Notes on IBEX and custom core

M. K. Dahl

March 25, 2025

1 Introduction

2 Quick start

3 Notes About Simple System

The simple system integrates the Ibex core as the host and connects to three slave devices. Below are the details of the system:

3.1 Host

The Ibex core acts as the host in the system. The host is defined as:

```
typedef enum logic {
    CoreD
} bus_host_e;
```

3.2 Slave Devices

The system is integrated with three slave devices, which can be accessed using the following addresses:

- `00`: RAM
- `01`: Simulation Control (`SimCtrl`)
- `10`: Timer

The slave devices are defined as:

```
typedef enum logic[1:0] {
    Ram,
    SimCtrl,
    Timer
} bus_device_e;
```

3.3 RAM

The RAM has the following characteristics:

- Depth: $\frac{1024 \times 1024}{4} = 256$k words, assuming each word is 4 bytes (32 bits).
- Total size: $1024 \times 1024 = 1$MB, divided by 4 to get 256k words.

The RAM file is set during simulation using the following command:

```
./build/lowrisc_ibex_ibex_simple_system_0/sim-verilator/Vibex_simple_system [-t] --meminit=r
```

Here, `<sw_elf_file>` should be the path to an ELF file (or alternatively a VMEM file).

The RAM is instantiated as:

```
ram_2p #(
    .Depth(1024*1024/4)
)
```

The device address mapping for the RAM is configured as follows:

```
// Device address mapping
assign cfg_device_addr_mask[Ram] = ~32'hFFFFF; // 1 MB
```

This masks the lower 20 bits, ensuring that everything above 12 bits is set to 0.

3.4 Other Peripherals

- **BUS:** A simple bus that connects the host to the slave devices.
- **Simulation Control (`SimCtrl`):** A peripheral used to write ASCII output to a file.
- **Timer:** A basic timer with interrupt capabilities.

3.5 Software Framework

The system includes a software framework to interact with the peripherals and manage the system.

4 Configurations

The following items can be configured in the Ibex core, enabled or disabled. There are 4 support IBEX configs: small, opentita, maxperf and maxperf-pmp-bmbalanced.

- **RV32E:** Determines whether the core uses the RV32E instruction set, which is a reduced version of the RISC-V instruction set with only 16 general-purpose registers (instead of 32 in RV32I). Set to 0 to disable RV32E and use the full RV32I instruction set.

- **RV32M:** Specifies the configuration of the multiplier/divider extension (RV32M). `ibex_pkg::RV32MFast` enables a fast implementation of the RV32M extension with a 3-cycle multiplier.
- **RV32B:** Specifies the configuration of the bit manipulation extension (RV32B). `ibex_pkg::RV32BNone` disables the RV32B extension.
- **RegFile:** Configures the type of register file. `ibex_pkg::RegFileFF` uses flip-flops for the register file implementation, which provides lower latency but uses more area.
- **BranchTargetALU:** Enables or disables a dedicated branch target ALU. Set to 0 to disable it, reducing area but increasing branch resolution latency.
- **WritebackStage:** Configures whether the core has a separate writeback stage in the pipeline. Set to 0 to disable it, simplifying the pipeline but potentially reducing performance.
- **ICache:** Enables or disables the instruction cache. Set to 0 to disable it, reducing area but increasing instruction fetch latency.
- **ICacheECC:** Enables or disables error-correcting code (ECC) for the instruction cache. Set to 0 to disable it, reducing area and power consumption.
- **ICacheScramble:** Enables or disables scrambling for the instruction cache. Set to 0 to disable it, simplifying the design.
- **BranchPredictor:** Enables or disables the branch predictor. Set to 0 to disable it, reducing area but increasing branch misprediction penalties.
- **DbgTriggerEn:** Enables or disables debug triggers. Set to 0 to disable them, reducing area.
- **SecureIbex:** Enables or disables security features in the Ibex core. Set to 0 to disable them, simplifying the design.
- **PMPEnable:** Enables or disables the Physical Memory Protection (PMP) feature. Set to 0 to disable it, reducing area.
- **PMPGranularity:** Configures the granularity of PMP regions. Set to 0 for the default granularity.
- **PMPNumRegions:** Specifies the number of PMP regions. Higher values allow for more memory regions to be protected, increasing flexibility but also area.
- **MHPMCounterNum:** Specifies the number of hardware performance monitoring counters. Set to 0 to disable them, reducing area.

- **MHPMCounterWidth:** Specifies the width (in bits) of the hardware performance monitoring counters. Wider counters allow for larger values to be recorded, reducing the risk of overflow during long-running measurements.