Report

Student Name | Student Number

Task:

Create a simple hello world program using OpenCL.

Description:

I have followed the instructions in section 3.3 from [1] and created a simple hello world program. I have installed latest Nvidia driver on Windows 10 platform in a x64 bit machine. I used a visual studio 2019 platform to build and run the simple hello world program.

Code Snippet:

hello.c:

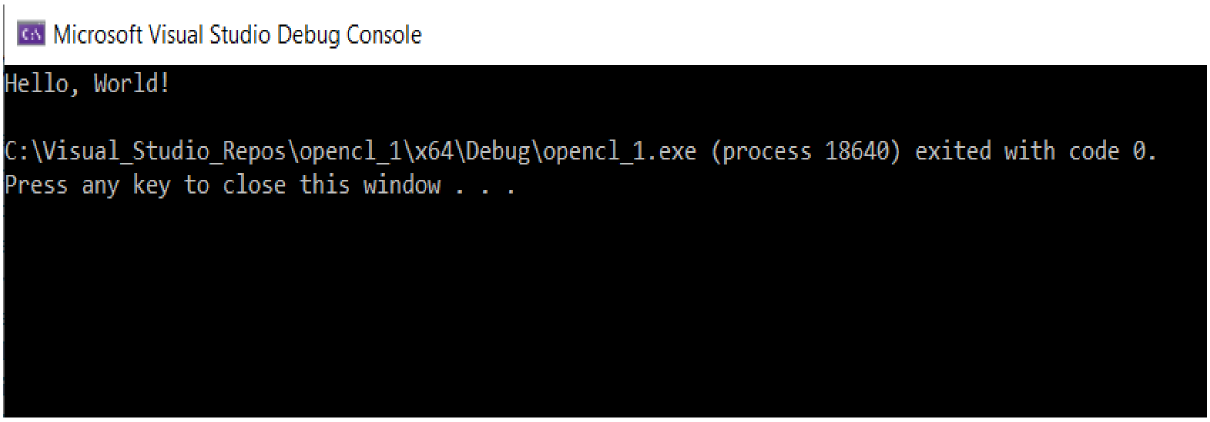
```
#include <stdio.h>
#include <stdlib.h>

#ifdef __APPLE__
#include <OpenCL/opencl.h>
#else
#include <CL/cl.h>
#endif

#define MEM_SIZE (128)
#define MAX_SOURCE_SIZE (0x001000)

int main()
{
    cl_device_id device_id = NULL;
    cl_context context = NULL;
    cl_command_queue command_queue = NULL;
    cl_mem memobj = NULL;
    cl_program program = NULL;
    cl_kernel kernel = NULL;
    cl_platform_id platform_id = NULL;
    cl_uint ret_num_devices;
    cl_uint ret_num_platforms;
    cl_int ret;

    char string[MEM_SIZE];

    FILE* fp;
    char fileName[] = "hello.cl";
    char* source_str;
    size_t source_size;

    /* Load the source code containing the kernel*/
    fopen_s(&fp, fileName, "r");
    if (!fp) {
        fprintf(stderr, "Failed to load kernel.\n");
        exit(1);
    }
    source_str = (char*)calloc(MAX_SOURCE_SIZE, 1);
    source_size = fread(source_str, 1, MAX_SOURCE_SIZE, fp);


    fclose(fp);

    /* Get Platform and Device Info */
    ret = clGetPlatformIDs(1, &platform_id, &ret_num_platforms);
    //printf("platform ret: %d\n", ret);
    ret = clGetDeviceIDs(platform_id, CL_DEVICE_TYPE_GPU, 1, &device_id,
&ret_num_devices);
    //printf("platform id: %d, device id: %d\n", platform_id, device_id);

    /* Create OpenCL context */
    context = clCreateContext(NULL, 1, &device_id, NULL, NULL, &ret);

    /* Create Command Queue */
    command_queue = clCreateCommandQueue(context, device_id, 0, &ret);

    /* Create Memory Buffer */
    memobj = clCreateBuffer(context, CL_MEM_READ_WRITE, MEM_SIZE * sizeof(char), NULL,
&ret);
    //printf("source str: %s\n", source_str);
    /* Create Kernel Program from the source */
    program = clCreateProgramWithSource(context, 1, (const char**)&source_str,
        (const size_t*)&source_size, &ret);

    /* Build Kernel Program */
    ret = clBuildProgram(program, 1, &device_id, NULL, NULL, NULL);
    //printf("%d\n", ret);
    if (ret == CL_BUILD_PROGRAM_FAILURE) {
        // Determine the size of the log
        size_t log_size;
        clGetProgramBuildInfo(program, device_id, CL_PROGRAM_BUILD_LOG, 0, NULL,
&log_size);

        // Allocate memory for the log
        char* log = (char*)malloc(log_size);

        // Get the log
        clGetProgramBuildInfo(program, device_id, CL_PROGRAM_BUILD_LOG, log_size, log,
NULL);

        // Print the log
        //printf("%s\n", log);
    }
    /* Create OpenCL Kernel */
    kernel = clCreateKernel(program, "hello", &ret);

    /* Set OpenCL Kernel Parameters */
    ret = clSetKernelArg(kernel, 0, sizeof(cl_mem), (void*)&memobj);

    /* Execute OpenCL Kernel */
    ret = clEnqueueTask(command_queue, kernel, 0, NULL, NULL);

    /* Copy results from the memory buffer */
    ret = clEnqueueReadBuffer(command_queue, memobj, CL_TRUE, 0,
        MEM_SIZE * sizeof(char), string, 0, NULL, NULL);

    /* Display Result */
    puts(string);
    //printf("%s\n", string);
    /* Finalization */
    ret = clFlush(command_queue);
    ret = clFinish(command_queue);
    ret = clReleaseKernel(kernel);
    ret = clReleaseProgram(program);
    ret = clReleaseMemObject(memobj);
    ret = clReleaseCommandQueue(command_queue);
    ret = clReleaseContext(context);

    free(source_str);

    return 0;
}
```

Hello.cl:

```
__kernel void hello(__global char* string)
{
       string[0] = 'H';
       string[1] = 'e';
       string[2] = 'l';
       string[3] = 'l';
       string[4] = 'o';
       string[5] = ',';
       string[6] = ' ';
       string[7] = 'W';
       string[8] = 'o';
       string[9] = 'r';
       string[10] = 'l';
       string[11] = 'd';
       string[12] = '!';
       string[13] = '\0';
}
```

Result:

The code snippet was build and run successfully on visual studio platform. It resulted in the following output.



```
Hello, World!

C:\Visual_Studio_Repos\opencl_1\x64\Debug\opencl_1.exe (process 18640) exited with code 0.
Press any key to close this window . . .
```