1. **OpenCores Simple Peripheral (e.g., SPI Slave):**
   * Complexity: Low
   * Estimated Nodes: 100-500
   * Estimated Blocks: Few (logic gates, registers, potentially analog-to-digital converters)
   * Estimated Macros: Might include pre-designed IP blocks for specific functionalities
   * Estimated Gates: Varies depending on peripheral type and complexity
   * Description: Simple peripheral circuit for interfacing with external devices.
2. **OpenCores Complex Peripheral (e.g., VGA Controller):**
   * Complexity: High
   * Estimated Nodes: > 5000
   * Estimated Blocks: Specialized video processing units, memory controllers
   * Estimated Macros: Likely includes pre-designed IP blocks for video generation
   * Estimated Gates: High number of gates due to complex processing needs
   * Description: Complex peripheral circuit for handling video display functionalities.
3. **OpenCores RISC-V Core (e.g., PicoRV32):**
   * **Complexity:** Moderate
   * **Estimated Nodes:** 1000-2000
   * **Estimated Blocks:** Processing units (ALU, register file), control unit
   * **Estimated Macros:** Might include a pre-designed memory controller macro
   * **Estimated Gates:** 1000-3000 (varies depending on core features)
   * **Description:** A simple RISC-V processor core with basic processing capabilities like performing ALU operations, load/store instructions, and control flow. This circuit provides an opportunity to evaluate placement for a processing unit, including its data path (ALU, registers) and control logic. It's more complex than the previous examples as it introduces an instruction set architecture and basic processing pipeline.
4. **OpenCores DMA Controller (e.g., Lite DMA Controller):**
   * **Complexity:** Moderate
   * **Estimated Nodes:** 1500-3000
   * **Estimated Blocks:** Dedicated logic for DMA channels, memory address generation, data transfer units
   * **Estimated Macros:** Might include pre-designed bus interface controller macros
   * **Estimated Gates:** 2000-5000 (varies depending on DMA controller complexity)
   * **Description:** A memory controller design for efficient data transfer between memory and processor. DMA (Direct Memory Access) controllers handle data movement autonomously, reducing processor workload. This circuit allows you to analyze placement for memory access patterns and the interaction between the DMA controller, memory, and processor. The complexity of DMA controllers varies depending on the number of supported channels, data transfer features, and bus interfaces.
5. **OpenCores Complex Microcontroller (e.g., OpenRISC 1200):**
   * **Complexity:** High
   * **Estimated Nodes:** > 10,000
   * **Estimated Blocks:** Processing units (ALU, multipliers, dividers), memory management units, peripherals
   * **Estimated Macros:** Likely includes pre-designed IP blocks for memory controllers, communication interfaces
   * **Estimated Gates:** Very high number of gates due to the comprehensive features (> 50,000)
   * **Description:** A complex microcontroller design with features beyond a simple RISC-V core. It might include additional processing units like multipliers and dividers, memory management units for handling virtual memory, and various peripherals for interfacing with external devices. This circuit represents a challenging benchmark for your DeepRL+MultipleHA approach, allowing you to assess its effectiveness in handling a complete system-on-chip (SoC) design with diverse processing needs and communication requirements.
6. **OpenCores Simple Peripheral (e.g., I2C Slave):**

* **Nodes:** Varies depending on the specific peripheral (typically 100-200)
* **Working Complexity:** Medium
* **Description:** This example uses an I2C Slave peripheral as a representative of simple peripherals. These peripherals handle communication with external devices. They introduce new placement considerations due to the presence of external interfaces alongside internal logic blocks.