Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчет по практикуму №1
по курсу «Архитектура ЭВМ»

Тема Разработка и отладка программ в вычислительном комплексе Тераграф

Студент Романов С. К.

Группа ИУ7-55Б

Оценка (баллы)

Преподаватель Дубровин Е.Н.

Москва — 2022 г.

ВВЕДЕНИЕ

Практикум посвящен освоению принципов работы вычислительного комплекса Терапраф и получению практических навыков решения задач обработки множеств на основе гетерогенной вычислительной структуры. В ходе практикума необходимо ознакомиться с типовой структурой двух взаимодействующих программ: хост-подсистемы и программного ядра sw_kernel. Участникам предоставляется доступ к удаленному серверу с ускорительной картой и настроенными средствами сборки проектов, конфигурационный файл для двухъядерной версии микропроцессора Леонард Эйлер, а также библиотека leonhard x64 xrt с открытым исходным кодом.

1 Задание

1.1 Постановка задачи

Устройство проверки прав доступа. По запросу от хост-подсистемы, содержащему 64-битный индекс и 64-битный ключ доступа необходимо выполнить поиск на наличие записи с указанным индексом в таблице прав доступа. Если такой индекс имеется, сравнить переданный ключ доступа с сохраненным, и при совпадение ответить хост системе утвердительно (значение 1). Если индекс сохранен, но ключи доступа не совпадают, ответить отрицательно (значение 0). Если индекс не найден, то создать новую запись с полученным индексом и ключом доступа.

1.2 Решение

Листинг 1: Проверка прав доступа

```
1   void check_access()
2  {
3   lnh_del_str_sync(TEST_STRUCTURE);
4
5   //Объявление переменных
6   unsigned int count = mq_receive();
7   unsigned int size_in_bytes = 2*count*sizeof(uint64_t);
8
9   //Создание буферадляприемапакета
10  uint64_t *buffer = (uint64_t*)malloc(size_in_bytes);
11
12  //Чтение пакетавRAM
13  buf_read(size_in_bytes, (char*)buffer);
14  for (int i = 0; i < count; i++)
15  {
16      uint64_t key = buffer[i*2];
```

```
        uint64_t value = buffer[i*2 + 1];

        // Поискключаkey вструктуреTEST_STRUCTURE ивыдачанайденногоключаизначенияvalue.
        // res = false - ключненайден
        // res = true - ключнайден
        bool res = lnh_search(TEST_STRUCTURE, key);
        if (!res) {
            //Вставка ключаkey изначенияvalue вструктуруTEST_STRUCTURE. Записьрезультатавочередь.
            lnh_ins_syncq(TEST_STRUCTURE, key, value);
            mq_send(lnh_core.result.status);
        } else if (value == lnh_core.result.value) {
            mq_send(1);
        } else if (value != lnh_core.result.value) {
            mq_send(0);
        }
    }
    lnh_sync();
    free(buffer);
}
```

Листинг 2: Проверка работы функции

```
    ...
    __foreach_core(group, core) {
        // Парыkey - value
        host2gpc_buffer[group][core][0] = 1;
        host2gpc_buffer[group][core][1] = 2;
        host2gpc_buffer[group][core][2] = 1;
        host2gpc_buffer[group][core][3] = 2;
        host2gpc_buffer[group][core][4] = 5;
        host2gpc_buffer[group][core][5] = 6;
        host2gpc_buffer[group][core][6] = 7;
        host2gpc_buffer[group][core][7] = 8;

        host2gpc_buffer[group][core][8] = 1;
        host2gpc_buffer[group][core][9] = 2;
        host2gpc_buffer[group][core][10] = 3;
        host2gpc_buffer[group][core][11] = 4;
        host2gpc_buffer[group][core][12] = 5;
        host2gpc_buffer[group][core][13] = 5;
        host2gpc_buffer[group][core][14] = 7;
```

```
        host2gpc_buffer[group][core][15] = 7;
    }

    __foreach_core(group, core) {
        lnh_inst.gpc[group][core]->start_async(__event__(check_access));
    }

        //DMA записьмассиваhost2gpc_buffer вглобальнуюпамять
    __foreach_core(group, core) {
        lnh_inst.gpc[group][core]->buf_write(BURST*2*sizeof(uint64_t),(char*)host2gpc_buffer[group][core]);
    }

    //Ожидание завершенияDMA
    __foreach_core(group, core) {
        lnh_inst.gpc[group][core]->buf_write_join();
    }



    // Получитьсигналы
    unsigned int signals[LNH_GROUPS_COUNT][LNH_MAX_CORES_IN_GROUP][BURST];
    printf("Полученные сигналы:\n");
    __foreach_core(group, core) {
        lnh_inst.gpc[group][core]->mq_send(8);
    }

    __foreach_core(group, core) {
        for (int i = 0; i < BURST; i++) {
            signals[group][core][i] = lnh_inst.gpc[group][core]->mq_receive();
            printf("Group: %d, Core: %d, signal: %d\n", group, core, signals[group][core][i]);
        }
    }

    bool error = false;
    // Проверкаданных
    // NOTE: 151062035 - кодстатуса
    __foreach_core(group, core) {
        error = signals[group][core][0] != 151062035 ||
                signals[group][core][1] != 1  ||
                signals[group][core][2] != 151062035 ||
```

```
60                      signals[group][core][3] != 151062035 ||
61                      signals[group][core][4] != 1 ||
62                      signals[group][core][5] != 151062035 ||
63                      signals[group][core][6] != 0 ||
64                      signals[group][core][7] != 0;
65
66      }
67
68      __foreach_core(group, core) {
69          free(host2gpc_buffer[group][core]);
70      }
71
72
73      if (!error)
74          printf("Тест пройден успешно!\n");
75      else
76          printf("Тест завершен с ошибкой!\n");
77
78  ...
```

Программа была протестирована. Данные остались в целостности, полученные сигналы корректны.

ЗАКЛЮЧЕНИЕ

В данной лабораторной работе были освоены принципы работы вычислительного комплекса Терапраф и получены практические навыки решения задач обработки множеств на основе гетерогенной вычислительной структуры.