MidSem Project
Warp State Analysis

Team members:

Ishan Mardani (21CS01023)

Akshit Dudeja (21CS01026)

Tushar Joshi(21CS01078)

Vishnu Tirth Bysani (21CS01077)

Joshua Dias Barreto (21CS01075)

Warp States

When a kenel executes on an SM, warps of the kenel can be in different states.

We classify the warps depending on their state of execution in a given cycle:

Issued:

Warps that issue an instruction to the execution pipeline are accounted here. It indicates the IPC of the SM and a high number of warps in this state indicate good performance

Waiting:

Warps waiting for an instruction to commit so that further dependent instructions can be issued to the pipeline are in this category.

Excess ALU (XALU):

Warps that are ready for execution of arithmetic operations, but cannot execute due to unavailability of resources are in this category. These are ready to execute warps and cannot issue because the scheduler can only issue a fixed number of instructions per cycle. Xalu indicates the excess warps ready for arithmetic execution.

Excess Memory (XMEM):

Warps that are ready to send an instruction to the Load/Store pipeline but are restricted are accounted here. These warps are restricted if the pipeline is stalled due to back pressure from memory or if the maximum number of instructions that can be issued to this pipeline have been issued. Xmem

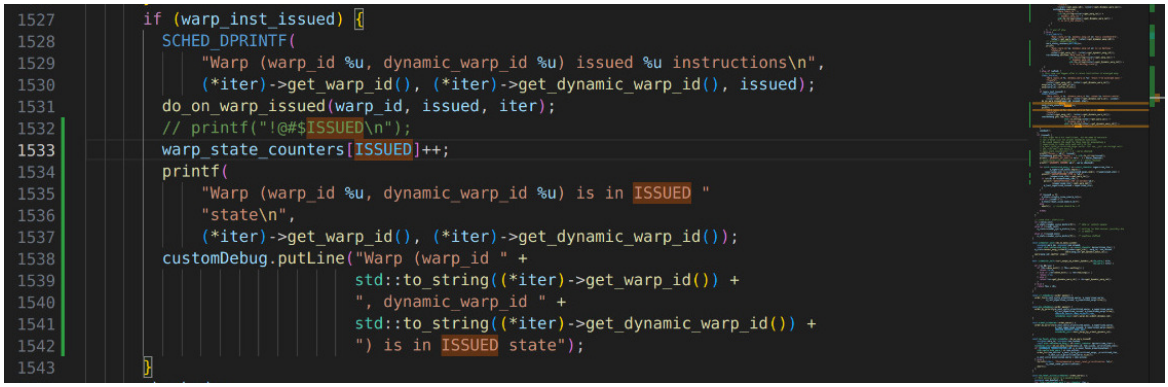
warps represents the excess warps that will increase the pressure on the memory subsystem from the current SM.

Other:

Warps waiting on a synchronisation instruction or warps that do not have their instructions in the instruction buffer are called Others. As there is no for these warps, their requirements is unknown.

Modifications in Codebase

Issued State Counter:



```
1527    if (warp_inst_issued) {
1528      SCHED_DPRINTF(
1529          "Warp (warp_id %u, dynamic_warp_id %u) issued %u instructions\n",
1530          (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id(), issued);
1531      do_on_warp_issued(warp_id, issued, iter);
1532      // printf("!@#$ISSUED\n");
1533      warp_state_counters[ISSUED]++;
1534      printf(
1535          "Warp (warp_id %u, dynamic_warp_id %u) is in ISSUED "
1536          "state\n",
1537          (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1538      customDebug.putLine("Warp (warp_id " +
1539                          std::to_string((*iter)->get_warp_id()) +
1540                          ", dynamic_warp_id " +
1541                          std::to_string((*iter)->get_dynamic_warp_id()) +
1542                          ") is in ISSUED state");
1543    }
```

Waiting State Counter:

```
1235        if (pI) {
1236          assert(valid);
1237          if (pc != pI->pc) {
1238            SCHED_DPRINTF(
1239                "Warp (warp_id %u, dynamic_warp_id %u) control hazard "
1240                "instruction flush\n",
1241                (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1242            // control hazard
1243            warp(warp_id).set_next_pc(pc);
1244            warp(warp_id).ibuffer_flush();
1245            warp_state_counters[WAITING]++;
1246            printf(
1247                "Warp (warp_id %u, dynamic_warp_id %u) is in Waiting "
1248                "state\n",
1249                (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1250            customDebug.putLine("Warp (warp_id " +
1251                                std::to_string((*iter)->get_warp_id()) +
1252                                ", dynamic_warp_id " +
1253                                std::to_string((*iter)->get_dynamic_warp_id()) +
1254                                ") is in Waiting state");
1255          } else {
```

```
1256            valid_inst = true;
1257            if (!m_scoreboard->checkCollision(warp_id, pI)) {
1258              SCHED_DPRINTF(
1259                  "Warp (warp_id %u, dynamic_warp_id %u) passes scoreboard\n",
1260                  (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1261              ready_inst = true;
1262
1263              const active_mask_t &active_mask =
1264                  m_shader->get_active_mask(warp_id, pI);
1265
1266              assert(warp(warp_id).inst_in_pipeline());
1267
1268              if ((pI->op == LOAD_OP) || (pI->op == STORE_OP) ||
1269                  (pI->op == MEMORY_BARRIER_OP) ||
1270                  (pI->op == TENSOR_CORE_LOAD_OP) ||
1271                  (pI->op == TENSOR_CORE_STORE_OP)) {
1272                if (m_mem_out->has_free(m_shader->m_config->sub_core_model,
```

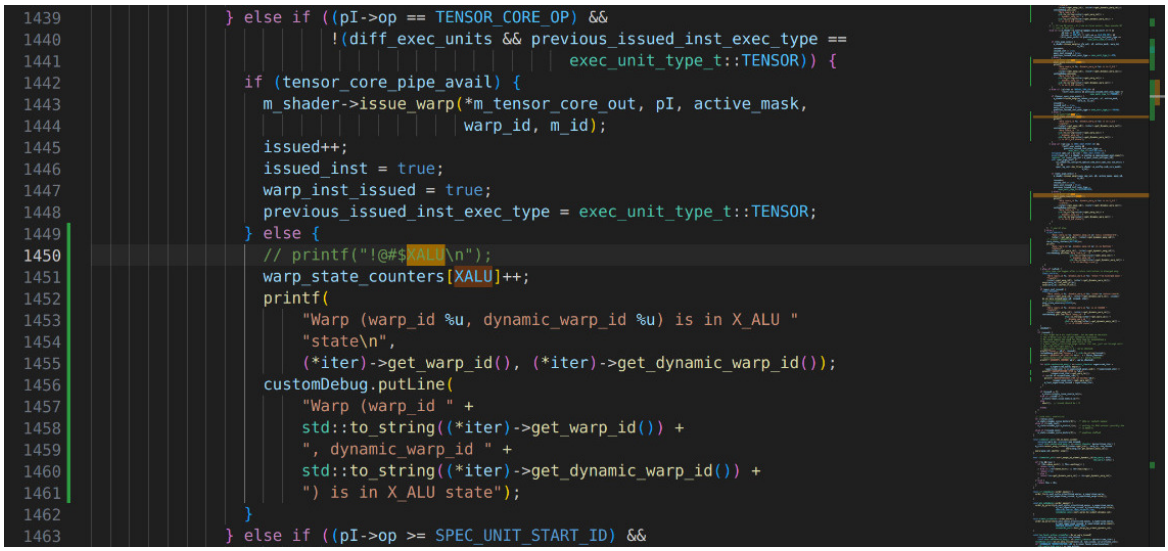
• • •

```
1500            }  // end of else
1501          } else {
1502            SCHED_DPRINTF(
1503                "Warp (warp_id %u, dynamic_warp_id %u) fails scoreboard\n",
1504                (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1505            // printf("!@#$WAITING\n");
1506            warp_state_counters[WAITING]++;
1507            printf(
1508                "Warp (warp_id %u, dynamic_warp_id %u) is in Waiting "
1509                "state\n",
1510                (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1511            customDebug.putLine("Warp (warp_id " +
1512                                std::to_string((*iter)->get_warp_id()) +
1513                                ", dynamic_warp_id " +
1514                                std::to_string((*iter)->get_dynamic_warp_id()) +
1515                                ") is in Waiting state");
1516          }
```

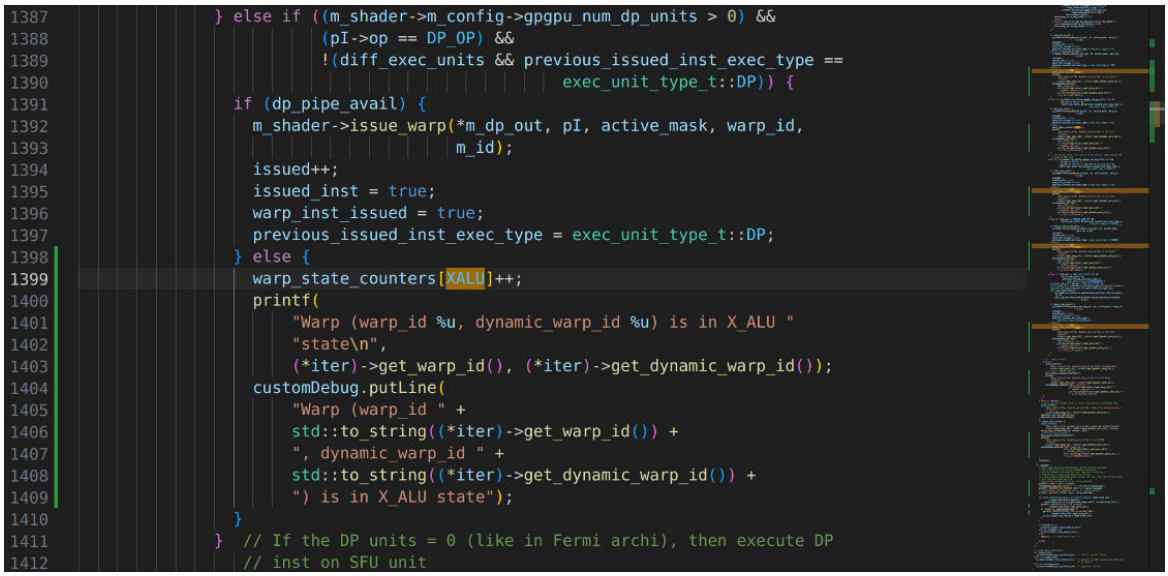
XALU State Counter:

```
1359            if (execute_on_SP) {
1360              m_shader->issue_warp(*m_sp_out, pI, active_mask, warp_id,
1361                                   m_id);
1362              issued++;
1363              issued_inst = true;
1364              warp_inst_issued = true;
1365              previous_issued_inst_exec_type = exec_unit_type_t::SP;
1366            } else if (execute_on_INT) {
1367              m_shader->issue_warp(*m_int_out, pI, active_mask, warp_id,
1368                                   m_id);
1369              issued++;
1370              issued_inst = true;
1371              warp_inst_issued = true;
1372              previous_issued_inst_exec_type = exec_unit_type_t::INT;
1373            } else {
1374              // printf("!@#$XALU\n");
1375              warp_state_counters[XALU]++;
1376              printf(
1377                  "Warp (warp_id %u, dynamic_warp_id %u) is in X_ALU "
1378                  "state\n",
1379                  (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1380              customDebug.putLine(
1381                  "Warp (warp_id " +
1382                  std::to_string((*iter)->get_warp_id()) +
1383                  ", dynamic_warp_id " +
1384                  std::to_string((*iter)->get_dynamic_warp_id()) +
1385                  ") is in X_ALU state");
1386            }
```

```
1476            if (spec_pipe_avail) {
1477              m_shader->issue_warp(*spec_reg_set, pI, active_mask, warp_id,
1478                                   m_id);
1479              issued++;
1480              issued_inst = true;
1481              warp_inst_issued = true;
1482              previous_issued_inst_exec_type =
1483                  exec_unit_type_t::SPECIALIZED;
1484            } else {
1485              // printf("!@#$XALU\n");
1486              warp_state_counters[XALU]++;
1487              printf(
1488                  "Warp (warp_id %u, dynamic_warp_id %u) is in X_ALU "
1489                  "state\n",
1490                  (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1491              customDebug.putLine(
1492                  "Warp (warp_id " +
1493                  std::to_string((*iter)->get_warp_id()) +
1494                  ", dynamic_warp_id " +
1495                  std::to_string((*iter)->get_dynamic_warp_id()) +
1496                  ") is in X_ALU state");
1497            }
```

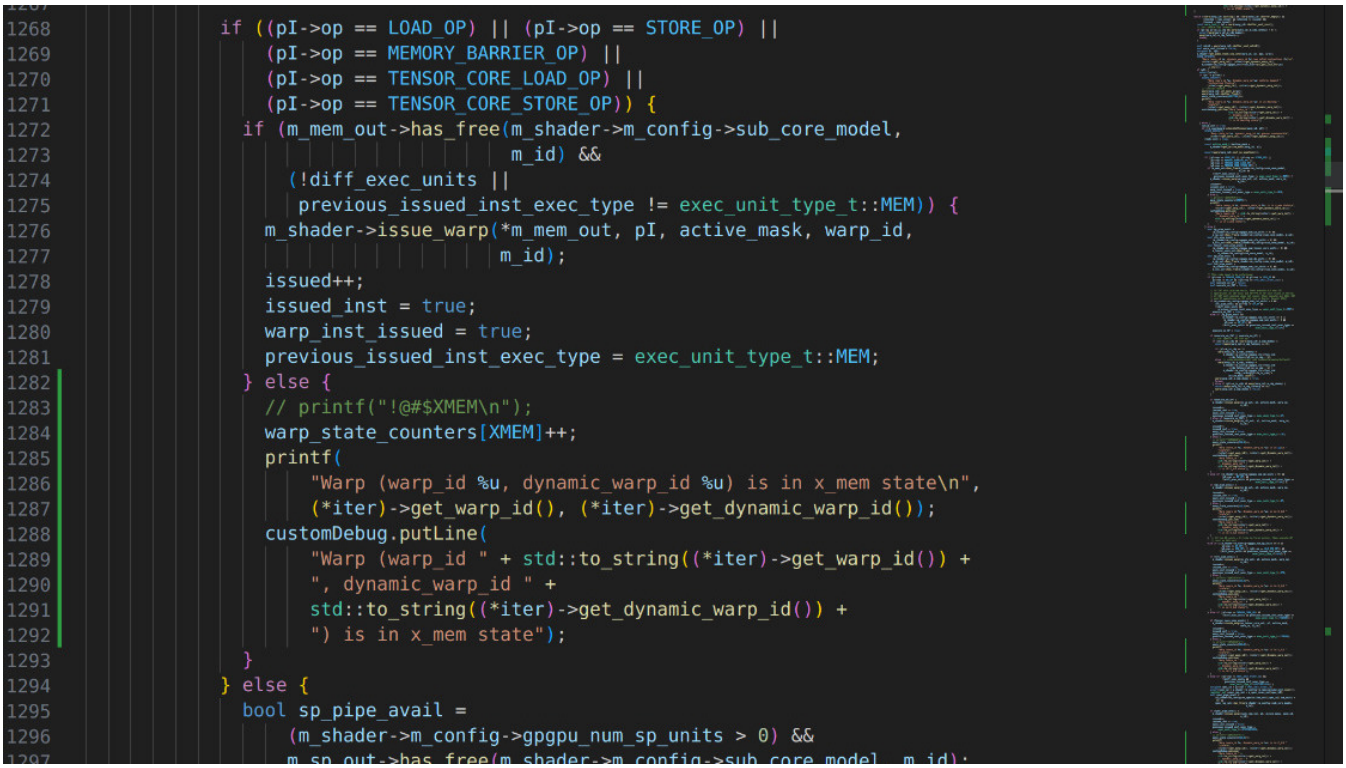


```
1439            } else if ((pI->op == TENSOR_CORE_OP) &&
1440                       !(diff_exec_units && previous_issued_inst_exec_type ==
1441                                            exec_unit_type_t::TENSOR)) {
1442              if (tensor_core_pipe_avail) {
1443                m_shader->issue_warp(*m_tensor_core_out, pI, active_mask,
1444                                     warp_id, m_id);
1445                issued++;
1446                issued_inst = true;
1447                warp_inst_issued = true;
1448                previous_issued_inst_exec_type = exec_unit_type_t::TENSOR;
1449              } else {
1450                // printf("!@#$XALU\n");
1451                warp_state_counters[XALU]++;
1452                printf(
1453                    "Warp (warp_id %u, dynamic_warp_id %u) is in X_ALU "
1454                    "state\n",
1455                    (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1456                customDebug.putLine(
1457                    "Warp (warp_id " +
1458                    std::to_string((*iter)->get_warp_id()) +
1459                    ", dynamic_warp_id " +
1460                    std::to_string((*iter)->get_dynamic_warp_id()) +
1461                    ") is in X_ALU state");
1462              }
1463            } else if ((pI->op >= SPEC_UNIT_START_ID) &&
```



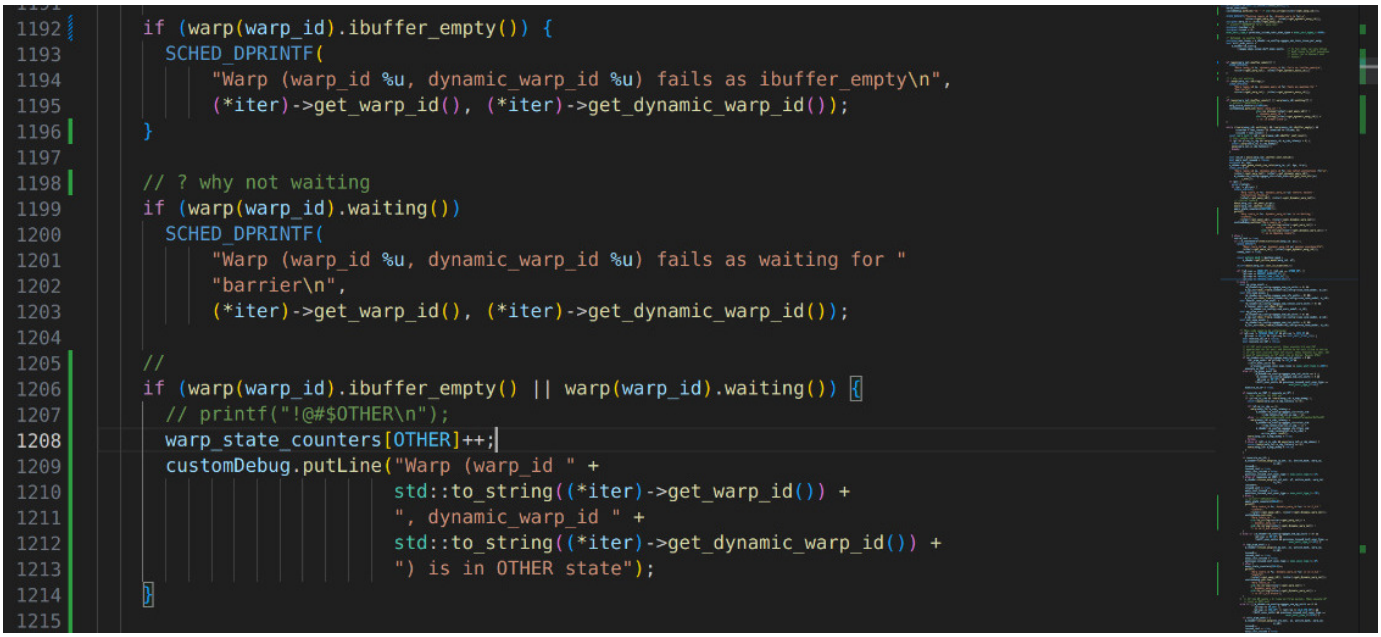
```
1387            } else if ((m_shader->m_config->gpgpu_num_dp_units > 0) &&
1388                       (pI->op == DP_OP) &&
1389                       !(diff_exec_units && previous_issued_inst_exec_type ==
1390                                            exec_unit_type_t::DP)) {
1391              if (dp_pipe_avail) {
1392                m_shader->issue_warp(*m_dp_out, pI, active_mask, warp_id,
1393                                     m_id);
1394                issued++;
1395                issued_inst = true;
1396                warp_inst_issued = true;
1397                previous_issued_inst_exec_type = exec_unit_type_t::DP;
1398              } else {
1399                warp_state_counters[XALU]++;
1400                printf(
1401                    "Warp (warp_id %u, dynamic_warp_id %u) is in X_ALU "
1402                    "state\n",
1403                    (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1404                customDebug.putLine(
1405                    "Warp (warp_id " +
1406                    std::to_string((*iter)->get_warp_id()) +
1407                    ", dynamic_warp_id " +
1408                    std::to_string((*iter)->get_dynamic_warp_id()) +
1409                    ") is in X_ALU state");
1410              }
1411            }  // If the DP units = 0 (like in Fermi archi), then execute DP
1412              // inst on SFU unit
```

XMEM State Counter:



```
1268            if ((pI->op == LOAD_OP) || (pI->op == STORE_OP) ||
1269                (pI->op == MEMORY_BARRIER_OP) ||
1270                (pI->op == TENSOR_CORE_LOAD_OP) ||
1271                (pI->op == TENSOR_CORE_STORE_OP)) {
1272              if (m_mem_out->has_free(m_shader->m_config->sub_core_model,
1273                                      m_id) &&
1274                  (!diff_exec_units ||
1275                   previous_issued_inst_exec_type != exec_unit_type_t::MEM)) {
1276                m_shader->issue_warp(*m_mem_out, pI, active_mask, warp_id,
1277                                     m_id);
1278                issued++;
1279                issued_inst = true;
1280                warp_inst_issued = true;
1281                previous_issued_inst_exec_type = exec_unit_type_t::MEM;
1282              } else {
1283                // printf("!@#$XMEM\n");
1284                warp_state_counters[XMEM]++;
1285                printf(
1286                    "Warp (warp_id %u, dynamic_warp_id %u) is in x_mem state\n",
1287                    (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1288                customDebug.putLine(
1289                    "Warp (warp_id " + std::to_string((*iter)->get_warp_id()) +
1290                    ", dynamic_warp_id " +
1291                    std::to_string((*iter)->get_dynamic_warp_id()) +
1292                    ") is in x_mem state");
1293              }
1294            } else {
1295              bool sp_pipe_avail =
1296                  (m_shader->m_config->gpgpu_num_sp_units > 0) &&
1297                  m_sp_out->has_free(m_shader->m_config->sub_core_model, m_id);
```

Other State Counter:



```
1192    if (warp(warp_id).ibuffer_empty()) {
1193      SCHED_DPRINTF(
1194          "Warp (warp_id %u, dynamic_warp_id %u) fails as ibuffer_empty\n",
1195          (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1196    }
1197
1198    // ? why not waiting
1199    if (warp(warp_id).waiting())
1200      SCHED_DPRINTF(
1201          "Warp (warp_id %u, dynamic_warp_id %u) fails as waiting for "
1202          "barrier\n",
1203          (*iter)->get_warp_id(), (*iter)->get_dynamic_warp_id());
1204
1205    //
1206    if (warp(warp_id).ibuffer_empty() || warp(warp_id).waiting()) {
1207      // printf("!@#$OTHER\n");
1208      warp_state_counters[OTHER]++;
1209      customDebug.putLine("Warp (warp_id " +
1210                          std::to_string((*iter)->get_warp_id()) +
1211                          ", dynamic_warp_id " +
1212                          std::to_string((*iter)->get_dynamic_warp_id()) +
1213                          ") is in OTHER state");
1214    }
1215
```

OUTPUT

We used the application Path Finder from GPU-Rodinia.

This application has three inputs: Row Length, Column Length, Pyramid Height

Input 1:

Row Length: 100

Column Length: 50

Pyramid Height: 20

Output 1:

CYCLE: 23514

ISSUED: 17632

ISSUED %: 9.529832

XALU: 2391

XALU %: 1.292300

XMEM: 447

XMEM %: 0.241597

WAITING: 101425

WAITING %: 54.818694

OTHER: 63124

OTHER %: 34.117577

Total : 185019

Input 1:

Row Length: 150

Column Length: 100

Pyramid Height: 35

Output 2:

CYCLE: 47004

ISSUED: 37338

ISSUED %: 9.982542

XALU :4950

XALU % :1.323413

XMEM :447

XMEM % :0.119508

WAITING :231269

WAITING % :61.831175

OTHER :100029

OTHER % :26.743362

Total : 374033

Input 3:

Row Length: 250

Column Length: 125

Pyramid Height: 50

Output 3:

CYCLE: 116820

ISSUED: 96410

ISSUED %: 10.338198

XALU: 12688

XALU %: 1.360554

XMEM: 894

XMEM %: 0.095865

WAITING: 612663

WAITING %: 65.696828

OTHER: 209906

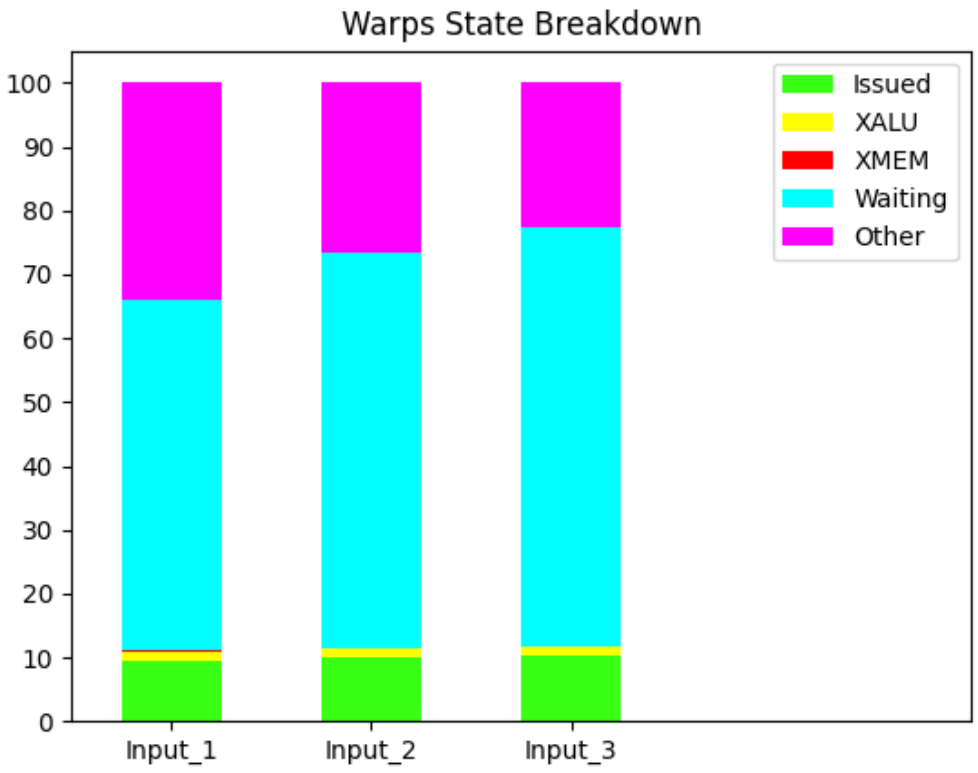
OTHER %: 22.508554

Total: 932561

Since we have 8 warps :

Cycle Count * 8 = Total State Count

Warps State Breakdown



We created a new header file warp-counters.h

```
#ifndef WARP_STATE_COUNTER
#define WARP_STATE_COUNTER

// Defining custom indexes for the counter array
enum counters { CYCLE, WAITING, ISSUED, XALU, XMEM, OTHER };

// Counter array size
#define NUM_COUNTERS (OTHER + 1)

// Declaring the counter array
extern unsigned long long warp_state_counters[NUM_COUNTERS];

#endif
```