MIPS32 Pipeline

A naive 5-stage pipeline MIPS CPU.

Quick Start

My hardware:

```
$ uname -a
Darwin MacBook-Pro.local 23.4.0 Darwin Kernel Version 23.4.0: Fri Mar 15 00:12:41 PDT 2024;
root:xnu-10063.101.17~1/RELEASE_ARM64_T8103 arm64
```

Set up the required toolchains:

```
$ brew install verilator llvm@17
$ echo 'export PATH="/opt/homebrew/opt/llvm@17/bin:$PATH"' >> ~/.zshrc
```

For better development experience, add the following two paths to the `includePath` in VSCode. Please note that the paths may vary based on your setup, but you should be able to locate the correct ones on your machine:

```
/opt/homebrew/Cellar/verilator/5.024/share/verilator/include
/opt/homebrew/Cellar/verilator/5.024/share/verilator/include/vltstd
```

Build Test Images

Set up the required toolchains based on the following list:

```
$ dpkg --list | grep mips
ii  binutils-mips-linux-gnu          2.38-1ubuntu1cross2                  amd64        GNU
binary utilities, for mips-linux-gnu target
ii  cpp-10-mips-linux-gnu            10.3.0-1ubuntu1cross2                amd64        GNU C
preprocessor
ii  cpp-mips-linux-gnu               4:10.2.0-1                           amd64        GNU C
preprocessor (cpp) for the mips architecture
ii  gcc-10-cross-base-mipsen         10.3.0-1ubuntu1cross2                all          GCC, the
GNU Compiler Collection (library base package)
ii  gcc-10-mips-linux-gnu            10.3.0-1ubuntu1cross2                amd64        GNU C
compiler (cross compiler for mips architecture)
ii  gcc-10-mips-linux-gnu-base:amd64 10.3.0-1ubuntu1cross2                amd64        GCC, the
GNU Compiler Collection (base package)
ii  gcc-mips-linux-gnu               4:10.2.0-1                           amd64        GNU C
compiler for the mips architecture
ii  libatomic1-mips-cross            10.3.0-1ubuntu1cross2                all          support
library providing __atomic built-in functions
ii  libc6-dev-mips-cross             2.35-0ubuntu1cross1                  all          GNU C
Library: Development Libraries and Header Files (for cross-compiling)
```

```
ii  libc6-mips-cross                2.35-0ubuntu1cross1                  all          GNU C
Library: Shared libraries (for cross-compiling)
ii  libgcc-10-dev-mips-cross        10.3.0-1ubuntu1cross2                all          GCC
support library (development files)
ii  libgcc-s1-mips-cross            10.3.0-1ubuntu1cross2                all          GCC
support library (mips)
ii  libgomp1-mips-cross             10.3.0-1ubuntu1cross2                all          GCC
OpenMP (GOMP) support library
ii  linux-libc-dev-mips-cross       5.15.0-18.18cross1                   all          Linux
Kernel Headers for development (for cross-compiling)
```

To build the test images, navigate to `./tests` and run:

```
make all
```

If you have all the required toolchains set up correctly, you will find `*.bin` files in the test case folders.

Run Tests

If everything goes well, you can run the tests with:

```
make test
```

and see the output:

```
[OK]            TEST1-JUMP.
[OK]            TEST2-BITWISE.
[OK]            TEST3-IMM.
[OK]            TEST4-OPs.
[OK]            TEST5-LOAD_STORE.
[OK]            TEST6-BRANCH.
[OK]            TEST7-LUI.
[OK]            TEST8-QSORT.
ACCEPTED.
```

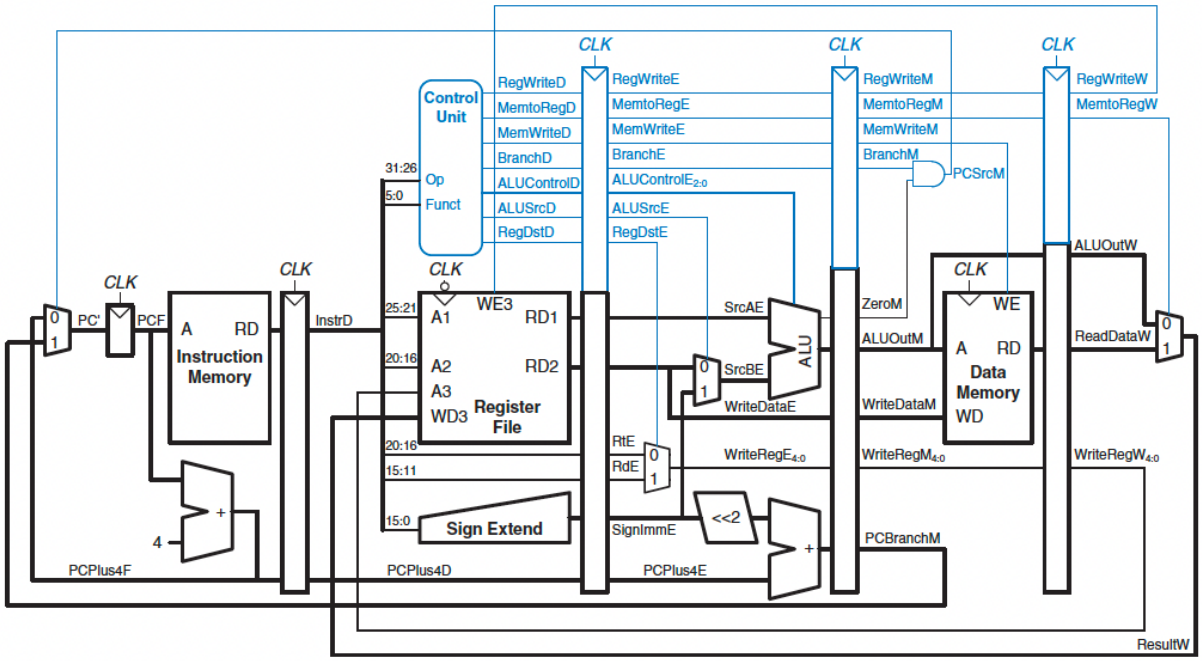
To debug the CPU step by step, try:

```
make run
```

You can change the target image in the Makefile under the root folder.

Overview

The CPU is built from the following reference architecture, with some fixtures and modifications made by me, including memory access latencies, bitwise operations, data forwarding, and more.



Some details of implementation can be found in the comments from the source code.