Computer Architecture
Assignment-2 Processor Simulation (Chipyard)

In this Assignment of Processor Simulation using **chipyard** was done. Chipyard is a collection of various tools that can be used for a variety of purposes, in this assignment it was used to simulate various riscv benchmark codes using the verilator on different numbers of cores.

1. Install and Setting up chipyard: For this chipyard was installed on the virtual instance and also docker was also used, to create the environment required for chipyard which is Ubuntu or Debian. After the installation and creation of the riscv-toolchain, the following output can be seen in the env.sh file. After viewing this it can be confirmed that chipyard is successfully installed.
There are two files provided in the zip folder named **chip.txt and gen.txt**. These files contain the directory structure obtained after installing chipyard and riscv-toolchain.

```
root@c21295631886: ~/chipyard
root@c21295631886:~/chipyard# ls
CHANGELOG.md     README.md    docs                fpga                                riscv-tools-install  target      variables.mk
CONTRIBUTING.md  build.sbt    env-esp-tools.sh    generators                          scripts              tests       vcs.mk
LICENSE          common.mk    env-riscv-tools.sh  init-submodules-no-riscv-tools.log  sims                 toolchains  vlsi
LICENSE.SiFive   dockerfiles  env.sh              project                             software             tools
root@c21295631886:~/chipyard# cat env.sh
# line auto-generated by init-submodules-no-riscv-tools.sh
__DIR="$(dirname "$(readlink -f "${BASH_SOURCE[0]:-${(%):-%x}}")")"
PATH=$__DIR/bin:$PATH
PATH=$__DIR/software/firemarshal:$PATH
# line auto-generated by build-toolchains.sh
source /root/chipyard/env-riscv-tools.sh
# line auto-generated by build-toolchains.sh
source /root/chipyard/env-esp-tools.sh
root@c21295631886:~/chipyard#
```

2. Build Stage processors: For building the processors with different stages.

- First go to the directory **chipyard/sims/verilator.**
- In the directory run the command make **CONFIG=Sodor1StageConfig**, to build one stage processor.
- Similarly change the R.H.S of command to **Sodor2StageConfig, Sodor3StageConfig, Sodor5StageConfig, SodorUCodeConfig** for creating two stage, three stage, five stage, and U code processor respectively.

After building the processors you can see that different directories are created for each processor and also there are different directories in generated-src for each processor.

Directories for each processor.

```
root@c21295631886: ~/chipyard/sims/verilator
-faligned-new -Wno-bool-operation -Wno-sign-compare -Wno-uninitialized -Wno-unused-but-set-variable -Wno-unused-parameter -Wno-unused-v
ariable -Wno-shadow     -O3 -std=c++11 -I/root/chipyard/riscv-tools-install/include -I/root/chipyard/tools/DRAMSim2 -I/root/chipyard/si
ms/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig    -D__STDC_FORMAT_MACROS -DTEST_HARNESS=VTestHarness -DVERILATOR -inc
lude /root/chipyard/sims/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig/chipyard.TestHarness.SodorUCodeConfig.plusArgs -
include /root/chipyard/sims/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig/verilator.h -include /root/chipyard/sims/veri
lator/generated-src/chipyard.TestHarness.SodorUCodeConfig/chipyard.TestHarness.SodorUCodeConfig/VTestHarness.h  -DVL_THREADED -std=gnu+
+14  -c -o VTestHarness__Syms.o VTestHarness__Syms.cpp
In file included from /usr/local/share/verilator/include/verilated.h:27:0,
                 from /usr/local/share/verilator/include/verilated_vcd_c.h:23,
                 from /root/chipyard/sims/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig/verilator.h:4,
                 from <command-line>:0:
/usr/local/share/verilator/include/verilatedos.h:252:0: warning: "__STDC_FORMAT_MACROS" redefined
 #define __STDC_FORMAT_MACROS

<command-line>:0:0: note: this is the location of the previous definition
ar -cr VTestHarness__ALL.a VTestHarness.o VTestHarness__1.o VTestHarness__2.o VTestHarness__3.o VTestHarness__4.o VTestHarness__5.o VTe
stHarness__6.o VTestHarness__7.o VTestHarness__8.o VTestHarness__9.o VTestHarness__10.o VTestHarness__11.o VTestHarness__12.o VTestHarn
ess__13.o VTestHarness__14.o VTestHarness__15.o VTestHarness__16.o VTestHarness__17.o VTestHarness__18.o VTestHarness__19.o VTestHarnes
s__20.o VTestHarness__21.o VTestHarness__22.o VTestHarness__23.o VTestHarness__24.o VTestHarness__25.o VTestHarness__26.o VTestHarness_
_27.o VTestHarness__28.o VTestHarness__29.o VTestHarness__30.o VTestHarness__31.o VTestHarness__32.o VTestHarness__33.o VTestHarness__3
4.o VTestHarness__35.o VTestHarness__36.o VTestHarness__37.o VTestHarness__38.o VTestHarness__39.o VTestHarness__40.o VTestHarness__41.
o VTestHarness___024unit.o VTestHarness__Slow.o VTestHarness__1__Slow.o VTestHarness__2__Slow.o VTestHarness__3__Slow.o VTestHarness__4
__Slow.o VTestHarness__5__Slow.o VTestHarness__6__Slow.o VTestHarness__7__Slow.o VTestHarness__8__Slow.o VTestHarness__9__Slow.o VTestH
arness__10__Slow.o VTestHarness___024unit__Slow.o VTestHarness__Dpi.o VTestHarness__Syms.o
ranlib VTestHarness__ALL.a
g++    SimDRAM.o SimDTM.o SimJTAG.o SimSerial.o SimUART.o emulator.o mm.o mm_dramsim2.o remote_bitbang.o testchip_tsi.o uart.o verilate
d.o verilated_dpi.o verilated_vpi.o VTestHarness__ALL.a   -L/root/chipyard/riscv-tools-install/lib -Wl,-rpath,/root/chipyard/riscv-tool
s-install/lib -L/root/chipyard/sims/verilator -L/root/chipyard/tools/DRAMSim2 -lfesvr -ldramsim  -pthread -lpthread -latomic -o /root/c
hipyard/sims/verilator/simulator-chipyard-SodorUCodeConfig
make[1]: Leaving directory '/root/chipyard/sims/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig/chipyard.TestHarness.Sodo
rUCodeConfig'
root@c21295631886:~/chipyard/sims/verilator# ls
Makefile        simulator-chipyard-MediumBoomConfig   simulator-chipyard-Sodor2StageConfig  simulator-chipyard-SodorUCodeConfig
generated-src   simulator-chipyard-RocketConfig        simulator-chipyard-Sodor3StageConfig
output          simulator-chipyard-Sodor1StageConfig  simulator-chipyard-Sodor5StageConfig
root@c21295631886:~/chipyard/sims/verilator# ls
Makefile        simulator-chipyard-MediumBoomConfig   simulator-chipyard-Sodor2StageConfig  simulator-chipyard-SodorUCodeConfig
generated-src   simulator-chipyard-RocketConfig        simulator-chipyard-Sodor3StageConfig
output          simulator-chipyard-Sodor1StageConfig  simulator-chipyard-Sodor5StageConfig
root@c21295631886:~/chipyard/sims/verilator#
```

Similarly for generated-src.

```
root@c21295631886: ~/chipyard/sims/verilator/generated-src
In file included from /usr/local/share/verilator/include/verilated.h:27:0,
                 from /usr/local/share/verilator/include/verilated_vcd_c.h:23,
                 from /root/chipyard/sims/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig/verilator.h:4,
                 from <command-line>:0:
/usr/local/share/verilator/include/verilatedos.h:252:0: warning: "__STDC_FORMAT_MACROS" redefined
 #define __STDC_FORMAT_MACROS

<command-line>:0:0: note: this is the location of the previous definition
ar -cr VTestHarness__ALL.a VTestHarness.o VTestHarness__1.o VTestHarness__2.o VTestHarness__3.o VTestHarness__4.o VTestHarness__5.o VTe
stHarness__6.o VTestHarness__7.o VTestHarness__8.o VTestHarness__9.o VTestHarness__10.o VTestHarness__11.o VTestHarness__12.o VTestHarn
ess__13.o VTestHarness__14.o VTestHarness__15.o VTestHarness__16.o VTestHarness__17.o VTestHarness__18.o VTestHarness__19.o VTestHarnes
s__20.o VTestHarness__21.o VTestHarness__22.o VTestHarness__23.o VTestHarness__24.o VTestHarness__25.o VTestHarness__26.o VTestHarness_
_27.o VTestHarness__28.o VTestHarness__29.o VTestHarness__30.o VTestHarness__31.o VTestHarness__32.o VTestHarness__33.o VTestHarness__3
4.o VTestHarness__35.o VTestHarness__36.o VTestHarness__37.o VTestHarness__38.o VTestHarness__39.o VTestHarness__40.o VTestHarness__41.
o VTestHarness___024unit.o VTestHarness__Slow.o VTestHarness__1__Slow.o VTestHarness__2__Slow.o VTestHarness__3__Slow.o VTestHarness__4
__Slow.o VTestHarness__5__Slow.o VTestHarness__6__Slow.o VTestHarness__7__Slow.o VTestHarness__8__Slow.o VTestHarness__9__Slow.o VTestH
arness__10__Slow.o VTestHarness___024unit__Slow.o VTestHarness__Dpi.o VTestHarness__Syms.o
ranlib VTestHarness__ALL.a
g++    SimDRAM.o SimDTM.o SimJTAG.o SimSerial.o SimUART.o emulator.o mm.o mm_dramsim2.o remote_bitbang.o testchip_tsi.o uart.o verilate
d.o verilated_dpi.o verilated_vpi.o VTestHarness__ALL.a   -L/root/chipyard/riscv-tools-install/lib -Wl,-rpath,/root/chipyard/riscv-tool
s-install/lib -L/root/chipyard/sims/verilator -L/root/chipyard/tools/DRAMSim2 -lfesvr -ldramsim  -pthread -lpthread -latomic -o /root/c
hipyard/sims/verilator/simulator-chipyard-SodorUCodeConfig
make[1]: Leaving directory '/root/chipyard/sims/verilator/generated-src/chipyard.TestHarness.SodorUCodeConfig/chipyard.TestHarness.Sodo
rUCodeConfig'
root@c21295631886:~/chipyard/sims/verilator# ls
Makefile        simulator-chipyard-MediumBoomConfig   simulator-chipyard-Sodor2StageConfig  simulator-chipyard-SodorUCodeConfig
generated-src   simulator-chipyard-RocketConfig        simulator-chipyard-Sodor3StageConfig
output          simulator-chipyard-Sodor1StageConfig  simulator-chipyard-Sodor5StageConfig
root@c21295631886:~/chipyard/sims/verilator# ls
Makefile        simulator-chipyard-MediumBoomConfig   simulator-chipyard-Sodor2StageConfig  simulator-chipyard-SodorUCodeConfig
generated-src   simulator-chipyard-RocketConfig        simulator-chipyard-Sodor3StageConfig
output          simulator-chipyard-Sodor1StageConfig  simulator-chipyard-Sodor5StageConfig
root@c21295631886:~/chipyard/sims/verilator# cd generated-sr
bash: cd: generated-sr: No such file or directory
root@c21295631886:~/chipyard/sims/verilator# cd generated-src
root@c21295631886:~/chipyard/sims/verilator/generated-src# ls
chipyard.TestHarness.MediumBoomConfig   chipyard.TestHarness.Sodor2StageConfig  chipyard.TestHarness.SodorUCodeConfig
chipyard.TestHarness.RocketConfig       chipyard.TestHarness.Sodor3StageConfig
chipyard.TestHarness.Sodor1StageConfig  chipyard.TestHarness.Sodor5StageConfig
root@c21295631886:~/chipyard/sims/verilator/generated-src#
```

3. To use different processors and check the working: For this, the command used is
make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Here the variable BMARKS is set to the path **chipyard/generated/riscv-sodor/riscv-bmarks** this directory contains the benchmark code of riscv that can be used for testing purposes. By running this command there are two files created in the output directory under the processor used, by the names of **towers.out and towers,log** which contain the instructions and the log of the command respectively.
The output of the above command is the number of cycles (mcycle) and the number of instructions (minstret) as shown below.

```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/towers.log)
mcycle = 6166
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use 2 stage processor:
make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/towers.log)
mcycle = 6166
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/towers.log)
mcycle = 6582
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use 3 stage processor:

make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/towers.log)
mcycle = 6166
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/towers.log)
mcycle = 6582
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/towers.log)
mcycle = 7414
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use 5 stage processor:

make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/towers.log)
mcycle = 6166
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/towers.log)
mcycle = 6582
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/towers.log)
mcycle = 7414
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/towers.log)
mcycle = 7000
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use UCode processor:

make CONFIG=SodorUCodeeConfig run-binary BINARY=/root/${BMARKS}/towers.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/towers.log)
mcycle = 6582
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/towers.log)
mcycle = 7414
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/towers.log)
mcycle = 7000
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=SodorUCodeConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-SodorUCodeConfig +permissive +dramsim +dramsim_ini_dir=/root/chip
yard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/ri
scv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeCo
nfig/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig/towers.log)
mcycle = 45051
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Using Another benchmark dhrystone.riscv

Running command to use 1 stage processor:

make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/towers.log)
mcycle = 7414
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/towers.log)
mcycle = 7000
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=SodorUCodeConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-SodorUCodeConfig +permissive +dramsim +dramsim_ini_dir=/root/chip
yard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/ri
scv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeCo
nfig/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig/towers.log)
mcycle = 45051
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 449
Dhrystones per Second:                      2227
mcycle = 224524
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use 2 stage processor:

make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/towers.log)
mcycle = 7000
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=SodorUCodeConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
mkdir -p /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-SodorUCodeConfig +permissive +dramsim +dramsim_ini_dir=/root/chip
yard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/ri
scv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeCo
nfig/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig/towers.log)
mcycle = 45051
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 449
Dhrystones per Second:                      2227
mcycle = 224524
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 515
Dhrystones per Second:                      1941
mcycle = 257527
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use 3 stage processor:

make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
yard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/ri
scv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeCo
nfig/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig/towers.log)
mcycle = 45051
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 449
Dhrystones per Second:                      2227
mcycle = 224524
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 515
Dhrystones per Second:                      1941
mcycle = 257527
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 655
Dhrystones per Second:                      1526
mcycle = 327533
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use 5-stage processor:

make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv

```
root@c21295631886: ~/chipyard/sims/verilator
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 449
Dhrystones per Second:                      2227
mcycle = 224524
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor2StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor2StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 515
Dhrystones per Second:                      1941
mcycle = 257527
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 655
Dhrystones per Second:                      1526
mcycle = 327533
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 589
Dhrystones per Second:                      1697
mcycle = 294534
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

Running command to use UCode processor:

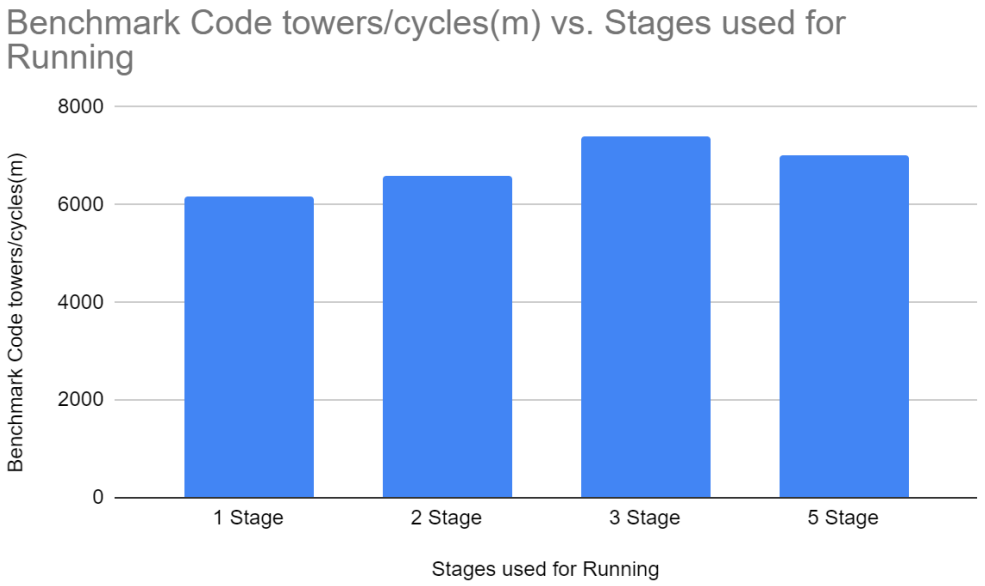
make CONFIG=SodorUCodeConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv

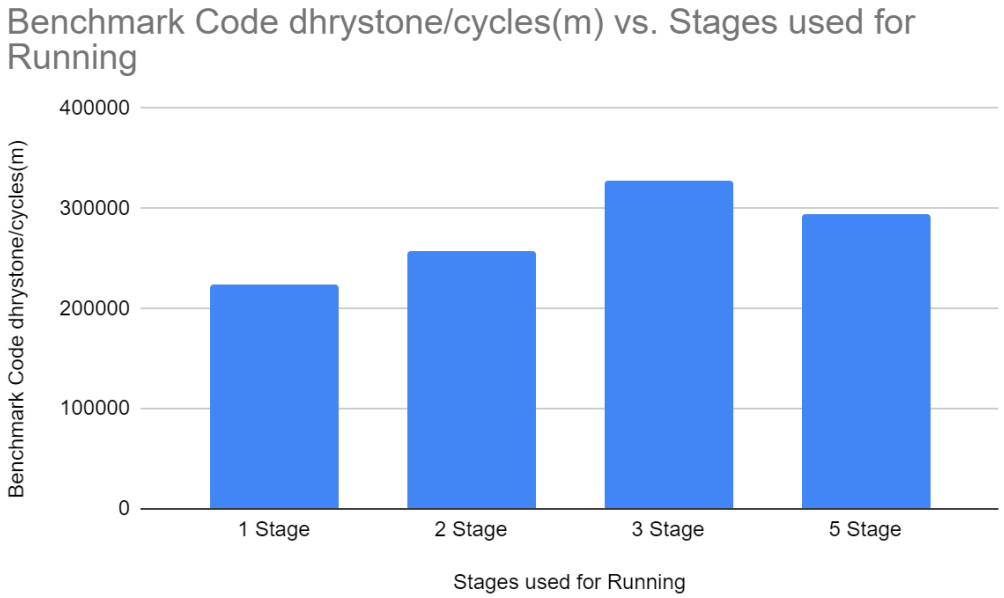
```
root@c21295631886: ~/chipyard/sims/verilator
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor2StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 515
Dhrystones per Second:                      1941
mcycle = 257527
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor3StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 655
Dhrystones per Second:                      1526
mcycle = 327533
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/r
iscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5St
ageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 589
Dhrystones per Second:                      1697
mcycle = 294534
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator# make CONFIG=SodorUCodeConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-SodorUCodeConfig +permissive +dramsim +dramsim_ini_dir=/root/chip
yard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard/generators/ri
scv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCod
eConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.SodorUCodeConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 3314
Dhrystones per Second:                      301
mcycle = 1657174
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@c21295631886:~/chipyard/sims/verilator#
```

After running the commands the output is summarized in the below-given table.

The given table summarises and compares the output which is the number of cycles and instructions count run on two different benchmarks application using the different stages of a processor. Also, graphs are drawn to easily visualize the comparison of different stages on the benchmark application.

S. No.	Stages used for Running	Benchmark Code towers		Benchmark Code dhrystone	
		cycles(m)	instructions(m)	cycles(m)	instructions(m)
1.	1 Stage	6166	6172	224524	224530
2.	2 Stage	6582	6172	257527	224530
3.	3 Stage	7414	6172	327533	224530
4.	5 Stage	7000	6172	294534	224530





By viewing the graphs we make some observations: The number of cycles for both the benchmark application increases till 3 stage and then decreases as stages increase to 5. All of these outputs are provided in the zip file under **part3**.

4. Tracing different instruction mixes for different processor stages and benchmark applications: For this number of stages is fixed to 3 that is **Sodor3StageConfig** is used and the benchmarks used are **dhrystone.riscv, median.riscv, multiply.riscv, qsort.riscv, rsort.riscv, towers.riscv, and vvadd.riscv**.
The command used is the same as above to create the .out and .log files which is
make CONFIG=Sodor3StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv after running the command there would be files created in **output/chipyard.TestHarness.Sodor3StageConfig** and we can use the tracer.py file which is in **chipyard/generators/rsicv-sodor/scripts/** and which is used to give the output in a much more understandable form that is it gives the different instruction count with CPI, IPC, etc.

The folder **output/chipyard.TestHarness.Sodor3StageConfig** and output files.

```
root@c21295631886: ~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig
chipyard.TestHarness.Sodor3StageConfig.memmap.json
chipyard.TestHarness.Sodor3StageConfig.plusArgs
chipyard.TestHarness.Sodor3StageConfig.rom.conf
chipyard.TestHarness.Sodor3StageConfig.top.anno.json
chipyard.TestHarness.Sodor3StageConfig.top.fir
chipyard.TestHarness.Sodor3StageConfig.top.mems.conf
chipyard.TestHarness.Sodor3StageConfig.top.mems.v
chipyard.TestHarness.Sodor3StageConfig.top.v
emulator.cc
firrtl_black_box_resource_files.harness.f
firrtl_black_box_resource_files.top.f
mm.cc
mm.h
mm_dramsim2.cc
mm_dramsim2.h
plusarg_reader.v
remote_bitbang.cc
remote_bitbang.h
sim_files.common.f
sim_files.f
testchip_tsi.cc
testchip_tsi.h
uart.cc
uart.h
verilator.h
root@c21295631886:~/chipyard/sims/verilator/generated-src/chipyard.TestHarness.Sodor3StageConfig# cd ..
root@c21295631886:~/chipyard/sims/verilator/generated-src# ls
chipyard.TestHarness.MediumBoomConfig   chipyard.TestHarness.Sodor2StageConfig  chipyard.TestHarness.SodorUCodeConfig
chipyard.TestHarness.RocketConfig       chipyard.TestHarness.Sodor3StageConfig
chipyard.TestHarness.Sodor1StageConfig  chipyard.TestHarness.Sodor5StageConfig
root@c21295631886:~/chipyard/sims/verilator/generated-src# cd ..
root@c21295631886:~/chipyard/sims/verilator# cd output
root@c21295631886:~/chipyard/sims/verilator/output# ls
chipyard.TestHarness.RocketConfig       chipyard.TestHarness.Sodor2StageConfig  chipyard.TestHarness.Sodor5StageConfig
chipyard.TestHarness.Sodor1StageConfig  chipyard.TestHarness.Sodor3StageConfig  chipyard.TestHarness.SodorUCodeConfig
root@c21295631886:~/chipyard/sims/verilator/output# cd  chipyard.TestHarness.Sodor3StageConfig
root@c21295631886:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig# ls
dhrystone.log  median,.log  median.log  multiply.log  qsort.log  rsort.log  towers.log  vvadd.log
dhrystone.out  median,.out  median.out  multiply.out  qsort.out  rsort.out  towers.out  vvadd.out
root@c21295631886:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig#
```

The output of towers.out using the command cat towers.out. The file towers.out contain the cycles and the respective instruction run in those cycles. This is very hard to read and thus the script tracer.py is used to get a better readable output.

Using **less output/chipyard.TestHarness.Sodor3StageConfig/towers.out**

```
root@c21295631886: ~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig
Cyc=      25817 [1] pc=[80001a90] W[r 0=80001a94][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[0000006f]     j       pc + 0x0
Cyc=      25818 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r 0][00001000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=      25819 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r31][fffff000] Op2=[r31][80001a98] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25820 [0] pc=[80001a90] W[r 0=80001a90][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25821 [1] pc=[80001a90] W[r 0=80001a94][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[0000006f]     j       pc + 0x0
Cyc=      25822 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r 0][00001000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=      25823 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r31][fffff000] Op2=[r31][80001a98] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25824 [0] pc=[80001a90] W[r 0=80001a90][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25825 [1] pc=[80001a90] W[r 0=80001a94][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[0000006f]     j       pc + 0x0
Cyc=      25826 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r 0][00001000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=      25827 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r31][fffff000] Op2=[r31][80001a98] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25828 [0] pc=[80001a90] W[r 0=80001a90][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25829 [1] pc=[80001a90] W[r 0=80001a94][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[0000006f]     j       pc + 0x0
Cyc=      25830 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r 0][00001000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=      25831 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r31][fffff000] Op2=[r31][80001a98] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25832 [0] pc=[80001a90] W[r 0=80001a90][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25833 [1] pc=[80001a90] W[r 0=80001a94][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[0000006f]     j       pc + 0x0
Cyc=      25834 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r 0][00001000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=      25835 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r31][fffff000] Op2=[r31][80001a98] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25836 [0] pc=[80001a90] W[r 0=80001a90][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  J  xor     zero, zero, ze
ro
Cyc=      25837 [1] pc=[80001a90] W[r 0=80001a94][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[0000006f]     j       pc + 0x0
Cyc=      25838 [0] pc=[80001a90] W[r 0=80001a98][0] Op1=[r 0][00001000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
*** PASSED *** Completed after 114925 cycles
root@c21295631886:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor3StageConfig#
```

Running the script tracer.py using the command **../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConfig/dhrystone.out**

And similarly using other benchmarks.

```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor1StageConf
ig/towers.out

Stats:

CPI          : 1.000
IPC          : 1.000
Cycles       : 19591
Instructions : 19592
Bubbles      : 0

Instruction Breakdown:
% Arithmetic  : 41.716 %
% Ld/St       : 42.180 %
% Branch/Jump : 15.389 %
% Misc.       : 0.715 %

root@c21295631886:~/chipyard/sims/verilator#
```

Similarly for other benchmarks.

```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor1StageConf
ig/towers.out

Stats:

CPI          : 1.000
IPC          : 1.000
Cycles       : 19591
Instructions : 19592
Bubbles      : 0

Instruction Breakdown:
% Arithmetic  : 41.716 %
% Ld/St       : 42.180 %
% Branch/Jump : 15.389 %
% Misc.       : 0.715 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/towers.out

Stats:

CPI          : 1.362
IPC          : 0.734
Cycles       : 25533
Instructions : 18748
Bubbles      : 6786

Instruction Breakdown:
% Arithmetic  : 42.671 %
% Ld/St       : 41.892 %
% Branch/Jump : 14.690 %
% Misc.       : 0.747 %

root@c21295631886:~/chipyard/sims/verilator#
```

```
root@c21295631886: ~/chipyard/sims/verilator
% Arithmetic  : 41.716 %
% Ld/St       : 42.180 %
% Branch/Jump : 15.389 %
% Misc.       : 0.715 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/towers.out

Stats:

CPI          : 1.362
IPC          : 0.734
Cycles       : 25533
Instructions : 18748
Bubbles      : 6786

Instruction Breakdown:
% Arithmetic  : 42.671 %
% Ld/St       : 41.892 %
% Branch/Jump : 14.690 %
% Misc.       : 0.747 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/dhrystone.out

Stats:

CPI          : 1.473
IPC          : 0.679
Cycles       : 358201
Instructions : 243186
Bubbles      : 115016

Instruction Breakdown:
% Arithmetic  : 40.567 %
% Ld/St       : 35.166 %
% Branch/Jump : 23.719 %
% Misc.       : 0.547 %

root@c21295631886:~/chipyard/sims/verilator#
```

root@c21295631886: ~/chipyard/sims/verilator

```
root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/dhrystone.out

Stats:

CPI          : 1.473
IPC          : 0.679
Cycles       : 358201
Instructions : 243186
Bubbles      : 115016

Instruction Breakdown:
% Arithmetic  : 40.567 %
% Ld/St       : 35.166 %
% Branch/Jump : 23.719 %
% Misc.       : 0.547 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/media.out
usage: tracer.py [-h] [-u] [file]
tracer.py: error: argument file: can't open 'output/chipyard.TestHarness.Sodor3StageConfig/media.out': [Errno 2] No such file or direct
ory: 'output/chipyard.TestHarness.Sodor3StageConfig/media.out'
root@c21295631886:~/chipyard/sims/verilator# clear
root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/median.out

Stats:

CPI          : 1.694
IPC          : 0.590
Cycles       : 27863
Instructions : 16449
Bubbles      : 11415

Instruction Breakdown:
% Arithmetic  : 32.537 %
% Ld/St       : 31.242 %
% Branch/Jump : 35.370 %
% Misc.       : 0.851 %
```

root@c21295631886: ~/chipyard/sims/verilator

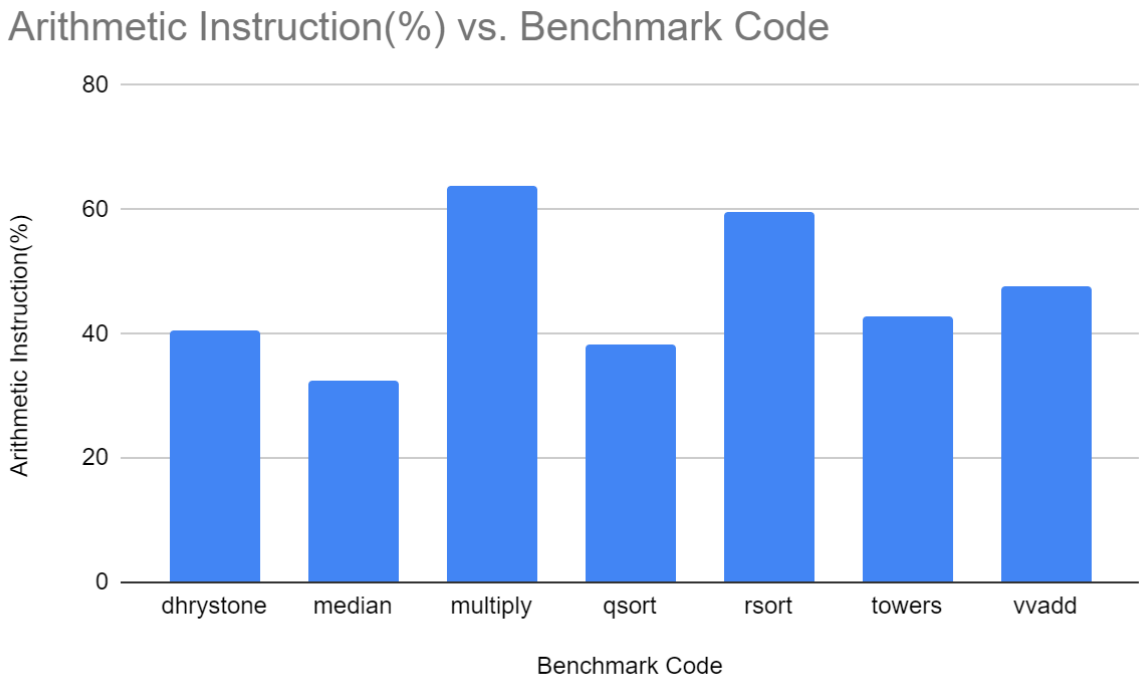
```
root@c21295631886:~/chipyard/sims/verilator# clear
root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/median.out

Stats:

CPI          : 1.694
IPC          : 0.590
Cycles       : 27863
Instructions : 16449
Bubbles      : 11415

Instruction Breakdown:
% Arithmetic  : 32.537 %
% Ld/St       : 31.242 %
% Branch/Jump : 35.370 %
% Misc.       : 0.851 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/multiply.out

Stats:

CPI          : 1.843
IPC          : 0.542
Cycles       : 92053
Instructions : 49935
Bubbles      : 42119

Instruction Breakdown:
% Arithmetic  : 63.879 %
% Ld/St       : 4.153 %
% Branch/Jump : 31.615 %
% Misc.       : 0.352 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/qsort.out

Stats:
```

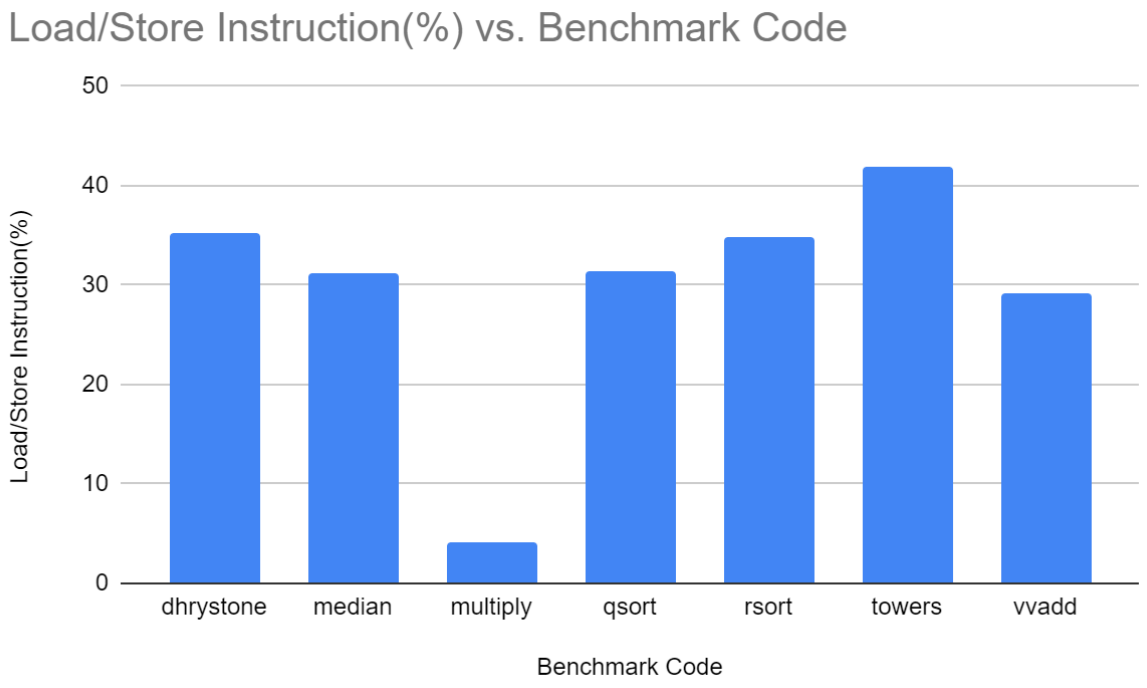
```
root@c21295631886: ~/chipyard/sims/verilator
root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/qsort.out

Stats:

CPI          : 1.617
IPC          : 0.618
Cycles       : 381371
Instructions : 235836
Bubbles      : 145536

Instruction Breakdown:
% Arithmetic  : 38.436 %
% Ld/St       : 31.407 %
% Branch/Jump : 29.834 %
% Misc.       : 0.323 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/rsort.out

Stats:

CPI          : 1.135
IPC          : 0.881
Cycles       : 424873
Instructions : 374409
Bubbles      : 50465

Instruction Breakdown:
% Arithmetic  : 59.668 %
% Ld/St       : 34.842 %
% Branch/Jump : 4.335 %
% Misc.       : 1.155 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/towers.out

Stats:

CPI          : 1.362
```

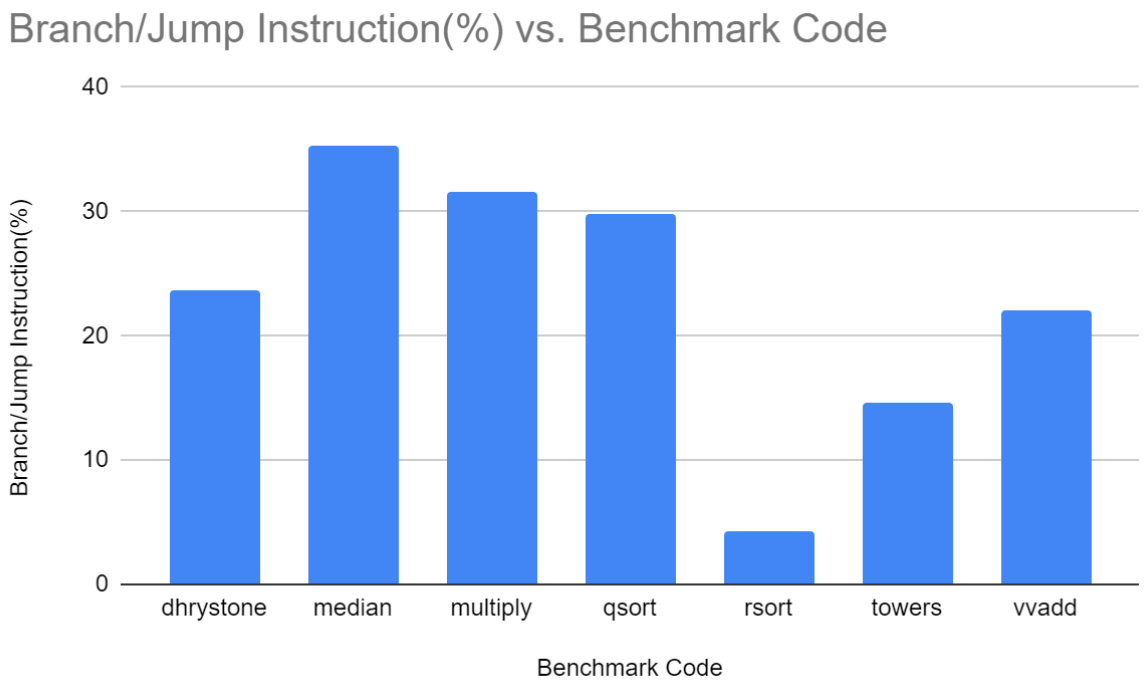
```
root@c21295631886: ~/chipyard/sims/verilator
% Arithmetic  : 59.668 %
% Ld/St       : 34.842 %
% Branch/Jump : 4.335 %
% Misc.       : 1.155 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/towers.out

Stats:

CPI          : 1.362
IPC          : 0.734
Cycles       : 25533
Instructions : 18748
Bubbles      : 6786

Instruction Breakdown:
% Arithmetic  : 42.671 %
% Ld/St       : 41.892 %
% Branch/Jump : 14.690 %
% Misc.       : 0.747 %

root@c21295631886:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor3StageConf
ig/vvadd.out

Stats:

CPI          : 1.516
IPC          : 0.660
Cycles       : 18411
Instructions : 12143
Bubbles      : 6269

Instruction Breakdown:
% Arithmetic  : 47.616 %
% Ld/St       : 29.227 %
% Branch/Jump : 22.004 %
% Misc.       : 1.153 %

root@c21295631886:~/chipyard/sims/verilator#
```

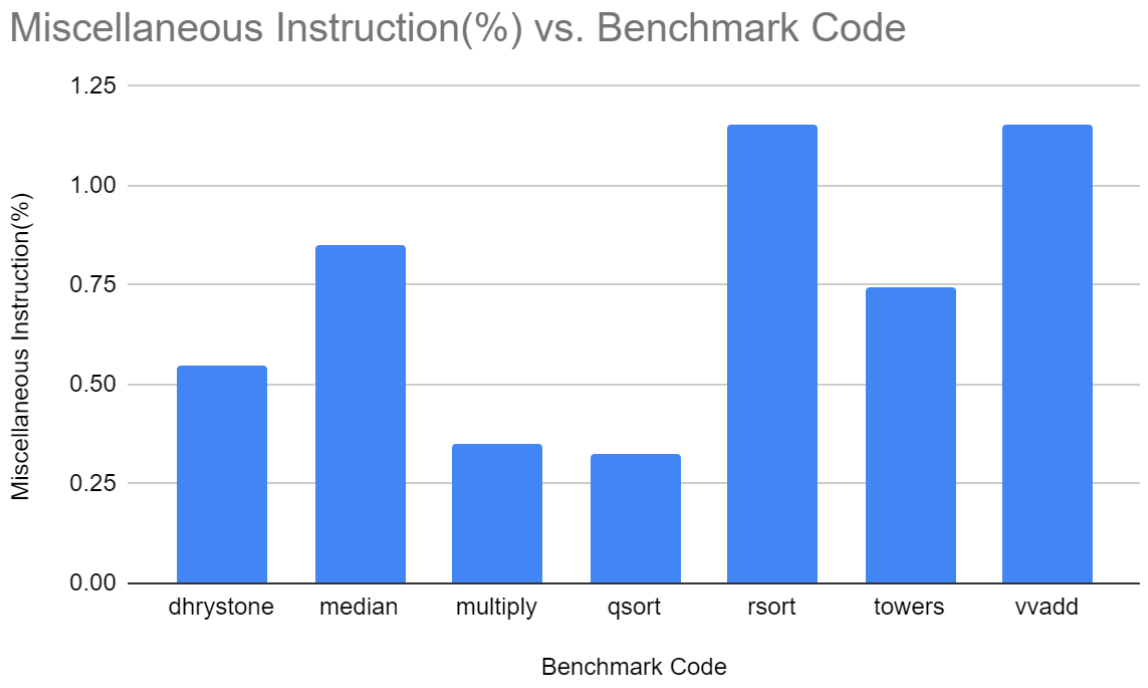
Finally, after running all the benchmarks for 3 stage processor the below table is created to summarise and compare the results of **tracer.py** for the different benchmarks. There are also graphs for different benchmark applications for a single instruction type and also the complete graph of all the instruction types with all benchmarks.

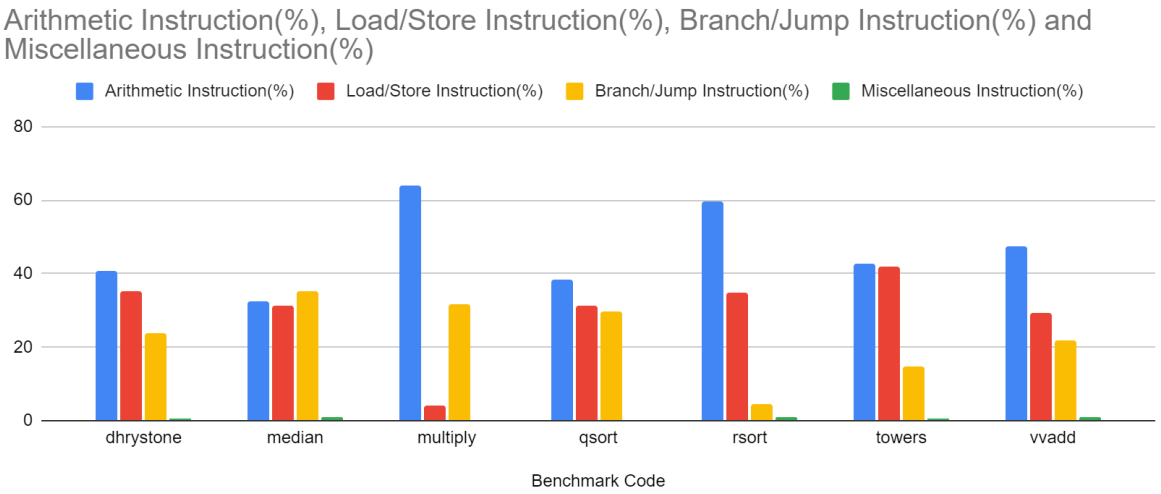
S.No.	Benchmark Code	Arithmetic Instruction(%)	Load/Store Instruction(%)	Branch/Jump Instruction(%)	Miscellaneous Instruction(%)
1.	dhrystone	40.567	35.166	23.719	0.547
2.	median	32.537	31.242	35.370	0.851
3.	multiply	63.879	4.153	31.615	0.352
4.	qsort	38.436	31.407	29.834	0.323
5.	rsort	59.668	34.842	4.335	1.155
6.	towers	42.671	41.892	14.690	0.747
7.	vvadd	47.616	29.227	22.004	1.153




Load/Store Instruction(%) vs. Benchmark Code
Load/Store Instruction(%)
50
40
30
20
10
0
dhrystone
median
multiply
qsort
rsort
towers
vvadd
Benchmark Code


Branch/Jump Instruction(%) vs. Benchmark Code
Branch/Jump Instruction(%)
40
30
20
10
0
dhrystone
median
multiply
qsort
rsort
towers
vvadd
Benchmark Code





From the graphs we can make some observations:

- **Multiply has the highest arithmetic intensity** with the lowest load/store and miscellaneous.
- The **second highest arithmetic intensity is in rsort** with the lowest branch and jumps instructions.
- The **towers has the highest memory load and store**. Thus it is mostly memory bound.
- In the case of **branch and jump, the median has the highest intensity**.

5. CPI Analysis using Five stage processor: For this analysis, we need to go to the directory **chipyard/generators/riscv-sodor/src/main/scala/rv32_5stage** in this directory there is a file **consts.scala**.

We can view the contents of **consts.scala** using the command **cat consts.scala**. Inside the file **USE_FULL_BYPASSING** which is currently set to **true**, set this variable to **false** to use the 5 stage without full bypassing.

First, we run the benchmarks and get their instruction distribution on **USE_FULL_BYPASSING =true**

```
root@a0a0533e3405: ~/chipyard/sims/verilator
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stag
eConfig/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/towers.log)
mcycle = 7000
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/vvadd.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/vvadd.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/vvadd.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/vvadd.log)
mcycle = 3020
minstret = 2418
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/dhrystone.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/dhrystone.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5S
tageConfig/dhrystone.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/dhrystone.log)
Microseconds for one run through Dhrystone: 589
Dhrystones per Second:                      1697
mcycle = 294534
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/median.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/median.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stag
eConfig/median.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/median.log)
mcycle = 6369
minstret = 4155
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator#
root@a0a0533e3405:~/chipyard/sims/verilator#
```

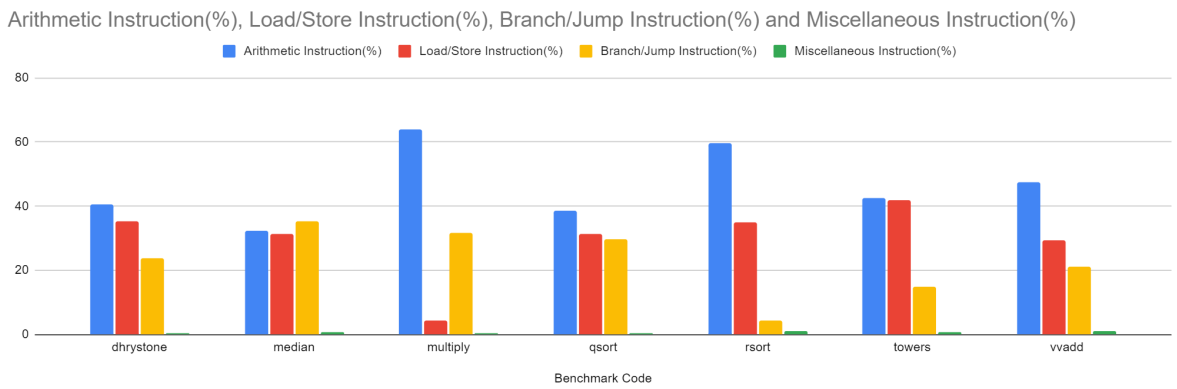
```
root@a0a0533e3405: ~/chipyard/sims/verilator
minstret = 224530
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/median.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/median.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stag
eConfig/median.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/median.log)
mcycle = 6369
minstret = 4155
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator#
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/multiply.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/multiply.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5St
ageConfig/multiply.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/multiply.log)
mcycle = 33136
minstret = 20902
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/qsort.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/qsort.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/qsort.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/qsort.log)
mcycle = 172780
minstret = 123509
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/${BMARKS}/rsort.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root//chipyard/generators/
riscv-sodor/riscv-bmarks/rsort.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/rsort.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/rsort.log)
mcycle = 182392
minstret = 171134
[UART] UART0 is here (stdin/stdout).
root@a0a0533e3405:~/chipyard/sims/verilator#
```

Then check the instruction distribution using **tracer.py.**

```
root@a0a0533e3405: ~/chipyard/sims/verilator
% Arithmetic  : 59.660 %
% Ld/St       : 34.844 %
% Branch/Jump : 4.341 %
% Misc.       : 1.155 %

root@a0a0533e3405:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor5StageConf
ig/towers.out

Stats:

CPI          : 1.250
IPC          : 0.800
Cycles       : 23604
Instructions : 18877
Bubbles      : 4728

Instruction Breakdown:
% Arithmetic  : 42.544 %
% Ld/St       : 41.935 %
% Branch/Jump : 14.780 %
% Misc.       : 0.742 %

root@a0a0533e3405:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor5StageConf
ig/vvadd.out

Stats:

CPI          : 1.351
IPC          : 0.740
Cycles       : 16519
Instructions : 12229
Bubbles      : 4291

Instruction Breakdown:
% Arithmetic  : 47.535 %
% Ld/St       : 29.397 %
% Branch/Jump : 21.923 %
% Misc.       : 1.145 %

root@a0a0533e3405:~/chipyard/sims/verilator#
```

Results are summarised in table and bar chart given below:

S.No.	Benchmark Code	Arithmetic Instruction(%)	Load/Store Instruction(%)	Branch/Jump Instruction(%)	Miscellaneous Instruction(%)
1.	dhrystone	40.503	35.199	23.725	0.546
2.	median	32.471	31.387	35.293	0.848
3.	multiply	63.801	4.227	31.620	0.352
4.	qsort	38.448	31.404	29.826	0.323
5.	rsort	59.664	34.840	4.341	1.155
6.	towers	42.544	41.935	14.780	0.742
7.	vvadd	47.535	29.397	21.293	1.145



After, we set **USE_FULL_BYPASSING = false**.

```
root@c21295631886: ~/chipyard/generators/riscv-sodor/src/main/scala/rv32_5stage
root@c21295631886:~/chipyard/generators/riscv-sodor/src/main/scala/rv32_5stage# vim consts.scala
root@c21295631886:~/chipyard/generators/riscv-sodor/src/main/scala/rv32_5stage# cat consts.scala
//**************************************************************************
// RISCV Processor Constants
//--------------------------------------------------------------------------
//
// Christopher Celio
// 2011 Feb 1

package sodor.stage5
package constants
{

import chisel3._
import chisel3.util._


trait SodorProcConstants
{

   //************************************
   // Machine Parameters
   val USE_FULL_BYPASSING = false  // turn on full bypassing (only stalls
                                   // on load-use). Otherwise rely
                                   // entirely on interlocking to handle
                                   // pipeline hazards.
}

trait ScalarOpConstants
{

   //************************************
   // Control Signals
   val Y        = true.B
   val N        = false.B

   // PC Select Signal
   val PC_4     = 0.asUInt(2.W)  // PC + 4
   val PC_BRJMP = 1.asUInt(2.W)  // brjmp_target
   val PC_JALR  = 2.asUInt(2.W)  // jump_reg_target
```

Running towers.riscv benchmark and vvadd.riscv benchmark.

```
root@903332e32e5f: ~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig
towers.log  towers.out
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# less
Missing filename ("less --help" for help)
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# less towers.out
using random seed 1651471380
This emulator compiled with JTAG Remote Bitbang client. To enable, use +jtag_rbb_enable=1.
Listening on port 44861
Cyc=          0 [0] pc=[00000000] W[r 0=00000000][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          1 [0] pc=[00000000] W[r 0=00000000][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          2 [0] pc=[00000000] W[r 0=00000000][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          3 [0] pc=[00000000] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          4 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          5 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          6 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          7 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          8 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=          9 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         10 [0] pc=[00010040] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         11 [1] pc=[00010040] W[r10=00010040][1] Op1=[r 0][00010040] Op2=[r 0][00000000] inst=[00000517]     auipc   a0, 0x0
Cyc=         12 [0] pc=[00010044] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         13 [0] pc=[00010044] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         14 [0] pc=[00010044] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         15 [0] pc=[00010044] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         16 [0] pc=[00010044] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
```

root@903332e32e5f: ~/chipyard/sims/verilator

```
Cyc=     541612 [0] pc=[00010008] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  B  xor     zero, zero, ze
ro
Cyc=     541613 [0] pc=[00010008] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541614 [1] pc=[00010008] W[r 8=00000008][0] Op1=[r10][00000000] Op2=[r 0][00000008] inst=[00050463]     beqz    a0, pc + 8
Cyc=     541615 [0] pc=[0001000c] W[r 0=00000000][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541616 [0] pc=[0001000c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541617 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541618 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541619 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541620 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541621 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541622 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541623 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541624 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541625 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541626 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541627 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541628 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541629 [0] pc=[00010010] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541630 [1] pc=[00010010] W[r12=02000004][1] Op1=[r11][02000000] Op2=[r 4][00000004] inst=[00458613]     addi    a2, a1, 4
Cyc=     541631 [0] pc=[00010014] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=     541632 [0] pc=[00010014] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r ^C
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# ls
```

root@903332e32e5f: ~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig

```
vvadd.ot: No such file or directory
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# less vvadd.out
Cyc=         268 [1] pc=[0001006c] W[r 0=00000000][0] Op1=[r 0][00000000] Op2=[r 5][00000000] inst=[10500073]     wfi
Cyc=         269 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         270 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         271 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         272 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         273 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         274 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]  B  xor     zero, zero, ze
ro
Cyc=         275 [0] pc=[00010070] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         276 [1] pc=[00010070] W[r 0=00010074][1] Op1=[r31][00000000] Op2=[r29][fffffffc] inst=[ffdff06f]     j       pc - 0x4
Cyc=         277 [0] pc=[00010074] W[r 0=00000000][0] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         278 [0] pc=[00010074] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         279 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         280 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         281 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         282 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         283 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         284 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         285 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         286 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
Cyc=         287 [0] pc=[0001006c] W[r 0=00000000][1] Op1=[r 0][00000000] Op2=[r 0][00000000] inst=[00004033]     xor     zero, zero, ze
ro
```

```
root@903332e32e5f: ~/chipyard/sims/verilator
        jr      ra
        .size   main, .-main
        .ident  "GCC: (GNU) 9.2.0"
root@903332e32e5f:~/chipyard/generators/riscv-sodor/riscv-bmarks# cd ..
root@903332e32e5f:~/chipyard/generators/riscv-sodor# cd ..
root@903332e32e5f:~/chipyard/generators# cd ..
root@903332e32e5f:~/chipyard# cd sims/verilator
root@903332e32e5f:~/chipyard/sims/verilator# ls
Makefile        output                                  simulator-chipyard-Sodor1StageConfig
generated-src   simulator-chipyard-MediumBoomConfig    simulator-chipyard-Sodor5StageConfig
root@903332e32e5f:~/chipyard/sims/verilator# make
make                        make-first-existing-target  makeconv                    makeinfo
root@903332e32e5f:~/chipyard/sims/verilator# make CONFIG=Sodor5StageConfig run-binary BINARY=/root/chipyard/${BMARKS}/vvadd.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor5StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard//generators/
riscv-sodor/riscv-bmarks/vvadd.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5Stage
Config/vvadd.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig/vvadd.log)
[UART] UART0 is here (stdin/stdout).
/root/chipyard/common.mk:175: recipe for target 'run-binary' failed
make: *** [run-binary] Error 2
root@903332e32e5f:~/chipyard/sims/verilator# cd output
root@903332e32e5f:~/chipyard/sims/verilator/output# ls
chipyard.TestHarness.Sodor1StageConfig  chipyard.TestHarness.Sodor5StageConfig
root@903332e32e5f:~/chipyard/sims/verilator/output# chipyard.TestHarness.Sodor5StageConfig/
bash: chipyard.TestHarness.Sodor5StageConfig/: Is a directory
root@903332e32e5f:~/chipyard/sims/verilator/output# ls
chipyard.TestHarness.Sodor1StageConfig  chipyard.TestHarness.Sodor5StageConfig
root@903332e32e5f:~/chipyard/sims/verilator/output# cd chipyard.TestHarness.Sodor5StageConfig/
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# ls
towers.log  towers.out  vvadd.log  vvadd.out
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# less vvadd.ot
vvadd.ot: No such file or directory
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# less vvadd.out
root@903332e32e5f:~/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor5StageConfig# cd ..
root@903332e32e5f:~/chipyard/sims/verilator/output# cd ..
root@903332e32e5f:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor5StageConf
ig/vvadd.out
Trace analyzer found no instructions. Are you passing in the correct trace file?
root@903332e32e5f:~/chipyard/sims/verilator#
```

The CPI and instruction mix are provided in **part5.txt** files and also summarised in the above table. The CPI observed was very much expected with 5 Stage processor.

As we can see that using **interlocking instead of bypassing**, the instruction count increases to a high extent and thus it would not be useful to use interlocking. The tracer.py file was also not able to provide the output.

Part 6: Design your own 5 Stage Processor

In this we need to modify the tracer.py to not use the li instruction for loading.

First, the Benchmarks were run on 1 stage with the original tracer.py.

```
root@551ea79e6215: ~/chipyard/sims/verilator
mcycle = 20896
minstret = 20902
[UART] UART0 is here (stdin/stdout).
root@551ea79e6215:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/chipyard/${BMARKS}/qsort.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard//generators/
riscv-sodor/riscv-bmarks/qsort.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/qsort.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/qsort.log)
mcycle = 123503
minstret = 123509
[UART] UART0 is here (stdin/stdout).
root@551ea79e6215:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/chipyard/${BMARKS}/rsort.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard//generators/
riscv-sodor/riscv-bmarks/rsort.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/rsort.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/rsort.log)
mcycle = 171128
minstret = 171134
[UART] UART0 is here (stdin/stdout).
root@551ea79e6215:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/chipyard/${BMARKS}/towers.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard//generators/
riscv-sodor/riscv-bmarks/towers.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stag
eConfig/towers.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/towers.log)
mcycle = 6166
minstret = 6172
[UART] UART0 is here (stdin/stdout).
root@551ea79e6215:~/chipyard/sims/verilator# make CONFIG=Sodor1StageConfig run-binary BINARY=/root/chipyard/${BMARKS}/vvadd.riscv
Running with RISCV=/root/chipyard/riscv-tools-install
(set -o pipefail &&  /root/chipyard/sims/verilator/simulator-chipyard-Sodor1StageConfig +permissive +dramsim +dramsim_ini_dir=/root/chi
pyard/generators/testchipip/src/main/resources/dramsim2_ini +max-cycles=10000000   +verbose +permissive-off /root/chipyard//generators/
riscv-sodor/riscv-bmarks/vvadd.riscv </dev/null 2> >(spike-dasm > /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1Stage
Config/vvadd.out) | tee /root/chipyard/sims/verilator/output/chipyard.TestHarness.Sodor1StageConfig/vvadd.log)
mcycle = 2412
minstret = 2418
[UART] UART0 is here (stdin/stdout).
root@551ea79e6215:~/chipyard/sims/verilator#
```

Running **tracer.py.**

```
root@551ea79e6215: ~/chipyard/sims/verilator
% Arithmetic  : 59.575 %
% Ld/St       : 34.872 %
% Branch/Jump : 4.401 %
% Misc.       : 1.152 %

root@551ea79e6215:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor1StageConf
ig/towers.out

Stats:

CPI          : 1.000
IPC          : 1.000
Cycles       : 19591
Instructions : 19592
Bubbles      : 0

Instruction Breakdown:
% Arithmetic  : 41.716 %
% Ld/St       : 42.180 %
% Branch/Jump : 15.389 %
% Misc.       : 0.715 %

root@551ea79e6215:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor1StageConf
ig/vvadd.out

Stats:

CPI          : 1.000
IPC          : 1.000
Cycles       : 12946
Instructions : 12947
Bubbles      : 0

Instruction Breakdown:
% Arithmetic  : 45.887 %
% Ld/St       : 30.548 %
% Branch/Jump : 22.484 %
% Misc.       : 1.081 %

root@551ea79e6215:~/chipyard/sims/verilator#
```

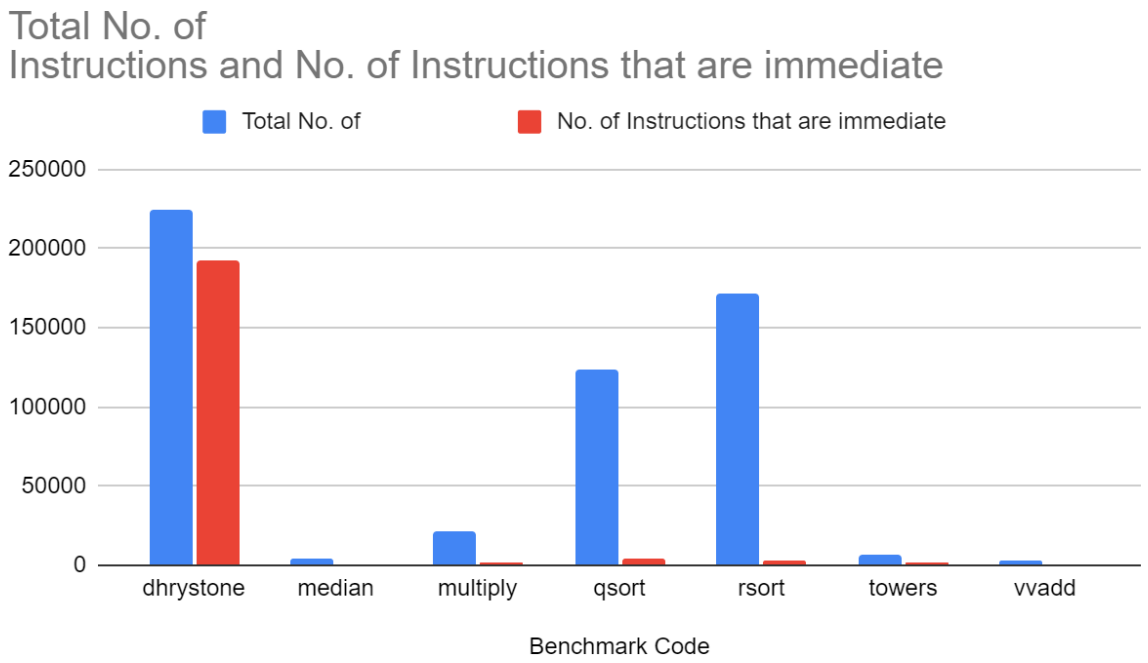
Now Modifying the scripts to see what changes are there in instruction count when we modify the processor so that no immediate load and store instruction can work. For this modification, we have not calculated the instructions that are immediate load and stores.

Running Modified script

```
root@551ea79e6215: ~/chipyard/sims/verilator
% Arithmetic  : 59.556 %
% Ld/St       : 34.872 %
% Branch/Jump : 4.401 %
% Misc.       : 1.171 %

root@551ea79e6215:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor1StageConf
ig/towers.out

Stats:

CPI          : 1.000
IPC          : 1.000
Cycles       : 19591
Instructions : 19592
Bubbles      : 0

Instruction Breakdown:
% Arithmetic  : 41.486 %
% Ld/St       : 42.180 %
% Branch/Jump : 15.389 %
% Misc.       : 0.944 %

root@551ea79e6215:~/chipyard/sims/verilator# ../../generators/riscv-sodor/scripts/tracer.py output/chipyard.TestHarness.Sodor1StageConf
ig/vvadd.out

Stats:

CPI          : 1.000
IPC          : 1.000
Cycles       : 12946
Instructions : 12947
Bubbles      : 0

Instruction Breakdown:
% Arithmetic  : 45.540 %
% Ld/St       : 30.548 %
% Branch/Jump : 22.484 %
% Misc.       : 1.429 %

root@551ea79e6215:~/chipyard/sims/verilator#
```

The given below tables summarises the result.

S. No.	Benchmark Code	Arithmetic Instruction (%)	Arithmetic Instruction Without Immediate (%)	Difference between percentages (%)	Total No. of Instructions	No. of Instructions that are immediate
1.	dhrystone	40.381	39.523	0.858	224530	192646
2.	median	31.843	31.581	0.262	4155	1088
3.	multiply	63.157	63.044	0.113	20902	2361
4.	qsort	38.379	38.349	0.03	123509	3705
5.	rsort	59.575	59.556	0.019	171134	3251
6.	towers	41.716	41.486	0.23	6172	1419
7.	vvadd	45.887	45.540	0.347	2418	839



All the files are provided in zip. The file includes **part6.txt** which contains the output of the commands and instructions which contain modified code for the instructions which are not using load and stores using immediate.

Load and store using immediate are calculated under arithmetic and when we remove them the no. of arithmetic instructions decreases refer the columns 2 and 3 of the above table. But at the same time load and store increase.
As we can see if we do not use the immediate then the number of arithmetic instructions decreases but at the same time the load and store would increase that does not use immediate the number by which load increases are given in the table above the last column, now since the time for load is much higher than arithmetic since it uses all stages, it would be better to use the previous design.
But in case the time taken for load is reduced by some means then the new design can also be used.