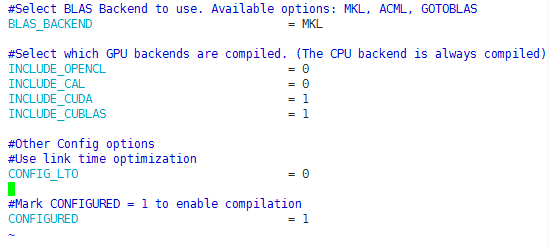
一、配置caldgemm (caldgemm的blas后端有mkl,gotoblas2,amd的acml；MPI后端有openmpi, mvpich,mvapich2；dgemm操作后端有cpu, opencl, cuda和CAL).

1. caldgemm下载：  
git clone <https://github.com/davidrohr/caldgemm.git>  
  
2，进入caldgemm目录下：

cp config\_options.sample config\_options.mak

cp caldgemm\_config.sample caldgemm\_config.h

修改config\_options.mak内容为：



此处默认BLAS后端为MKL，dgemm后端为CUDA。

3. vim ~/.bashrc

# CUDA

export LOCAL\_DIR=/usr/local

#export CUDA\_65\_HOME=$LOCAL\_DIR/cuda-6.5

#export CUDA\_70\_HOME=$LOCAL\_DIR/cuda-7.0

export CUDA\_75\_HOME=$LOCAL\_DIR/cuda-7.5

#export CUDA\_80\_HOME=$LOCAL\_DIR/cuda-8.0

export CUDA\_HOME=$CUDA\_75\_HOME

export PATH=$CUDA\_HOME/bin:$PATH

export LD\_LIBRARY\_PATH=$CUDA\_HOME/lib64:$CUDA\_HOME/lib:$LD\_LIBRARY\_PATH

#export PATH=/home/th/TensorRT-2.1.2/bin:$PATH

#export LD\_LIBRARY\_PATH=/home/th/TensorRT-2.1.2/lib:$LD\_LIBRARY\_PATH

#export PATH=/home/th/cmake-3.8.2/build/bin:$PATH

export PATH=/opt/rocm/hip/bin:$PATH

export PATH=/usr/bin:$PATH

export LD\_LIBRARY\_PATH=/usr/lib:/usr/lib32:/usr/lib/x86\_64-linux-gnu:$LD\_LIBRARY\_PATH

#openmpi

#export PATH=/home/th/openmpi-1.6.5/build/bin:$PATH

#export LD\_LIBRARY\_PATH=/home/th/openmpi-1.6.5/build/lib:$LD\_LIBRARY\_PATH

#GotoBLAS2

#export LD\_LIBRARY\_PATH=/home/th/GotoBLAS2:$LD\_LIBRARY\_PATH

#gotoblas\_path

#export LD\_LIBRARY\_PATH=/home/th/caldgemm/gotoblas\_patch:$LD\_LIBRARY\_PATH

#mpich2

export PATH=/home/th/MPI/mpich2-1.3/bin:$PATH

export LD\_LIBRARY\_PATH=/home/th/MPI/mpich2-1.3/lib:$LD\_LIBRARY\_PATH

export MPICH2\_LIBRARIES=$MPICH2\_INSTALL\_PATH/lib

export LD\_LIBRARY\_PATH=/home/th/hpl/src/cuda:$LD\_LIBRARY\_PATH

#mkl

export PATH=/opt/intel/bin:$PATH

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/opt/intel/lib/intel64:/opt/intel/mkl/lib/intel64

source /opt/intel/compilers\_and\_libraries\_2018.1.163/linux/mkl/bin/mklvars.sh intel64

export OPENCL\_ROOT=/opt/rocm/opencl

export PATH=/opt/rocm/bin:$PATH

#GotoBLAS2

#export BLAS\_INSTALL\_PATH=/home/th/GotoBLAS2

#export BLAS\_INSTALL\_PATH=/home/th/caldgemm/gotoblas\_patch

#export CUBLAS\_INSTALL\_PATH=/usr/local/cuda-7.5

#export BLAS\_INCLUDES=$BLAS\_INSTALL\_PATH

#export BLAS\_LIBRARIES=$BLAS\_INSTALL\_PATH

#export CUBLAS\_INCLUDES=/usr/local/cuda-7.5/include

#export CUBLAS\_LIBRARIES=/usr/local/cuda-7.5/lib64

（如用acml后端则修改如下：

#export BLAS\_INSTALL\_PATH=/home/th/acml5.3.1/gfortran64\_mp

#gotoblas no path to include&lib

#export BLAS\_INCLUDES=$BLAS\_INSTALL\_PATH/include

#export BLAS\_LIBRARIES=$BLAS\_INSTALL\_PATH/lib

#Base path to Intel software

export INTELPATH=/opt/intel

#Path to the Intel MKL BLAS library (usually inside $INTELPATH)

export MKL\_PATH=$INTELPATH/mkl

#Path to libiomp (as part of ICC or standalone)

export ICC\_PATH=$INTELPATH

#Path to Intel TBB library inside MKL (Comment out if you want to use MKL TBB instead of having hpl-gpu compile tbb)

export TBB\_PATH=$INTELPATH/tbb

#Path to the GotoBLAS BLAS library

#export GOTOBLAS\_PATH=/home/th/caldgemm/gotoblas\_patch

#Path to AMD ACML BLAS library

#export ACML\_PATH=$HOME/acml/gfortran64\_mp

#Path to the CBLAS interface (required for ACML BLAS Slibrary)

#export CBLAS\_PATH=$HOME/CBLAS

#Path to NVIDIA CUDA SDK

export CUDA\_PATH=$CUDA\_HOME

###Add all required paths to $LD\_LIBRARY\_PATH

#We want to use the most recent AMD OpenCL library. Usually this comes with the driver. If the SDK is newer than the driver, outcomment the next line.

#export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:$AMDAPPSDKROOT/lib/x86\_64

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:$CUDA\_PATH/lib64

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:$ICC\_PATH/lib/intel64:$MKL\_PATH/lib/intel64

#export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:$ACML\_PATH/lib

#export LD\_LIBRARY\_PATH=$OPENMPI\_PATH/lib:$LD\_LIBRARY\_PATH

#We need one library path at the very beginning that overrides all others for preloading libraries

export LD\_LIBRARY\_PATH=$HOME/lib:$LD\_LIBRARY\_PATH

###Add OpenMPI to $PATH for mpirun command

#export PATH=$OPENMPI\_PATH/bin:$PATH

###Set some environment variables for AMD GPUs and for Headless X Setup

export GPU\_FORCE\_64BIT\_PTR=1

export GPU\_NUM\_COMPUTE\_RINGS=1

export DISPLAY=:0

export COMPUTE=:0

最后 source ~/.bashrc  
  
4,修改程序里内容  
benchmark.cpp  
71     bool mem\_gpu\_access = true;  
76     int use\_opencl\_not\_cal = 2;  
239         Config->Debug = true;

**注意：** 如果此处caldgemm里blas用的后端是GotoBLAS2, 则需要下载GotoBLAS2并重命名为GotoBLAS2-org放在$HOME/caldgemm目录下，再

cd gotoblas\_patch

执行: patch -p0 < gotoblas.patch

执行完后需要把/home/th/caldgemm/gotoblas\_patch/driver/others/blas\_server.c 里的setThreadtime() 相关的三行给注释掉

最后在gotoblas\_patch文件夹下编译新的gotoblas2，即 make TARGET=NEHALEM CC=gcc FC=gfortran BINARY=64

如果报错有can’t find -l -l，则修改f\_check的第298行如下:

"print MAKEFILE "FEXTRALIB=$linker\_L -lgfortran -lm -lquadmath -lm -lc $linker\_a\n";"

最后vi config\_options.mak把blas改为GOTOBLAS 再make 即可。

**注意：**如果需要编译acml后端，则需要在<http://developer.amd.com/tools-and-sdks/acml-downloads-resources/#download> 下载

|  |  |
| --- | --- |
| 窗体顶端  acml-5-3-1-gfortran-64bit.tgz  窗体底端 |  |

解压后执行如下脚本：解压后执行如下脚本：

./install-acml-5-3-1-gfortran-64bit.sh

再在<http://www.netlib.org/blas/blast-forum/cblas.tgz下载cblas>并解压，并把上一步编译完acml在/home/th/acml5.3.1/gfortran64\_mp/lib下生成的libacml\_mp.和.so文件都复制到$HOME/CBLAS/testing目录下，同时在$HOME/CBLAS目录下执行：

cp Makefile.LINUX Makefile.in

vi Makefile.in

把 BLLIB=libblas.a 改为 libacml\_mp.so

然后执行 make all

此时会在 $HOME/CBLAS/lib/LINUX下生成 cblas\_LINUX.a,

然后 cp cblas\_LINUX.a ../

最后进入$HOME/caldgemm下修改config\_options.make的blas为ACML再 make.

编译caldgemm:

make 后会生成dgemm\_bench,运行./dgemm\_bench没报错即可。

二、配置HPL-GPU

1，配置caldgemm

2，下载 hpl\_gpu 到caldgemm同级目录下

git clone <https://github.com/davidrohr/hpl-gpu.git>

3,

[hplgpuxll@gpu200 ~]$ ls

caldgemm  hpl-gpu

[hplgpuxll@gpu200 ~]$ cd hpl-gpu/

[hplgpuxll@gpu200 hpl-gpu]$ ln -s ../caldgemm

[hplgpuxll@gpu200 hpl-gpu]$ cp setup/Make.Generic ./

[hplgpuxll@gpu200 hpl-gpu]$ cp setup/Make.Generic.Options ./

[hplgpuxll@gpu200 hpl-gpu]$ vi Make.Generic.Options

修改Make.Generic.Options如下内容：

HPL\_CONFIG\_MPI = 0

HPL\_CONFIG\_VERBOSE = 3

HPL\_CALDGEMM\_BACKEND = cuda

修改caldgemm/caldgemm.cpp  
3996行：   retVal &= Config->InitializeBackendOptions();

在hpl-gpu/caldgemm目录下重新编译caldgemm  
在hpl-gpu目录下：

./build.sh 编译

./run.sh 测试

\*\*caldgemm需要先编译通过，才能继续编译hpl\_gpu，未解决的问题：编译hpl-gpu时报错: unreferenced to ‘fatbindata’.