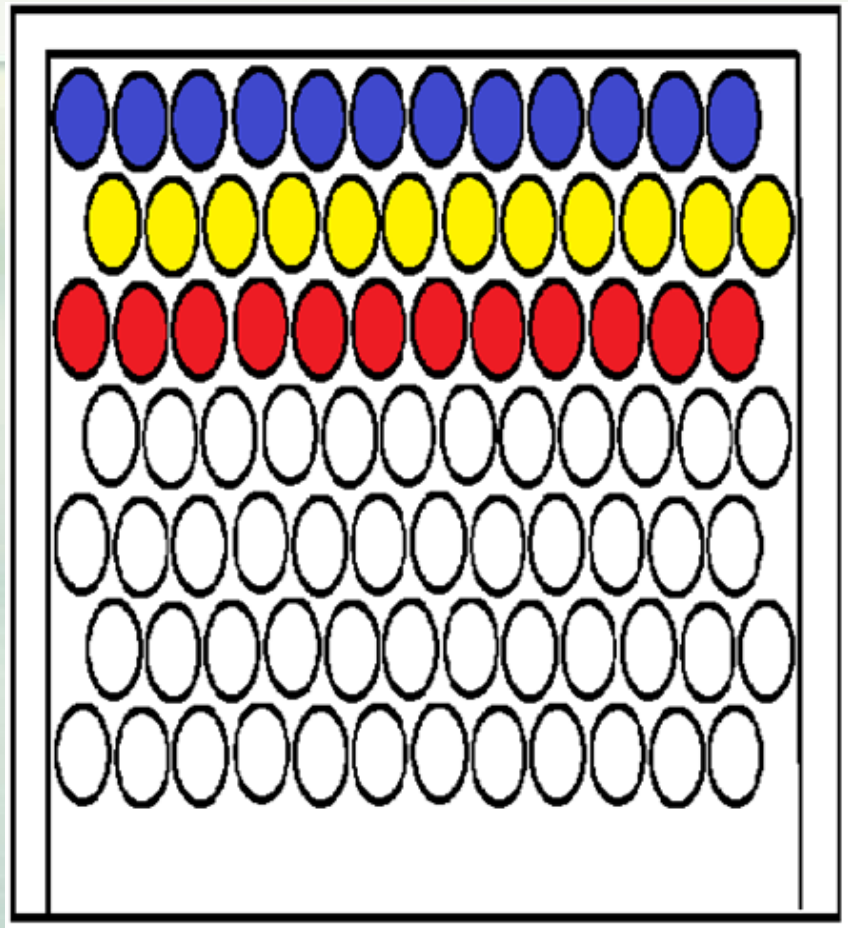
Architecture & Timing Design
Software

- Every mode have a special image



Architecture & Timing Design

Game Logic



1 1 1 1 1 1 1 1 1 1 1

2 2 2 2 2 2 2 2 2 2 2

3 3 3 3 3 3 3 3 3 3 3

0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0

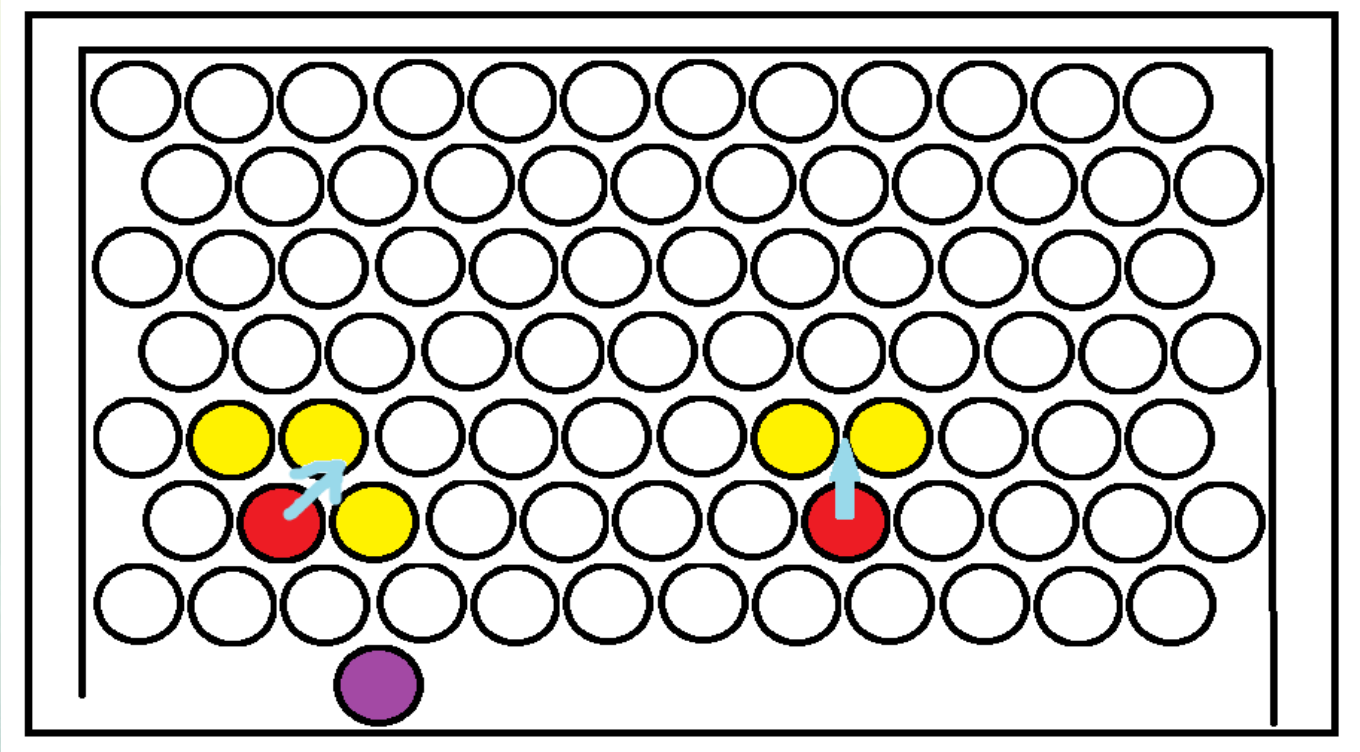
0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0

Divide the play-region into 180 ball regions and updating ball regions data by NIOS

Architecture & Timing Design

Game Logic



- We use this model to do the stop ball, shot ball and clean ball functions.

Experience and Issues in Implementation

- Timing issue
- Effective use the hardware source
- Software architecture

Summary & Lessons Learned

- Communication between hardware and software

  Team work

Thanks for your attention!!!