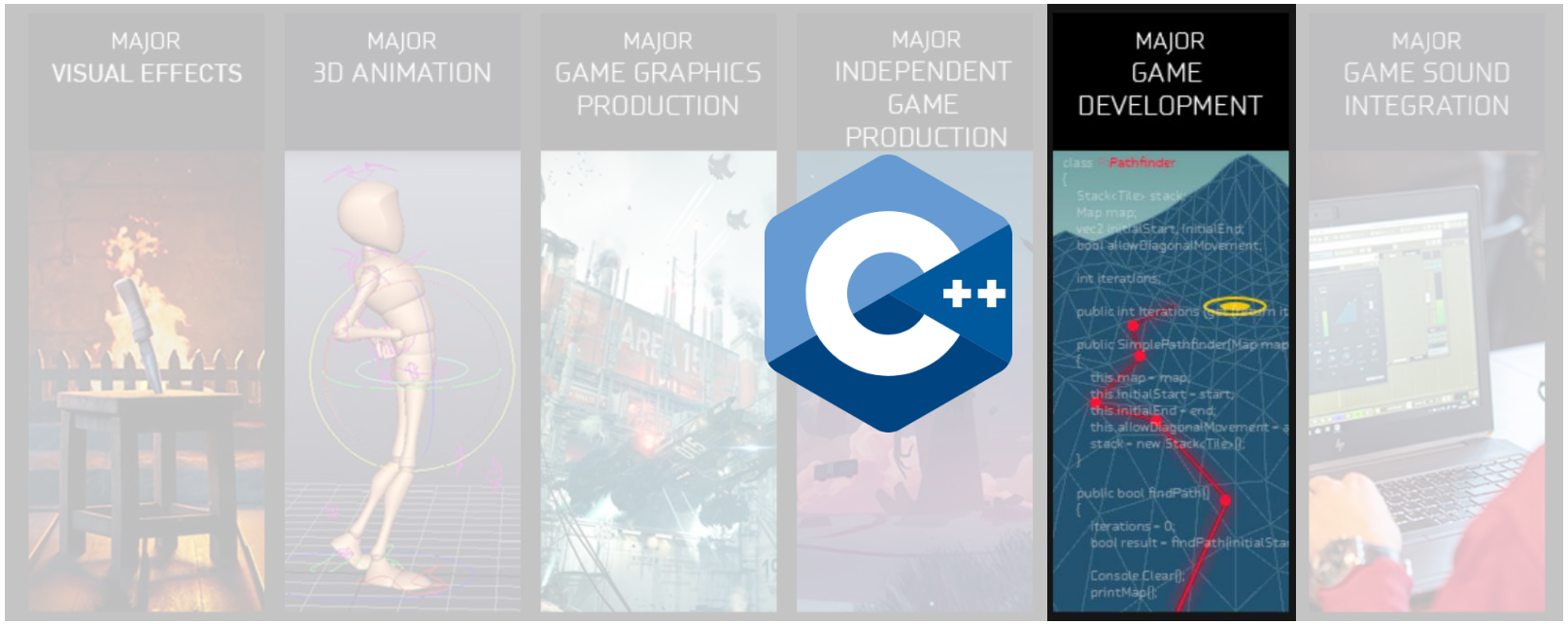
# From Pure ISO C++20 To Compute Shaders

KOEN SAMYN


# About me

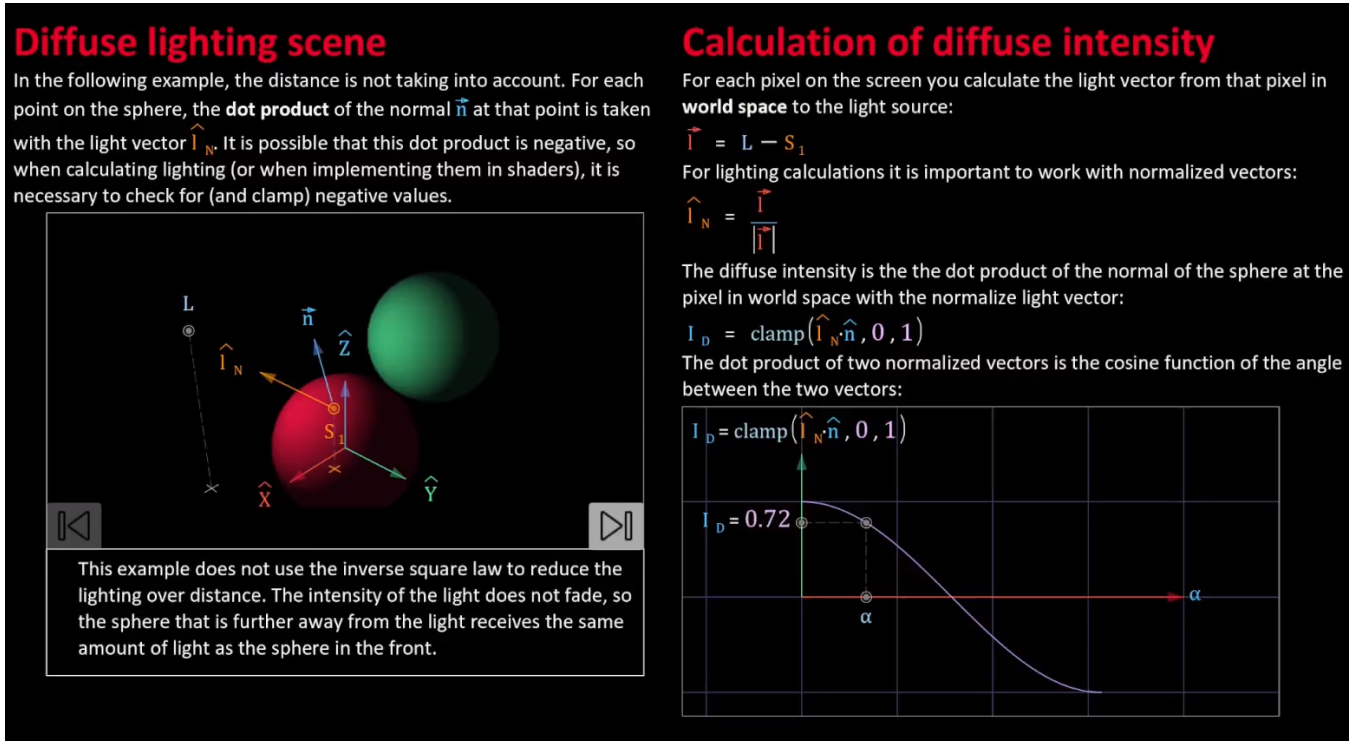
- Koen Samyn
- Program Coordinator Game Development at DAE




https://digitalartsandentertainment.be/

# About me

- Koen Samyn
- Program Coordinator Game Development at DAE



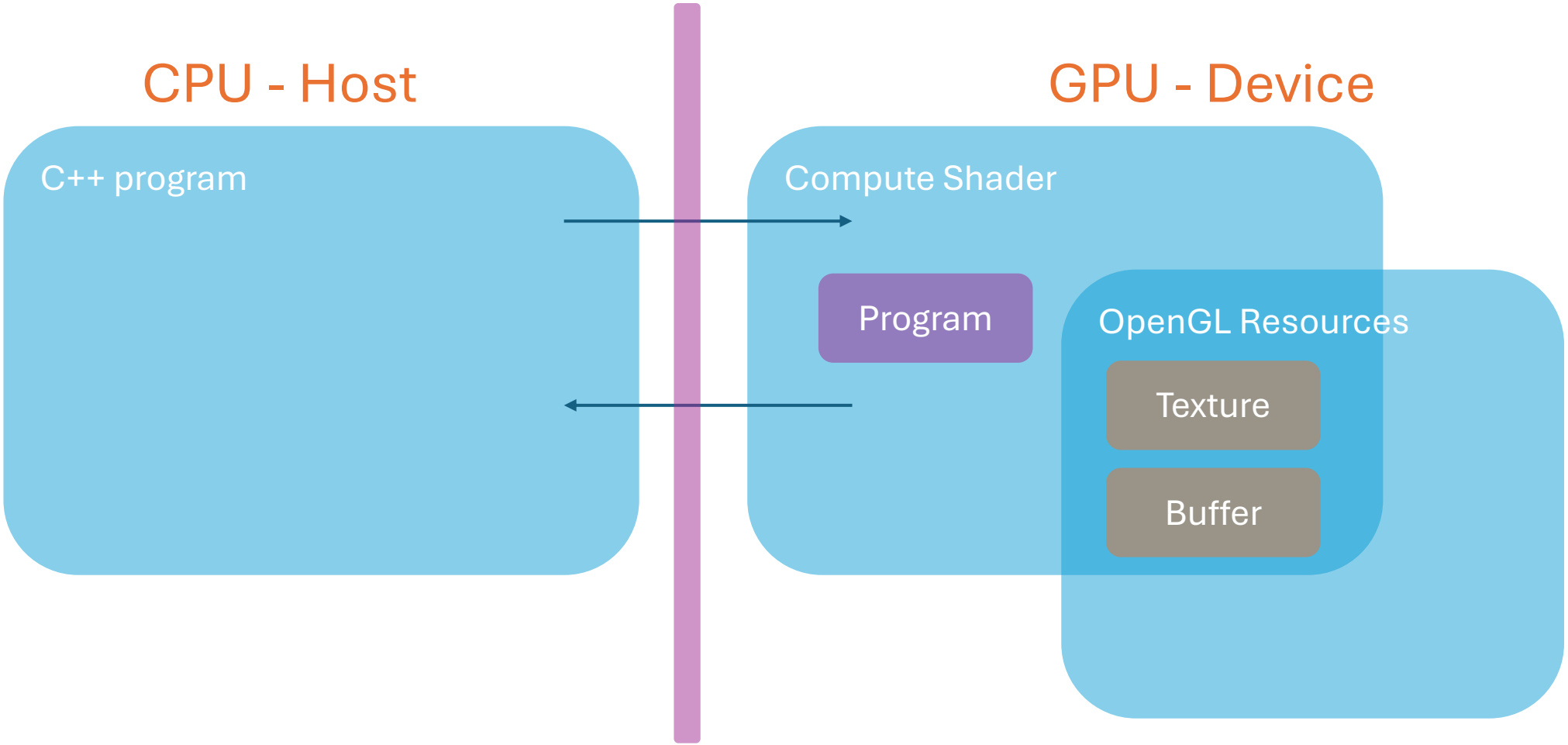
# Why this project?

- Mental load to develop compute shaders is heavy
  - Still developing C++ skills.
  - Management layer needed for GPU resources.
  - Debugging GPU programs is hard.
- Programming model is different
  - Workgroups / threads
  - Homogeneous parallelism: every thread performs the same task
  - Existing C++ code can be hard to convert

# Is this new?

- A lof of movement in this area and existing frameworks
  - CUDA /HIP
  - SyCL
  - CompuShady (Python)
  - Slang
  - OpenMP
- Niche I am trying to fill:
  - Non-proprietary
  - Lightweight
  - **Educational focus**: teach transferable skills.
  - Focus on techniques that are transferable to graphics programming shaders: vertex/fragment/mesh shaders

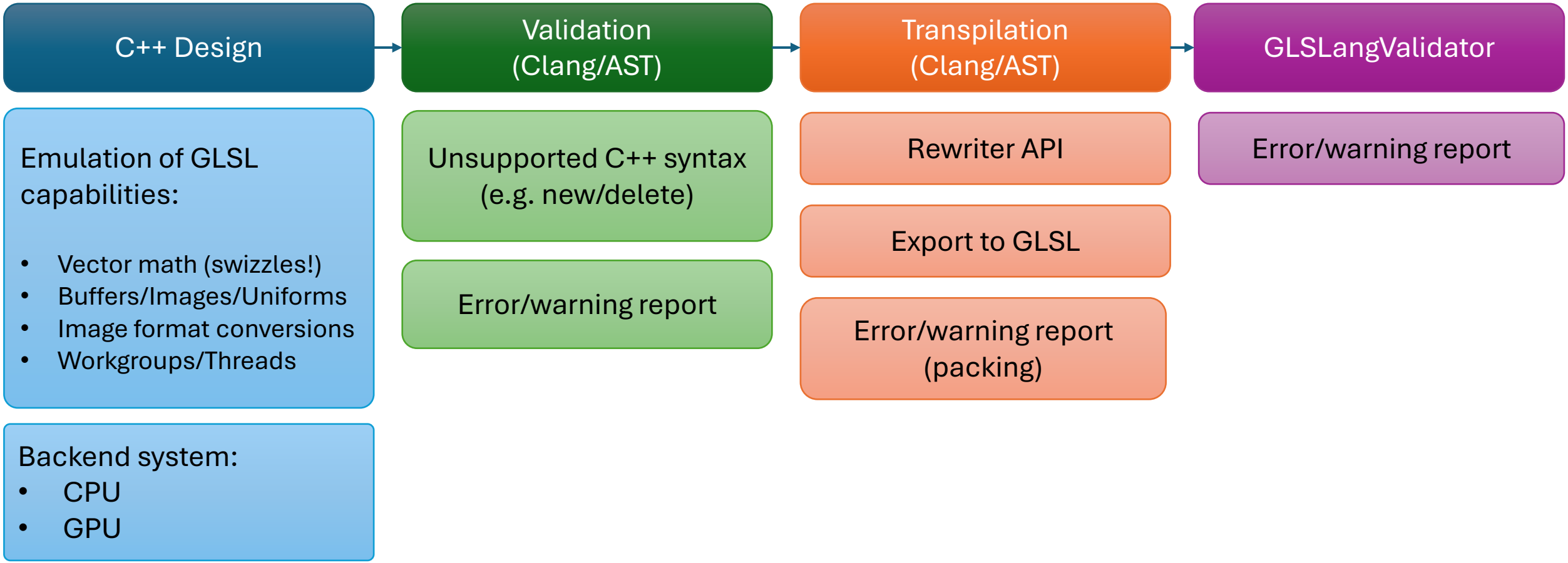
# What is a compute shader?



# Design goals

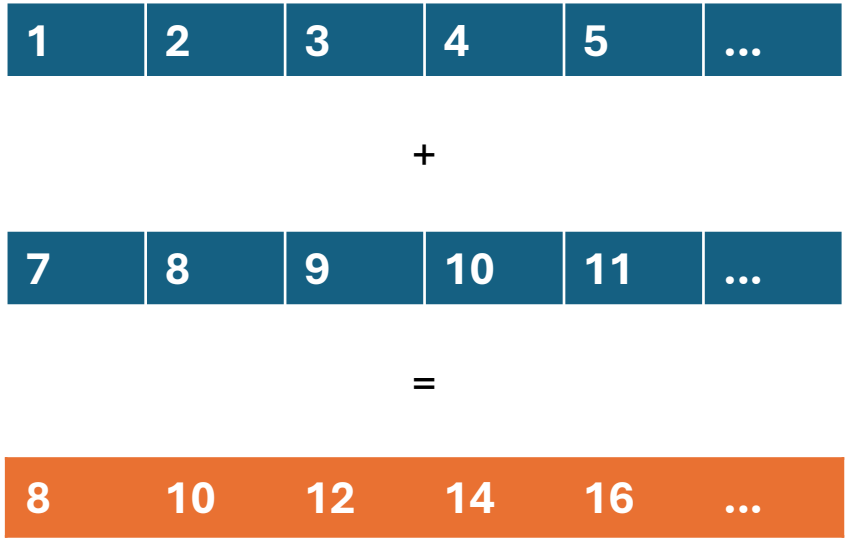
- Start with C++20 → no new keywords
- Compact code
- Close mapping with GPU parallelism
  - Workgroups/threads
- Equivalence between CPU and GPU execution

# Overview

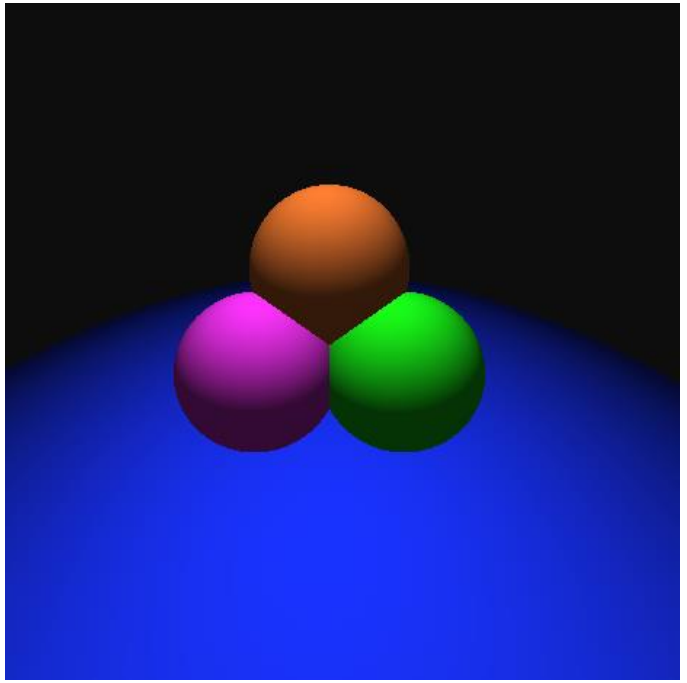

C++ Design
Validation (Clang/AST)
Transpilation (Clang/AST)
GLSLangValidator
Emulation of GLSL capabilities:
• Vector math (swizzles!)
• Buffers/Images/Uniforms
• Image format conversions
• Workgroups/Threads
Backend system:
• CPU
• GPU
Unsupported C++ syntax (e.g. new/delete)
Error/warning report
Rewriter API
Export to GLSL
Error/warning report (packing)
Error/warning report

# Cases

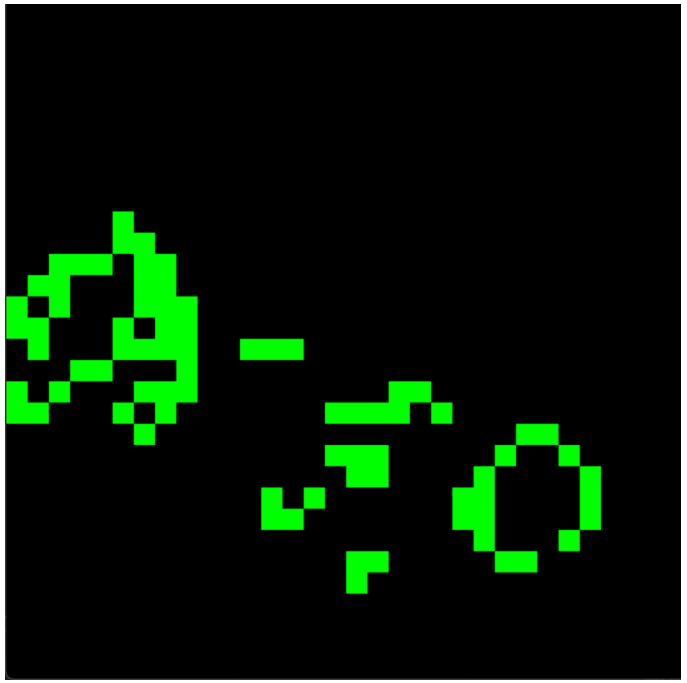
Adding two vectors of floats




Raytracing



Game of life



# Case 1: Adding floats – Compute Shader

```
1  #version 430
2
3
4
5
6
7
8
9
10
11
12
13
14
15
```

Compute shader compatible with OpenGL version 4.3

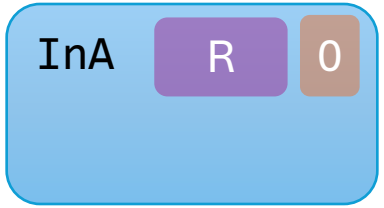
# Case 1: Adding floats – Compute Shader

```
#version 430
layout(local_size_x = 256) in;
```

**Workgroup** size → defines the number of threads that will be created for one **workgroup**.

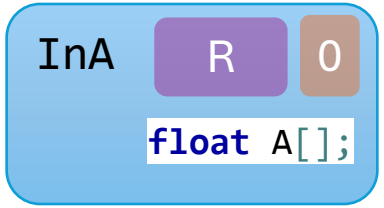
# Case 1: Adding floats – Compute Shader

```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4
5   };
6
7
8
9
10
11
12
13
14
15
```



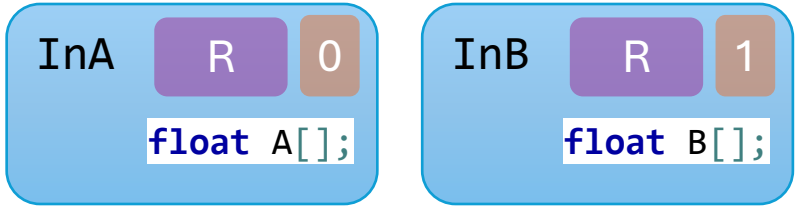
# Case 1: Adding floats – Compute Shader

```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4       float A[];
5   };
6
7
8
9
10
11
12
13
14
15
```



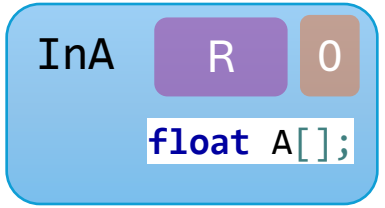
# Case 1: Adding floats – Compute Shader

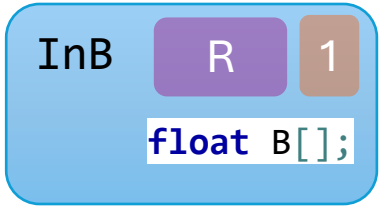
```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4       float A[];
5   };
6   layout( binding = 1 ) readonly buffer InB {
7       float B[];
8   };
9
10
11
12
13
14
15
```

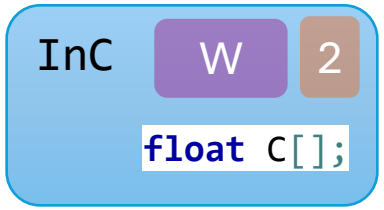


# Case 1: Adding floats – Compute Shader

```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4       float A[];
5   };
6   layout( binding = 1 ) readonly buffer InB {
7       float B[];
8   };
9   layout( binding = 2 ) writeonly buffer OutC
10  {   float C[];
11  };
12
13
14
15
```

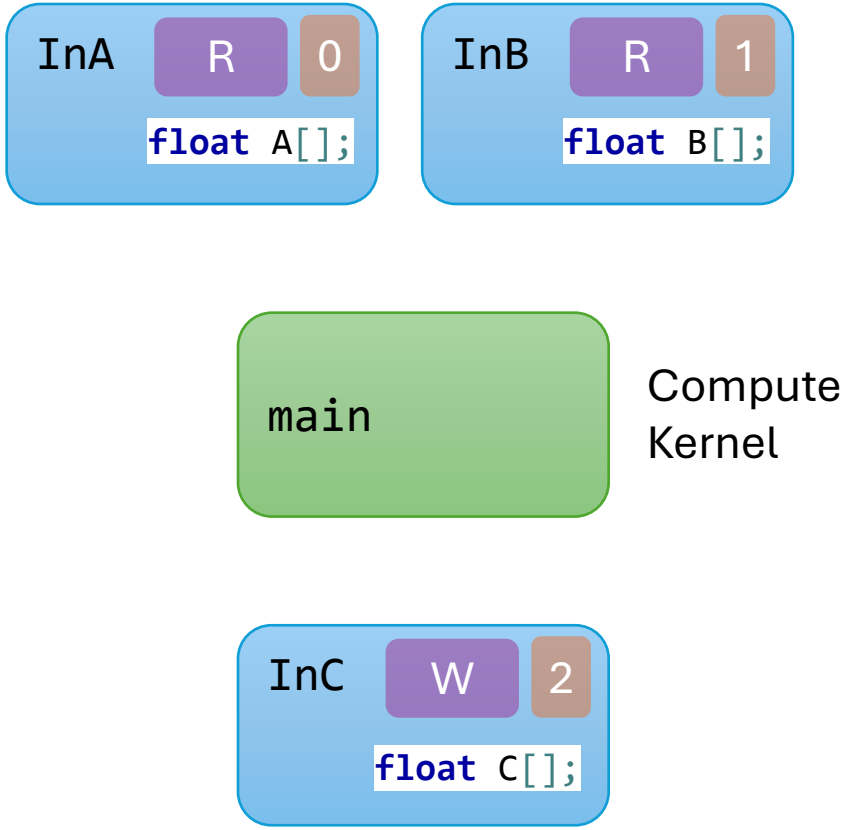


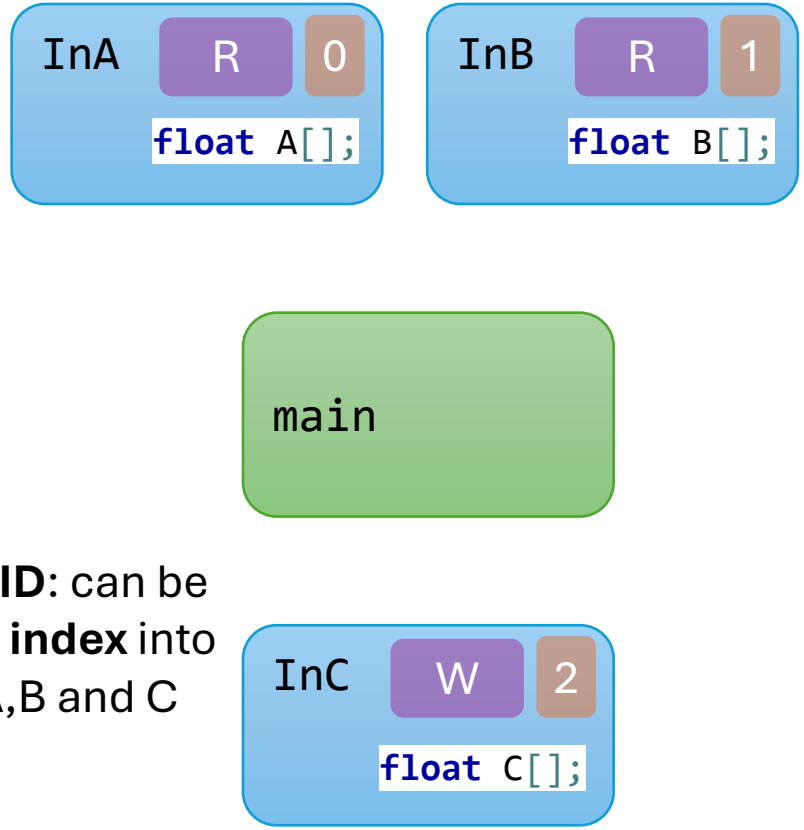
# Case 1: Adding floats – Compute Shader

```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4       float A[];
5   };
6   layout( binding = 1 ) readonly buffer InB {
7       float B[];
8   };
9   layout( binding = 2 ) writeonly buffer OutC {
10      float C[];
11  };
12  void main() {
13
14
15  };
```



# Case 1: Adding floats – Compute Shader

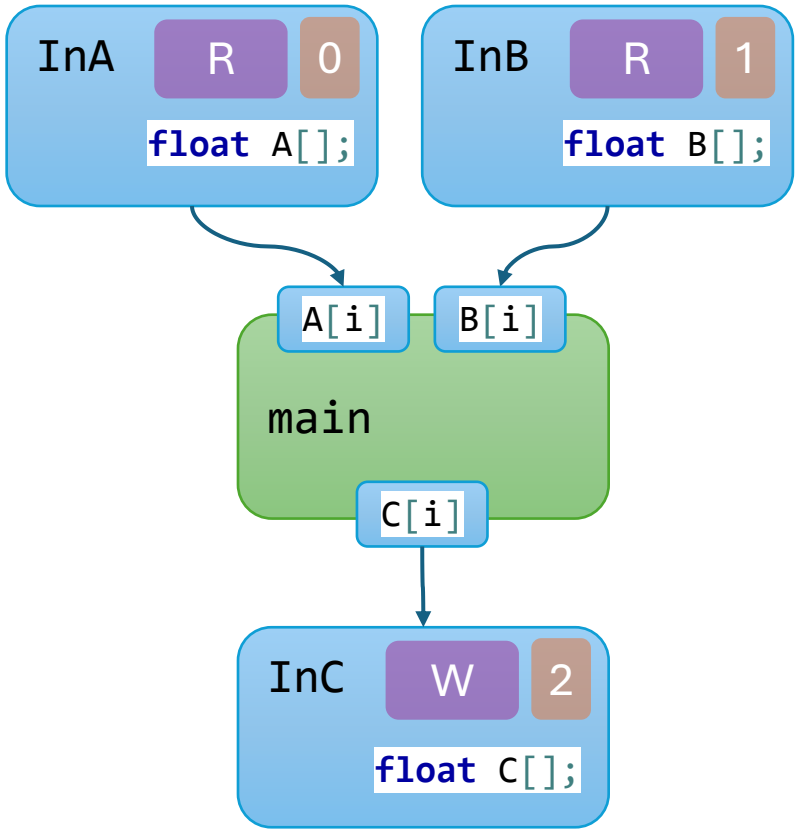
```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4       float A[];
5   };
6   layout( binding = 1 ) readonly buffer InB {
7       float B[];
8   };
9   layout( binding = 2 ) writeonly buffer OutC {
10      float C[];
11  };
12  void main() {
13      uint i = gl_GlobalInvocationID.x;
14
15  };
```




**Thread ID**: can be used as **index** into arrays A,B and C

# Case 1: Adding floats – Compute Shader

```
1   #version 430
2   layout(local_size_x = 256) in;
3   layout( binding = 0 ) readonly buffer InA {
4       float A[];
5   };
6   layout( binding = 1 ) readonly buffer InB {
7       float B[];
8   };
9   layout( binding = 2 ) writeonly buffer OutC {
10      float C[];
11  };
12  void main() {
13      uint i = gl_GlobalInvocationID.x;
14      C[i] = A[i] + B[i];
15  };
```



# Case 1: C++ equivalent

```
1   #include "vec.hpp"
2
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5
6
7
8
9
10
11
12
13
14  };
```

`vec.hpp` : for vector types: `uvecn`, `ivecn`, `vecn`, `dvecn`, `nvecn`

`Kernel annotation:` clang annotation to find kernels in the AST (see further)

C++ Design

# Case 1: C++ equivalent

```
1   #include "vec.hpp"
2
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5         static constexpr char fileLocation[] = “floatadder";
6
7
8
9
10
11
12
13
14  };
```

Where to store the generated glsl file.

C++ Design

# Case 1: C++ equivalent

```
1   #include "vec.hpp"
2
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5         static constexpr char fileLocation[] = “floatadder";
6         uvec3 local_size{ 256, 1, 1 };
7
8
9
10
11
12
13
14  };
```

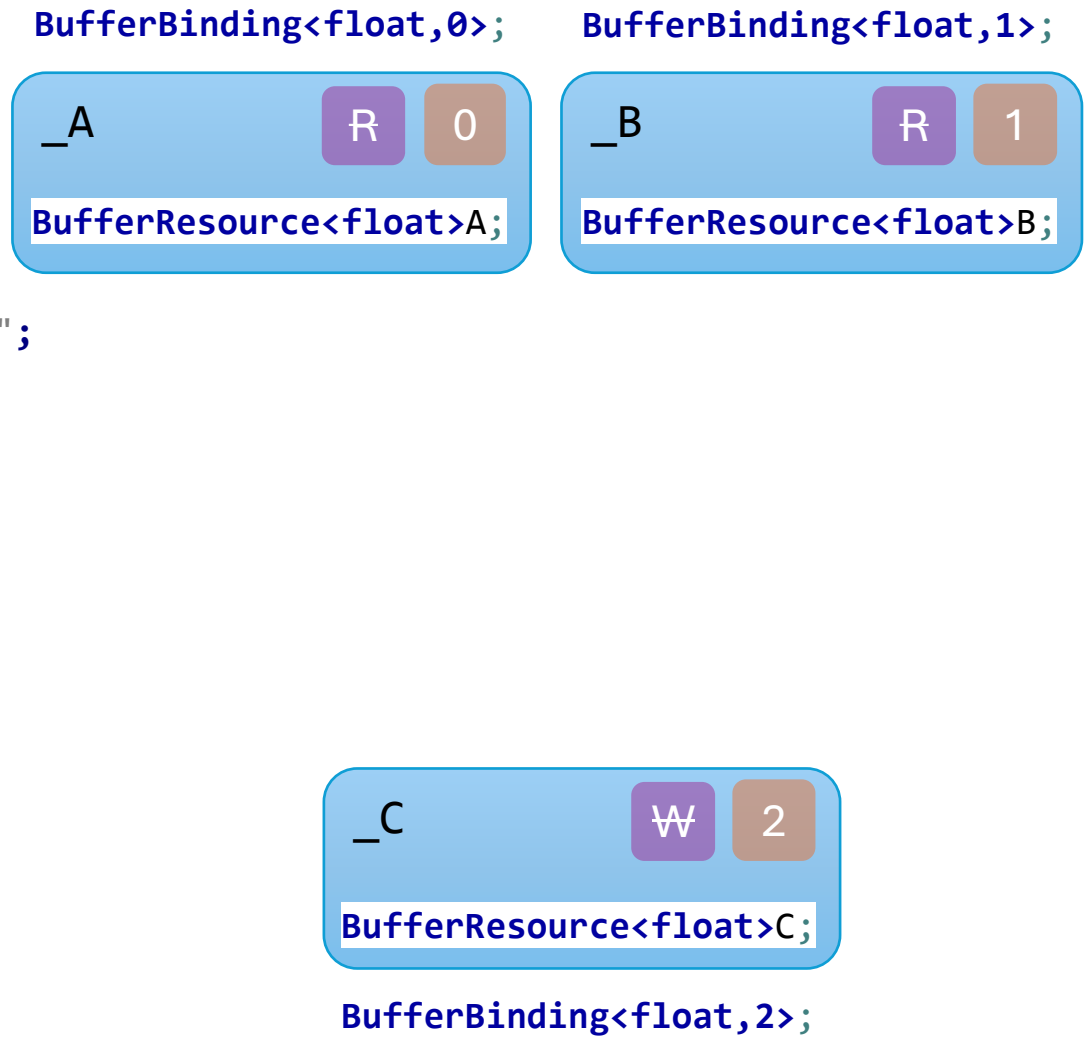
Corresponds with :

```
layout(local_size_x = 256) in;
```

C++ Design

# Case 1: C++ equivalent

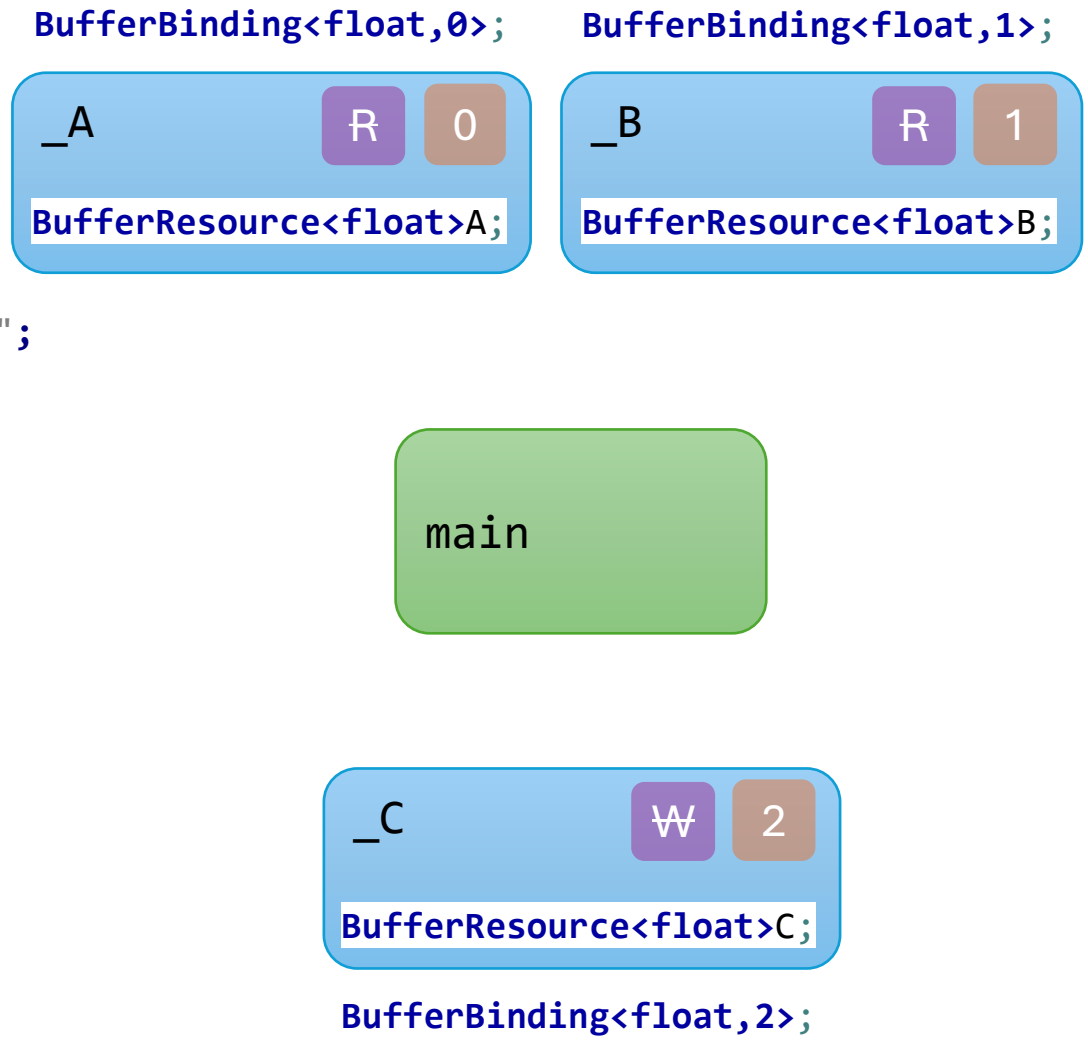
```
1   #include "vec.hpp"
2   #include "computebackend.hpp"
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5         static constexpr char fileLocation[] = “floatadder";
6         uvec3 local_size{ 256, 1, 1 };
7         BufferBinding< float , 0> A;
8         BufferBinding< float , 1> B;
9         BufferBinding< float , 2> C;
10
11
12
13
14  };
```




C++ Design

# Case 1: C++ equivalent

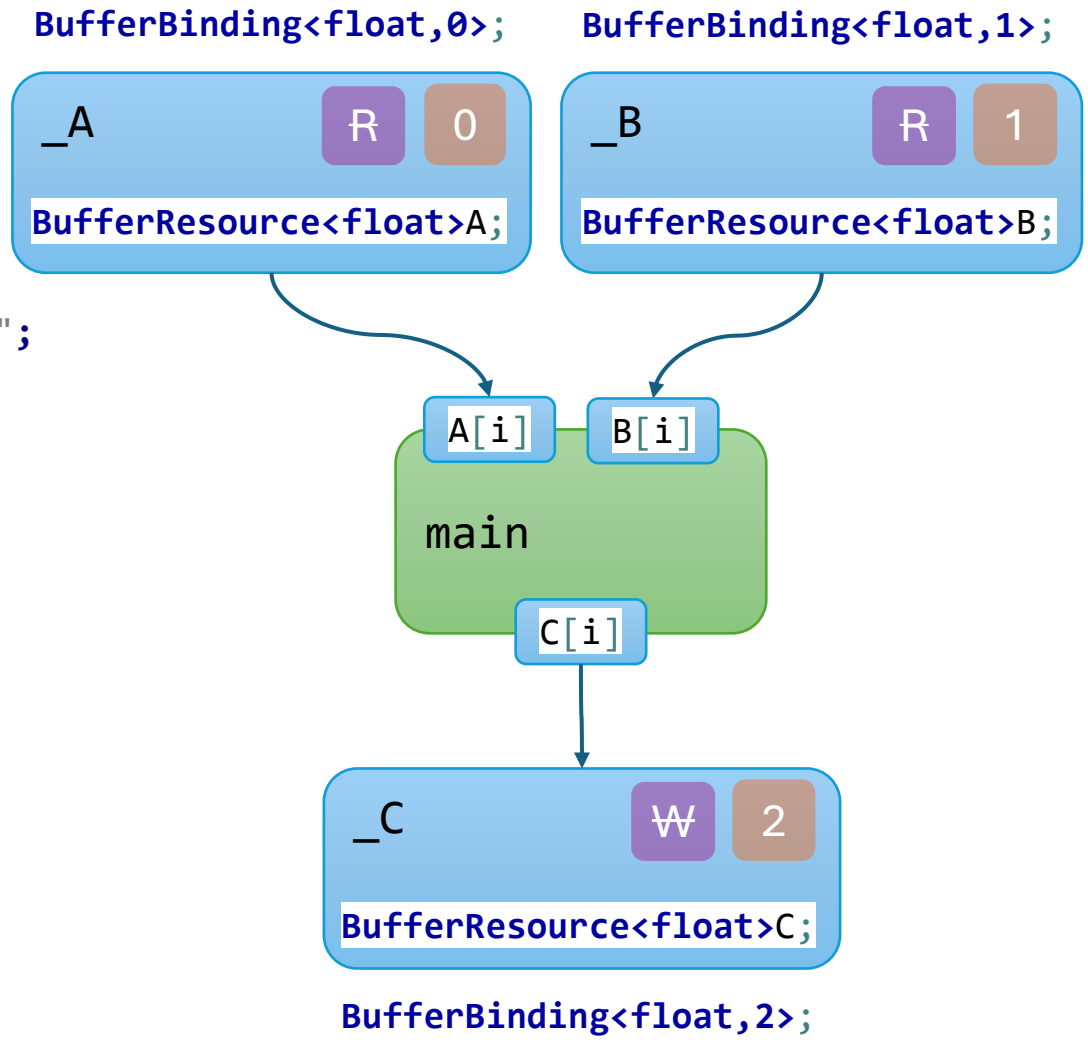
```
1   #include "vec.hpp"
2   #include "computebackend.hpp"
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5         static constexpr char fileLocation[] = “floatadder";
6         uvec3 local_size{ 256, 1, 1 };
7         BufferBinding< float , 0> A;
8         BufferBinding< float , 1> B;
9         BufferBinding< float , 2> C;
10        void main() {
11
12
13        }
14  };
```




C++ Design

# Case 1: C++ equivalent

```
1   #include "vec.hpp"
2   #include "computebackend.hpp"
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5         static constexpr char fileLocation[] = “floatadder";
6         uvec3 local_size{ 256, 1, 1 };
7         BufferBinding< float , 0> A;
8         BufferBinding< float , 1> B;
9         BufferBinding< float , 2> C;
10        void main() {
11              tc::uint i = tc::gl_GlobalInvocationID.x;
12              C[i] = A[i] + B[i];
13        }
14  };
```




C++ Design

# Case 1 : Validation

- Clang 22.0 : ***matchers*** and ***visitors***
- Custom C++ tool called via CMake

```
-CXXRecordDecl 0x26d24bde080 <line:24:1, line:104:1> line:24:38 struct GameOfLifeKernel definition
 |-DefinitionData pass_in_registers aggregate standard_layout trivially_copyable has_constexpr_non_copy_move_ctor
 | |-DefaultConstructor exists non_trivial constexpr needs_implicit defaulted_is_constexpr
 | |-CopyConstructor simple trivial has_const_param needs_implicit implicit_has_const_param
 | |-MoveConstructor exists simple trivial needs_implicit
 | |-CopyAssignment simple trivial has_const_param needs_implicit implicit_has_const_param
 | |-MoveAssignment exists simple trivial needs_implicit
 | `-Destructor simple irrelevant trivial constexpr needs_implicit
 |-AnnotateAttr 0x26d24bde1b8 <col:10, col:34> "kernel"
 |-CXXRecordDecl 0x26d24bde258 <col:1, col:38> col:38 implicit struct GameOfLifeKernel
 |-VarDecl 0x26d24bde358 <line:26:2, col:41> col:24 fileLocation 'const char[11]' static inline constexpr cinit
 | |-value: Array size=11
 | | |-elements: Int 103, Int 97, Int 109, Int 101
 | | |-elements: Int 111, Int 102, Int 108, Int 105
 | | |-elements: Int 102, Int 101
 | | `-filler: 1 x Int 0
 | `-StringLiteral 0x26d24bde440 <col:41> 'const char[11]' "gameoflife"
 |-FieldDecl 0x26d24bde558 <line:27:2, col:32> col:12 referenced local_size 'sf::uvec3'
 | `-CXXConstructExpr 0x26d24bded38 <col:22, col:32> 'sf::uvec3' 'void (uint32_t, uint32_t, uint32_t)' list
 |   |-ImplicitCastExpr 0x26d24bdecf0 <col:24> 'uint32_t':'unsigned int' <IntegralCast>
 |   | `-IntegerLiteral 0x26d24bdec20 <col:24> 'int' 18
 |   |-ImplicitCastExpr 0x26d24bded08 <col:27> 'uint32_t':'unsigned int' <IntegralCast>
 |   | `-IntegerLiteral 0x26d24bdec48 <col:27> 'int' 18
 |   `-ImplicitCastExpr 0x26d24bded20 <col:30> 'uint32_t':'unsigned int' <IntegralCast>
 |     `-IntegerLiteral 0x26d24bdec70 <col:30> 'int' 1
 |-FieldDecl 0x26d24bdea90 <line:28:2, col:16> col:16 t 'Test<float, 3>'
 `-CXXMethodDecl 0x26d24bdeb48 <line:29:2, line:32:2> line:29:7 main 'void ()' implicit-inline
   `-CompoundStmt 0x26d24be96a0 <col:13, line:32:2>
     `-DeclStmt 0x26d24be9688 <line:31:3, col:30>
       `-VarDecl 0x26d24bded98 <col:3, col:29> col:12 idx 'unsigned int' cinit
```

Validation (Clang/AST)

# Find kernel structs via matchers

```
1   #include "vec.hpp"
2   #include "computebackend.hpp"
3   struct [[clang::annotate("kernel")]] FloatAdder
4   {
5       static constexpr char fileLocation[] = “floatadder";
6       uvec3 local_size{ 256, 1, 1 };
7       BufferBinding< float , 0> A;
8       BufferBinding< float , 1> B;
9       BufferBinding< float , 2> C;
10      void main() {
11          tc::uint i = tc::gl_GlobalInvocationID.x;
12          C[i] = A[i] + B[i];
13      }
14  };
```

```
auto kernelStructMatcher = cxxRecordDecl(
        isDefinition(),

        hasAttr(attr::Annotate),

        hasMethod(
                cxxMethodDecl(hasName("main"))
        )
        hasDescendant(
                varDecl(

                hasName("fileLocation"),
                        isConstexpr()
                ).bind("fileLocation")
        ),
        hasDescendant(
                fieldDecl(
                        hasName("local_size")
                ).bind("localSizeVar")
        ),
).bind("kernelStruct");
```

Validation (Clang/AST)

# Clang validation

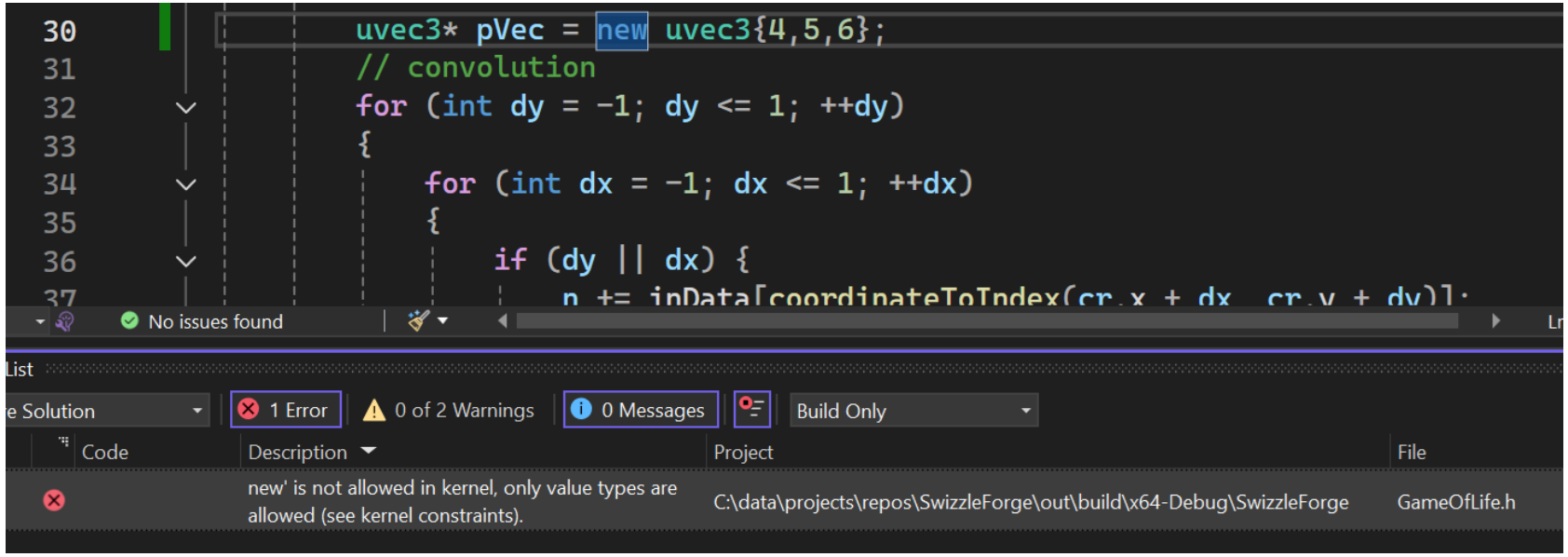
- C++ AST library
- Visit methods
  - VisitFieldDecl(clang::FieldDecl* FD)
  - VisitCXXNewExpr(clang::CXXNewExpr* E)
  - …
- Report errors back to the IDE

**Note**: only look within the found kernel structs

Validation (Clang/AST)

# Clang validation – example 1

```
bool VisitCXXNewExpr(clang::CXXNewExpr* E){
        llvm::errs()
            << E->getBeginLoc().printToString(Context.getSourceManager())
            << " Error: 'new' is not allowed in kernel, only value types are allowed.\n";

        reportError(E->getBeginLoc(), dynamicMemory);
        Valid = false;
        return true;
}
```




Validation (Clang/AST)

# Case 1: Transpilation

```
#include "vec.hpp"
#include "computebackend.hpp"
struct [[clang::annotate("kernel")]] FloatAdder
{
      static constexpr char fileLocation[] = "…";
      uvec3 local_size{ 256, 1, 1 };
      BufferBinding< float , 0> A;
      BufferBinding< float , 1> B;
      BufferBinding< float , 2> C;
      void main() {
            tc::uint i = tc::gl_GlobalInvocationID.x;
            C[i] = A[i] + B[i];
      }
};
```

Transpilation (Clang/AST)

```
#version 430
layout (
            local_size_x = 256,
            local_size_y = 1,
            local_size_z = 1)
in;
layout(set = 0, binding = 0) buffer _ALayout {
            float A[];
};
layout(set = 0, binding = 1) buffer _BLayout {
            float B[];
};
layout(set = 0, binding = 2) buffer _CLayout {
            float C[];
};
void main() {
            uint i = gl_GlobalInvocationID.x;
            C[i] = A[i] + B[i];
}
```

# Case 1: Usage

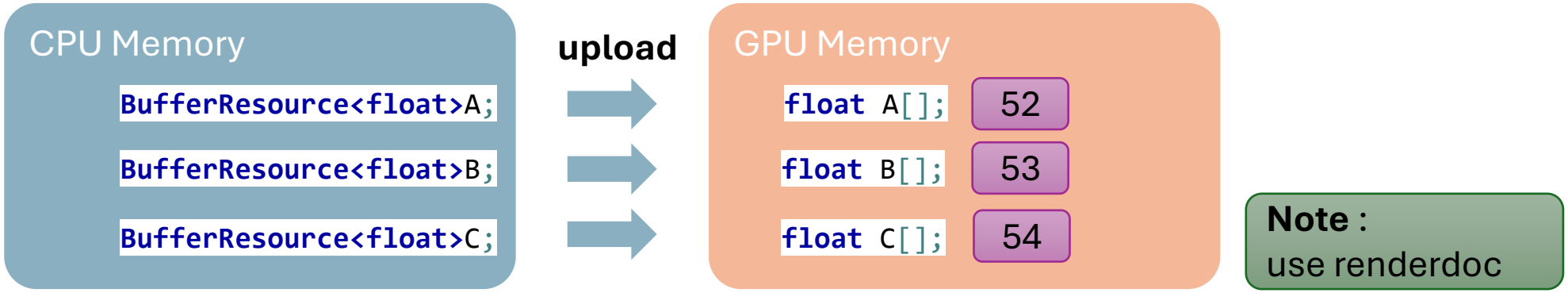
```
using Backend = tc::gpu::GPUBackend;
Backend compute;
```

# Case 1: Usage

```
using Backend = tc::gpu::GPUBackend;
Backend compute;

compute.uploadBuffer(m_pA.get());
compute.uploadBuffer(m_pB.get());
compute.uploadBuffer(m_pC.get());
```

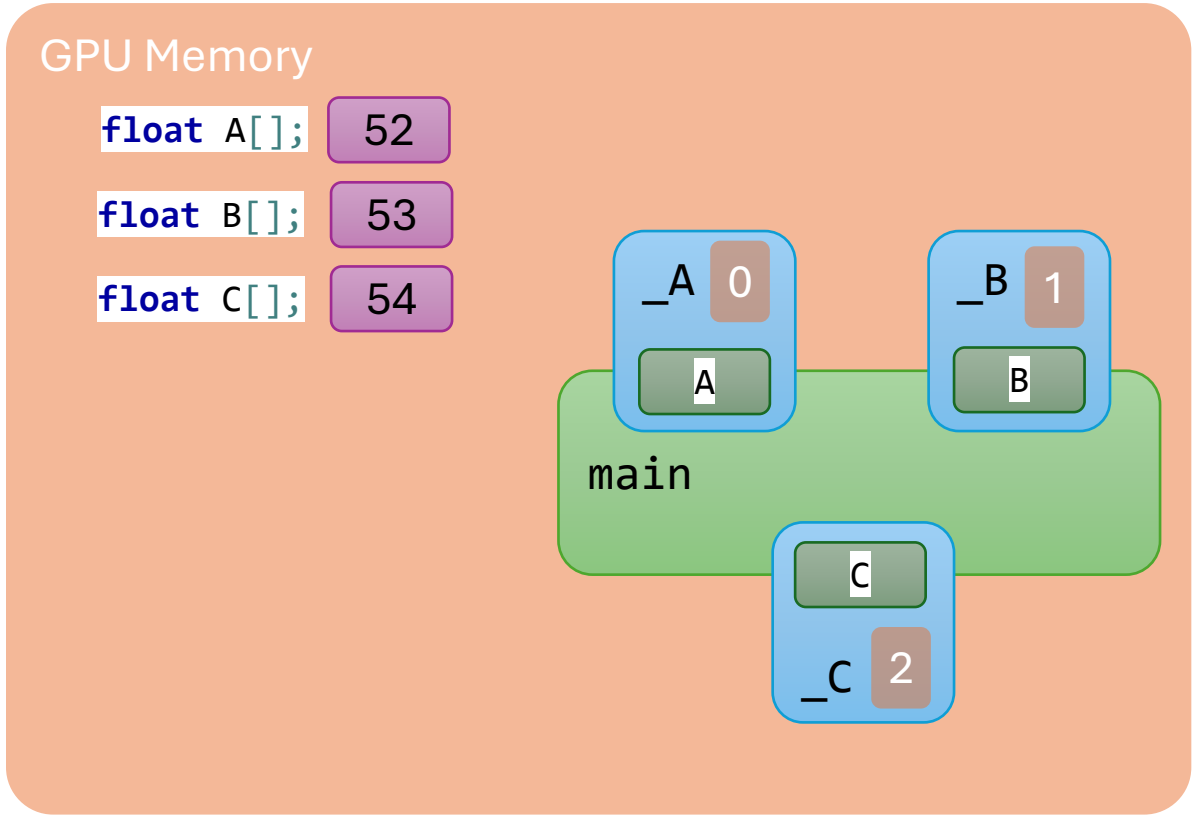
With the definitions for `m_pA`, `m_pB` and `m_pC` given as:

```
std::unique_ptr<tc::BufferResource<float>> m_pA;
std::unique_ptr<tc::BufferResource<float>> m_pB;
std::unique_ptr<tc::BufferResource<float>> m_pC;
```



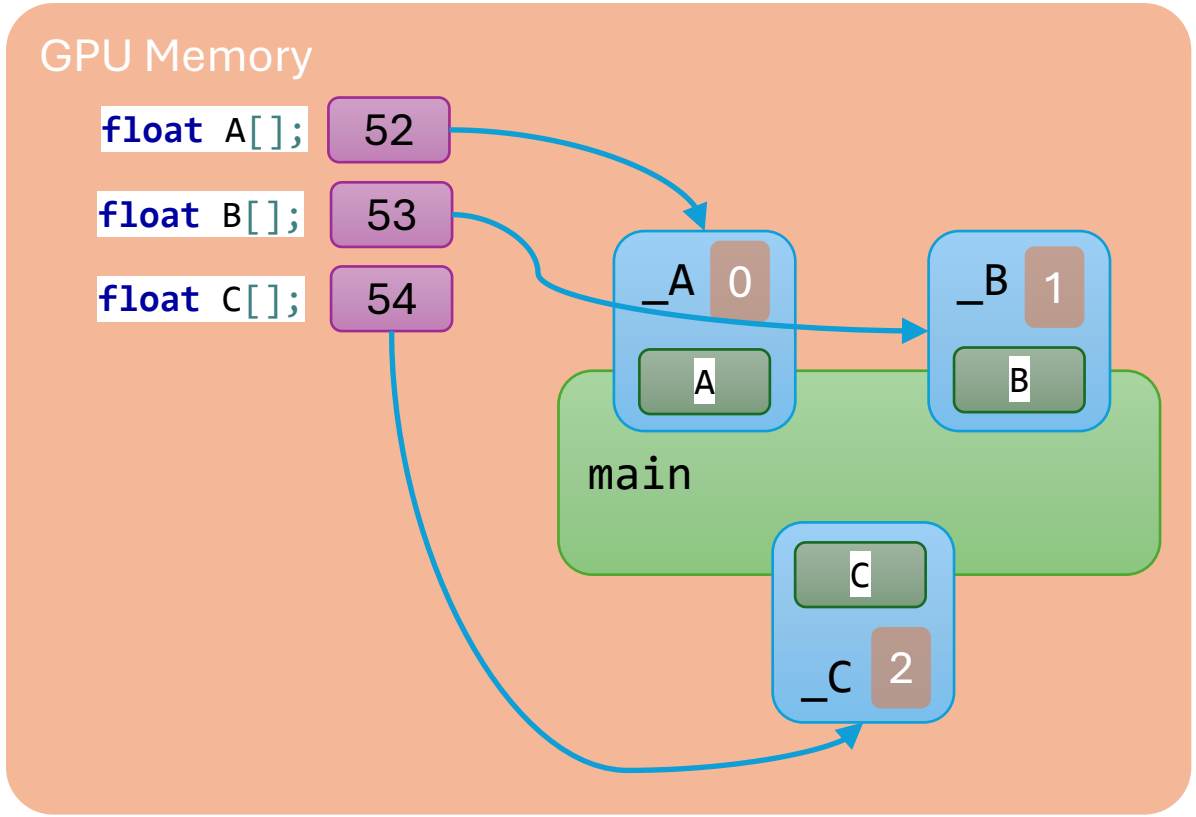
# Case 1: Usage

```
using Backend = tc::gpu::GPUBackend;
Backend compute;

compute.uploadBuffer(m_pA.get());
compute.uploadBuffer(m_pB.get());
compute.uploadBuffer(m_pC.get());

compute.useKernel(m_FloatAdder);
```



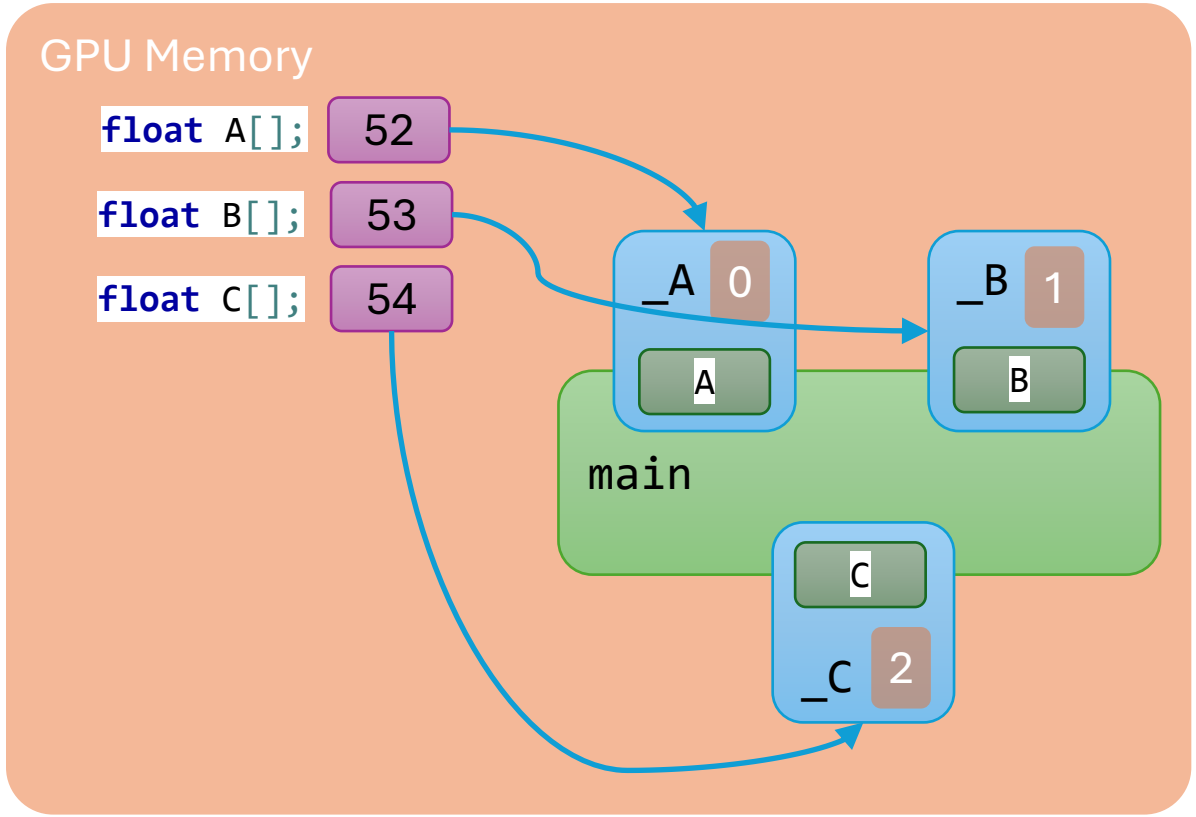
# Case 1: Usage

```
using Backend = tc::gpu::GPUBackend;
Backend compute;

compute.uploadBuffer(m_pA.get());
compute.uploadBuffer(m_pB.get());
compute.uploadBuffer(m_pC.get());

compute.useKernel(m_FloatAdder);

compute.bindBuffer(m_FloatAdder.A);
compute.bindBuffer(m_FloatAdder.B);
compute.bindBuffer(m_FloatAdder.C);
```



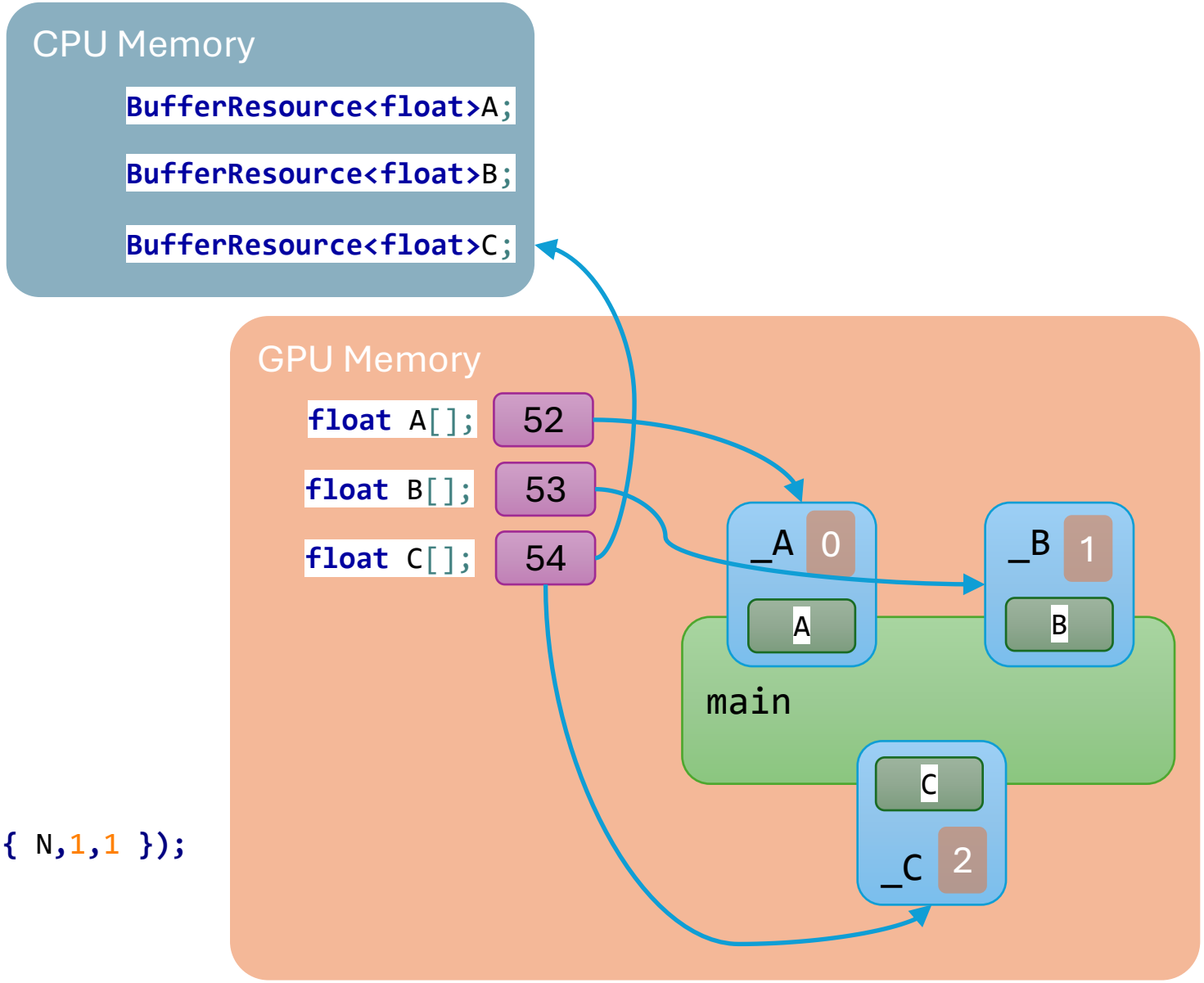
# Case 1: Usage

```
using Backend = tc::gpu::GPUBackend;
Backend compute;

compute.uploadBuffer(m_pA.get());
compute.uploadBuffer(m_pB.get());
compute.uploadBuffer(m_pC.get());

compute.useKernel(m_FloatAdder);

compute.bindBuffer(m_FloatAdder.A);
compute.bindBuffer(m_FloatAdder.B);
compute.bindBuffer(m_FloatAdder.C);
compute.execute(m_FloatAdder, tc::uvec3{ N,1,1 });
```



# Case 1: Usage

```
using Backend = tc::gpu::GPUBackend;
Backend compute;

compute.uploadBuffer(m_pA.get());
compute.uploadBuffer(m_pB.get());
compute.uploadBuffer(m_pC.get());

compute.useKernel(m_FloatAdder);

compute.bindBuffer(m_FloatAdder.A);
compute.bindBuffer(m_FloatAdder.B);
compute.bindBuffer(m_FloatAdder.C);
compute.execute(m_FloatAdder, tc::uvec3{ N,1,1 });
compute.downloadBuffer(m_pC.get());
```



# Case 1: Usage

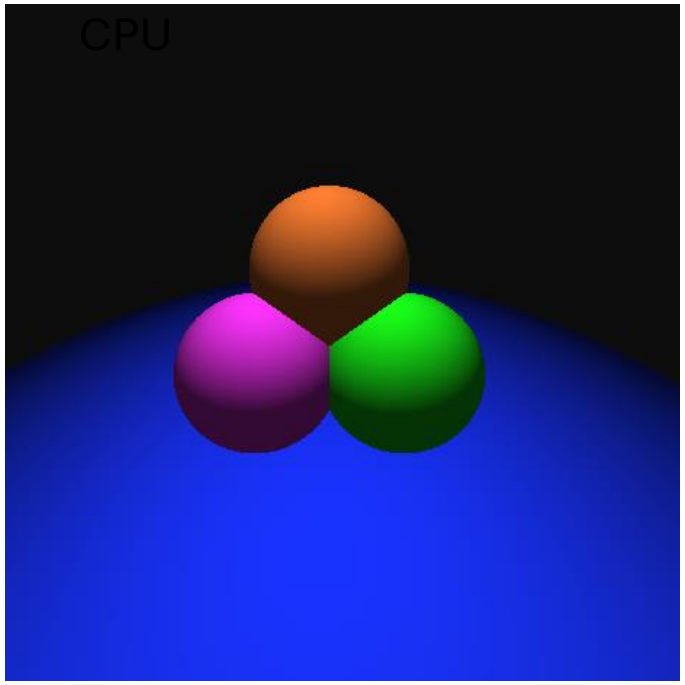
```
using Backend = tc::gpu::GPUBackend;
Backend compute;

compute.uploadBuffer(m_pA.get());
compute.uploadBuffer(m_pB.get());
compute.uploadBuffer(m_pC.get());

compute.useKernel(m_FloatAdder);

compute.bindBuffer(m_FloatAdder.A);
compute.bindBuffer(m_FloatAdder.B);
compute.bindBuffer(m_FloatAdder.C);
compute.execute(m_FloatAdder, tc::uvec3{ N,1,1 });
compute.downloadBuffer(m_pC.get());
```

CPU Memory

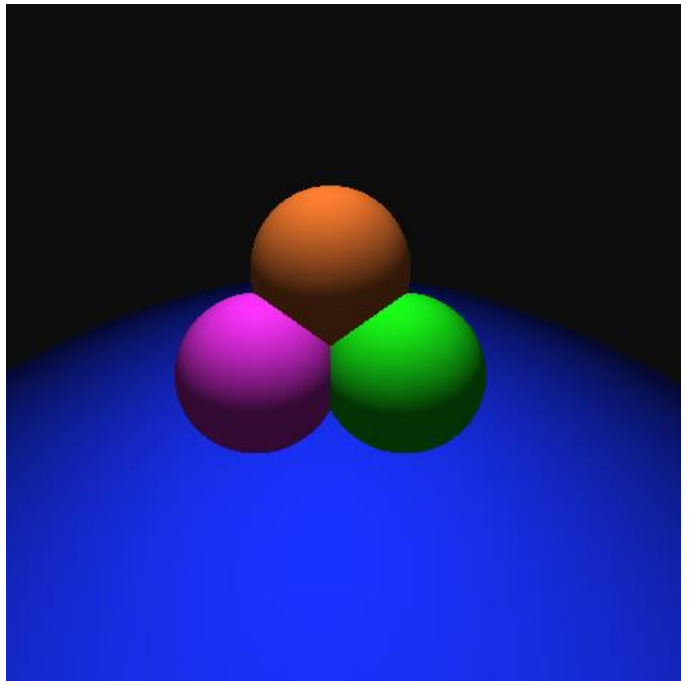
```
BufferResource<float>A;
BufferResource<float>B;
BufferResource<float>C;
```

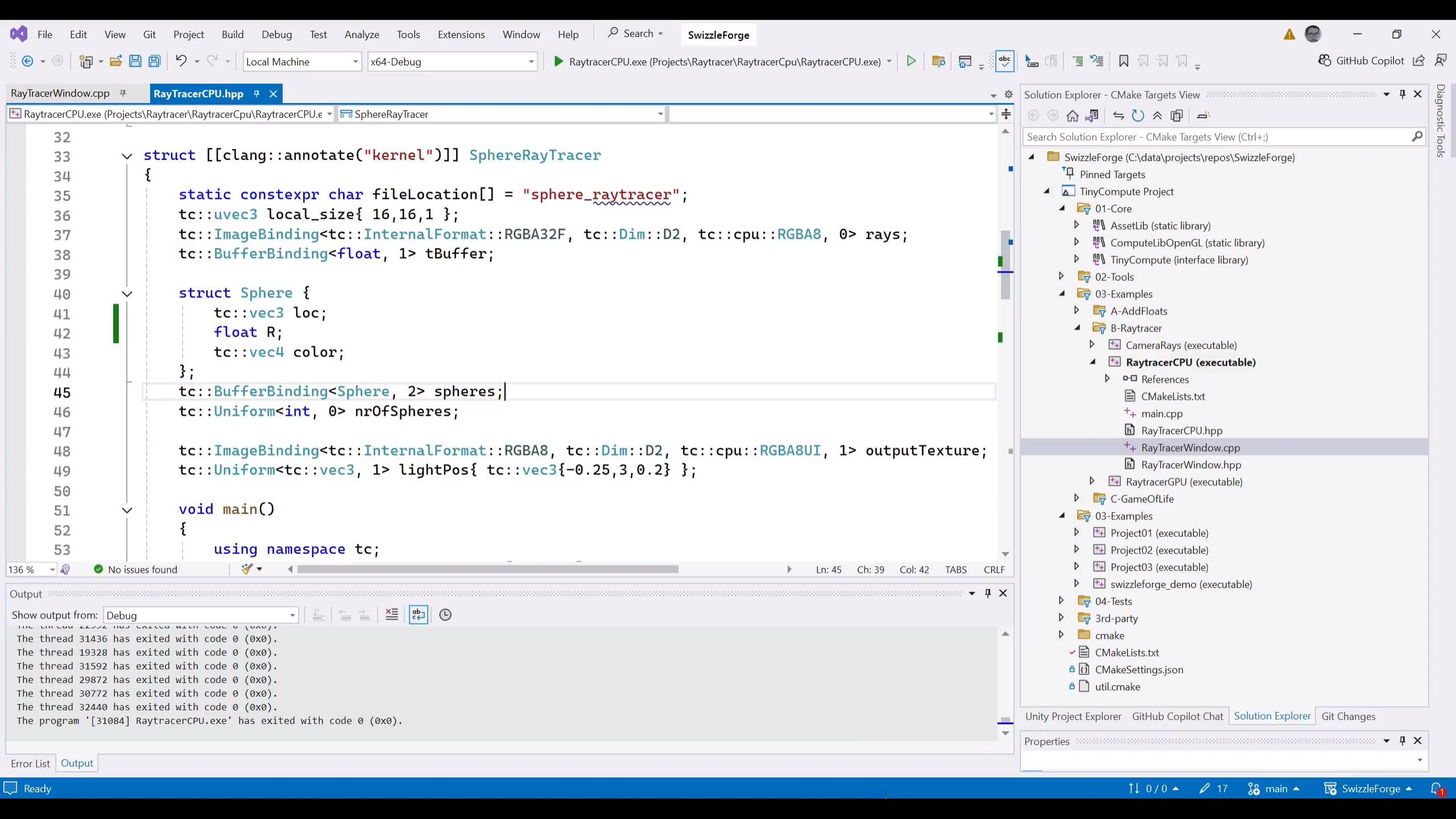
Everything is already in the correct memory space → log error if buffer not bound

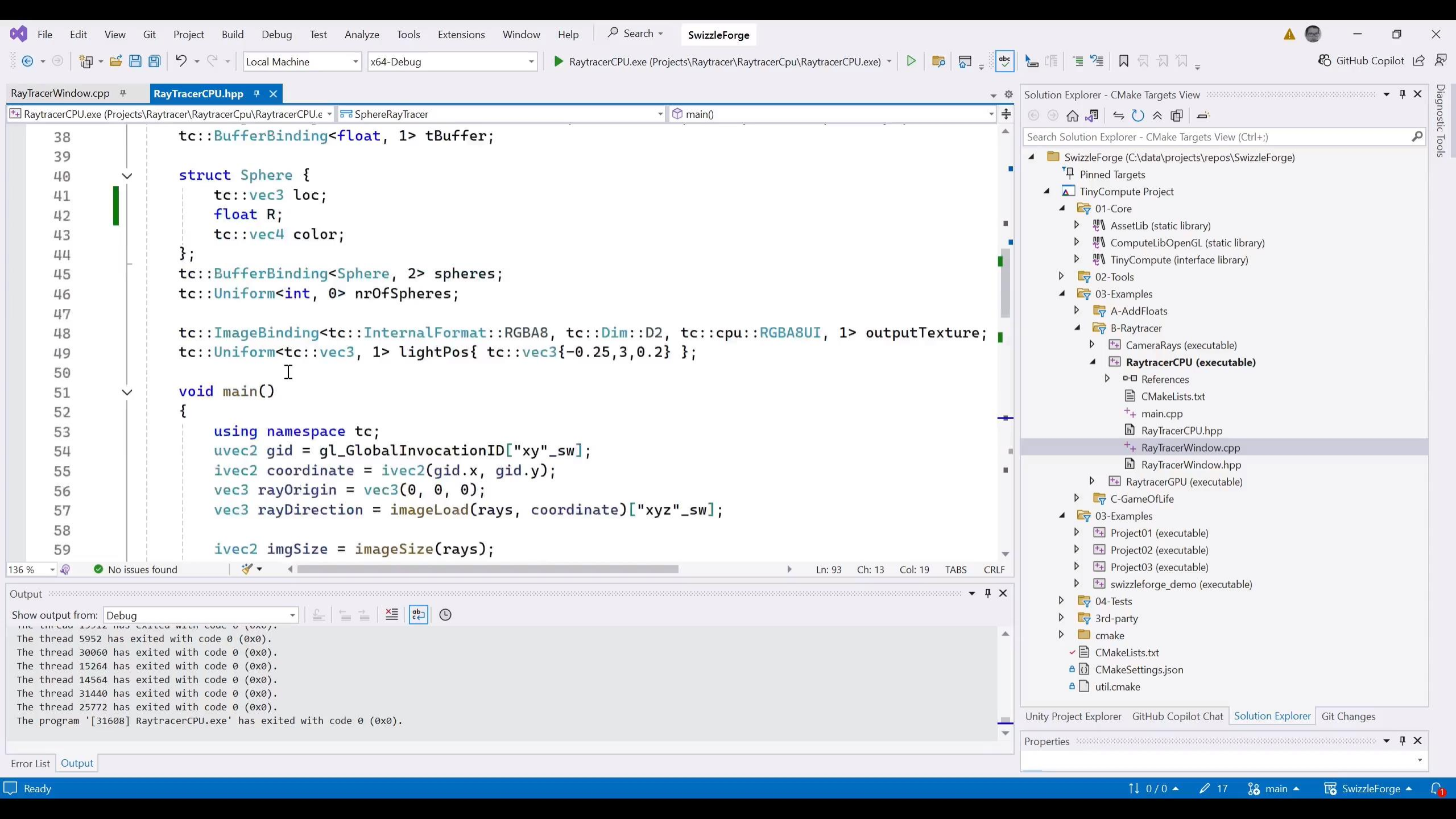
# Raytracer - demo

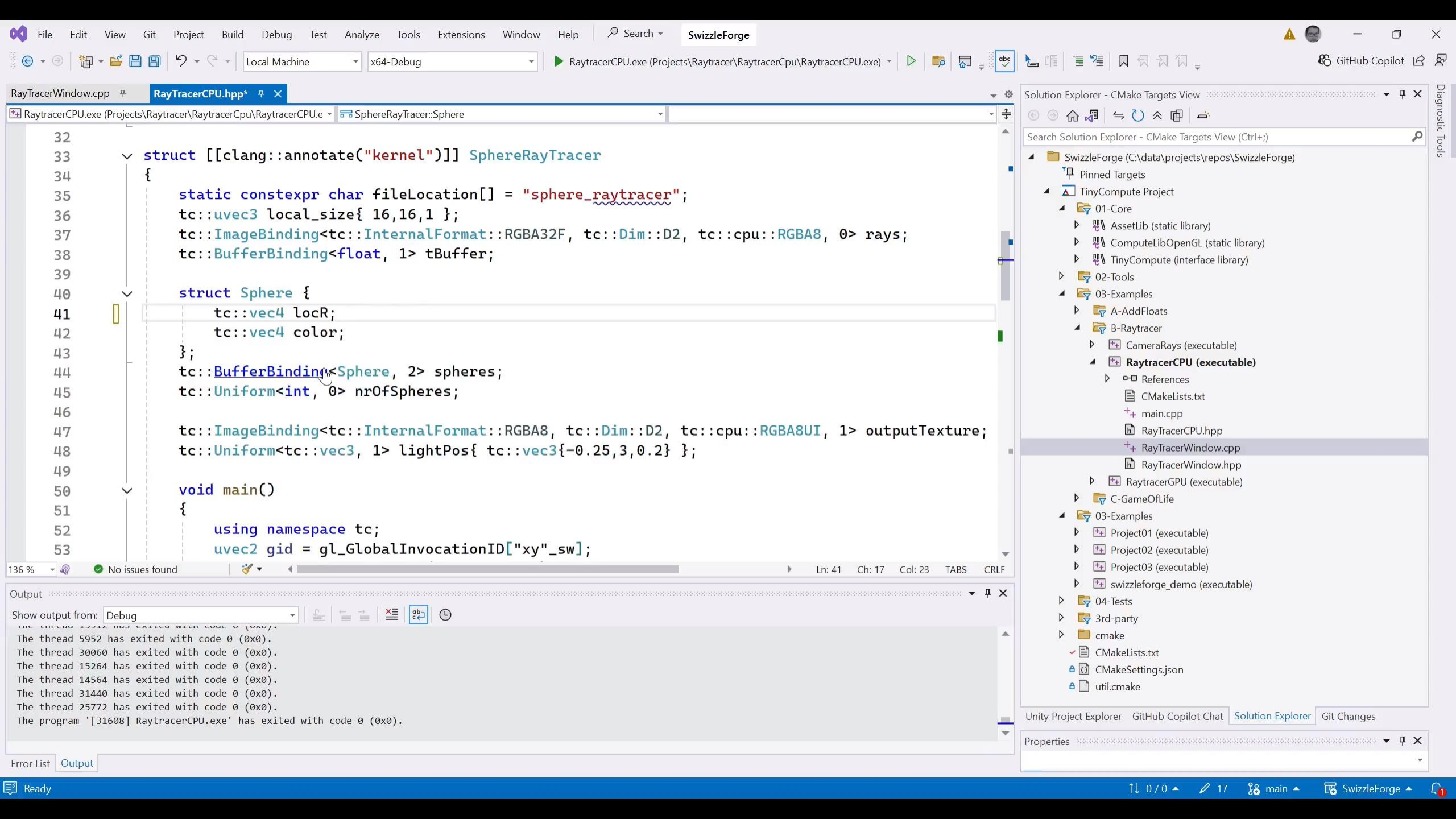


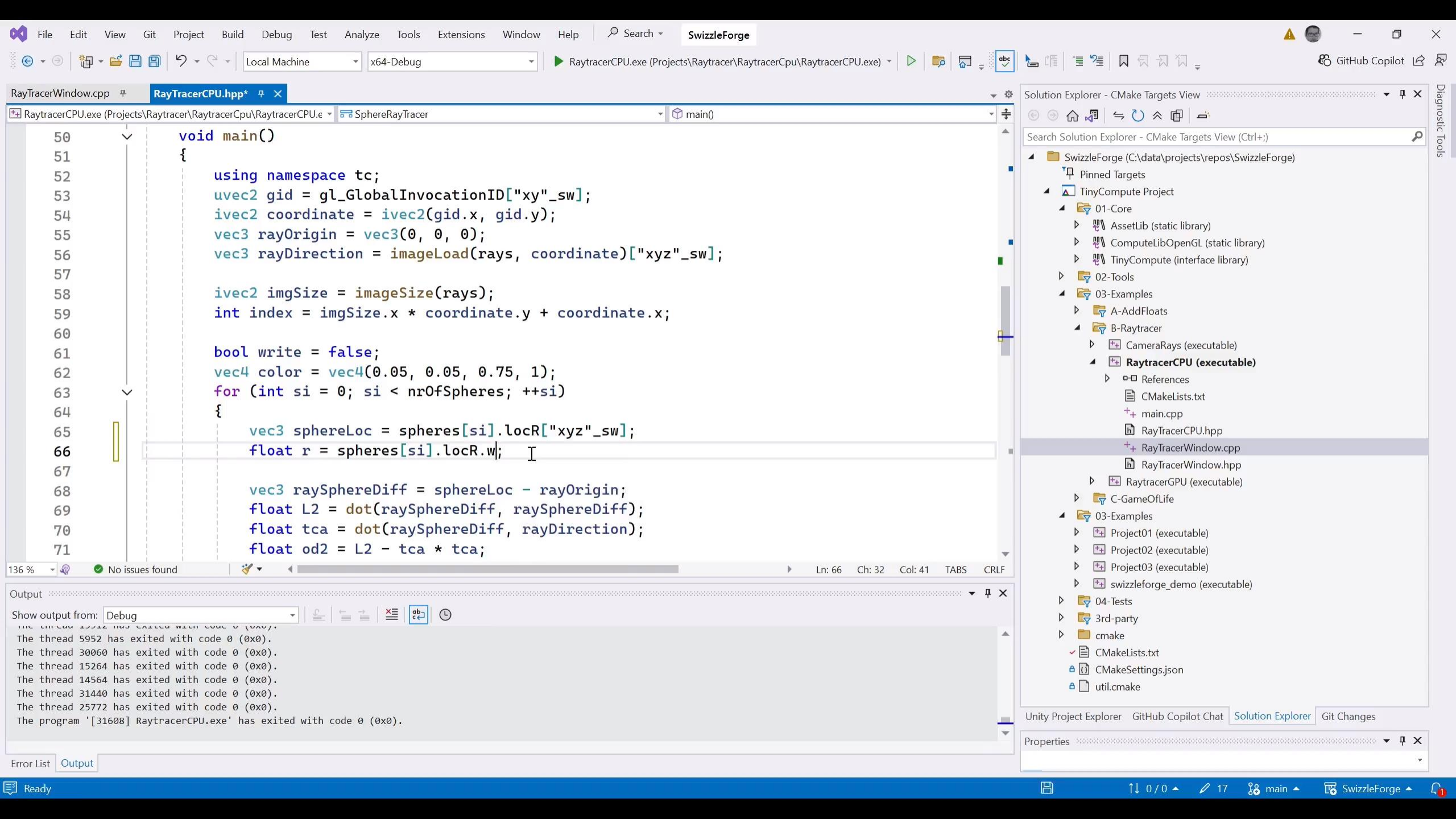

GPU




File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
SwizzleForge
Local Machine
x64-Debug
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.exe)
GitHub Copilot
RayTracerWindow.cpp
RayTracerCPU.hpp
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.e
SphereRayTracer
32
33 struct [[clang::annotate("kernel")]] SphereRayTracer
34 {
35 static constexpr char fileLocation[] = "sphere_raytracer";
36 tc::uvec3 local_size{ 16,16,1 };
37 tc::ImageBinding<tc::InternalFormat::RGBA32F, tc::Dim::D2, tc::cpu::RGBA8, 0> rays;
38 tc::BufferBinding<float, 1> tBuffer;
39
40 struct Sphere {
41 tc::vec3 loc;
42 float R;
43 tc::vec4 color;
44 };
45 tc::BufferBinding<Sphere, 2> spheres;
46 tc::Uniform<int, 0> nrOfSpheres;
47
48 tc::ImageBinding<tc::InternalFormat::RGBA8, tc::Dim::D2, tc::cpu::RGBA8UI, 1> outputTexture;
49 tc::Uniform<tc::vec3, 1> lightPos{ tc::vec3{-0.25,3,0.2} };
50
51 void main()
52 {
53 using namespace tc;
136 %
No issues found
Ln: 45 Ch: 39 Col: 42 TABS CRLF
Output
Show output from: Debug
The thread 31436 has exited with code 0 (0x0).
The thread 19328 has exited with code 0 (0x0).
The thread 31592 has exited with code 0 (0x0).
The thread 29872 has exited with code 0 (0x0).
The thread 30772 has exited with code 0 (0x0).
The thread 32440 has exited with code 0 (0x0).
The program '[31084] RaytracerCPU.exe' has exited with code 0 (0x0).
Error List
Output
Solution Explorer - CMake Targets View
Search Solution Explorer - CMake Targets View (Ctrl+;)
SwizzleForge (C:\data\projects\repos\SwizzleForge)
Pinned Targets
TinyCompute Project
01-Core
AssetLib (static library)
ComputeLibOpenGL (static library)
TinyCompute (interface library)
02-Tools
03-Examples
A-AddFloats
B-Raytracer
CameraRays (executable)
RaytracerCPU (executable)
References
CMakeLists.txt
main.cpp
RayTracerCPU.hpp
RayTracerWindow.cpp
RayTracerWindow.hpp
RaytracerGPU (executable)
C-GameOfLife
03-Examples
Project01 (executable)
Project02 (executable)
Project03 (executable)
swizzleforge_demo (executable)
04-Tests
3rd-party
cmake
CMakeLists.txt
CMakeSettings.json
util.cmake
Unity Project Explorer
GitHub Copilot Chat
Solution Explorer
Git Changes
Properties
Diagnostic Tools
Ready
0 / 0
17
main
SwizzleForge

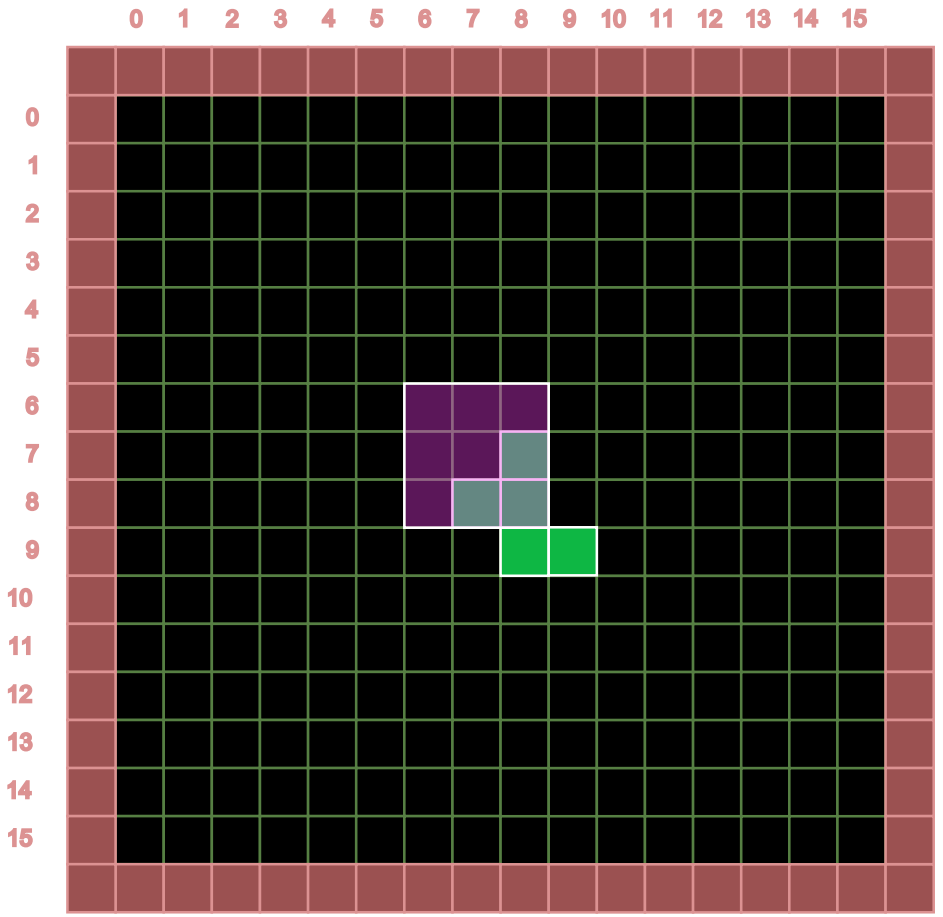

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
SwizzleForge
Local Machine
x64-Debug
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.exe)
GitHub Copilot
RayTracerWindow.cpp
RayTracerCPU.hpp
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.e
SphereRayTracer
main()
38 tc::BufferBinding<float, 1> tBuffer;
39
40 struct Sphere {
41 tc::vec3 loc;
42 float R;
43 tc::vec4 color;
44 };
45 tc::BufferBinding<Sphere, 2> spheres;
46 tc::Uniform<int, 0> nrOfSpheres;
47
48 tc::ImageBinding<tc::InternalFormat::RGBA8, tc::Dim::D2, tc::cpu::RGBA8UI, 1> outputTexture;
49 tc::Uniform<tc::vec3, 1> lightPos{ tc::vec3{-0.25,3,0.2} };
50
51 void main()
52 {
53 using namespace tc;
54 uvec2 gid = gl_GlobalInvocationID["xy"_sw];
55 ivec2 coordinate = ivec2(gid.x, gid.y);
56 vec3 rayOrigin = vec3(0, 0, 0);
57 vec3 rayDirection = imageLoad(rays, coordinate)["xyz"_sw];
58
59 ivec2 imgSize = imageSize(rays);
136 %
No issues found
Ln: 93 Ch: 13 Col: 19 TABS CRLF
Output
Show output from: Debug
The thread 5952 has exited with code 0 (0x0).
The thread 30060 has exited with code 0 (0x0).
The thread 15264 has exited with code 0 (0x0).
The thread 14564 has exited with code 0 (0x0).
The thread 31440 has exited with code 0 (0x0).
The thread 25772 has exited with code 0 (0x0).
The program '[31608] RaytracerCPU.exe' has exited with code 0 (0x0).
Error List
Output
Solution Explorer - CMake Targets View
Search Solution Explorer - CMake Targets View (Ctrl+;)
SwizzleForge (C:\data\projects\repos\SwizzleForge)
Pinned Targets
TinyCompute Project
01-Core
AssetLib (static library)
ComputeLibOpenGL (static library)
TinyCompute (interface library)
02-Tools
03-Examples
A-AddFloats
B-Raytracer
CameraRays (executable)
RaytracerCPU (executable)
References
CMakeLists.txt
main.cpp
RayTracerCPU.hpp
RayTracerWindow.cpp
RayTracerWindow.hpp
RaytracerGPU (executable)
C-GameOfLife
03-Examples
Project01 (executable)
Project02 (executable)
Project03 (executable)
swizzleforge_demo (executable)
04-Tests
3rd-party
cmake
CMakeLists.txt
CMakeSettings.json
util.cmake
Unity Project Explorer
GitHub Copilot Chat
Solution Explorer
Git Changes
Properties
Diagnostic Tools
Ready
0 / 0
17
main
SwizzleForge


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
SwizzleForge
Local Machine
x64-Debug
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.exe)
GitHub Copilot
RayTracerWindow.cpp
RayTracerCPU.hpp*
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.e
SphereRayTracer::Sphere
32
33 struct [[clang::annotate("kernel")]] SphereRayTracer
34 {
35 static constexpr char fileLocation[] = "sphere_raytracer";
36 tc::uvec3 local_size{ 16,16,1 };
37 tc::ImageBinding<tc::InternalFormat::RGBA32F, tc::Dim::D2, tc::cpu::RGBA8, 0> rays;
38 tc::BufferBinding<float, 1> tBuffer;
39
40 struct Sphere {
41 tc::vec4 locR;
42 tc::vec4 color;
43 };
44 tc::BufferBinding<Sphere, 2> spheres;
45 tc::Uniform<int, 0> nrOfSpheres;
46
47 tc::ImageBinding<tc::InternalFormat::RGBA8, tc::Dim::D2, tc::cpu::RGBA8UI, 1> outputTexture;
48 tc::Uniform<tc::vec3, 1> lightPos{ tc::vec3{-0.25,3,0.2} };
49
50 void main()
51 {
52 using namespace tc;
53 uvec2 gid = gl_GlobalInvocationID["xy"_sw];
136 %
No issues found
Ln: 41 Ch: 17 Col: 23 TABS CRLF
Output
Show output from: Debug
The thread 5952 has exited with code 0 (0x0).
The thread 30060 has exited with code 0 (0x0).
The thread 15264 has exited with code 0 (0x0).
The thread 14564 has exited with code 0 (0x0).
The thread 31440 has exited with code 0 (0x0).
The thread 25772 has exited with code 0 (0x0).
The program '[31608] RaytracerCPU.exe' has exited with code 0 (0x0).
Error List
Output
Solution Explorer - CMake Targets View
Search Solution Explorer - CMake Targets View (Ctrl+;)
SwizzleForge (C:\data\projects\repos\SwizzleForge)
Pinned Targets
TinyCompute Project
01-Core
AssetLib (static library)
ComputeLibOpenGL (static library)
TinyCompute (interface library)
02-Tools
03-Examples
A-AddFloats
B-Raytracer
CameraRays (executable)
RaytracerCPU (executable)
References
CMakeLists.txt
main.cpp
RayTracerCPU.hpp
RayTracerWindow.cpp
RayTracerWindow.hpp
RaytracerGPU (executable)
C-GameOfLife
03-Examples
Project01 (executable)
Project02 (executable)
Project03 (executable)
swizzleforge_demo (executable)
04-Tests
3rd-party
cmake
CMakeLists.txt
CMakeSettings.json
util.cmake
Unity Project Explorer
GitHub Copilot Chat
Solution Explorer
Git Changes
Properties
Diagnostic Tools
Ready
0 / 0
17
main
SwizzleForge

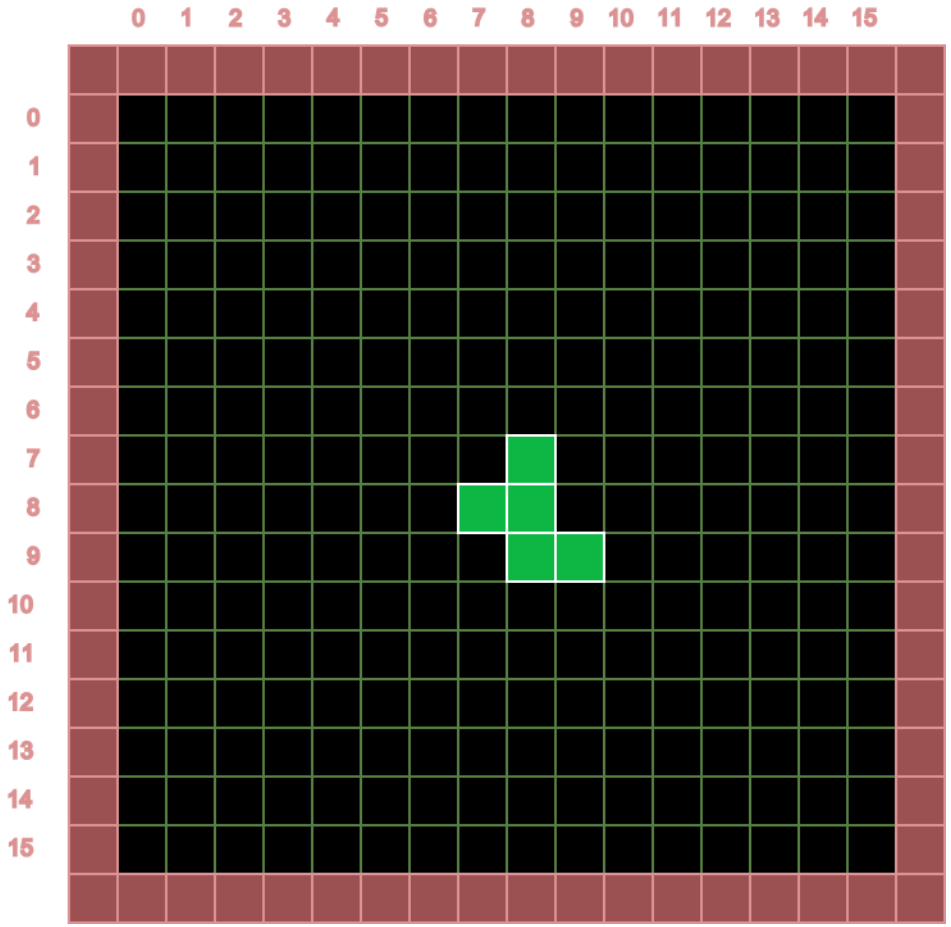

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
SwizzleForge
Local Machine
x64-Debug
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.exe)
GitHub Copilot
RayTracerWindow.cpp
RayTracerCPU.hpp*
RaytracerCPU.exe (Projects\Raytracer\RaytracerCpu\RaytracerCPU.e
SphereRayTracer
main()
50 void main()
51 {
52 using namespace tc;
53 uvec2 gid = gl_GlobalInvocationID["xy"_sw];
54 ivec2 coordinate = ivec2(gid.x, gid.y);
55 vec3 rayOrigin = vec3(0, 0, 0);
56 vec3 rayDirection = imageLoad(rays, coordinate)["xyz"_sw];
57
58 ivec2 imgSize = imageSize(rays);
59 int index = imgSize.x * coordinate.y + coordinate.x;
60
61 bool write = false;
62 vec4 color = vec4(0.05, 0.05, 0.75, 1);
63 for (int si = 0; si < nrOfSpheres; ++si)
64 {
65 vec3 sphereLoc = spheres[si].locR["xyz"_sw];
66 float r = spheres[si].locR.w;
67
68 vec3 raySphereDiff = sphereLoc - rayOrigin;
69 float L2 = dot(raySphereDiff, raySphereDiff);
70 float tca = dot(raySphereDiff, rayDirection);
71 float od2 = L2 - tca * tca;
136 %
No issues found
Ln: 66 Ch: 32 Col: 41 TABS CRLF
Output
Show output from: Debug
The thread 5952 has exited with code 0 (0x0).
The thread 30060 has exited with code 0 (0x0).
The thread 15264 has exited with code 0 (0x0).
The thread 14564 has exited with code 0 (0x0).
The thread 31440 has exited with code 0 (0x0).
The thread 25772 has exited with code 0 (0x0).
The program '[31608] RaytracerCPU.exe' has exited with code 0 (0x0).
Error List
Output
Solution Explorer - CMake Targets View
Search Solution Explorer - CMake Targets View (Ctrl+;)
SwizzleForge (C:\data\projects\repos\SwizzleForge)
Pinned Targets
TinyCompute Project
01-Core
AssetLib (static library)
ComputeLibOpenGL (static library)
TinyCompute (interface library)
02-Tools
03-Examples
A-AddFloats
B-Raytracer
CameraRays (executable)
RaytracerCPU (executable)
References
CMakeLists.txt
main.cpp
RayTracerCPU.hpp
RayTracerWindow.cpp
RayTracerWindow.hpp
RaytracerGPU (executable)
C-GameOfLife
03-Examples
Project01 (executable)
Project02 (executable)
Project03 (executable)
swizzleforge_demo (executable)
04-Tests
3rd-party
cmake
CMakeLists.txt
CMakeSettings.json
util.cmake
Unity Project Explorer
GitHub Copilot Chat
Solution Explorer
Git Changes
Properties
Diagnostic Tools
Ready
0 / 0
17
main
SwizzleForge

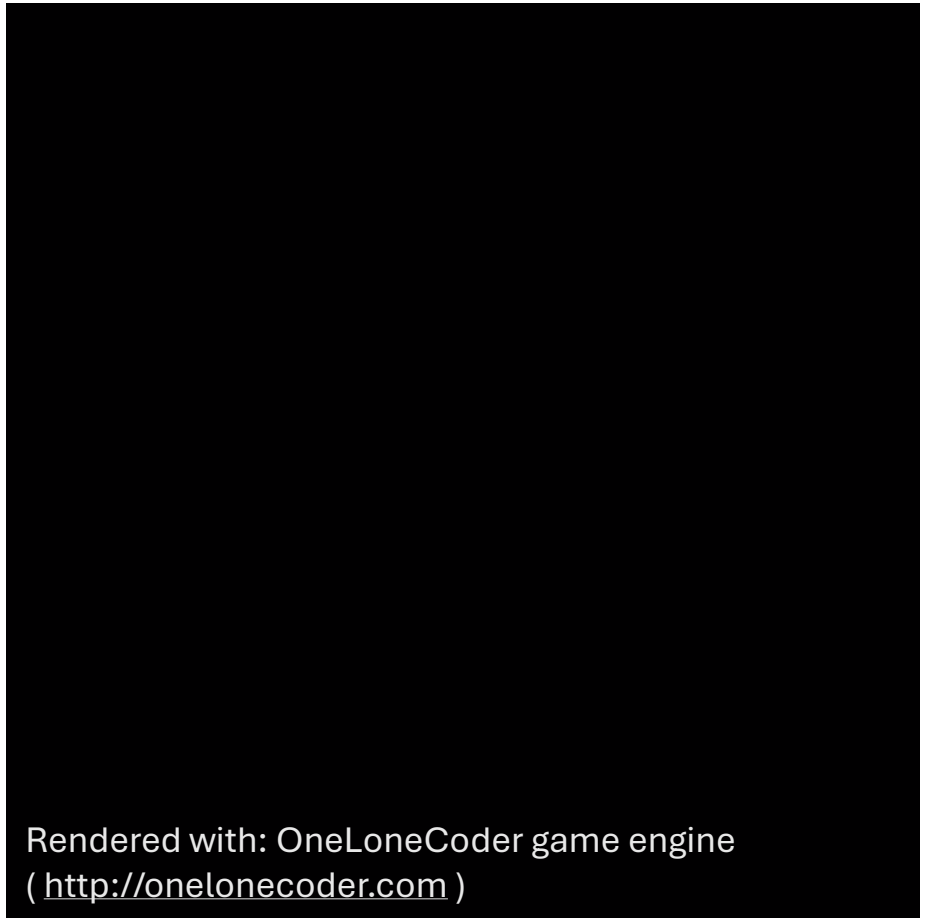
# Case 3 : Game of life

- Image
  - Source Formats
  - External type
  - Destination Formats
  - Internal type
- GLSL Image operations
  - imageLoad
  - ImageStore

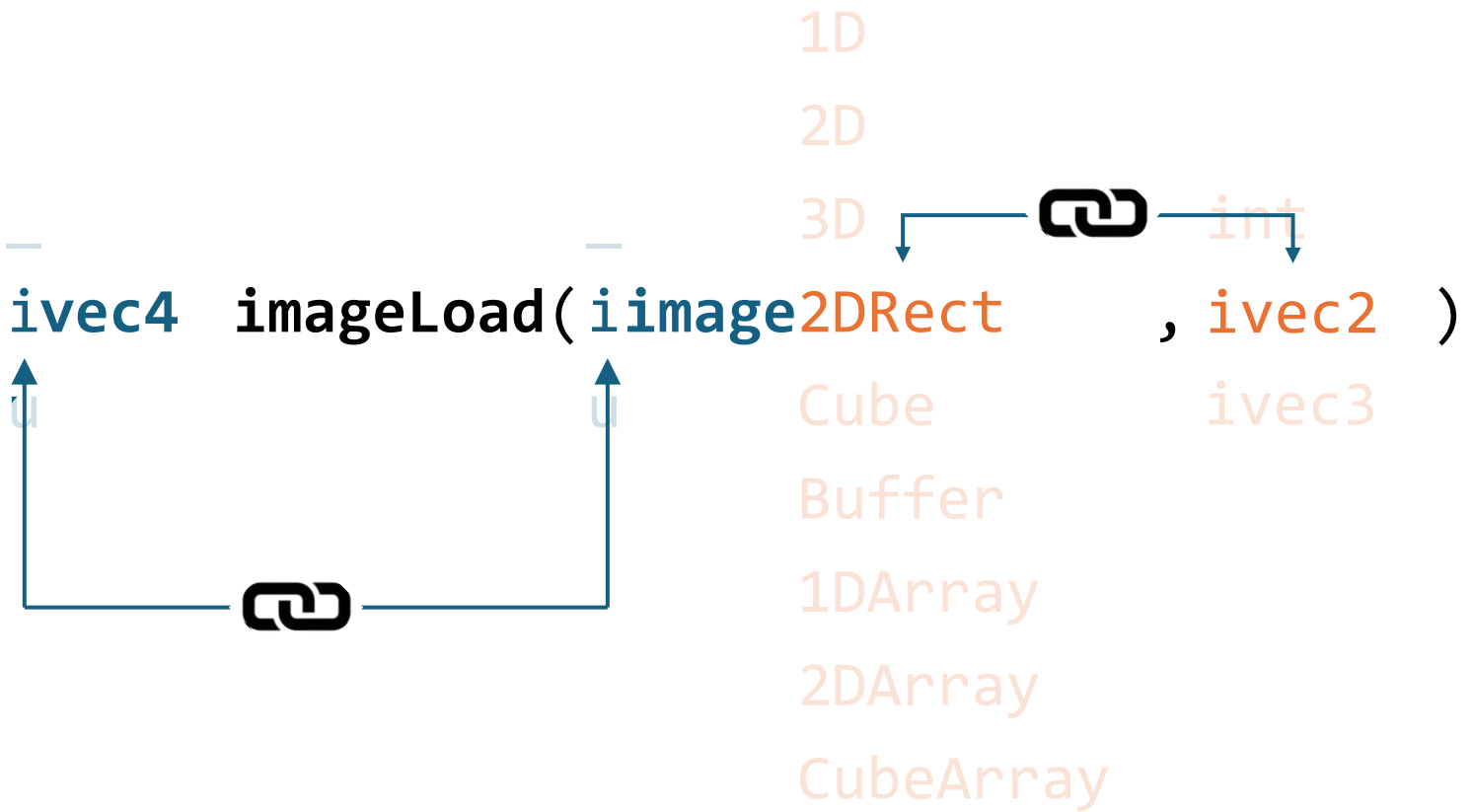


# Case 3: Game of life



# ImageLoad - GLSL

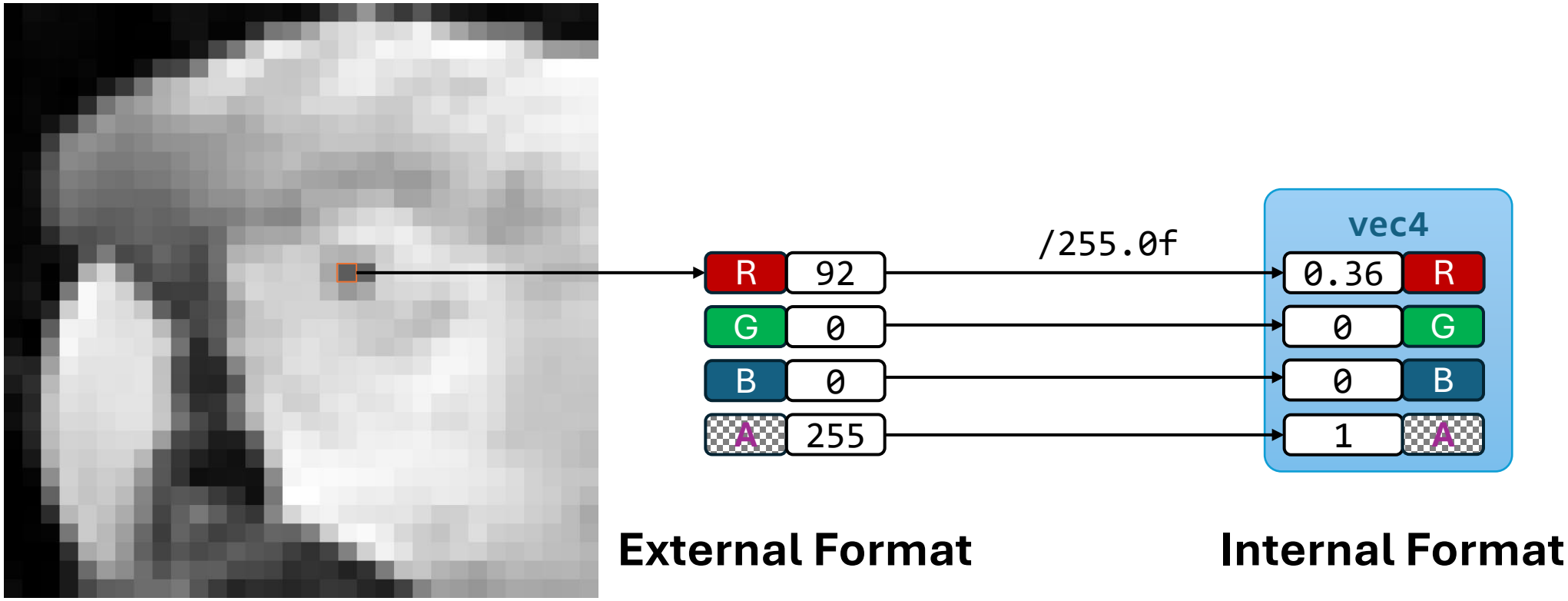



27 combinations!

# But wait! There's more

**Source**: grayscale , uint8_t (R8)

**Destination**: image2D , **RGBA float**




**External Format**

**Internal Format**

# Images in GLSL

```
1   #version 430

2   layout (local_size_x = 16, local_size_y = 16, local_size_z = 1) in;

3   layout(binding=0, r8ui ) uniform uimage2D inData;
4   layout(binding=1, r8ui ) uniform uimage2D outData;

5   void main()
6   {
7           uvec4 pixelIn = imageLoad( inData, coordinate );
8           // convolution
9           imageStore( outData, coordinate, uvec4(1,0,0,0) );
10  }
```

Internal format

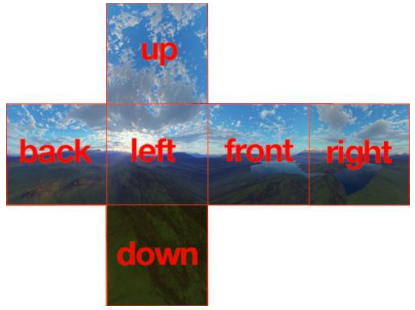
# imageLoad – C++

Dimension

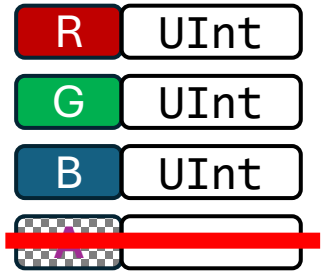
1D

2D

3D

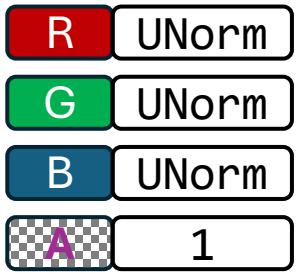


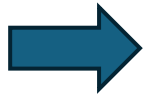

External format
(channels +type)




Convert/Pad

Internal format
(channels +type)



Templates
Traits
`if constexpr`

Can we create **one** version of the **imageLoad** free function?

**Without** runtime branching ?

C++ Design

# imageLoad - BufferResource

```
template<typename T, Dim D = Dim::D1>
BufferResource
    BufferResource(dimType bufferSize)
    :m_Data(/* bufferSize (nD) → size */)
    {
    }

    const T& operator[](dimType index) const
    {
        return m_Data[ /* index (nD) → 1D */  ];
    }
private:
    dimType m_BufferSize;
    std::vector<T> m_Data;  ——→ 1D
```

| D | dimType |
|---|---|
| D1 | int32_t |
| D2 | ivec2 |
| D3 | ivec3 |

***GLSL**: signed int is type used in **imageLoad***

C++ Design

# Dimension - Traits

```
enum class Dim : uint8_t {
        D1 = 1,
        D2 = 2,
        D3 = 3
};
```

```
template<Dim D>
struct DimTraits;
```

```
1  template<>
2  struct DimTraits< Dim::D1 > {
3       using IndexType =  int32_t;
4       static constexpr int32_t flatten( IndexType coord, IndexType /*size*/) {
5            return coord;
6       }
7       static constexpr int32_t product( IndexType d ) {
8            return d;
9       }
10 };
```

C++ Design

# Dimension – 2D

```
enum class Dim : uint8_t {
        D1 = 1,
        D2 = 2,
        D3 = 3
};
```

```
template<Dim D>
struct DimTraits;
```

```
1   template<>
2   struct DimTraits< Dim::D2 > {
3       using IndexType =  ivec2;
4       static constexpr int32_t flatten( IndexType c, IndexType size ) {
5           return c.y * size.x + c.x;
6       }
7       static constexpr int32_t product( IndexType d ) {
8           return d.x * d.y;
9       }
10  };
```

C++ Design

# Dimension – 3D

```
enum class Dim : uint8_t {
        D1 = 1,
        D2 = 2,
        D3 = 3
};
```

```
template<Dim D>
struct DimTraits;
```

```
1   template<>
2   struct DimTraits< Dim::D3 > {
3       using IndexType =  ivec3;
4       static constexpr int32_t flatten( IndexType c, IndexType size ) {
5           return (coord.z * size.y + coord.y) * size.x + coord.x;
6       }
7       static constexpr int32_t product( IndexType d ) {
8           return d.x * d.y * d.z;
9       }
10  };
```

C++ Design

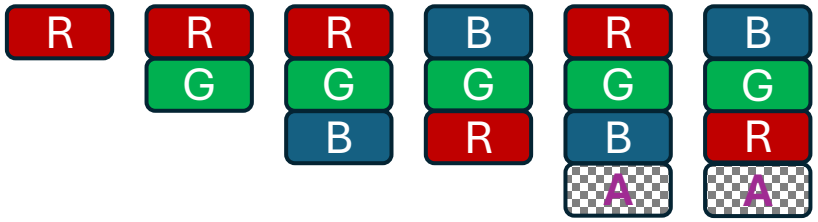
# imageLoad - BufferResource

```
template<typename T, Dim D = Dim::D1>
BufferResource
public:
    using dimType = typename Traits::IndexType;

    BufferResource(dimType bufferSize)
    :m_Data( Traits::product(bufferSize) )
    {
    }

    const T& operator[](dimType index) const
    {
        return m_Data[ Traits::flatten(index) ];
    }

private:
    using Traits = DimTraits<D>;

    dimType m_BufferSize;

    std::vector<T> m_Data;
```

C++ Design

# imageLoad – external format

## Channels




## Semantics

`Literal` — `float or integer`

`Signed normal` — `[-1,1]`

`Unsigned normal` — `[0,1]`

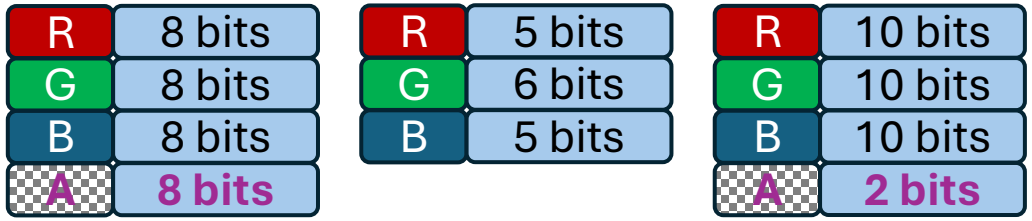
`_INTEGER`

## Representation

`uint8_t` `int8_t` `uint16_t` `int16_t` `float16_t` …

C++23!

## Packing

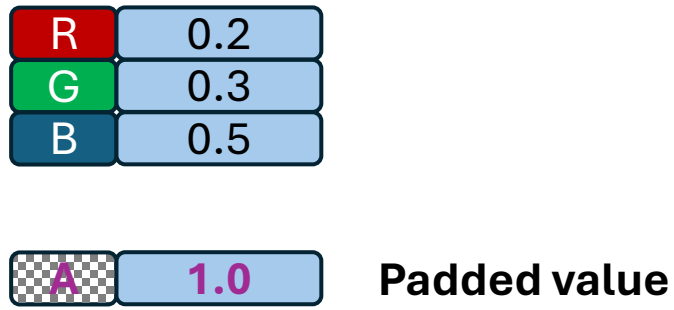



…

C++ Design

# Pixel concept

```
1   using namespace tc;
2   enum class Channel{ R=0, G=1, B=2, A=3 };
3
4   RGB8 px{ 0.2, 0.3, 0.5 };
5   float r = px.get<Channel::R>();
6   float g = px.get<Channel::G>();
7   float b = px.get<Channel::B>();
8   float a = px.get<Channel::A>();
9
10  EXPECT_EQ(r, 0.2);
11  EXPECT_EQ(g, 0.3);
12  EXPECT_EQ(b, 0.5);
13  EXPECT_EQ(a, 1.0);
```



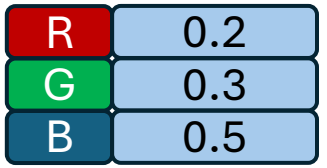

**Padded value**

Simple pixel: **r**, **g** and **b** have the same type → `float`

C++ Design

# Pixel concept

```
using r_t = tc::cpu::RGB8::ChannelType<Channel::R>;

using g_t = tc::cpu::RGB8::ChannelType<Channel::G>;

using b_t = tc::cpu::RGB8::ChannelType<Channel::B>;

using a_t = tc::cpu::RGB8::ChannelType<Channel::A>;
```




It is possible to have a different type for each channel.

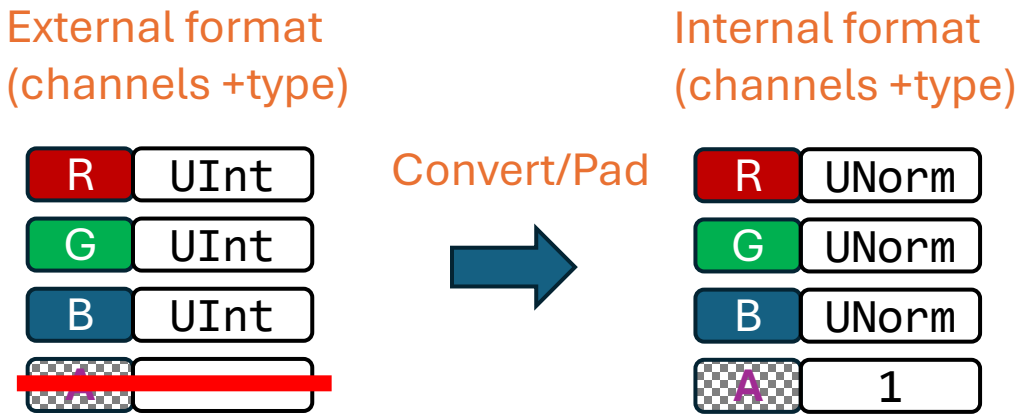
C++ Design

# Pixel Concept

```
template<class P, Channel C>
concept ChannelConcept = requires(P p) {

      typename P::template ChannelType<C>;

      { p.template get<C>() }
               -> std::convertible_to<typename P::template ChannelType<C>>;

      { p.template set<C>( p.template get<C>() ) };

};

template<class P>
concept PixelConcept =
         ChannelConcept<P, Channel::R> &&
         ChannelConcept<P, Channel::G> &&
         ChannelConcept<P, Channel::B> &&
         ChannelConcept<P, Channel::A>;
```

C++ Design

# Conversions




```
template<class Src, class Dst>
struct ChannelConverter;

template<class T>
struct ChannelConverter<T, T> {
        static constexpr T apply(T v) noexcept { return v; }
};

template<>
struct ChannelConverter<std::uint8_t, float> {
        static constexpr float apply(std::uint8_t v) noexcept {
                return float(v) * (1.0f / 255.0f);
        }
};
```

C++ Design

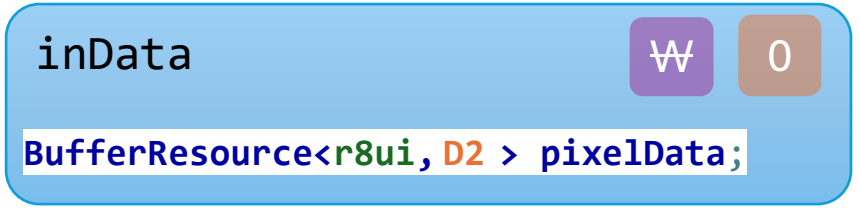
# ImageBinding

```
1   #version 430

2   layout (local_size_x = 16, local_size_y = 16, local_size_z = 1) in;

3   layout(binding=0, r8ui ) uniform uimage2D inData;
4   layout(binding=1, r8ui ) uniform uimage2D outData;

5   void main()
6   {
7           uvec4 pixelIn = imageLoad( inData, coordinate );
8           // convolution
9           imageStore( outData, uvec4(1,0,0,0) );
10  }
```

Reminder:


```
BufferBinding<float,2>;
```

```
ImageBinding< r8ui , D2 , r8ui ,  0 >;
```




C++ Design

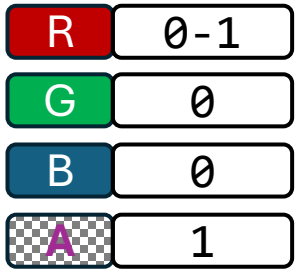
# Internal Format

External format
(channels +type)



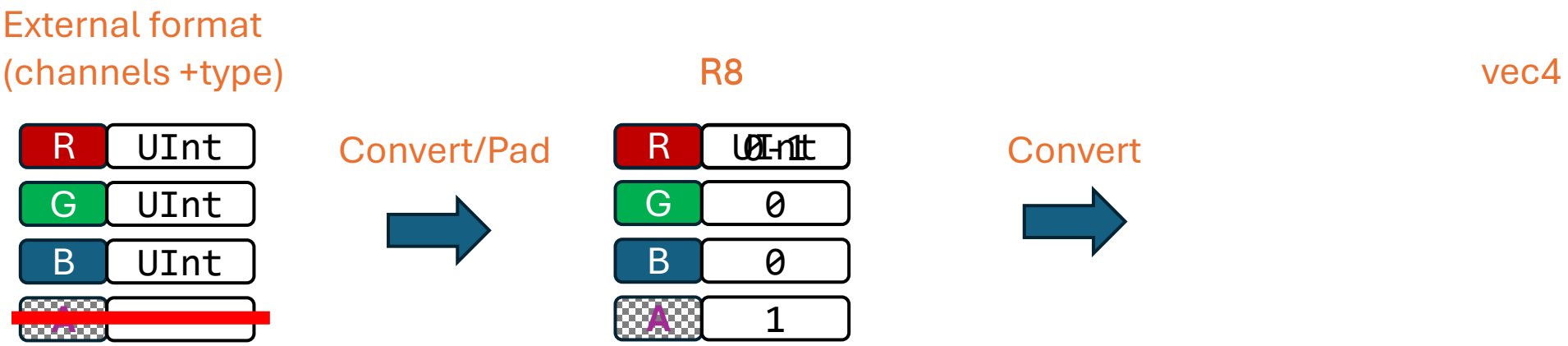

Convert/Pad

R8




Pad with internal format **defaults**

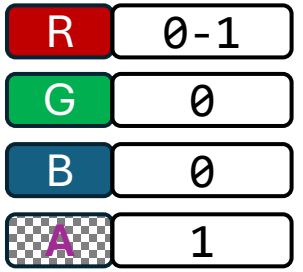
# Internal Format



# Internal Format

```
template<>
struct GPUFormatTraits<tc::gpu::R32F> {
        using ChannelType = float;
        using VectorType = tc::vec4;
        static inline constexpr std::array<tc::Channel, 4> channels{
              Channel::R,
              Channel::Min,
              Channel::Min,
              Channel::Max
        };
};
```

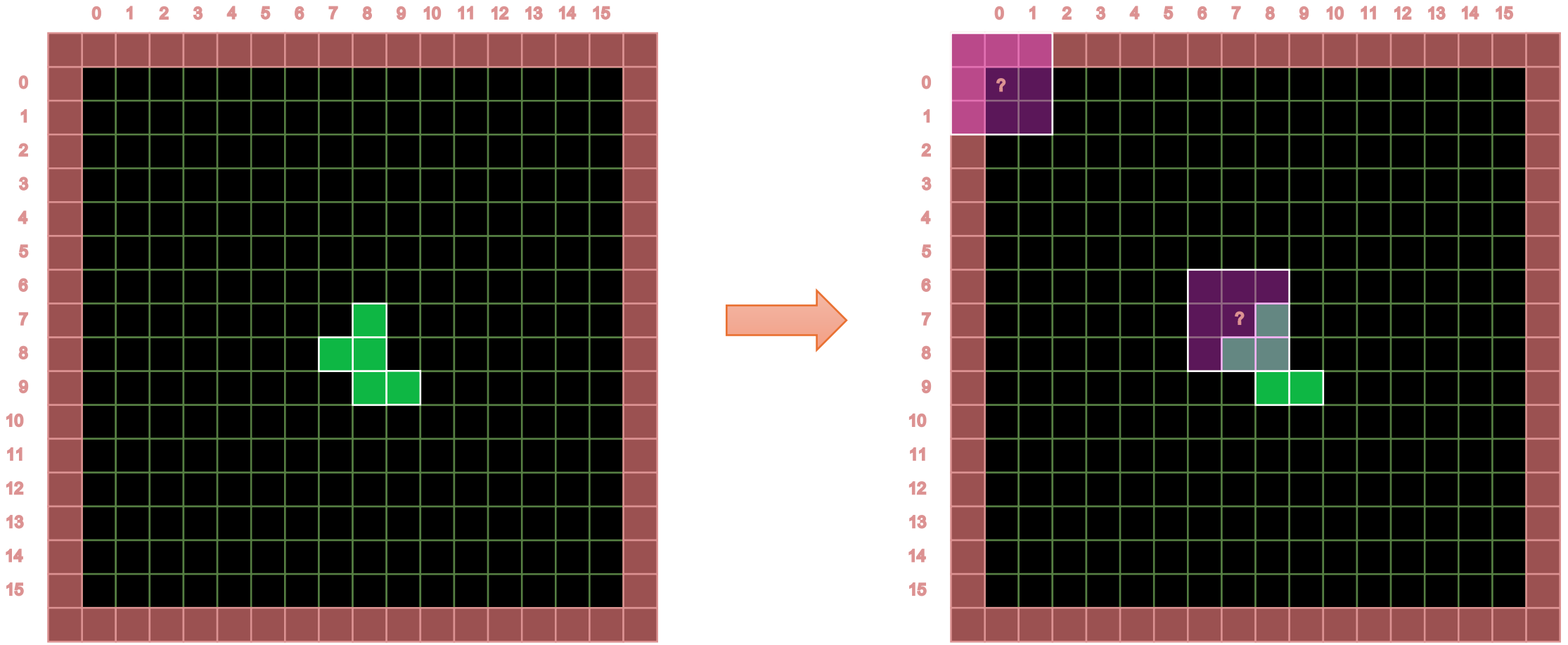
R8



# ImageLoad

```
template<tc::InternalFormat G, tc::Dim D, tc::cpu::PixelConcept P>
auto imageLoad(const ImageBinding<G, D, P>& image, tcVec<D> texCoord)
{
        const auto* buf = image.getBufferData();
        const P& px = (*buf)[texCoord];
        using gpuTraits = GPUFormatTraits<G>;
        using dst_t = typename gpuTraits::ChannelType;
        using vec_t = typename gpuTraits::VectorType;

        dst_t r = loadChannel<gpuTraits::channels[0], dst_t, P>(px);
        dst_t g = loadChannel<gpuTraits::channels[1], dst_t, P>(px);
        dst_t b = loadChannel<gpuTraits::channels[2], dst_t, P>(px);
        dst_t a = loadChannel<gpuTraits::channels[3], dst_t, P>(px);
        return vec_t{ r,g,b,a };
 }

```

# Conway’s game of life



# Compute Shader – C++

```
struct [[clang::annotate("kernel")]] GameOfLifeKernel
{
      ImageBinding<gpu::R8UI, tc::Dim::D2, cpu::R8UI, 0> inData;
      ImageBinding<gpu::R8UI, tc::Dim::D2, cpu::R8UI, 1> outData;
      Uniform<int32_t, 1> pad{ 1 };
      std::array<ivec2, 8> kIndices {
          ivec2{-1,-1}, ivec2{0,-1}, ivec2{1,-1},
          ivec2{-1, 0},              ivec2{1, 0},
          ivec2{-1, 1}, ivec2{0, 1}, ivec2{1, 1}
      };
```

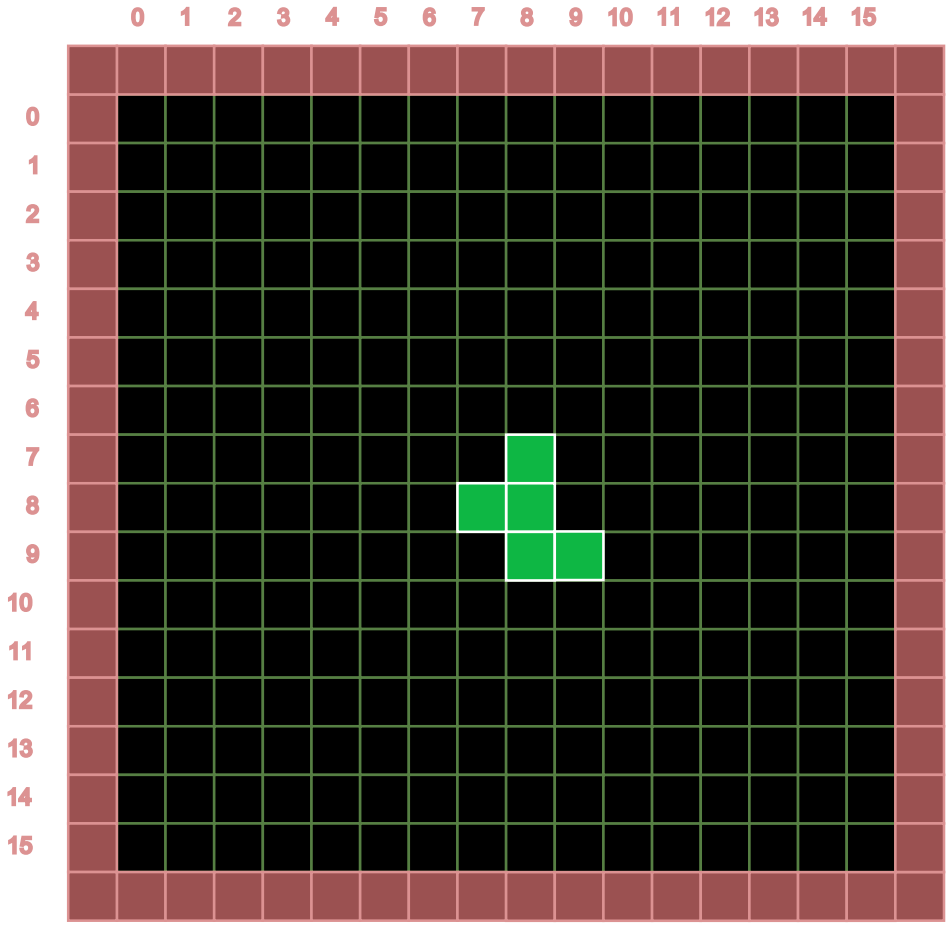
```
void main() {
      uvec2 gId = tc::gl_GlobalInvocationID["xy"_sw];
      ivec2 coordinate = ivec2(gId.x+pad, gId.y+pad);
      uint n = 0;
      bool alive = imageLoad(inData, coordinate).x;
      for (int ki = 0; ki < kernelIndices.size(); ++ki) {
          n += imageLoad(inData, coordinate + kIndices[ki]).x;
      }
      bool newState = (n == 3) || (alive && n == 2);
      imageStore(outData, coordinate, uvec4(newState));
}


};
```

# Compute Shader – GLSL

```
struct [[clang::annotate("kernel")]] GameOfLifeKernel
{
     layout(binding=0,r8ui) uniform uimage2D inData; 0> inData;
     layout(binding=1,r8ui) uniform uimage2D outData;1> outData;
     layout(location=1) uniform int pad;
     const ivec2 kIndices[8] = ivec2[8] (
         ivec2(-1,-1), ivec2(0,-1),ivec2(1,-1),,
         ivec2(-1, 0),             ivec2(1, 0),,
         ivec2(-1, 1), ivec2(0, 1),ivec2(1, 1)}
     );
```

```
void main() {
     uvec2 gId = gl_GlobalInvocationID.xy;["xy"_sw];
     ivec2 coordinate = ivec2(gId.x+pad, gId.y+pad);
     uint n = 0;
     bool alive = bool( imageLoad(inData, coordinate).x );
     for (int ki = 0; ki < kIndices.size(); ++ki) {
         n += imageLoad(inData, coordinate + kIndices[ki]).x;
     }
     bool newState = (n == 3) || (alive && n == 2);
     imageStore(outData, coordinate, uvec4(newState));
}


};
```

# Game of life GPU




# Conclusions

- 3 vertical slices → now go horizontal
- Generic programming reduces code:
  - Traits
  - Constexpr
  - Concepts
  - …
- Compile time programming → no performance penalty for very flexible code
- I need to upgrade to C++26
  - float16_t
  - Dimensional spans
  - Reflection!