Институт информационных и вычислительных технологий МОН РК

Казахский Национальный Университет имени аль-Фараби

Университет Туран

Люблинский технический университет, Польша

«Ғылым ордасы»

МАТЕРИАЛЫ

IV международной научно-практической конференции "Информатика и прикладная математика", посвященной 70-летнему юбилею профессоров Биярова Т.Н., Вальдемара Вуйцика и 60-летию профессора Амиргалиева Е.Н. 25-29 сентябрь 2019, Алматы, Казахстан

Часть 1

Алматы 2019

ИСПОЛЬЗОВАНИЕ ТЕХНОЛОГИИ VULKAN ДЛЯ 3D-ВИЗУАЛИЗАЦИЙ БОЛЬШИХ ВЫЧИСЛИТЕЛЬНЫХ ДАННЫХ, ИЗМЕНЯЮЩИХСЯ СО ВРЕМЕНЕМ

[1]Мустафин М.Б., [2]Турар О.Н., [3]Ахмед-Заки Д.Ж.
e-mail: mustafin.mb@gmail.com
[1,2]Казахский Национальный Университет имени Аль-Фараби, Казахстан,
[3]Университет международного бизнеса

Аннотация. *В данной работе было разработано высокопроизводительное приложение для визуализации сеточных моделей больших размеров (около млн. ячеек), с использованием технологий Vulkan. Для работы приложения без прерывания было использовано двойная буферизация буфера вершин и многопоточность процессора. Для визуализаций 2D и 3D модели были взяты результаты уравнения Пуассона, а именно данные каждой итерации. Используя вышеуказанные методы, приведены примеры сеточной модели уравнения Пуассона. В результате данной работы был разработан и представлен прототип визуализатора, которую можно использовать для любых результатов численного математического моделирования на структуированных и неструктурированных 3D сетках.*

1 Введение

Vulkan – это API для графических и вычислительных устройств [1]. Как и OpenGL [2], Vulkan позволяют с высокой производительностью отображать пиложения 3D-графикой в реальном времени [3][4], а так же более высокую производительность и меньшую нагрузку на центральный процессор. Технология Vulkan был основан на технологиях AMD Mantle [5]. По сравнению с технологией OpenGL, Vulkan является низкоуровневым API и обладает в целом анологичными возможностями. Это позволяет использовать всю возможность графического процессора для вычисления, получить низкоуровневый доступ к GPU и контроля его работы[6].

В отличий от OpenGL, шейдеры в Vulkan представлены, бинарным промежуточным представлением программ SPIR-V [7]. SPIR-V – это единственный поддерживаемый шейдерный язык для Vulkan. Он принимается на уровне API и используется для создания конвейеров, предназначенных для управления устройства для выполнения работы приложением [8][6].

В данной статье предлагается визуализация в реальном времени больших сеточных моделей с использованием технологий Vulkan на типовых персональных компьютерах, оснащенных дискретной графической картой [9]. Для хранения и обновления данных было использовано двойная буферизация буферов вершин [10] и многопоточность C++11 [11]. Предлагаемый подход к визуализации сеточной модели оптимизирует скорость отрисовки.

Отметим, что представленное приложение может быть использован для любых результатов численного математического моделирования на структуированных и неструктурированных 3D сетках.

2 Обзор литературы

В настоящее время Язык шейдеров OpenGL (GLSL) является основной частью программирования с использованием библиотеки OpenGL [2][7][15]. С помощю GLSL можно пользоваться мощью графического процессора для отображения и вычислений. На основе GLSL была создана библиотека RGL, которая предлагает трехмерную визуализацию в реальном времени [3][4]. В работе [10][16] были использованы возможности графического процессора и языка шейдеров библиотеки OpenGL, а также провизуализированы результаты численного математического моделирования.

В последнее время технология Vulkan набирает большую популярность в сфере визуализаций. Vulkan – это кроссплатформенный графический и вычислительный API разрабатываемый консорциумом Kronos Group [5]. Vulkan является приемником OpenGL, но очень сильно отличается. Основным отличием является то, что раньше делал драйвер OpenGL, является обязанностью программиста [6][18]. Еше одно отличие технологий Vulkan от OpenGL, это шейдерный язык. Vulkan поддерживает единственный шейдерный язык SPIR-V[8]. В литературах [14][17][19] можно ознакомится со спецификациями технологии Vulkan.

В работе [11] описывается языки и библиотеки для многопоточного програмирования, и как развить навыки программирования, а также описываются и демонстрируются различные методы тестирования и отладки, разработанные для многопоточных программ за последние 20 лет.

3 Материалы и методы

Целью данной работы была разработка высокопроизводительного приложения для визуализации результатов численного математического моделирования, используя технологию Vulkan.

В более старых API, таких как OpenGL, драйвер управляет синхронизацией и памятью, во время выполнения приложения проверяет на ошибки. Это удобно для программистов, но на все это тратится время CPU. В Vulkan передаются все обязанности в руки программиста, т.е. практически все отлеживания состояния, управление памятью и синхронизацией [14].

3.1 Системы координат

Vulkan работает с отрезками и треугольниками, представляя их вершины как точки в трехмерном пространстве. Они называются вершинами [15].

Для начала была создана сетка с размером 50x50 и заполнена случайными цветами. Каждый квадрат сетки состоит из двух треугольников, которые задаются координатами [16]. В функций initVertices() был заполнен вектор, в котором содержатся координаты и цвет каждой вершины треугольника.

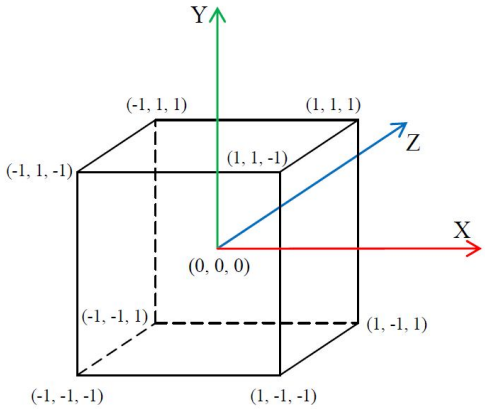

Рис. 1 Система координат вершин

3.2 Память и ресурсы

Практически для всех вычислительных систем, вкючая Vulkan, память является крайне важным. В Vulkan существует два базовых типа памяти: память CPU (host memory) и память GPU (device memory). При созданий нового объекта в Vulkan понадобится память для хранения данных. Для этого используется обычная память CPU.

Vulkan работает с данными, а данные хранятся в ресурсах. В Vulkan есть два базовых типа ресурсов: буферы и изображения [6]. Буфер – это простой линейный кусок памяти, который может быть использован для различных целей. Они используются для хранения линейных структурированных и неструктурированных данных, которые могут иметь формат или просто быть байтами в памяти.

3.3 Управление памятью устройства

Когда Vulkan работает с данными, эти данные должны храниться в памяти устройства (device memory). На рисунке 1 Схема памяти GPU и CPU, каждый со своей памятью [6].

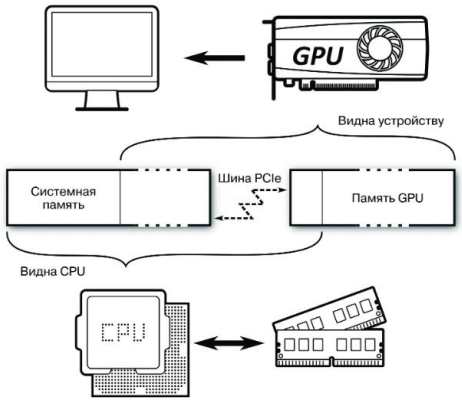

Рис. 2 Память CPU и GPU

3.4 Выделение памяти устройства

Выделение памяти устройства представлено как объект VkDeviceMemory, создаваемый с помощью функций vkAllocateMemory(), прототип которого приводится ниже [17][6][18][19]:

```
VkResult vkAllocateMemory (
VkDevice  device,
const VkMemoryAllocateInfo*     pAllocateInfo,
const VkAllocationCallbacks*    pAllocator,
VkDeviceMemory*                 pMemory);
```

После выделения памяти устройства, ее можно использовать для хранения данных. Таким образом, был создан буфер под названием vertexBuffer и использован для хранения данных вершин. После проведения всех работ получаем следующее:

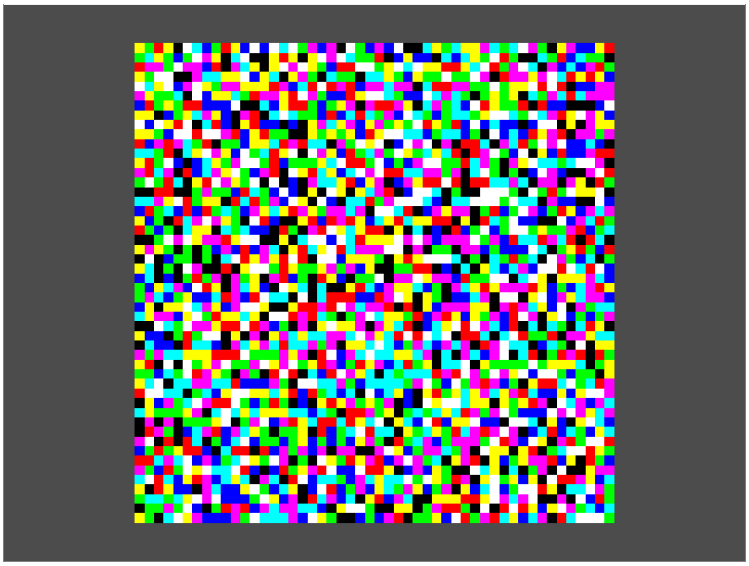

Рис. 3 Сетка с размером 50x50

При визуализации в реальном времени будут поступать новые данные, а это данные цветов каждой ячейки. Так как координаты и цвет ячеек записаны в одном буфере, координаты будут перезаписываться, а это трата драгоценного времени CPU. Чтобы решить эту проблему был создан отдельный буфер для данных цветов. В результате получим два буффера: posBuffer – для хранения данных координат и colBuffer – для хранения данных цветов. Программа будет работать в реальном времени, а значит данные будут поступать постоянно. Создаем новый буфер colBuffer1 для хранения новых данных.

3.5 Обновление буфера

Для обновления данных внури буфера была использована функция vkCmdUpdateBuffer [19]. vkCmdUpdateBuffer() копируют данные напрямую из памяти CPU в память буфера. Данные забираются из памяти CPU при вызове, и по

возращении из vkCmdUpdateBuffer() можно освободить эту память или записать в нее новые данные.

3.6 Двойная буферизация буфера вершин

Двойная буферизация – метод подготовки данных, обеспечивающий возможность отдачи готового результата без прерывания [10]. Этот метод повсеместно используется при работе с буфером кадра. Данная статья описывает использование похожего подхода для буферизации данных цветов на модели пласта. Метод заключается в следующем. Создается второй буфер, и данные заполняются только на нем. Как только завершается процесс чтения, буферы меняются местами, и вывод данных начнет осуществляться из второго буфера, а новые данные заполняться в первый буфер. Это похоже на двойную буферизацию кадров и используется для буферов вершин. На рисунке 4 и 5 приведены примеры отрисовки каждого буфера.

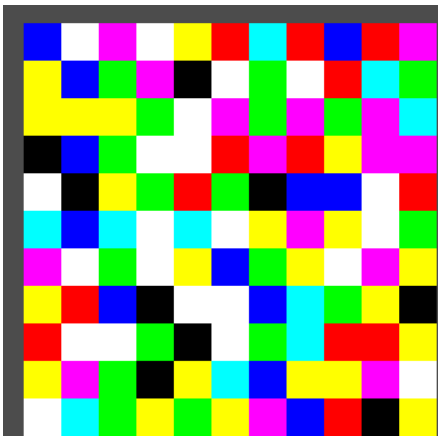

Рис. 4 Отрисовка colBuffer

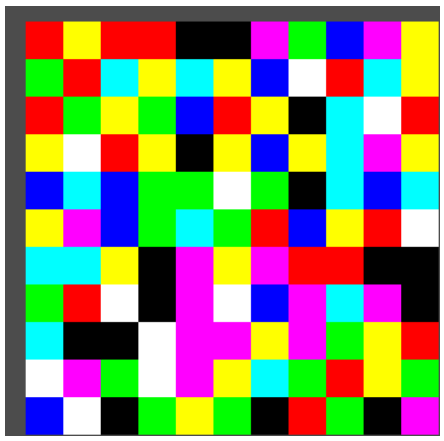

Рис. 5 Отрисовка colBuffer1

При чтении и записи новых данных функция отрисовки drawFrame() останавливается и программа не будет реагировать пока данные не будут инициализированы. Несмотря на то, что на Vulkan процессы на CPU и GPU происходят асинхронно, интерактивное управление моделью такое, как поворот и перенос модели с помощью мыши и клавиатуры, должно осуществляться на CPU. Если все действия на центральном процессоре будут происходит в одном потоке, на время записи в буфер мы не будем иметь возможность взаимодействовать с трехмерной моделью. Например, если мы будем вызывать функцию чтения и записи каждые десять секунд, а эта функция будет выполняться за шесть секунд, в результате мы получим программу, которая не реагирует в момент инициализации и работает только на оставшееся время. Визуальный пример приведен ниже на рисунке 6.

Рис. 6 - Отрисовка, - Копирование (A-host to host, B – host to device)

Чтобы входные данные записывались, не останавливая работу отрисовки, ее нужно выполнять параллельно. Для этого будем использовать многопоточность C++ 11 [11]. Современные компьютеры имеют многоядерные процессоры, в них многопоточность осуществляется тем, что на разных ядрах исполняются несколько процессов. Используя возможность многопоточности мы передаем команду чтения и заполнения буфера в другой поток, и она будет выполняться параллельно, не влияя на отрисовку. То есть, когда на экране отображается входные данные из первого буфера, параллельно будет обновляться второй буфер. Как только второй буфер будет готов, буфера меняются местами. Визуальный пример приведен на рисунке 7.

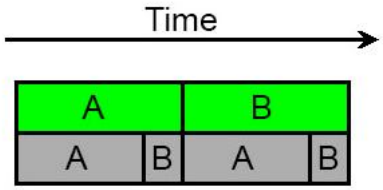

Рис. 7 - Отрисовка, - Копирование (A-host to host, B – host to device)

Как видим на рисунке 6 и 7, инициализация данных проходит в два этапа: чтение и запись даных в системную память (A), и копирование из системной памяти в память GPU (B). Большее время инициализации занимает чтение и запись новых данных в системную память, т.е. host to host.

4 Результаты

Для получения результата использовался персональный компьютер (Core i7 3770 3.40 GHz, 8Gb DDR3), оснащенный дискретной графической картой (nVidia GeForce GTX650Ti, 1Gb GDDR5). В качестве входных данных были взяты результаты каждой 100-ой итераций уравнения Пуассона с размером 1000x1000 для 2D и 33x33x11 для 3D, количество итераций 20000 [12]. Для визуализации трехмерной модели был использован фотмат GRDECL [20]. Чтобы инициализировать данные с файла считываем данные построчно и вместо случайных цветов в вершинах заполняем полученными данными.

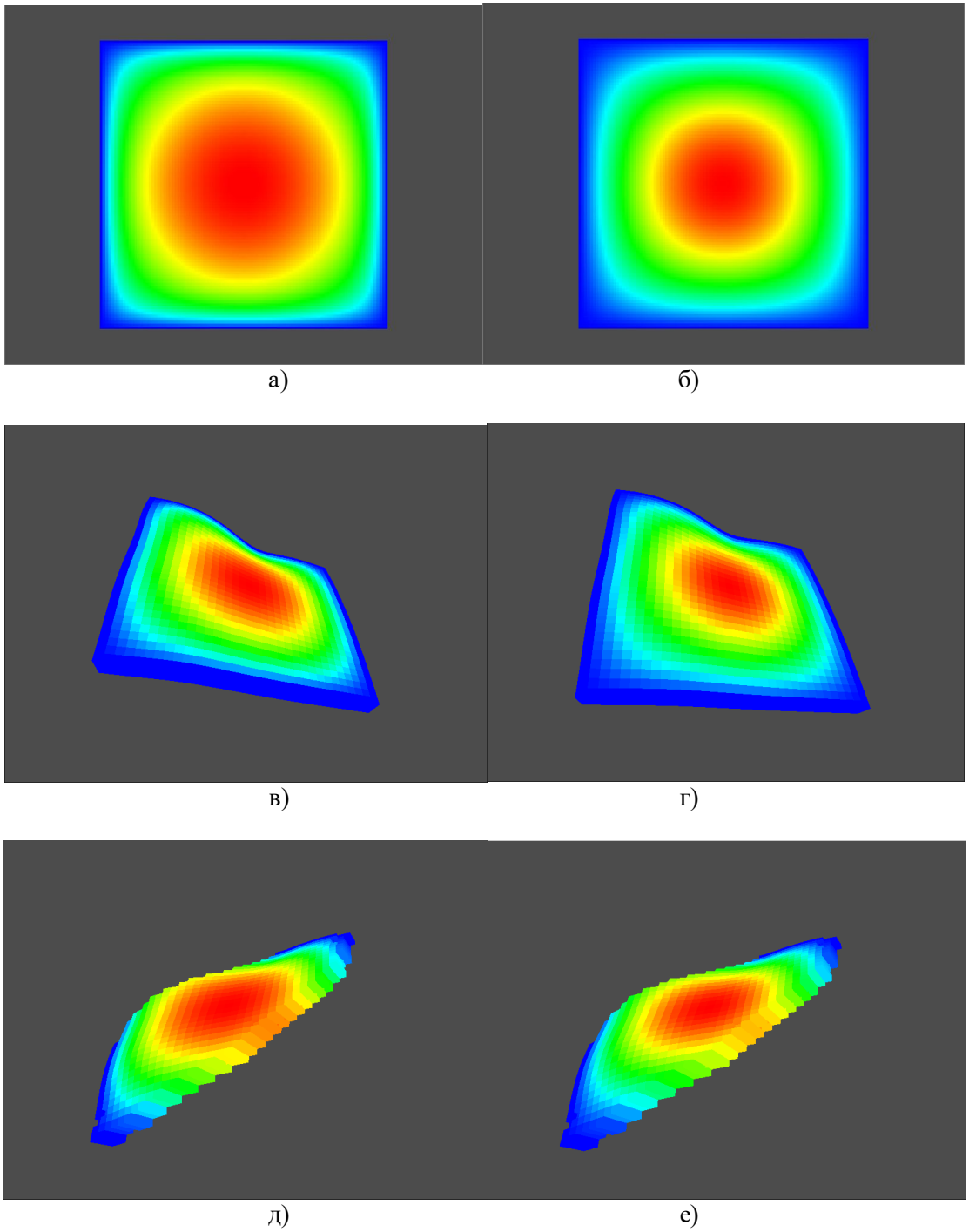

Рис. 8 Примеры визуализаций сеточной модели уравнения Пуассона: а) 2D модель самой первой итераций, б) 2D модель самой последней итераций, в) 3D модель самой первой итераций, г) 3D модель самой последней итераций, д) 3D модель с неакивными участками самой первой итераций, е) 3D модель с неактивными участками самой последней итераций

На рисунке 8 представлен прототип приложения, визуализирующий результат сеточной модели уравнения Пуассона. Были произведены замеры и получены результаты представленные на таблице 1

Таблица 1. Результаты замера

Размерность	2D,1000x1000	3D, 33x33x11
Количество ячеек	10^6	11979
FPS (кадры в секунду)	1750	4415
copy host to host, мс	8438	97
copy host to device, мс	14	0

При запуске приложения для сеточной модели с вышеуказанными размерами большее время занимает копирование host to host, т.е. чтение и запись данных в системную память, которая зависит от ресурсов компьютера: скорость чтения и записи жесткого диска, скорость передачи данных через интернет. Копирование данных с системной памяти в память устройства зависит от таких характеристик графического процессора: шина памяти, интерфейс памяти, скорость передачи данных памяти, пропускная способность памяти, тип выделенной видеопамяти.

Таблица 2. Характеристика графической карты nVidia Geforce GTX650Ti

Шина памяти	PCI Express x16 Gen3
Интерфейс памяти	128 бит
Скорость передачи данных памяти	5400 МГц
Пропускная способность памяти	86.40 ГБ/с
Тип выделенной видеопамяти	1024 МБ GDDR5

Заключение

Используя технологию Vulkan, было разработано высокопроизводительное приложение, которое визуализирует в реальном времени сеточные модели численного математического моделирования. Для того, чтобы приложение работало без прерывания, была использована двойная буферизация буферов вершин и многопоточность C++11. Метод двойной буферизации буферов вершин использовалось для буферизации данных цветов на модели пласта. Данный подход к визуализации сеточной модели оптимизировала скорость работы приложения. Представлено приложение, визуализирующее 3D-графику в реальном времени, с высокой производительностью и меньшей нагрузкой на центральный процессор. Для примера были взяты результаты уравнения Пуассона 2D и 3D. Готовое приложение будет применяться для визуализаций моделей результатов численного математического моделирования.

Литература

1. Khronos Group. SPIR Overview. – https : //www.khronos.org/spir/. (13.07.2017).

2. Wolff D., OpenGL 4.0 Shading Language Cookbook. Packt Publishing Ltd. Birmingham (2011) 17. Wolfram, S.: Mathematica Book. [D. Wolff, OpenGL 4.0 Shading

Language Cookbook. Packt Publishing Ltd. Birmingham (2011) 17. Wolfram, S.: Mathematica Book.]

3. Daniel Adler, Oleg Nenadic, Walter Zucchini, RGL: A R-library for 3D visualization with OpenGL

4. T. Mullen *et al*., "Real-time modeling and 3D visualization of source dynamics and connectivity using wearable EEG," *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Osaka, 2013, pp. 2184-2187.

5. "Vulkan." Nvidia developer, accessed – https://developer.nvidia.com/Vulkan/. (09.12.2017).

6. Селлерс Г. Vulkan. Руководство разработчика. Официальное руководство / пер. С англ. А. Б. Борескова. – М.:ДМК Пресс, 2017. – 394с.:ил.

7. Sellers G., Wright R.S. Jr., Haemel N. OpenGL SuperBible: Comprehensive Tutorial and Reference (6th Edi- tion). Addison-Wesley Professional; 6 edition (July 31, 2013) 2013 P. 848. [G. Sellers, R.S. Jr. Wright, N. Haemel, OpenGL SuperBible: Comprehensive Tutorial and Reference (6th Edi- tion). Addison-Wesley Professional; 6 edition (July 31, 2013) 2013 P. 848.]

8. Khronos Group. SPIR Overview – https : / / www . khronos . org / spir/. (06.01.2018).

9. Loix T., De Breucker S., Vanassche P., Van den Keybus J., Driesen J. and Visscher K., "Layout and performance of the power electronic converter platform for the VSYNC project," *2009 IEEE Bucharest PowerTech*, Bucharest, 2009, pp. 1-8.

10. Бадретдинов М. Р. , Бадретдинов Т. Р. , Борщук М. С., Применение библиотеки opengl для визуализации результатов численного математ ического моделирования на сетках большой размерности. Вестник УГАТУ, 2015. № 4. 84-94

11. Richard H. Carver, Kuo-Chung Tai. Modern multithreading: implementing, testing, and debugging multithreaded Java and C++/Pthreads/Win32 programs / by Richard H. Carver and Kuo-Chung Tai.

12. Вержбицкий В.М. Численные методы (математический анализ и обыкновенные дифференциальные уравнения). – М.: Высшая школа, 2001.

13. Самарский А.А., Николаев Е.С. Методы решения сеточных уравнений – М.: Наука, 1987. – С.130.

14. “C++ Abstraction library for Vulkan API”, L. O. Tolo. – https://github.com/larso0/bp. (03.04.2018).

15. OpenGL programming guide : the official guide to learning OpenGL, version 4.3 / Dave Shreiner, Graham Sellers, John Kessenich, Bill Licea-Kane ; the Khronos OpenGL ARB Working Group.---Eighth edition.

16. Abraham F., Celes W. Distributed Visualization of Com- plex Black Oil Reservoir Models // Eurographics Symposium on Parallel Graphics and Visualization (2009), EGPGV 2009, Mu- nich, Germany P. 87–94. [F. Abraham, W. Celes, “Distributed Visualization of Complex Black Oil Reservoir Models”, Eurographics Symposium on Parallel Graphics and Visualiza- tion (2009), EGPGV 2009, Munich, Germany P. 87–94.]

17. Pawel Lapinski, Vulkan Coockbook. Work through recipes to unlock the full potential of the next generation graphics API—Vulkan. Packt Publishing Ltd. Birmingham (2017).

18. Khronos Vulkan Working Group. Vulkan 1.0.98 - A Specification (with KHR extensions). 1.0.98. Jan. 2019.

19. Schlumberger. Eclipse reference manual: technical description, 2002 P. 2683. [Schlumberger. Eclipse reference manual: technical description, 2002 P. 2683.]

ЕДИНЫЙ ИНТРАНЕТ ПОРТАЛ ДЛЯ ПОДДЕРЖКИ ПРЕТЕНДЕНТОВ НА ГОСУДАРСТВЕННУЮ СЛУЖБУ РК

Мухаев Д.
Казахский Национальный Университет им. аль-Фараби
daryn.mukhayev@gmail.com

Аннотация. *В работе проанализированы недостатки существующей системы отбора претендентов на государственную службу РК, выявлены возможные причины сбоев системы, указаны пути их устранения. В целях оптимизации систем отбора претендентов предложено использовать преимущества технологий дистанционного обучения.*

На XV Съезде партии «Нұр-Отан» было обозначено 100 конкретных шагов по реализации пяти институциональных реформ:

- формирование профессионального государственного аппарата;
- обеспечение верховенства закона;
- индустриализация и экономический рост;
- идентичность и единство;
- формирование подотчетного государства.

Если остановиться на каждой из пяти институциональных реформ, то первое, это первоочередная задача государственных служащих в профессиональном государственном аппарате - это обслуживание населения; второе, если не будет работать система закона, то не будет и цивилизации; третье, если будет индустрия, то изменится мышление нашего народа, изменится взгляд на государство, изменится само государство; четвертое, идентичность и единство – это такие задачи, как дружба и единство всего народа Казахстана; пятое - подотчетное государство, то есть государство должно давать отчет перед гражданами данного государства [0].

Система государственной службы постоянно совершенствуется в соответствии с требованиями времени и пяти институциональных реформ. В 2016 году вступил в силу новый Закон «О государственной службе Республики Казахстан» [1]. Он направлен на обеспечение открытого конкурсного отбора, продвижение по службе на основе компетентности и уровня вознаграждения в зависимости от результатов. Прием на государственную службу начнется с низовых