Flappy Bird

Wei Zheng Gaoyuan Zhang
Yen Hsi Lin Junhui Zhang

Overview and Objectives

- Motivated by the flappy bird on cell phone
- Strategies:

  press the key to fly the bird

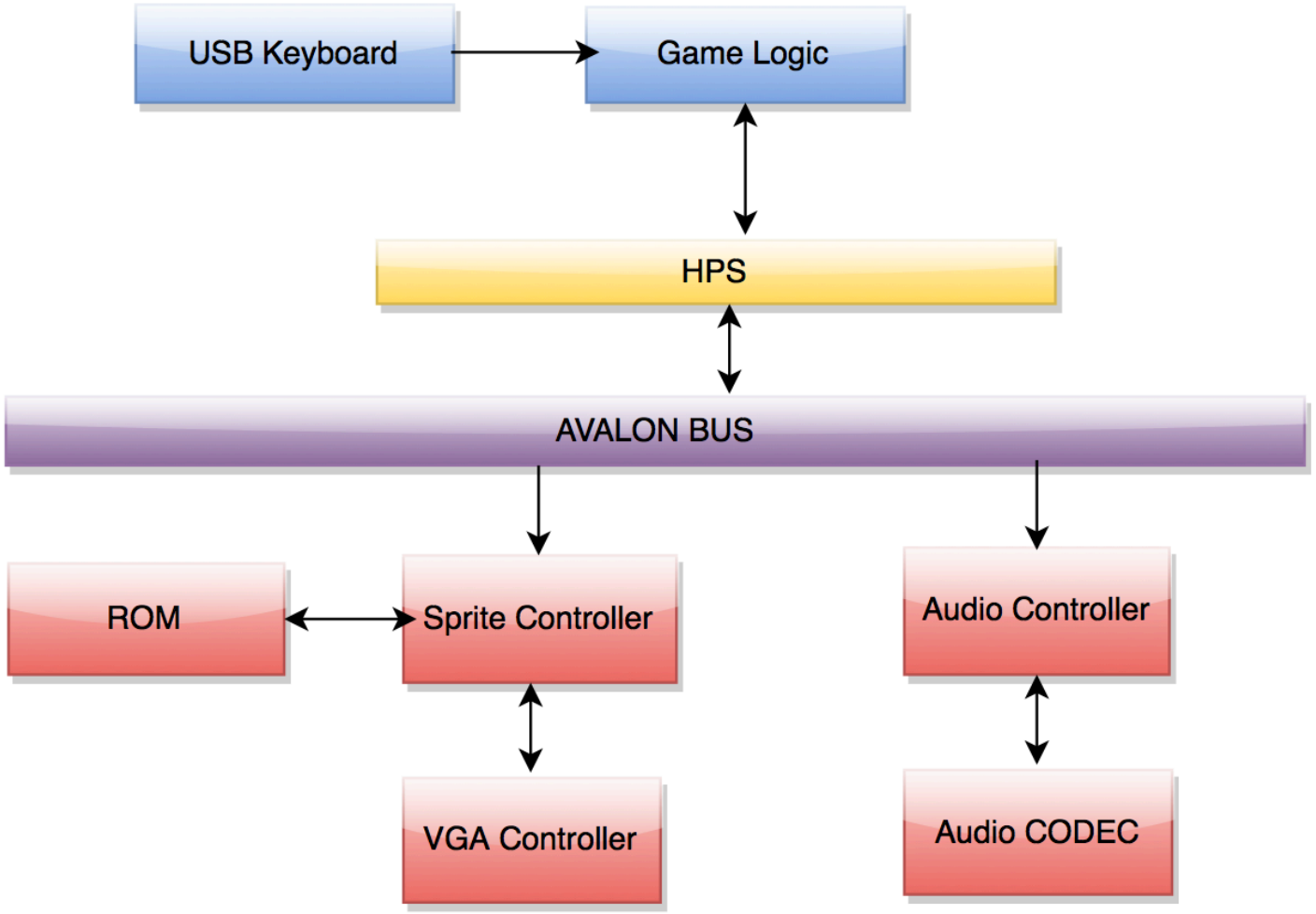
  the bird automatically falls down

  control the bird through the pipes with random heights

  fly as far as possible to get high scores
- Combine the elements of other games
- Objectives:

  make all things work together without bugs

Overview and Objectives



Overview and Objectives

- Image processing

  pre-process the images to use

  Generate a memory initialization file for each image

  Single-port ROM memory blocks

  24-bit index color

- Audio processing

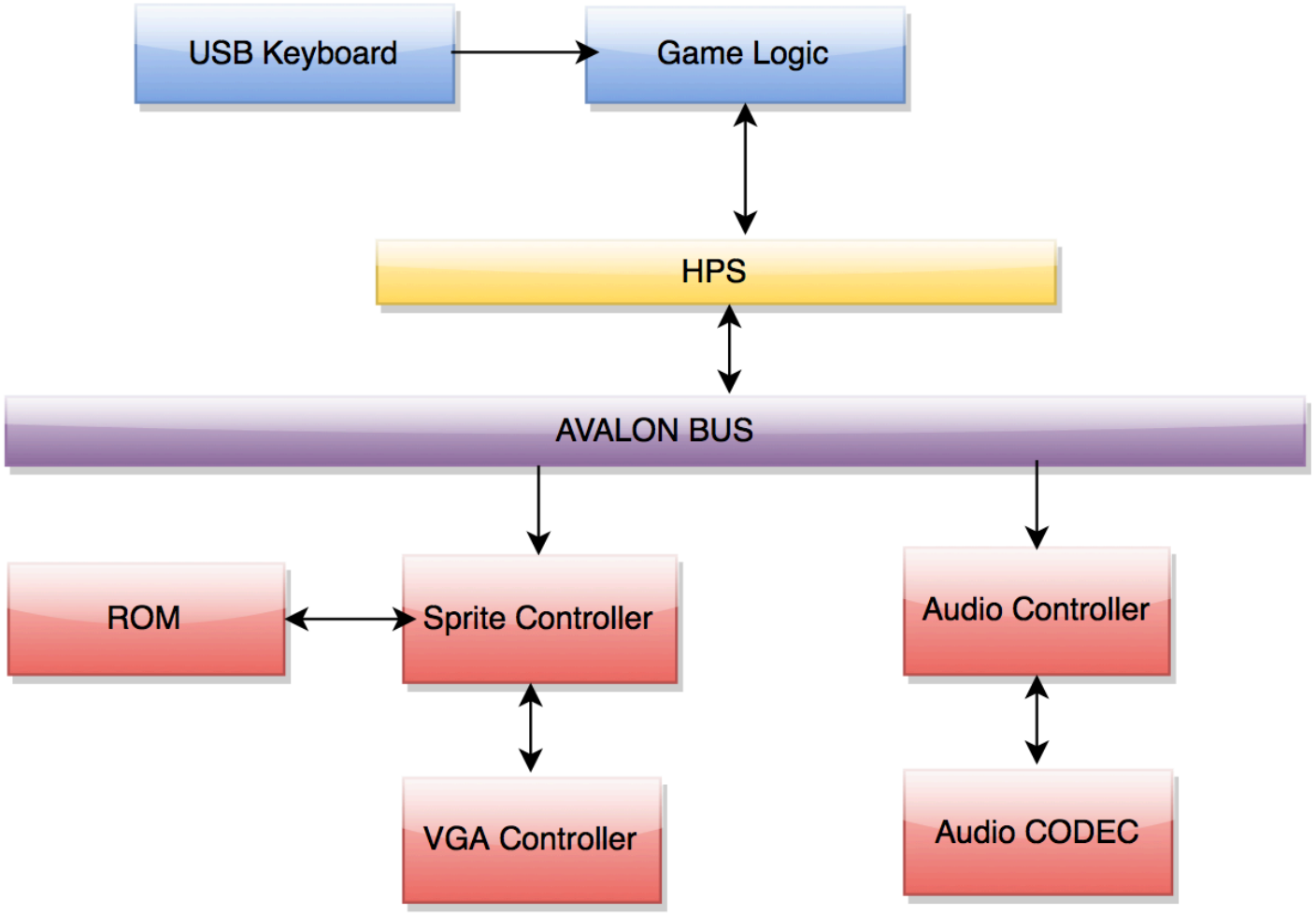
  pre-process the audios

  Generate a memory initialization file for each audio

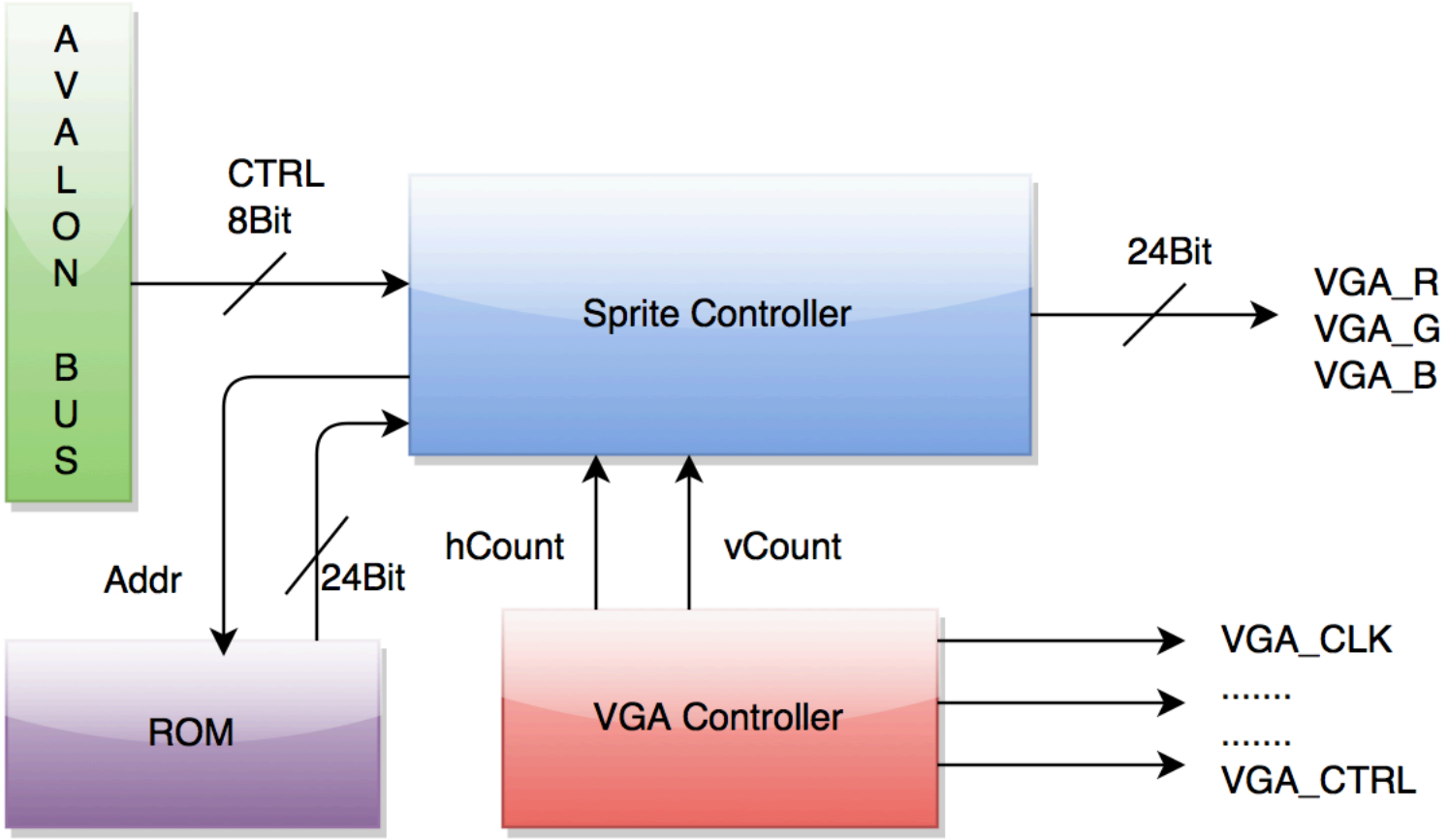
  Single-port ROM memory blocks

  44100Hz sampling rate 16-bit quantization

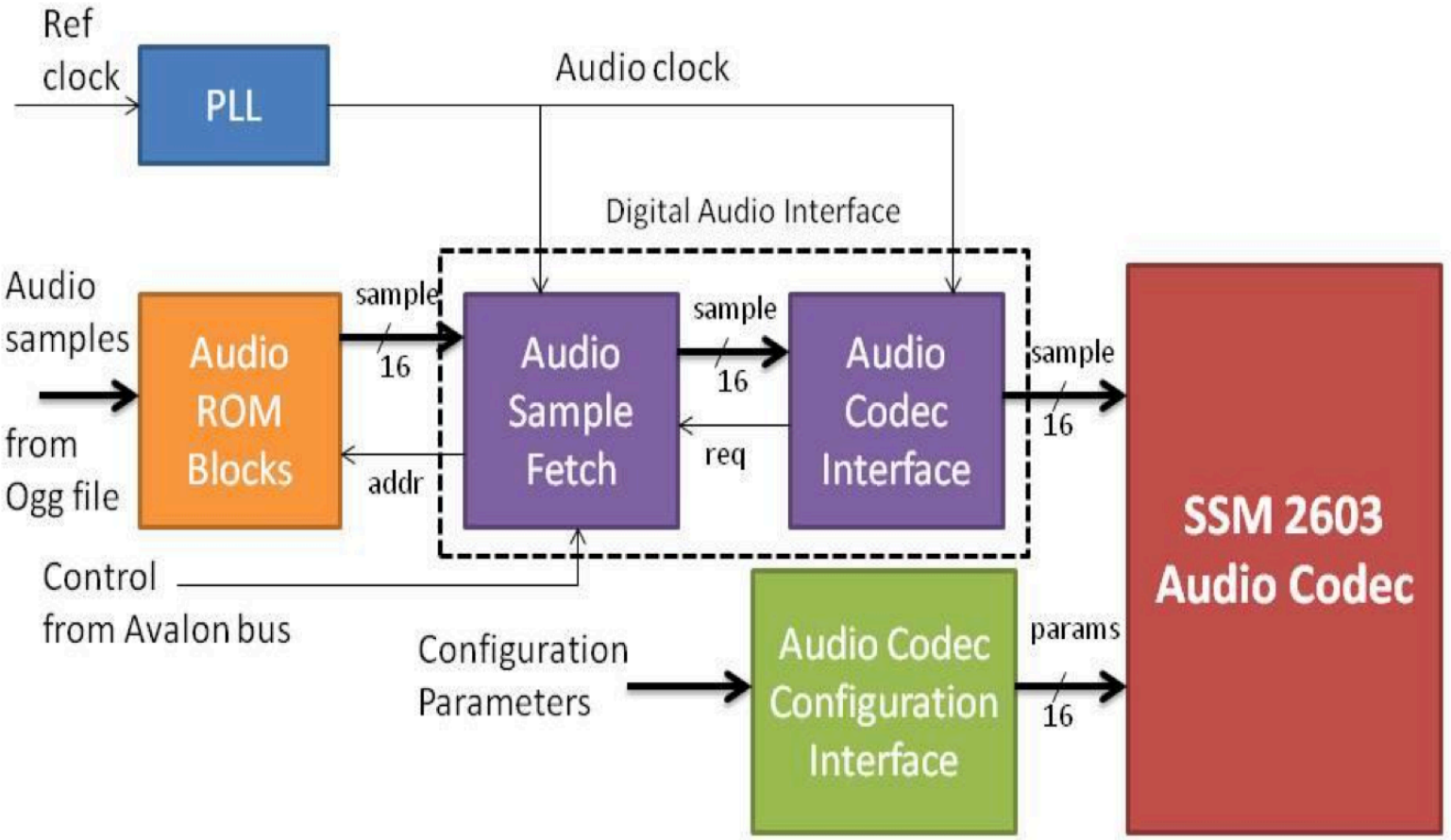
Architecture



VGA DISPLAY MODULE



Audio Controller



Keyboard Controller

- Use libusb C library to communicate with USB keyboard
- Spawn one thread to receive signal from Keyboard, leaving the main program to handle the sprite control

Game Logic

- Interaction between user and hardware
  - User: keyborad
  - Hardware: vga and audio controller
- Do the computation and control the game
  - Generate the height of pipes randomly
  - Control the up and down of bird
  - Control when the game is over

Experience and issues in implementation

- **Sprite background elimination (when displaying scores, bird, and sun)**
  - Simple color: check RGB value in ROM if it is equal to background value.
  - Complex color: No efficient way to eliminate the background color.

Experience and issues in implementation

- **Sprite for pillars**
  - In order to realize pillars with different length using only one sprite, we capture a sprite of "partial pillar", with relatively small height, and combine a certain number of such sprites to form a pillar whose length is what we want.

- **Priority of sprites**
  - We set the priority of each sprites so that the screen will show the correct image when sprites overlap with each other.

- **Frame update synchronization**
  - Receive and update coordination during synchronization *vcount*>480

Experience and issues in implementation

- **Control signal to display the title**

  We use one bit (first bit) of the 8-bit input as the signal to control if the title "flappy bird" is displayed or not, so that we can add more control signals by only introducing one input signal.

- **Problem**
  - Don't have enough space of RAM to add more signals.

Experience and issues in implementation

- **Game logic : Time**
  - System clock data type is not available in calculation, ex: clock()
  - Implement counter: loop number as time unit collaborating with delay, ex: usleep()
- **Game logic : Status**
  - Status variable: record jumping and falling
  - Status variable cooperates with initial velocity supporting continuous jumping without multi-thread

Experience and issues in implementation

- **Game logic : pillar display**
  - We combine the display of pillar with the check condition (whether the pillar is on the screen or not), and reset the coordinate of the pillar once the condition is triggered.

Lesson learned

- Sprite implementation
- Hardware and software collaboration
- Time management
- Architecture design of SoCKit board