Notes on IBEX and custom core

M. K. Dahl

April 2, 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, opentitan, maxperf and maxperf-pmp-bmbalanced.

Name	Description	Possible Values	Value in Cille
RV32E	Determines whether the core uses the RV32E instruction set (16 general-purpose registers instead of 32).	0 (disabled), 1 (enabled)	0
RV32M	Configures the multiplier/divider extension.	`ibex_pkg::RV32MNone`, `RV32MSingleCycle`, `RV32MFast`	`RV32MSingleCycle`
RV32B	Configures the bit manipulation extension.	`ibex_pkg::RV32BNone`, `RV32BOTEarlGrey`	`RV32BOTEarlGrey`
RegFile	Type of register file used in the core.	`ibex_pkg::RegFileFF`, `RegFileLatch`	`RegFileFF`
BranchTargetALU	Enables a dedicated ALU for branch target calculation.	0 (disabled), 1 (enabled)	1
WritebackStage	Enables a separate writeback stage in the pipeline.	0 (disabled), 1 (enabled)	1
ICache	Enables the instruction cache.	0 (disabled), 1 (enabled)	1
ICacheECC	Enables error-correcting code (ECC) for the instruction cache.	0 (disabled), 1 (enabled)	1
ICacheScramble	Enables scrambling for instruction cache contents.	0 (disabled), 1 (enabled)	1
BranchPredictor	Enables the branch predictor for speculative execution.	0 (disabled), 1 (enabled)	1
DbgTriggerEn	Enables hardware debug triggers.	0 (disabled), 1 (enabled)	1
SecureIbex	Enables security features in the Ibex core.	0 (disabled), 1 (enabled)	1
PMPEnable	Enables the Physical Memory Protection (PMP) feature.	0 (disabled), 1 (enabled)	1
PMPGranularity	Defines the granularity of PMP regions (0 = default, higher values for finer granularity).	Integer values	0
PMPNumRegions	Specifies the number of PMP regions available.	0, 8, 16	16
MHPMCounterNum	Number of hardware performance monitoring counters.	0, 4, 10	10
MHPMCounterWidth	Bit width of hardware performance monitoring counters.	32, 64	32

Table 1: Configuration options for the Cille configuration of the Ibex core.