

Run in background job

Submit the job with:

```
cd gpu-computing
qsub k20.pbs
```

Example output:

```
2878521.master01.hydra.brussel.vsc
```

you can check the state of the job with:

```
qstat
```

output:

Job ID	Name	User	Time Use	S	Queue
2878519.master01.hydra.brusse	k20.pbs	aabouzai	00:00:01	C	gpu

The S column is the state, here C means complete ([more infos on states and qstat](#)).

When the job is complete two files containing the `stderr` and `stdout` outputs are created.

```
-bash-4.2$ cat k20.pbs.o2878521
Show all GPUS:
- Tesla K20Xm on NVIDIA CUDA: 14 cores 732MHz vector width=1
```

Show all devices:

Platform 0

```
Name:      NVIDIA CUDA
Vendor:     NVIDIA Corporation
Profile:    FULL_PROFILE
Version:    OpenCL 1.2 CUDA 10.1.236
```

Device: 0

```
Name                : Tesla K20Xm
OpenCL C Version     : OpenCL C 1.2
#Compute Units (cores) : 14
Native vector width  : 1
2D Image limits      : 16384x16384
3D Image limits      : 4096x4096x16384
Maximum buffer size [MB]: 1425
Local memory size [KB] : 48
Maximum workgroup size : 1024
Timer resolution     : 1000
Clock frequency       : 732
```

to use others gpus

Some gpu are less available (probably are already in use by other users). Files are: * `gtx.pbs` GTX 1080ti * `k20.pbs` Tesla K20Xm * `p100.pbs` Tesla P100

[more details on gpu types](#)

Run in interactive mode.

Change the `feature=kepler` to (`geforce` for GTX 1080 or `pascal` for P100) Other parameters: [link](#)

```
qsub -I -l nodes=1:ppn=1:gpus=1 -l walltime=00:05:00 -l pmem=1gb -l feature=kepler
module purge
module load CMake/3.15.3-GCCcore-8.3.0
module load NCCL/2.2.13-CUDA-9.2.148.1
```

Now you can compile OpenCL code, for example:

```
cd gpu-computing
mkdir build
cd build
cmake ../src
make
../bin/showDevices
cd ../bin && ./sumInts -p 0 -d 0 -s 1048576
# or cd sumInts && ../../bin/sumInts -p 0 -d 0 -s 1048576
# the ocl file should be in the current dir
exit # when finished
```