Clover Status Update

Tom Stellard

Advanced Micro Devices, Inc.

September 24, 2013

Agenda

- Introduction to Clover
- OpenCL™ mini-tutorial
- What Clover can do
- Plans for the future
- Related projects

What is Clover?

- CLover: Computing Language over Gallium
- History
  - Dec 2008 - Initial work by Zach Rusin at Tungsten Graphics
  - August 2011 - GSoC Project by Denis Steckelmacher
  - November 2011 - EVoC Project by Francisco Jerez
  - May 2012 - Clover merged into Mesa
- What is OpenCL™ ?
  - API enabling general purpose computing on GPUs (GPGPU) and other devices
  - Well suited for certain kinds of parallel computations
    - Hash Cracking (e.g SHA, MD5, etc.)
    - Image processing
    - Simulations

More about OpenCL™

- Key Terms
  - Device - GPU, CPU, FPGA, etc.
  - Work Item - Thread
  - Work Group - Group of Work Items
  - Memory Spaces
    - Private - Work item memory
    - Local - Memory shared by work items in a work group
    - Global - Memory shared by all work items
    - Constant - Read-only global memory
- OpenCL™ Runtime
  - Device creation
  - Buffer management
  - Kernel dispatch
  - etc.
- OpenCL™ C
  - C99 Based
  - Vector Types
  - Builtin Library

Clover Dependencies

- Clang
  - Provides OpenCL™ C compiler frontend
  - Generates LLVM IR
  - Clover uses libclang and not the standalone compiler
- LLVM
  - Modular compiler library
  - LLVM IR optimization passes
  - Code generation
- libclc
  - Implementation of the OpenCL™ C standard library
  - LLVM bytecode library
  - Linked at runtime

Hello World

```
int main(int argc, char **argv)
{
    clGetPlatformIDs([...]);

    clGetDeviceIDs([...]);

    clCreateContext([...]);

    clCreateCommandQueue([...]);

    clCreateProgramWithSource([...]);

    clBuildProgram([...]);

    clCreateKernel([...]);

    clCreateBuffer([...]);

    clSetKernelArg([...]);

    clEnqueueNDRangeKernel([...]);

    clFinish([...]);

    clEnqueueReadBuffer([...]);
}
```

Hello World

```
clGetPlatformIDs(1, &platform_id, &total_platforms);
```

- Query system for avaiable platforms
- Multiple platforms can be used with the ICD extension

```
clGetDeviceIDs(platform_id, CL_DEVICE_TYPE_GPU, 1,
               &device_id, &total_gpu_devices);
```

- Queries the system for available devices
- Uses gallium pipe-loader to discover devices
- Creates a pipe_screen object for each device

Hello World

```
context = clCreateContext(
    NULL,           /* Properties */
    1,              /* Number of devices */
    &device_id,     /* Device pointer */
    NULL,           /* Callback for reporting errors */
    NULL,           /* User data to pass to error callback */
    &error);        /* Error code */
```

- Creates a new context with pipe_screen::context_create()

```
command_queue = clCreateCommandQueue(
    context,
    device_id,
    0,              /* Command queue properties */
    &error);        /* Error code */
```

- Setup a command queue to manage events

Hello World

```
const char * program_src =
"__kernel\n"
"void pi(__global float * out) \n"
"{\n"
"   out[0] = 3.14159f;\n"
"}\n";

program = clCreateProgramWithSource(
    context,
    1,              /* Number of strings */
    &program_src,
    NULL,           /* String lengths: NULL means all the
                     * strings are NULL terminated. */
    &error);
```

- Program is a group of kernels and other functions

Hello World

```
clBuildProgram(program,
    1,      /* Number of devices */
    &device_id,
    NULL,   /* options */
    NULL,   /* callback function when compile is complete */
    NULL); /* user data for callback */
```

- OpenCL™ C compiled to LLVM IR
- Linked with libclc
- Kernel enumeration

```
kernel = clCreateKernel(program, "pi", &error);
```

- Create a kernel object

Hello World

```
out_buffer = clCreateBuffer(context,
    CL_MEM_WRITE_ONLY, /* Flags */
    sizeof(float),     /* Size of buffer */
    NULL,              /* Pointer to the data */
    &error);           /* error code */
```

- pipe_screen::resource_create()

```
clSetKernelArg(kernel,
    0, /* Arg index */
    sizeof(cl_mem),
    &out_buffer);
```

Hello World

```
clEnqueueNDRangeKernel(command_queue,
    kernel,
    1,      /* Number of dimensions */
    NULL,   /* Global work offset */
    &global_work_size,
    &local_work_size,
    0,      /* Events in wait list */
    NULL,   /* Wait list */
    NULL); /* Event object for this event */
```

- pipe_context::create_compute_state()
- pipe_context::bind_compute_state()
- pipe_context::set_compute_sampler_states()
- pipe_context::set_compute_sampler_views()
- pipe_context::set_compute_resources()
- pipe_context::set_global_binding()
- pipe_context::launch_grid()

Hello World

```
clFinish(command_queue);
```

- pipe_screen::fence_signalled()
- pipe_context::flush()
- pipe_screen::fence_reference()
- pipe_screen::fence_finish()

```
clEnqueueReadBuffer(command_queue,
    out_buffer,
    CL_TRUE,          /* TRUE means it is a blocking read. */
    0,                /* Buffer offset to read from. */
    sizeof(float),    /* Bytes to read */
    &out_value,       /* Pointer to store the data */
    0,                /* Events in wait list */
    NULL,             /* Wait list */
    NULL);            /* Event object */
```

- pipe_screen::transfer_map()
- pipe_screen::transfer_unmap()

What can Clover do?

- Supported Hardware
  - AMD Evergreen (HD5000) through Southern Islands (HD7000)
- Current Features (AMD Drivers)
  - Most runtime API features
  - 32-bit data types
  - Constant/Global/Local memory spaces well supported
- Supported Applications (AMD Drivers)
  - Bitcoin Mining
  - Piglit
  - OpenCV - 50% pass rate of testsuite
  - GEGL/GIMP - Many filters work
  - Possibly Others???

Testing

- Piglit
  - 1327 tests
    - AMD Evergreen/NI GPU passes 1241
  - 3 Types of tests:
    - cl-program-tester
    - Program tests
    - Custom tests
- Challenges:
  - Lack of test applications
  - Applications often require domain specific knowledge
  - Low margin for error

cl-program-tester

```
/*!
[config]
name: Add and subtract          # Name of the test
clc_version_min: 10             # Minimum requirec OpenCL C version
clc_version_max: 12             # Maximum required OpenCL C version
build_options: -D DEF           # Build options for the program
kernel_name: add                # Default kernel to run
dimensions: 1                   # Number of dimensions for ND kernel
global_size: 1 1 1            # Global work size for ND kernel (defaul
local_size:  1 1 1            # Local work size for ND kernel (default


# Execution tests #

[test]
arg_out: 0 buffer float[1]  3.0 tolerance 0.1
arg_in: 1 float 1.0
arg_in: 2 float 2.0

kernel void sub(global float* out, global float x, float y) {
        out[0] = x + y;
}
```

Future Work

- OpenCV
  - Current focus for AMD drivers
- OpenCL™ ICD
  - Targeting Mesa 9.3
- Render Nodes
  - New in 3.12 kernel
  - Lets us avoid DRM authentication issues with clover
- Image Support
  - Prototype for r600g
- Support more hardware
  - AMD Sea Islands
  - nouveau
  - CPUs via llvmpipe

Future Work (cont.)

- LLVM to TGSI
  - TGSI backend for LLVM
  - Alternative: simple LLVM IR lowering pass
- Add PIPE_CONTEXT_USAGE_COMPUTE flag to gallium
  - Enable drivers to create a lightweight compute only context.
- Piglit
  - More tests!
  - Improving the framework

Piglit Builtin Tests

```
/*!
[config]
dimensions: 1
global_size: 1 0 0

[test]
name: char
arg_out: 0 buffer char[1] 2
arg_in:  1 char 1
arg_in:  2 char 2
kernel_name: test_char

[test]
name: uchar
arg_out 0 buffer uchar[1] 2
arg_in: 1 char 1
arg_in: 2 char 2
kernel_name: test_uchar

[...]

!*/

kernel void test(global char *out, char in, char in2) {
        out[0] = max(a, b);
}

kernel void test(global uchar *out, uchar in, uchar in2) {
        out[0] = max(a, b);
}

[...]
```

Piglit Builtin Tests

```
/*!
[config]
dimensions: 1
global_size: 1 0 0
kernel_name: test
gentype: char uchar short ushort int uint float 1 2 3 4 8 16

[test]
name: A
arg_out: 0 buffer gentype[2] 1 2
arg_in:  1 gentype 1
arg_in:  2 gentype 2

!*/

kernel void test(global _PIGLIT_GENTYPE *out, _PIGLIT_GENTYPE a, _PIGLIT_GENTYPE b) {
        out[0] = max(a, b);
}
```

Related Projects

- POCL
  - Currently targets only CPUs: PPC32, PPC64, X86_64, ARMv7
  - libcuda backend may be merged soon
  - Proof of concept Gallium backend
  - ICD Support
- Beignet
  - Targets Intel GPUs
- Opportunities for collaboration
  - Piglit
  - OpenCL™ C standard library
  - OpenCL™ Runtime