

Goal

- ❑ Develop a hardware accelerator for sprite graphics rendering in an FPGA.
- ❑ Utilize an ARM processor to work in tandem with the accelerator by issuing instructions.
- ❑ Demonstrate the complete platform by creating and running a sample game.

Motivations and Objectives

- ❑ Motivations
 - Demonstrate and further develop knowledge of computer architecture, digital system design, embedded software development, and system design and testing
 - Develop an entertaining product on which to demonstrate this project
- ❑ Objectives
 - Design hardware that can cycle through 512 sprites to generate 640 x 480 resolution frames
 - Have system do so at a rate of 60 frames per second
 - Maximize usage of 140 block RAM instances available on development board and 32MB NOR Flash memory in order to store frames and assets respectively

Research Challenges

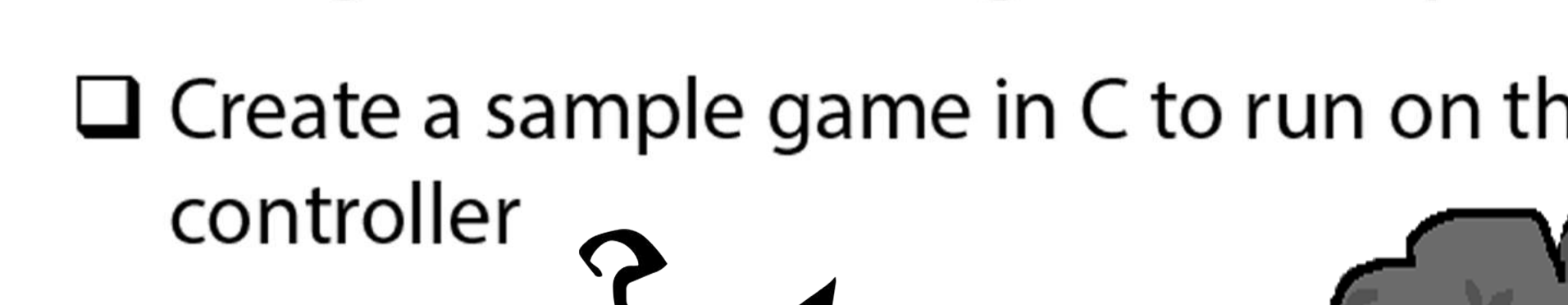
- ❑ Managing the memory available on the board in order to store frames being generated as well as sprites assets
- ❑ Perform communication between on-board processor and FPGA in order to coordinate generation of appropriate frames
- ❑ Designing hardware components such as the instruction register handler, SPI Flash controller, and frame buffer generator utilizing VHDL and Xilinx's Vivado

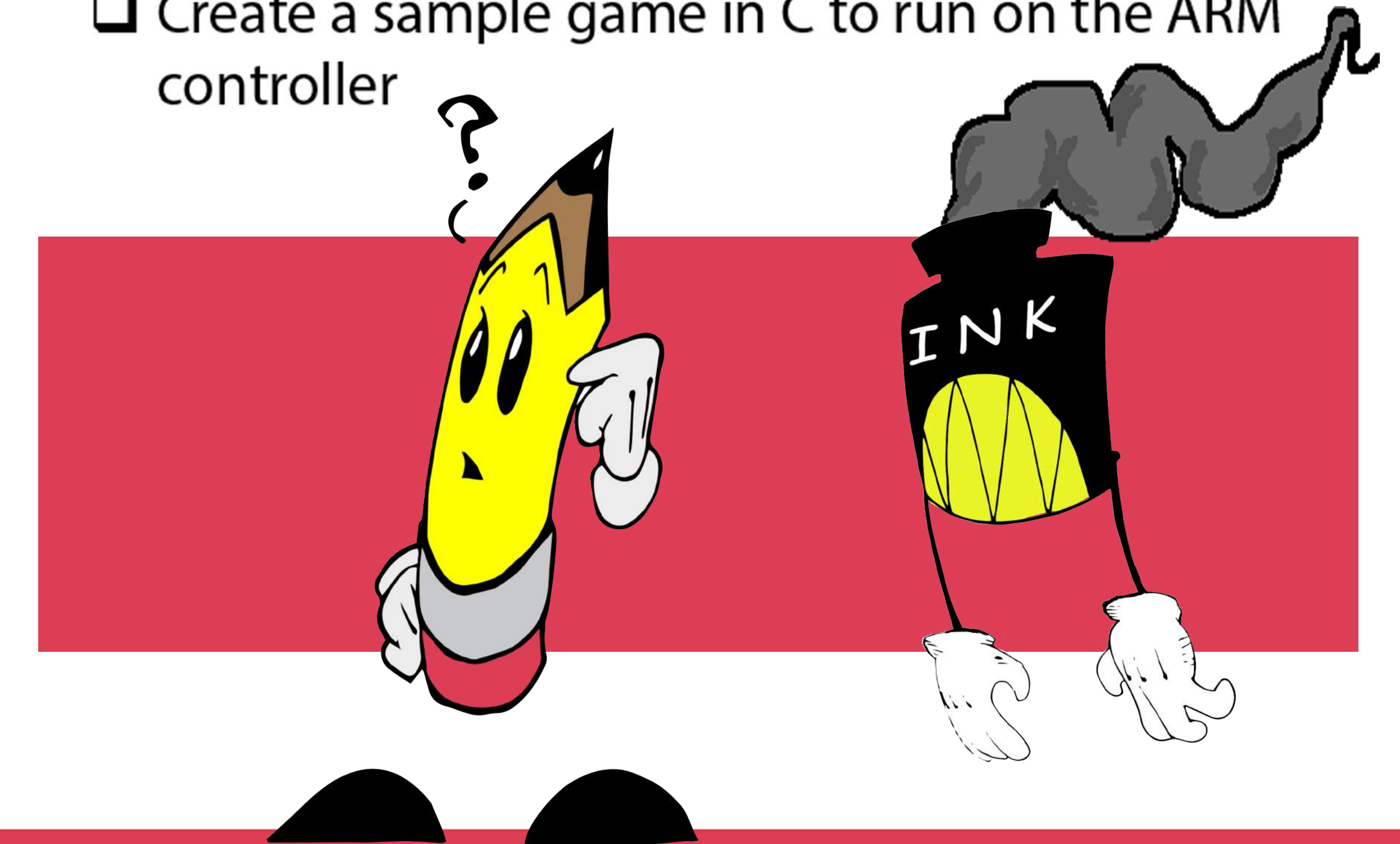
Acknowledgement

We would like to thank Philip Southard for advising us on this project

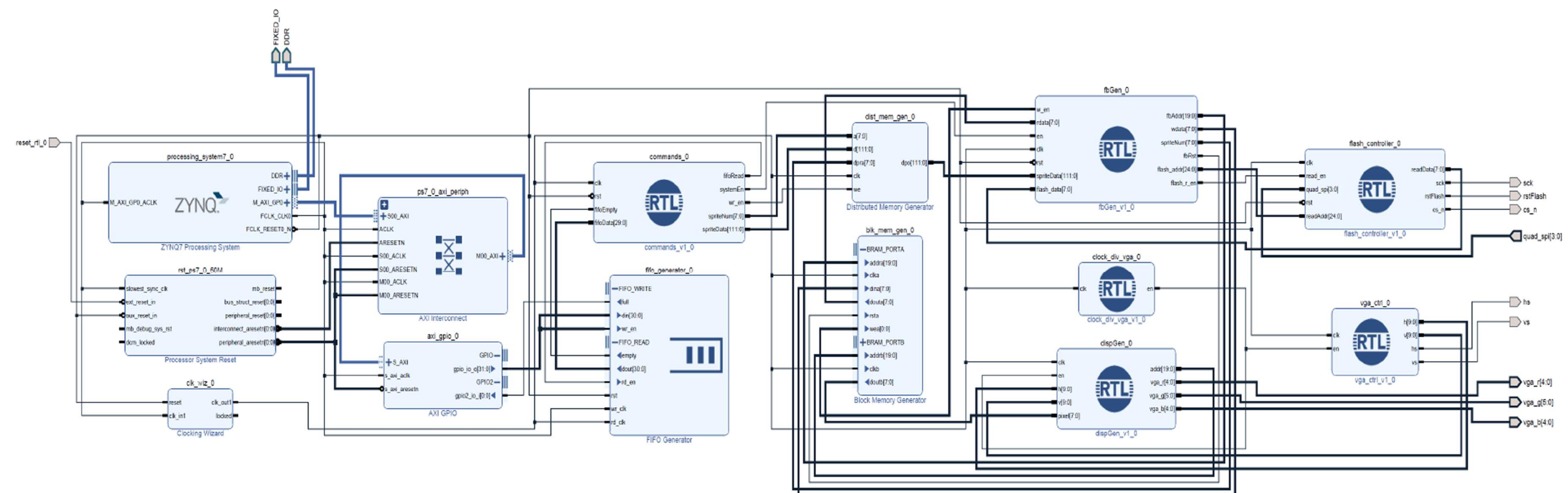
We would also like to thank Aaron Nell Millado of the Rutgers School of Arts and Sciences for providing artistic direction, as well as the sprites assets utilized for this project

Methodology

- ❑ Design a set of data attributes needed by each hardware sprite in order to be functionally complete
 - ❑ Develop a rendering algorithm that iterates across hardware sprites and uses their metadata in tandem with a central memory to generate the display frame
 - ❑ Detail a communications interface between the ARM controller and the sprite rendering accelerator
 - ❑ Create a memory controller to interface with a SPI NOR Flash as a central memory in order to retrieve game assets in a constrained time period
 - ❑ Develop the entire system using the custom IP and Xilinx IP cores
 - ❑ Design art assets for a game concept
 - ❑ Create a sample game in C to run on the ARM controller



Results



- ❑ Design requires a combination of 6 custom RTL designs, 5 IP cores, and the ARM Processor
- ❑ Block Diagram for top level design generated using Xilinx Vivado's IP Integrator tool

References

- [1] A Brown. (2014, June). An FPGA Sprite Graphics Accelerator with a 180MHz STM32F429 Controller and 640 x 360 LCD. Retrieved from <http://andybrown.me.uk/2014/06/01/ase/>