Компьютерная графика

Лекция 13: состояние OpenGL (напоминание), матрицы проекций (напоминание), рендеринг в cubemap, дистрибуция приложений на OpenGL

2021

Настройка графического конвейера

- Графический конвейер (pipeline) - набор всех операций, происходящих с данными от момента вызова `glDraw*` до появления пикселей на экране (или текстуре/рендербуфере)

Настройка графического конвейера

- Графический конвейер (pipeline) - набор всех операций, происходящих с данными от момента вызова `glDraw*` до появления пикселей на экране (или текстуре/рендербуфере)
- Графический конвейер = programmable pipeline + fixed-function pipeline

Настройка графического конвейера

- Графический конвейер (pipeline) - набор всех операций, происходящих с данными от момента вызова `glDraw*` до появления пикселей на экране (или текстуре/рендербуфере)
- Графический конвейер = programmable pipeline + fixed-function pipeline
- Настройка programmable pipeline: шейдеры (шейдерная программа)

Настройка графического конвейера

- Графический конвейер (pipeline) - набор всех операций, происходящих с данными от момента вызова `glDraw*` до появления пикселей на экране (или текстуре/рендербуфере)
- Графический конвейер = programmable pipeline + fixed-function pipeline
- Настройка programmable pipeline: шейдеры (шейдерная программа)
- Настройка fixed-function pipeline: включение/выключение (`glEnable/glDisable`) конкретных операций и их специфическая настройка

Настройка fixed-function pipeline

- Depth clamp
  - По умолчанию, все примитивы обрезаются по уравнению $z \leq |w|$
  - Можно заменить обрезание clamping'ом через `glEnable(GL_DEPTH_CLAMP)`

Настройка fixed-function pipeline

- Depth clamp
  - По умолчанию, все примитивы обрезаются по уравнению $z \leq |w|$
  - Можно заменить обрезание clamping'ом через `glEnable(GL_DEPTH_CLAMP)`
- Culling
  - Можно не рисовать back-facing или front-facing полигоны
  - Включить: `glEnable(GL_CULL_FACE)`
  - Настроить, что **не** рисуется: `glCullFace`
  - Настроить, что считается back-facing, а что front-facing: `glFrontFace`

Настройка fixed-function pipeline

- Depth clamp
  - По умолчанию, все примитивы обрезаются по уравнению $z \leq |w|$
  - Можно заменить обрезание clamping'ом через `glEnable(GL_DEPTH_CLAMP)`
- Culling
  - Можно не рисовать back-facing или front-facing полигоны
  - Включить: `glEnable(GL_CULL_FACE)`
  - Настроить, что **не** рисуется: `glCullFace`
  - Настроить, что считается back-facing, а что front-facing: `glFrontFace`
- Viewport
  - Настроить перевод из NDC (normalized device coordinates, [-1..1]) в пиксельные координаты: `glViewport`
  - Обычно нужно делать каждый раз при изменении размеров окна или при переключении фреймбуферов

Настройка fixed-function pipeline

- Depth test
  - Можно не рисовать пиксели, находящиеся сзади уже нарисованных пикселей
  - Включить: `glEnable(GL_DEPTH_TEST)`
  - Настроить: `glDepthFunc`
  - Настроить преобразование из NDC в [0, 1]: `glDepthRangef`
  - Включить/выключить запись значений глубины: `glDepthMask`

Настройка fixed-function pipeline

- Depth test
  - Можно не рисовать пиксели, находящиеся сзади уже нарисованных пикселей
  - Включить: `glEnable(GL_DEPTH_TEST)`
  - Настроить: `glDepthFunc`
  - Настроить преобразование из NDC в [0, 1]: `glDepthRangef`
  - Включить/выключить запись значений глубины: `glDepthMask`
- Stencil test
  - Включить: `glEnable(GL_STENCIL_TEST)`
  - Настроить: `glStencilFunc`, `glStencilOp`, `glStencilMask`

Настройка fixed-function pipeline

- Depth test
  - Можно не рисовать пиксели, находящиеся сзади уже нарисованных пикселей
  - Включить: `glEnable(GL_DEPTH_TEST)`
  - Настроить: `glDepthFunc`
  - Настроить преобразование из NDC в [0, 1]: `glDepthRangef`
  - Включить/выключить запись значений глубины: `glDepthMask`
- Stencil test
  - Включить: `glEnable(GL_STENCIL_TEST)`
  - Настроить: `glStencilFunc`, `glStencilOp`, `glStencilMask`
- Scissor test
  - Можно не рисовать пиксели, находящиеся вне некоторого прямоугольника
  - Включить: `glEnable(GL_SCISSOR_TEST)`
  - Настроить: `glScissor`

Настройка fixed-function pipeline

- Color mask
  - Настроить запись в конкретные цветовые каналы: `glColorMask`

Настройка fixed-function pipeline

- Color mask
  - Настроить запись в конкретные цветовые каналы: `glColorMask`
- Blending
  - Можно записывать значение некоторой функции от входного цвета и уже записанного цвета
  - Включить: `glEnable(GL_BLEND)`
  - Настроить: `glBlendFunc`/`glBlendFuncSeparate`, `glBlendEquation`, `glBlendColor`

Настройка fixed-function pipeline

- Color mask
  - Настроить запись в конкретные цветовые каналы: `glColorMask`
- Blending
  - Можно записывать значение некоторой функции от входного цвета и уже записанного цвета
  - Включить: `glEnable(GL_BLEND)`
  - Настроить: `glBlendFunc`/`glBlendFuncSeparate`, `glBlendEquation`, `glBlendColor`
- Color logical operation
  - Можно записывать результат некоторой побитовой операции от входного цвета и уже записанного цвета
  - Выключает blending
  - Включить: `glEnable(GL_COLOR_LOGIC_OP)`
  - Настроить: `glLogicOp`

Проекции

- Ничто не мешает делать с вершинами абсолютно любые преобразования в вершинном шейдере

Проекции

- Ничто не мешает делать с вершинами абсолютно любые преобразования в вершинном шейдере
- Обычно используют ортографическую или перспективную проекцию, так как они выражаются матрицами (с учётом perspective divide)

Проекции

- Ничто не мешает делать с вершинами абсолютно любые преобразования в вершинном шейдере
- Обычно используют ортографическую или перспективную проекцию, так как они выражаются матрицами (с учётом perspective divide)
- Ортографическая проекция: **не использует** perspective divide, последняя строка равна (0 0 0 1)

Проекции

- Ничто не мешает делать с вершинами абсолютно любые преобразования в вершинном шейдере
- Обычно используют ортографическую или перспективную проекцию, так как они выражаются матрицами (с учётом perspective divide)
- Ортографическая проекция: **не использует** perspective divide, последняя строка равна (0 0 0 1)
- Перспективная проекция: **использует** perspective divide, последняя строка содержит зависимость от X, Y или Z

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z
  - Центр C переходит в центр экрана ((0, 0, 0) в NDC)

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z
  - Центр C переходит в центр экрана ((0, 0, 0) в NDC)
  - Точки, отличающиеся на вектор параллельный X, после проекции отличаются только X-координатой

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z
  - Центр C переходит в центр экрана ((0, 0, 0) в NDC)
  - Точки, отличающиеся на вектор параллельный X, после проекции отличаются только X-координатой
  - Точки, отличающиеся на вектор параллельный Y, после проекции отличаются только Y-координатой

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z
  - Центр C переходит в центр экрана ((0, 0, 0) в NDC)
  - Точки, отличающиеся на вектор параллельный X, после проекции отличаются только X-координатой
  - Точки, отличающиеся на вектор параллельный Y, после проекции отличаются только Y-координатой
  - Точки, отличающиеся на вектор параллельный Z, после проекции попадают в один пиксель (и отличаются только Z-координатой)

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z
  - Центр C переходит в центр экрана ((0, 0, 0) в NDC)
  - Точки, отличающиеся на вектор параллельный X, после проекции отличаются только X-координатой
  - Точки, отличающиеся на вектор параллельный Y, после проекции отличаются только Y-координатой
  - Точки, отличающиеся на вектор параллельный Z, после проекции попадают в один пиксель (и отличаются только Z-координатой)
- N.B.: можно понимать её как композицию аффинного преобразования, двигающего камеру в точку C с осями XYZ, и стандартной ортографической проекции (с единичной матрицей)

Ортографическая проекция

- Проецирует параллельно некому вектору: точки пространства, отличающиеся на вектор, параллельный направлению взгляда, проецируются в одну точку экрана
- Видимая область пространства - параллелепипед
- Видимый размер объектов **не зависит** от расстояния до них
- Удобно описать центром видимой области C и осями X, Y, Z
  - Центр C переходит в центр экрана ((0, 0, 0) в NDC)
  - Точки, отличающиеся на вектор параллельный X, после проекции отличаются только X-координатой
  - Точки, отличающиеся на вектор параллельный Y, после проекции отличаются только Y-координатой
  - Точки, отличающиеся на вектор параллельный Z, после проекции попадают в один пиксель (и отличаются только Z-координатой)
- N.B.: можно понимать её как композицию аффинного преобразования, двигающего камеру в точку C с осями XYZ, и стандартной ортографической проекции (с единичной матрицей)
- Матрица проекции: см. лекцию 4

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче
- Удобно описать параметрами проекции near, far, fovx, fovy и аффинными преобразованием, двигающим камеру (как с ортографической проекцией)

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче
- Удобно описать параметрами проекции near, far, fovx, fovy и аффинными преобразованием, двигающим камеру (как с ортографической проекцией)
- Обычно за направление взгляда берут ось -Z, чтобы получить левую систему координат

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче
- Удобно описать параметрами проекции near, far, fovx, fovy и аффинными преобразованием, двигающим камеру (как с ортографической проекцией)
- Обычно за направление взгляда берут ось -Z, чтобы получить левую систему координат
  - near: всё ближе этого значения (по Z-координате) будет отсекаться

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче
- Удобно описать параметрами проекции near, far, fovx, fovy и аффинными преобразованием, двигающим камеру (как с ортографической проекцией)
- Обычно за направление взгляда берут ось -Z, чтобы получить левую систему координат
  - near: всё ближе этого значения (по Z-координате) будет отсекаться
  - far: всё дальше этого значения (по Z-координате) будет отсекаться

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче
- Удобно описать параметрами проекции near, far, fovx, fovy и аффинными преобразованием, двигающим камеру (как с ортографической проекцией)
- Обычно за направление взгляда берут ось -Z, чтобы получить левую систему координат
  - near: всё ближе этого значения (по Z-координате) будет отсекаться
  - far: всё дальше этого значения (по Z-координате) будет отсекаться
  - fovx, fovy - угол обзора по X и Y

Перспективная проекция

- Проецирует из некой точки: точки пространства, лежащие на одной прямой с центром проекции, проецируются в одну точку экрана
- Видимая область пространства - усечённая пирамида
- Видимый размер объектов **зависит** от расстояния до них: чем объект дальше, тем он мельче
- Удобно описать параметрами проекции near, far, fovx, fovy и аффинными преобразованием, двигающим камеру (как с ортографической проекцией)
- Обычно за направление взгляда берут ось -Z, чтобы получить левую систему координат
  - near: всё ближе этого значения (по Z-координате) будет отсекаться
  - far: всё дальше этого значения (по Z-координате) будет отсекаться
  - fovx, fovy - угол обзора по X и Y
- Матрица проекции: см. лекцию 4

Cubemaps

- Cubemap-текстура: (концептуально) набор из шести 2D текстур, понимаемых как грани *виртуального* куба

Cubemaps

- Cubemap-текстура: (концептуально) набор из шести 2D текстур, понимаемых как грани *виртуального* куба
- Удобно хранить изображения, натянутые на куб/сферу

Cubemaps

- Cubemap-текстура: (концептуально) набор из шести 2D текстур, понимаемых как грани *виртуального* куба
- Удобно хранить изображения, натянутые на куб/сферу
- Текстурная координата в шейдере - вектор направления; вычисляется пересечение луча из центра куба в этом направлении с поверхностью куба (не зависит от длины вектора направления и от размеров куба), по этому пересечению вычисляется пиксель в конкретной грани cubemap текстуры

Cubemaps: примеры использования

- Хочется нарисовать некое бесконечно удалённое окружение сцены (skybox) - небо, далёкие горы, и т.п.: нужна текстура для всех направлений $\Rightarrow$ cubemaps!

Cubemaps: примеры использования

- Хочется нарисовать некое бесконечно удалённое окружение сцены (skybox) - небо, далёкие горы, и т.п.: нужна текстура для всех направлений $\Rightarrow$ cubemaps!
- Environment mapping (как skybox, только для отражений) - нужно знать, как выглядит сцена в некотором направлении из отражающей точки $\Rightarrow$ cubemaps!

Cubemaps: примеры использования

- Хочется нарисовать некое бесконечно удалённое окружение сцены (skybox) - небо, далёкие горы, и т.п.: нужна текстура для всех направлений $\Rightarrow$ cubemaps!
- Environment mapping (как skybox, только для отражений) - нужно знать, как выглядит сцена в некотором направлении из отражающей точки $\Rightarrow$ cubemaps!
- Отражения (как environment mapping, только динамический) $\Rightarrow$ cubemaps!

Cubemaps: примеры использования

- Хочется нарисовать некое бесконечно удалённое окружение сцены (skybox) - небо, далёкие горы, и т.п.: нужна текстура для всех направлений $\Rightarrow$ cubemaps!
- Environment mapping (как skybox, только для отражений) - нужно знать, как выглядит сцена в некотором направлении из отражающей точки $\Rightarrow$ cubemaps!
- Отражения (как environment mapping, только динамический) $\Rightarrow$ cubemaps!
- Shadow maps от точечного источника - источник светит во все стороны из фиксированной точки: нужно знать расстояние до ближайшего объекта в каждом направлении $\Rightarrow$ cubemaps!

Рендеринг в cubemap

- Cubemap - это текстура $\Rightarrow$ чтобы рисовать в неё, нужен фреймбуфер

Рендеринг в cubemap

- Cubemap - это текстура $\Rightarrow$ чтобы рисовать в неё, нужен фреймбуфер
- Cubemap целиком нельзя сделать attachment'ом фреймбуфера, можно только отдельные её грани

Рендеринг в cubemap

- Cubemap - это текстура $\Rightarrow$ чтобы рисовать в неё, нужен фреймбуфер
- Cubemap целиком нельзя сделать attachment'ом фреймбуфера, можно только отдельные её грани
- `glFramebufferTexture2D` - принимает (в отличие от `glFramebufferTexture`) дополнительный параметр `textarget`, который можно (в том числе) использовать как индикатор конкретной грани cubemap текстуры:
  - `GL_TEXTURE_CUBE_MAP_POSITIVE_X`
  - `GL_TEXTURE_CUBE_MAP_NEGATIVE_X`
  - `GL_TEXTURE_CUBE_MAP_POSITIVE_Y`
  - `GL_TEXTURE_CUBE_MAP_NEGATIVE_Y`
  - `GL_TEXTURE_CUBE_MAP_POSITIVE_Z`
  - `GL_TEXTURE_CUBE_MAP_NEGATIVE_Z`

Рендеринг в cubemap

- Нужно 6 фреймбуферов, к каждому нужно присоединить соответствующую грань cubemap текстуры

Рендеринг в cubemap

- Нужно 6 фреймбуферов, к каждому нужно присоединить соответствующую грань cubemap текстуры
- Рисуем сцену 6 раз, по одному разу для каждой грани

Рендеринг в cubemap

- Нужно 6 фреймбуферов, к каждому нужно присоединить соответствующую грань cubemap текстуры
- Рисуем сцену 6 раз, по одному разу для каждой грани
- Для каждой грани нужна перспективная проекция
  - Центр - точка, с точки зрения которой рисуется cubemap (для теней - позиция источника света, для отражений - координаты отражающей точки)

Рендеринг в cubemap

- Нужно 6 фреймбуферов, к каждому нужно присоединить соответствующую грань cubemap текстуры
- Рисуем сцену 6 раз, по одному разу для каждой грани
- Для каждой грани нужна перспективная проекция
  - Центр - точка, с точки зрения которой рисуется cubemap (для теней - позиция источника света, для отражений - координаты отражающей точки)
  - fovx = fovy = 90° (из геометрии куба)

Рендеринг в cubemap: псевдокод

```
// Инициализация:
GLuint fbos[6];
...
for (i in 0..5) {
  glBindFramebuffer(GL_DRAW_FRAMEBUFFER, fbos[i]);
  glFramebufferTexture2D(GL_DRAW_FRAMEBUFFER,
    GL_COLOR_ATTACHMENT0,
    GL_TEXTURE_CUBE_MAP_POSITIVE_X + i, cubemap, 0);
}

// Рендеринг:
...
glViewport(0, 0, cubemap_size, cubemap_size);
set_uniform("projection", cubemap_projection);
for (i in 0..5) {
  glBindFramebuffer(GL_DRAW_FRAMEBUFFER, fbos[i]);
  set_uniform("view", cubemap_view[i]);
  draw_scene();
}
```

Дистрибуция программ на C++

- Чтобы программа запустилась на конкретном компьютере, нужно, чтобы:

Дистрибуция программ на C++

- Чтобы программа запустилась на конкретном компьютере, нужно, чтобы:
  - Операционная система понимала формат исполняемого файла

Дистрибуция программ на C++

- Чтобы программа запустилась на конкретном компьютере, нужно, чтобы:
  - Операционная система понимала формат исполняемого файла
  - Бинарный код подходил для используемого процессора и операционной системы

Дистрибуция программ на C++

- Чтобы программа запустилась на конкретном компьютере, нужно, чтобы:
  - Операционная система понимала формат исполняемого файла
  - Бинарный код подходил для используемого процессора и операционной системы
  - Присутствовали все необходимые динамические библиотеки

Форматы исполняемых файлов

- Windows: PE (Portable Executable)
- Linux: ELF (Executable and Linkable Format)
- MacOS: Mach-O (Mach Object)

Форматы исполняемых файлов

- Содержат:

Форматы исполняемых файлов

- Содержат:
  - Magic number для идентификации формата

Форматы исполняемых файлов

- Содержат:
  - Magic number для идентификации формата
  - Метаинформация: битность (32/64), CPU (набор инструкций - ISA), ABI, endianess, etc

Форматы исполняемых файлов

- Содержат:
  - Magic number для идентификации формата
  - Метаинформация: битность (32/64), CPU (набор инструкций - ISA), ABI, endianess, etc
  - Блоки с данными и кодом

Форматы исполняемых файлов

- Содержат:
  - Magic number для идентификации формата
  - Метаинформация: битность (32/64), CPU (набор инструкций - ISA), ABI, endianess, etc
  - Блоки с данными и кодом
  - Таблицы релокации (для функций из динамических библиотек)

Форматы исполняемых файлов

- Содержат:
  - Magic number для идентификации формата
  - Метаинформация: битность (32/64), CPU (набор инструкций - ISA), ABI, endianess, etc
  - Блоки с данными и кодом
  - Таблицы релокации (для функций из динамических библиотек)
  - Список зависимостей - динамических библиотек

Форматы исполняемых файлов

- Обычно компилятор по умолчанию генерирует исполняемый файл для системы, на которой происходит сборка (host)

Форматы исполняемых файлов

- Обычно компилятор по умолчанию генерирует исполняемый файл для системы, на которой происходит сборка (host)
- Кросс-компиляция: компиляция программы, предназначенной для запуска на системе (target), отличной от той, на которой происходит сборка (host)

Форматы исполняемых файлов

- Обычно компилятор по умолчанию генерирует исполняемый файл для системы, на которой происходит сборка (host)
- Кросс-компиляция: компиляция программы, предназначенной для запуска на системе (target), отличной от той, на которой происходит сборка (host)
- Кросс-компиляция с Linux (host) на Windows (target): MinGW

Форматы исполняемых файлов

- Обычно компилятор по умолчанию генерирует исполняемый файл для системы, на которой происходит сборка (host)
- Кросс-компиляция: компиляция программы, предназначенной для запуска на системе (target), отличной от той, на которой происходит сборка (host)
- Кросс-компиляция с Linux (host) на Windows (target): MinGW
- Кросс-компиляция с Windows (host) на Linux (target): Cygwin + crosstool-ng

Форматы исполняемых файлов

- Обычно компилятор по умолчанию генерирует исполняемый файл для системы, на которой происходит сборка (host)
- Кросс-компиляция: компиляция программы, предназначенной для запуска на системе (target), отличной от той, на которой происходит сборка (host)
- Кросс-компиляция с Linux (host) на Windows (target): MinGW
- Кросс-компиляция с Windows (host) на Linux (target): Cygwin + crosstool-ng
- Кросс-компиляция на MacOS (target): нет
  - Apple требует, чтобы программы для MacOS собирались на MacOS, на железе от Apple, и были подписаны платными сертификатами разработчика

CPU

- Скомпилированный код - байт-код для исполнения процессором

CPU

- Скомпилированный код - байт-код для исполнения процессором
- Если конкретный процессор не понимает этот байт-код, программу не запустить

CPU

- Скомпилированный код - байт-код для исполнения процессором
- Если конкретный процессор не понимает этот байт-код, программу не запустить
- Обычно линейки процессоров сохраняют обратную совместимость: программу для старого процессора можно запустить на более новом процессоре

CPU

- Скомпилированный код - байт-код для исполнения процессором
- Если конкретный процессор не понимает этот байт-код, программу не запустить
- Обычно линейки процессоров сохраняют обратную совместимость: программу для старого процессора можно запустить на более новом процессоре
- Компиляторы позволяют настраивать то, под какой процессор генерируется код, e.g. для GCC: `gcc -march=haswell ...` (полный список для GCC: gcc.gnu.org/onlinedocs/gcc/x86-Options.html)

ABI

- Скомпилированный код содержит вызовы внешних функций (системные вызовы и функции из динамически загруженных библиотек)

ABI

- Скомпилированный код содержит вызовы внешних функций (системные вызовы и функции из динамически загруженных библиотек)
- На уровне ассемблера нет понятия вызова функции $\Rightarrow$ нужны соглашения о вызове (calling conventions) - как передаются аргументы в функции, как возвращается результат, как передаётся параметр `this`, и т.п.

ABI

- Скомпилированный код содержит вызовы внешних функций (системные вызовы и функции из динамически загруженных библиотек)
- На уровне ассемблера нет понятия вызова функции $\Rightarrow$ нужны соглашения о вызове (calling conventions) - как передаются аргументы в функции, как возвращается результат, как передаётся параметр `this`, и т.п.
- Скомпилированный код содержит работу со структурами данных

ABI

- Скомпилированный код содержит вызовы внешних функций (системные вызовы и функции из динамически загруженных библиотек)
- На уровне ассемблера нет понятия вызова функции $\Rightarrow$ нужны соглашения о вызове (calling conventions) - как передаются аргументы в функции, как возвращается результат, как передаётся параметр `this`, и т.п.
- Скомпилированный код содержит работу со структурами данных
- На уровне ассемблера нет структур данных $\Rightarrow$ нужно соглашение о том, как выглядят структуры в памяти (memory layout)

ABI

- Скомпилированный код содержит вызовы внешних функций (системные вызовы и функции из динамически загруженных библиотек)
- На уровне ассемблера нет понятия вызова функции $\Rightarrow$ нужны соглашения о вызове (calling conventions) - как передаются аргументы в функции, как возвращается результат, как передаётся параметр `this`, и т.п.
- Скомпилированный код содержит работу со структурами данных
- На уровне ассемблера нет структур данных $\Rightarrow$ нужно соглашение о том, как выглядят структуры в памяти (memory layout)
- (для C++) нужно соглашение о деталях обработки исключений, разрешения namespace'ов и перегрузки функций

ABI

- Скомпилированный код содержит вызовы внешних функций (системные вызовы и функции из динамически загруженных библиотек)
- На уровне ассемблера нет понятия вызова функции $\Rightarrow$ нужны соглашения о вызове (calling conventions) - как передаются аргументы в функции, как возвращается результат, как передаётся параметр `this`, и т.п.
- Скомпилированный код содержит работу со структурами данных
- На уровне ассемблера нет структур данных $\Rightarrow$ нужно соглашение о том, как выглядят структуры в памяти (memory layout)
- (для C++) нужно соглашение о деталях обработки исключений, разрешения namespace'ов и перегрузки функций
- За всё это отвечает ABI (Application Binary Interface)

ABI

- GCC: начиная с 3ей версии использует стандартизованный Itanium ABI

ABI

- GCC: начиная с 3ей версии использует стандартизованный Itanium ABI
- MSVC: начиная с Visual Studio 2015 сохраняет ABI-совместимость кода, скомпилированного разными версиями MSVC

ABI

- GCC: начиная с 3ей версии использует стандартизованный Itanium ABI
- MSVC: начиная с Visual Studio 2015 сохраняет ABI-совместимость кода, скомпилированного разными версиями MSVC
- Clang: почти ABI-совместим с GCC; умеет генерировать код, ABI-совместимый с MSVC

ABI

- GCC: начиная с 3ей версии использует стандартизованный Itanium ABI
- MSVC: начиная с Visual Studio 2015 сохраняет ABI-совместимость кода, скомпилированного разными версиями MSVC
- Clang: почти ABI-совместим с GCC; умеет генерировать код, ABI-совместимый с MSVC
- Код может быть ABI-совместим с другим кодом с точки зрения C, но не с точки зрения C++

Динамические библиотеки

- Стандартная библиотека C (`libc.so` в Linux, `msvcrt.dll` в Windows) и C++ (`libstdc++.so` у GCC, `libc++.so` у Clang, `msvcp140.dll` и т.п. у Visual Studio, etc.)

Динамические библиотеки

- Стандартная библиотека C (`libc.so` в Linux, `msvcrt.dll` в Windows) и C++ (`libstdc++.so` у GCC, `libc++.so` у Clang, `msvcp140.dll` и т.п. у Visual Studio, etc.)
- Библиотека математических функций (`libm.so`)

Динамические библиотеки

- Стандартная библиотека C (`libc.so` в Linux, `msvcrt.dll` в Windows) и C++ (`libstdc++.so` у GCC, `libc++.so` у Clang, `msvcp140.dll` и т.п. у Visual Studio, etc.)
- Библиотека математических функций (`libm.so`)
- Библиотека с реализацией OpenGL (`libGL.so` в Linux, `opengl32.dll` в Windows)

Динамические библиотеки

- Стандартная библиотека C (`libc.so` в Linux, `msvcrt.dll` в Windows) и C++ (`libstdc++.so` у GCC, `libc++.so` у Clang, `msvcp140.dll` и т.п. у Visual Studio, etc.)
- Библиотека математических функций (`libm.so`)
- Библиотека с реализацией OpenGL (`libGL.so` в Linux, `opengl32.dll` в Windows)
- Вспомогательные библиотеки (SDL, загрузчик OpenGL, etc)

Динамические библиотеки

- Некоторые библиотеки можно слинковать статически - их код будет влит в код исполняемого файла, и при исполнении они уже не нужны

Динамические библиотеки

- Некоторые библиотеки можно слинковать статически - их код будет влит в код исполняемого файла, и при исполнении они уже не нужны
  - Стандартная библиотека C++

Динамические библиотеки

- Некоторые библиотеки можно слинковать статически - их код будет влит в код исполняемого файла, и при исполнении они уже не нужны
  - Стандартная библиотека C++
  - Загрузчик OpenGL

Динамические библиотеки

- Некоторые библиотеки можно слинковать статически - их код будет влит в код исполняемого файла, и при исполнении они уже не нужны
  - Стандартная библиотека C++
  - Загрузчик OpenGL
- Остальные нужно где-то найти!

Динамические библиотеки

- Многие библиотеки (стандартная библиотека C, математические функции, реализация OpenGL) есть в любой системе и лежат по каким-то стандартным путям (`/usr/lib` в Linux, `C:\Program Files` в Windows)

Динамические библиотеки

- Многие библиотеки (стандартная библиотека C, математические функции, реализация OpenGL) есть в любой системе и лежат по каким-то стандартным путям (`/usr/lib` в Linux, `C:\Program Files` в Windows)
- Поиск дополнительных зависимостей специфичен для системы:

Динамические библиотеки

- Многие библиотеки (стандартная библиотека C, математические функции, реализация OpenGL) есть в любой системе и лежат по каким-то стандартным путям (`/usr/lib` в Linux, `C:\Program Files` в Windows)
- Поиск дополнительных зависимостей специфичен для системы:
  - Windows: если библиотеки не найдены в системных путях, они ищутся в директории с исполняемым файлом

Динамические библиотеки

- Многие библиотеки (стандартная библиотека C, математические функции, реализация OpenGL) есть в любой системе и лежат по каким-то стандартным путям (`/usr/lib` в Linux, `C:\Program Files` в Windows)
- Поиск дополнительных зависимостей специфичен для системы:
  - Windows: если библиотеки не найдены в системных путях, они ищутся в директории с исполняемым файлом
  - Linux, MacOS: у исполняемых файлов есть свойство RPATH (runtime path) - список путей, по которым нужно искать зависимости; в качестве RPATH можно указать текущую директорию (`.` для Linux, специальные макросы для MacOS)

Динамические библиотеки

- Многие библиотеки (стандартная библиотека C, математические функции, реализация OpenGL) есть в любой системе и лежат по каким-то стандартным путям (`/usr/lib` в Linux, `C:\Program Files` в Windows)
- Поиск дополнительных зависимостей специфичен для системы:
  - Windows: если библиотеки не найдены в системных путях, они ищутся в директории с исполняемым файлом
  - Linux, MacOS: у исполняемых файлов есть свойство RPATH (runtime path) - список путей, по которым нужно искать зависимости; в качестве RPATH можно указать текущую директорию (`.` для Linux, специальные макросы для MacOS)
- С таким подходом зависимости можно поставлять в собранном виде рядом с исполняемым файлом