Class Report 3: FPro System Development

Samuel Hussey

1 Introduction

In this project a blinking-led core is incorporated into the FPro system development tutorial. The core has a 4-bit output signal corresponding to four discrete LEDs on the Nexys4, each LED blinks at a millisecond interval specified by one of four data registers interfaced to the FPro bus. The blinking-led circuit, its slot interface, and a wrapping circuit are synthesized to provide the hardware platform for driver development in the Vitis IDE. A blinking-led class provides the slot interface offsets and drivers to write the millisecond interval values to the core's data registers. A test program then demonstrates the ability to set a specific period for each of the LEDs by implementing a binary counting pattern. Source files for SystemVerilog and C++ can be found here and a video demonstration here.

2 Implementation

Implementation began with adding the MicroBlaze MCS soft-core processor IP to the project. A number of HDL files were then added to instantiate the CPU, bridge to the FPro bus, and IO subsystem. The MMIO subsystem contains the controller for interfacing with the IO bus as well as instantiations of the IO cores and their respective memory slots. The cores added include those for GPI, GPO, timer, UART, and the custom blinking-led core in slot 4.

Construction of the blinking-led core entailed a similar procedure as used for the provided example cores, starting with the design of the custom blinking-led circuit shown in figure 1. The IO register map for the slot interface was simple to determine, four 32-bit data registers were allocated on top of the base address offset. The wrapping circuit compliant with the slot interface specification was added to interface the FPro bus and decode the write operation and addressing. Finally, the software driver to utilize the new hardware was developed in the Vitis IDE by creating a custom class for the blinking-led core with its data register offsets and bare-metal write function. The testing program simply instantiates a BlinkingLedCore object and uses its write method to set the on/off intervals for each individual LED such that they count in binary to 15.

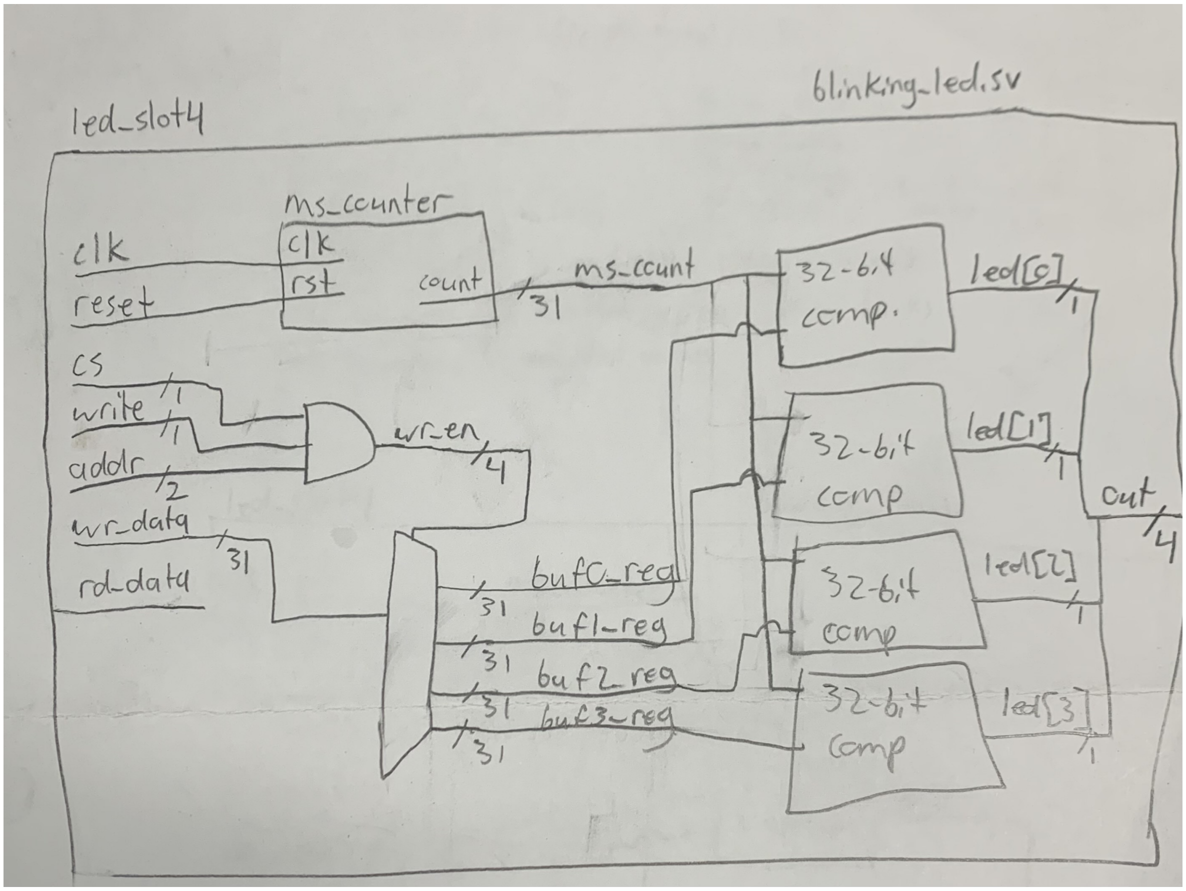

Figure 1: Design sketch for blinking LED circuit.

3 Results

Results are trivial. Once the hardware platform is imported from Vivado and the application project created in Vitis, the project is built. After specifying the bit and memory map information files that inform the IDE how to properly route instructions between the CPU and custom hardware, the device is programmed. The new IO core is exercised by writing to the register corresponding to the desired LED through a bare-metal interface, resulting in the toggling of LEDs at the specified rates.