Github Repo:  
  
<https://github.com/OliverJarvis1/Class_Report_3>

To develop this project, a block diagram was created for a core that controls a singular LED. Four of these were then combined into a larger core that controlled 4 LEDs.

A diagram of a light switch

Description automatically generated

A diagram of a circuit

Description automatically generated

After these block diagrams were created, the system was designed using hardware and software in the same order used for the block diagrams (1 led blinker made first, then multiplied into 4). Note that all code is provided by the fpga\_mcs\_sv\_src folder. The GPO Core module in this folder uses a 32 bit input to determine the base address. A GpoCore::write function allows data to be written to different LEDs by receiving a 32 bit input for data. If we input two integers, then we can interface with individual LEDs. We can achieve only 4 LEDs being interacted with by only writing to 4 different LEDs instead of every LED. The frequency of the blinking LEDs can be altered using longer or smaller sleep commands.

11.9.4 Blinking-LED core A blinking-LED core can turn LEDs on and off at specific rates. The core has a four-bit output signal connected to four discrete LEDs. It has four 16- bit registers that specify the values of the individual blinking intervals in milliseconds. With the blinking-LED core, the processor only needs to write the registers. The basic design and verification procedure is as follows:

* Design the blinking circuit for one LED and duplicate it four times.
* Determine the register map and derive the wrapping circuit.
* Derive the HDL code.
* Derive the device driver.
* Expand the vanilla MMIO subsystem to include a blinking-LED core in slot 4.
* Modify the vanilla FPro system to connect the led signal to the blinking-LED core and synthesize the new system.
* Derive a testing program and verify its operation.