**CSE 4117 Final Project.v2**

In this project, you will use Bird-CPU to read keystrokes from a ps2 keyboard and draw a scene on a VGA monitor.

Your system will draw

* A spaceship bitmap
* and a planet bitmap
* on a green background.

Spaceship bitmap will be 16x16 bits, and will be drawn at the center of the screen at the startup. Its design is given below (or, you can design your own). It will be red.

0000000010000000  
0000000111000000  
0000000111000000  
0000000111000000  
0000000111000000  
0000001111100000  
0000011111110000  
0000111111111000  
0011111111111110  
0000000111000000  
0000000111000000  
0000000111000000  
0000000111000000  
0000001111100000  
0000011111110000  
0000000111000000

Planet bitmap is again 16x16, and it is of shape circle. Design one of your own. It is yellow. It will also start at the center of the VGA screen.

You can see how bitmaps are used by computer graphics to draw figures on the screen by examining the pseudocode given below..

**Motion of spaceship**

The motions of spaceship will be controlled by the keyboard input:

--When the key W is pressed, the spaceship will move 4 pixels up.

--When the key S is pressed, the spaceship will move 4 pixels down.

--When the key A is pressed, the spaceship will move 4 pixels left.

--When the key D is pressed, the spaceship will move 4 pixels right.

-- spaceship should not pass the edges of the screen.

When the keys are released, no extra movement will occur (ie, you will disregard the second scancode which comes when you release the key).

**Motion of the planet**

The planet will move 10 pixels/second up in the vertical direction and 20 pixels/second to the right in the horizontal direction (these numbers are only approximate, do not try to fit them exactly). Whenever the planet hits left or right edge, its horizontal speed reverses. Whenever the planet hits top or bottom edge, its vertical speed reverses.

**Hardware you will build**

You can use the Verilog codes for the keyboard host controller interface and Mammal CPU without change from the website. The CPU will poll the keyboard for keypresses.

VGA module has the registers

Logic [15:0] x\_spaceship, y\_spaceship, x\_planet, y\_planet

Logic [15:0] spaceship\_bitmap[0:15]

Logic [15:0] planet\_bitmap[0:15]

VGA module will print the contents of the spaceship\_bitmap and planet\_bitmap to screen at every frame in such a way that the upper left hand corner of planet\_bitmap will be at pixel (x\_planet, y\_planet) and the upper left hand corner of spaceship\_bitmap will be at pixel (x\_spaceship, y\_spaceship). The main algorithm of the VGA module will be

Infinite loop {  
 For each pixel (x,y) in the visible part of VGA screen  
 If ((x\_spaceship <= x ) && (x<x\_spaceship+16)   
 && (y\_spaceship <= y ) && (y<y\_spaceship+16)  
 && ( spaceship\_bitmap[ x-x\_spaceship, y-y\_spaceship] ==1))  
 --paint the pixel to the colour of the spaceship (red)  
 elseif ((x\_planet <= x ) && (x<x\_planet+16)   
 && (y\_planet <= y ) && (y<y\_planet+16)  
 && ( planet\_bitmap[ x-x\_planet , y-y\_planet] ==1))  
 --paint the pixel to the colour of the planet (yellow)  
 else  
 --paint the pixel to the background color (green)  
}

As the registers x\_spaceship, y\_spaceship, x\_planet and y\_planet are changed by CPU, these objects move through the screen.

Note that the spaceship will always be at the front of the planet.

CPU will have two duties:

* During initialization, CPU will initialize spaceship\_bitmap and planet\_bitmap in VGA to the appropriate figures. This will require 16 store operations for each bitmap. Also don’t forget that stack and IDT must be also initialized.
* During operation, CPU will change the registers x\_spaceship, y\_spaceship, x\_planet, y\_planet for each frame, hence cause the spaceship and the planet to move.

Communicarion between the Mammal CPU and the VGA module will be via interrupts. If CPU modifies the registers x\_spaceship, y\_spaceship, x\_planet, y\_planet while the VGA draws the frame, artefacs could occur. Hence the modification of these registers should be done during the vertical flyback. When the vga module starts vertical flyback (ie, when vsync go from 0 to 1), it will send an interrupt signal to the CPU. CPU, on receiving the interrupt, will write new values into the registers x\_spaceship, y\_spaceship, x\_planet and y\_planet in the ISR. When any of these registers is written, the VGA module will lower the interrupt signal. In all the other times, the CPU will poll the keyboard.

You could use the codes for the PS2 keyboard module and Mammal CPU without change (or, with very little change). You have to modify the top module which "glues" the CPU, keyboard, and monitor significantly. You are also required to modify the VGA module. Assign addresses to the keyboard and VGA registers.

On the software side, you have to write the assembly code which manages the hardware for the given task. Modify the assembler codes given in the lecture notes suitably, and fill in the blank parts to assemble your code.