

Лекция 5

- CUDA Driver API.

```
/Ла69> ls -l
```

```
total 16
```

```
-rw-r--r-- 1 malkov users 2111 авг 23 2024 cda.cpp  
-rw-r--r-- 1 malkov users 122 окт 26 13:48 cda.cu  
-rw-r--r-- 1 malkov users 1273 окт 28 12:57 cda.py  
-rw-r--r-- 1 malkov users 1775 авг 7 2024 cuda_driver.py
```

```
extern "C"
```

```
__global__ void gStub(float* a, float* b){  
    int tid=threadIdx.x+blockIdx.x*blockDim.x;  
    a[tid]=a[tid]+b[tid];  
}
```

cda.cu

```
/Лаб9> nvcc -ptx cda.cu
```

```
/Лаб9> ls -ltr
```

```
total 20
```

```
-rw-r--r-- 1 malkov users 1775 авг 7 2024 cuda_driver.py
```

```
-rw-r--r-- 1 malkov users 2111 авг 23 2024 cda.cpp
```

```
-rw-r--r-- 1 malkov users 121 авг 6 15:23 cda.cu
```

```
-rw-r--r-- 1 malkov users 1314 авг 6 15:25 cda.py
```

```
-rw-r--r-- 1 malkov users 813 авг 6 15:33 cda.ptx
```

```
.version 8.5  
.target sm_52  
.address_size 64
```

cda.ptx

```
.visible .entry gStub(  
    .param .u64 gStub_param_0,  
    .param .u64 gStub_param_1  
)  
{  
    .reg .f32    %f<4>;  
    .reg .b32    %r<5>;  
    .reg .b64    %rd<8>;  
  
    ld.param.u64    %rd1, [gStub_param_0];  
    ld.param.u64    %rd2, [gStub_param_1];
```

```
cvta.to.global.u64    %rd3, %rd2;  
cvta.to.global.u64    %rd4, %rd1;  
mov.u32               %r1, %tid.x;  
mov.u32               %r2, %ctaid.x;  
mov.u32               %r3, %ntid.x;  
mad.lo.s32            %r4, %r2, %r3, %r1;  
mul.wide.s32          %rd5, %r4, 4;  
add.s64               %rd6, %rd4, %rd5;  
ld.global.f32         %f1, [%rd6];  
add.s64               %rd7, %rd3, %rd5;  
ld.global.f32         %f2, [%rd7];  
add.f32               %f3, %f1, %f2;  
st.global.f32         [%rd6], %f3;  
ret;
```

```
}
```

cda.cpp

```
#include <cuda.h>
#include <cuda_runtime.h>
#include <stdio.h>
#include <malloc.h>
```

```
int main(){
    cuInit(0);
    CUdevice cuDevice;
    CUresult res = cuDeviceGet(&cuDevice, 0);
    if (res != CUDA_SUCCESS){
        printf("cannot acquire device 0\n");
        exit(1);
    }
}
```

```
CUcontext cuContext;  
res = cuCtxCreate(&cuContext, 0, cuDevice);  
if (res != CUDA_SUCCESS){  
    printf("cannot create context\n");  
    exit(1);  
}
```

```
int N=2048;  
float* a=(float*)calloc(N, sizeof(float));  
float* b=(float*)calloc(N, sizeof(float));
```

```
for(int i=0; i<N; i++){  
    a[i]=2*i;  
    b[i]=2*i+1;  
}
```

```
float *a_d, *b_d;  
cudaMalloc((void**)&a_d, N*sizeof(float));  
cudaMalloc((void**)&b_d, N*sizeof(float));
```

```
cudaMemcpy(a_d, a, N*sizeof(float), cudaMemcpyHostToDevice);  
cudaMemcpy(b_d, b, N*sizeof(float), cudaMemcpyHostToDevice);
```



```
//gStub<<<N/128,128>>>(a_d,b_d);  
//cudaDeviceSynchronize();
```

```
CUmodule cuModule = (CUmodule)0;  
cuModuleLoad(&cuModule, "cda.ptx");
```

```
//cuModuleLoad(&cuModule, "cda.cubin");
```

```
nvcc -arch=sm_70 -cubin cda.cu
```

```
CUfunction gStub;  
cuModuleGetFunction(&gStub, cuModule, "gStub");
```

```
void* args[] = {&a_d, &b_d};  
cuLaunchKernel(gStub, N/128, 1, 1, 128, 1, 1, 0, 0, args, 0);  
cuCtxSynchronize ();
```

```
cudaMemcpy(a, a_d, N*sizeof(float), cudaMemcpyDeviceToHost);
```

```
for(int i=0; i<N; i+=N/16)  
    printf("%g\n",a[i]);
```

```
cuCtxDestroy(cuContext);  
return 0;
```

```
}
```

```
/Лаб6> g++ -I/usr/local/cuda/include -L/usr/local/cuda/lib64  
-lcudart -lcuda cda.cpp -o cda
```

```
/Лаб9> ls -ltr
```

```
total 40
```

```
-rw-r--r-- 1 malkov users 1775 авг 7 2024 cuda_driver.py  
-rw-r--r-- 1 malkov users 2111 авг 23 2024 cda.cpp  
-rw-r--r-- 1 malkov users 121 авг 6 15:23 cda.cu  
-rw-r--r-- 1 malkov users 1314 авг 6 15:25 cda.py  
-rw-r--r-- 1 malkov users 813 авг 6 15:33 cda.ptx  
-rwxr-xr-x 1 malkov users 16440 авг 6 15:47 cda
```

/Лаб9> **./cda**

1

513

1025

1537

2049

2561

.....

6657

7169

7681

```
from ctypes import *  
import sys
```

cuda_driver.py

```
if 'linux' in sys.platform:  
    cuda = CDLL('libcuda.so')  
elif 'win' in sys.platform:  
    cuda = CDLL('nvcuda.dll')
```

```
CUDA_ERRORS = {0 : 'CUDA_SUCCESS', 1 : 'CUDA_ERROR_INVALID_VALUE',  
200 : 'CUDA_ERROR_INVALID_IMAGE', 201 :  
'CUDA_ERROR_INVALID_CONTEXT ', 400 : 'CUDA_ERROR_INVALID_HANDLE' }
```

```
culnit = cuda.culnit  
culnit.argtypes = [c_uint]  
culnit.restype = int
```

.....

```
cuDeviceGet = cuda.cuDeviceGet  
cuDeviceGet.argtypes = [POINTER(c_int), c_int]  
cuDeviceGet.restype = int
```

```
cuCtxCreate = cuda.cuCtxCreate  
cuCtxCreate.argtypes = [c_void_p, c_uint, c_int]  
cuCtxCreate.restype = int
```

```
cuModuleLoad = cuda.cuModuleLoad  
cuModuleLoad.argtypes = [c_void_p, c_char_p]  
cuModuleLoad.restype = int
```

```
.....  
cuModuleGetFunction = cuda.cuModuleGetFunction  
cuModuleGetFunction.argtypes = [c_void_p, c_void_p, c_char_p ]  
cuModuleGetFunction.restype = int  
.....
```

```
import numpy as np
from cuda_driver import *
```

```
N=2048
```

```
culnit(0)
```

```
cnt = c_int(0)
cuDeviceGetCount(byref(cnt))
if cnt.value == 0:
    raise Exception('No GPU device found!')
```

```
cuDevice = c_int(0)
cuDeviceGet(byref(cuDevice), 0)
```

```
cuContext = c_void_p()
cuCtxCreate(byref(cuContext), 0, cuDevice)
```

cda.py

```
a_a = np.linspace(0, 2*(N-1), N, dtype=np.float32)
a = a_a.ctypes.data_as(POINTER(c_float))
b_a = np.linspace(1, 2*N-1, N, dtype=np.float32)
b = b_a.ctypes.data_as(POINTER(c_float))
```

```
a_d = c_void_p(0)
cuMemAlloc(byref(a_d), c_size_t(N*sizeof(c_float)))
b_d = c_void_p(0)
cuMemAlloc(byref(b_d), c_size_t(N*sizeof(c_float)))
```

```
cuMemcpyHtoD(a_d, a, c_size_t(N*sizeof(c_float)))
cuMemcpyHtoD(b_d, b, c_size_t(N*sizeof(c_float)))
```



```
cuModule = c_void_p()
cuModuleLoad(byref(cuModule), c_char_p(b'./cda.ptx'))

gStub_kern = c_void_p(0)
cuModuleGetFunction(byref(gStub_kern), cuModule, c_char_p(b'gStub'))

gStub_args=[c_void_p(addressof(a_d)), c_void_p(addressof(b_d))]
gStub_params = (c_void_p * len(gStub_args))(*gStub_args)
cuLaunchKernel(gStub_kern, int(N/128), 1, 1, 128, 1, 1, 0, 0, gStub_params, 0)
cuCtxSynchronize()

cuMemcpyDtoH( a, a_d, c_size_t(N*sizeof(c_float)))
print(a_a)

cuMemFree(a_d)
cuMemFree(b_d)
cuCtxDestroy(cuContext)
```

```
/Лаб9> python cda.py
```

```
1.0
```

```
513.0
```

```
1025.0
```

```
1537.0
```

```
2049.0
```

```
2561.0
```

```
.....
```

```
6657.0
```

```
7169.0
```

```
7681.0
```

<https://pypi.org/project/pycuda/>

~/Lab9> pip install pycuda

```
import numpy as np
import pycuda
from pycuda import gpuarray
import pycuda.autoinit
```

cda_pc.py

```
N=2048
cuModule = pycuda.driver.module_from_file('./cda.ptx')
gStub_kern = cuModule.get_function('gStub')
a_d = gpuarray.to_gpu(np.linspace(0, 2*N-2, N, dtype=np.float32))
b_d = gpuarray.to_gpu(np.linspace(1, 2*N-1, N, dtype=np.float32))

gStub_kern(a_d, b_d, grid=(int(N/128),1,1), block=(128,1,1))
print(a_d.get())
```