GHENT
UNIVERSITY

FACULTY OF ENGINEERING AND ARCHITECTURE

DEPARTMENT OF FLOW, HEAT AND COMBUSTION MECHANICS
RESEARCH GROUP FLUID MECHANICS

HANDS-ON: BUILDING OWN SOLVERS & LIBRARIES

Joris Degroote

GHENT UNIVERSITY

Vlaams Supercomputer Centrum

OUTLINE

- Basics about compiling, executables and libraries
- Make your own solver, based on existing solver
- Make your own library and use it in existing solver

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES

Compiling = source files(.C) → object files (.o)

human readable → machine readable

Libraries = collections of object files which cannot be executed directly

→ Can be used by multiple executables

COMPILING, EXECUTABLES AND LIBRARIES

Compiling = source files(.C) → object files (.o)

human readable → machine readable

↓

Linking = combining object files (.o), static libraries (.a) or shared libraries (.so) to create executable (binary, no extension on Linux)

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES

Static linking = include (part of) static library (.a) in executable

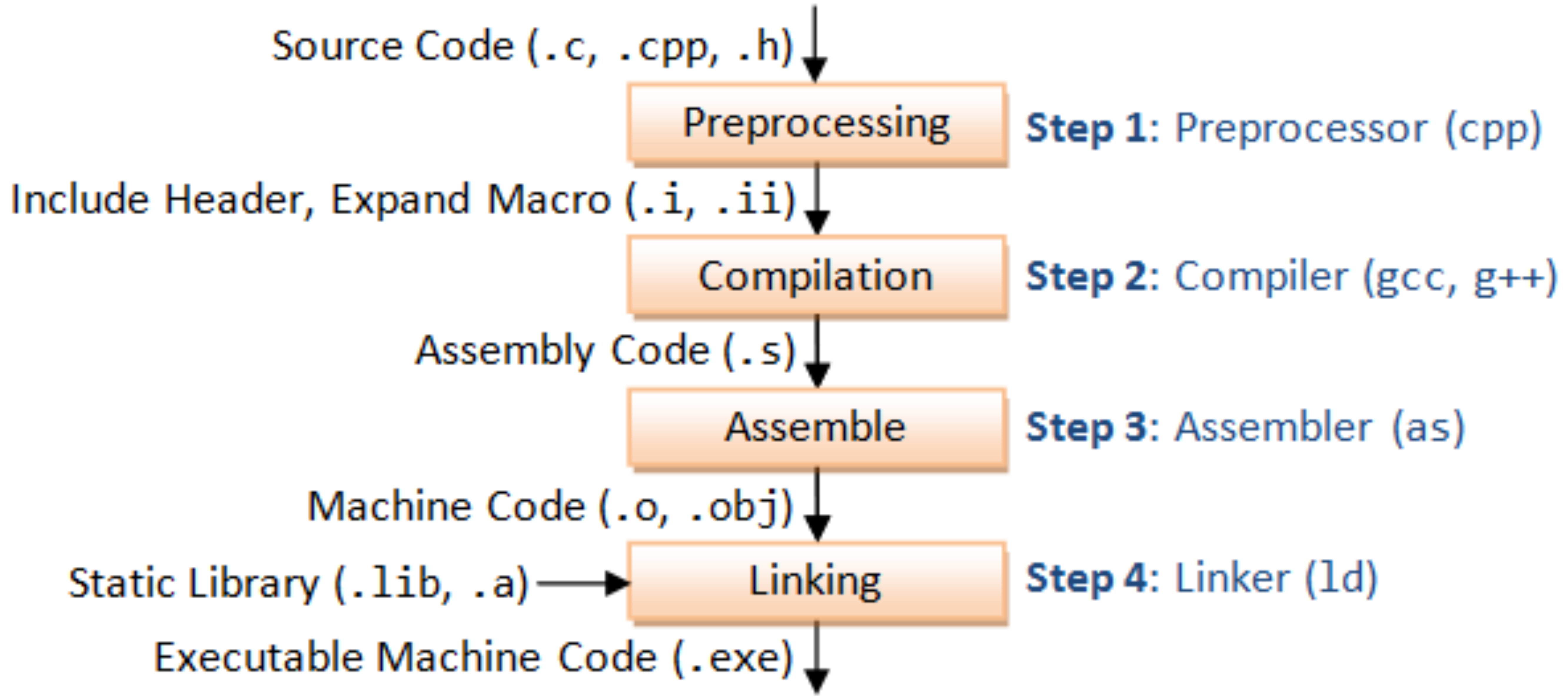
→ Large executable

Dynamic linking = create link to shared library (.so) in executable, so functions can be found

→ Small executable

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES


Source Code (.c, .cpp, .h)
Preprocessing
Step 1: Preprocessor (cpp)
Include Header, Expand Macro (.i, .ii)
Compilation
Step 2: Compiler (gcc, g++)
Assembly Code (.s)
Assemble
Step 3: Assembler (as)
Machine Code (.o, .obj)
Static Library (.lib, .a)
Linking
Step 4: Linker (ld)
Executable Machine Code (.exe)

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES

Example

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES

./src/Piece.h King.h Queen.h Tower.h … Chess.C

King.C Queen.C Tower.C

↓ Compile

King.o Queen.o Tower.o … Chess.o

↓ Link

./bin/chess

GHENT UNIVERSITY

COMPILING, EXECUTABLES AND LIBRARIES

```
./src/Piece.h    King.h   …  Chess.C  ...  Screen.C …
                 King.C
    ↓ Compile
                 King.o   …  Chess.o …     Screen.o …
    ↓ Link
./bin/chess
./lib/libgraphics.so
```

COMPILING, EXECUTABLES AND LIBRARIES

OpenFOAM uses dynamic linking with shared libraries

`wmake`	= compile all required source code and link as executable (binary) → Typically depends on several libraries
`wmake libso`	= compile all required source code and package as shared library → Can depend on other libraries

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

1. Interactive job
2. Copy existing solver
3. Change name
4. Change settings
5. Compile
6. Test run

MAKE YOUR OWN SOLVER

```
qsub -I -l walltime=00:59:59

module load OpenFOAM/4.1-intel-2017a

module list

source $FOAM_BASH
```

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

```
echo $FOAM_APPBIN
```

→ Location of official binaries

```
echo $FOAM_USER_APPBIN
```

→ Location of own binaries

MAKE YOUR OWN SOLVER

OpenFOAM structure

```
cd $WM_PROJECT_DIR
```

src/	→ source code of libraries
applications/solvers	→ source code of solvers
applications/utilities	→ source code of utilities
platforms/	→ binaries and libraries

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

Create same structure in own directory

```
mkdir -p $WM_PROJECT_USER_DIR
cd $WM_PROJECT_USER_DIR
```

(typically $VSC_HOME/OpenFOAM/username-version)

run/	→ simulation cases and results
src/	→ source code of own libraries
applications/solvers	→ source code of own solvers
applications/utilities	→ source code of own utilities
platforms/	→ binaries and libraries

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

```
cd $WM_PROJECT_DIR/applications/solvers

cd incompressible/

cp -r icoFoam $WM_PROJECT_USER_DIR/applications/solvers/myFoam

cd $WM_PROJECT_USER_DIR/applications/solvers/myFoam
```

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

```
mv icoFoam.C myFoam.C
```

Edit “myFoam.C”

```
  Info<< "Bye bye from myFoam\nEnd\n" << endl;
```

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

```
cd Make
ls
```

files

options

MAKE YOUR OWN SOLVER

Edit “files”

```
myFoam.C

EXE = $(FOAM_USER_APPBIN)/myFoam
```

MAKE YOUR OWN SOLVER

Edit “options”

```
EXE_INC = \
  -I$(LIB_SRC)/finiteVolume/lnInclude \
  -I$(LIB_SRC)/meshTools/lnInclude
```

→ Headers to be included when compiling executable

```
EXE_LIBS = \
    -lfiniteVolume \
    -lmeshTools
```

→ Libraries to be included when linking executable

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

```
cd ..

wmake

ls $FOAM_USER_APPBIN
```

GHENT UNIVERSITY

MAKE YOUR OWN SOLVER

```
cd $FOAM_RUN

cp -r $FOAM_TUTORIALS/incompressible/icoFoam/cavity/cavity myCavity

cd myCavity

blockMesh

myFoam
```

GHENT UNIVERSITY

MAKE YOUR OWN LIBRARY

GHENT UNIVERSITY

MAKE YOUR OWN LIBRARY

1. Interactive job
2. Copy part of existing library
3. Change name
4. Change settings
5. Compile
6. Test run

MAKE YOUR OWN LIBRARY

```
echo $FOAM_LIBBIN
```

→ Locations of official libraries

```
echo $FOAM_USER_LIBBIN
```

→ Locations of own libraries

MAKE YOUR OWN LIBRARY

```
cd $WM_PROJECT_DIR/src

cd functionObjects/utilities

cp -r writeDictionary $WM_PROJECT_USER_DIR/src/myWriteDictionary

cd $WM_PROJECT_USER_DIR/src/myWriteDictionary
```

GHENT
UNIVERSITY

MAKE YOUR OWN LIBRARY

```
mv writeDictionary.C myWriteDictionary.C
mv writeDictionary.H myWriteDictionary.H
```

Edit both files and replace “writeDictionary” by “myWriteDictionary”

Edit “myWriteDictionary.C”

```
Foam::functionObjects::myWriteDictionary::~myWriteDictionary()
{
    Info<< "Bye bye from myWriteDictionary" << endl;
}
```

GHENT UNIVERSITY

MAKE YOUR OWN LIBRARY

```
cp -r $WM_PROJECT_DIR/src/functionObjects/utilities/Make .
cd Make
ls
```

files

options

GHENT UNIVERSITY

MAKE YOUR OWN LIBRARY

Edit “files”

```
myWriteDictionary.C

LIB = $(FOAM_USER_LIBBIN)/libmyWriteDictionary
```

GHENT
UNIVERSITY

MAKE YOUR OWN LIBRARY

Edit “options”

```
EXE_INC = \
    -I$(LIB_SRC)/finiteVolume/lnInclude

LIB_LIBS = \
    -lfiniteVolume
```

GHENT
UNIVERSITY

MAKE YOUR OWN LIBRARY

```
cd ..

wmake libso

ls $FOAM_USER_LIBBIN
```

GHENT UNIVERSITY

MAKE YOUR OWN LIBRARY

```
cd $FOAM_RUN/myCavity
```

Edit “system/controlDict”

```
functions
{
    writeDictionary1
    {
        type    myWriteDictionary;

        libs    ("libmyWriteDictionary.so");

        dictNames (controlDict);
    }
}
```

GHENT
UNIVERSITY

MAKE YOUR OWN LIBRARY

```
myFoam
```

Check output

```
…
Bye bye from myFoam
End

Bye bye from myWriteDictionary
```

TIPS

Use variables for paths, do not hard code them

Use binaries and libraries only on cluster that has been used for compiling

Use ldd to check dependency on shared libraries

Study C++ (Stroustrup, …)

GHENT UNIVERSITY

REFERENCES

[1] H. Jasak, Introduction to OpenFOAM: Programming in OpenFOAM. 2016. https://www.youtube.com/playlist?list=PLqxhJj6bcnY9RoIgzeF6xDh5L9bbeK3BL

FACULTY OF ENGINEERING
AND ARCHITECTURE

Joris Degroote
Associate professor

DEPARTMENT OF FLOW, HEAT AND
COMBUSTION MECHANICS

E	joris.degroote@ugent.be
T	+32 9 264 95 22

www.ugent.be

Ghent University
@ugent
Ghent University

GHENT
UNIVERSITY