BIAS – CMake – Tutorial

09/02/2010

BIAS Version 2.7.0 (http://www.mip.informatik.uni-kiel.de/BIAS)
CMake Version 2.6 patch 2 (www.cmake.org)

Ingo Schiller
Multimedia Information Processing
Institute of Computer Science
Christian-Albrechts-University of Kiel
ischiller@mip.informatik.uni-kiel.de

Outline

1. What is CMake ?
2. Finding external libraries
3. Setting Variables (BIAS_HAVE_ …)
   BIASConfig.cmake (BIASConfig.cmake.in)
4. Debug/Release Builds
5. Libraries
6. Testing
7. Using BIAS in other Projects

1. What is CMake ?

- CMake is a cross-platform open-source build system
- Produces Makefiles / Eclipse projects / Visual Studio Projects / KDevelop …

With CMake you can:

- Search for other include files and libraries
- Set flags and test flags
- Specify own libraries and executables
- Include / Exclude subdirectories/files
- Build shared or static libs
- Set build types such as Debug/Release/…
- Hand over variables to other projects
- Automatically test software
- …much more

1. Preparations

- CMake uses files CMakeLists.txt in all directories
- BIAS uses own find scripts for extern libraries located in CMakeModules
  Find scripts are used to find include directories and libraries

- You have to set the environment variable: CMAKE_MODULE_PATH
  and point it to the directory of the CMakeModules directory.

Example:

Linux: file /home/xx/.bashrc :
  export CMAKE_MODULE_PATH = /home/xx/devel/CMakeModules

WinXP: set as variable in MyComputer->(right click)Properties->
  Advanced->Environment Variables

- Using the FindScripts (next slide) will first search in the CMakeModule path
  and then in the CMake default search paths

2. Finding External Libraries 1

- Libraries are searched for using a “Findxxx.cmake” – script file

Principle:

- Set search path for libraries and headers.

```
Example: SET (OpenCV_POSSIBLE_ROOT_DIRS
                “$ENV{OPENCV_DIR}”
                “$ENV{OpenCV_ROOT_DIR}”
                “$ENV{EXTERN_LIBS_DIR}/OpenCV”
                /usr/local
                /usr)
```

Windows: All external libraries are in our Bazaar Repository “ExternLibs”
specified by environment variable EXTERN_LIBS_DIR

- Select one path from the list above

```
Example : FIND_PATH(OpenCV_ROOT_DIR
                NAMES
                cv/include/cv.h
                include/opencv/cv.h
                include/cv/cv.h
                PATHS ${OpenCV_POSSIBLE_ROOT_DIRS}
                NO_DEFAULT_PATH)
```

2. Finding External Libraries 2

- Find library:

Example: FIND_LIBRARY(OpenCV_CV_LIBRARY
NAMES cv opencv cv0.9.7
PATHS ${OpenCV_ROOT_DIR}
PATH_SUFFIXES ${OpenCV_LIBDIR_SUFFIXES}
NO_CMAKE_SYSTEM_PATH)

- Set variable that you found your external package to use later

Example: SET(OpenCV_FOUND ON)

- Set variables for includes and libs to link against: (Multiple libs/dirs, specified by ${NAME}.)

Example: LIST(APPEND OpenCV_INCLUDE_DIRS
${OpenCV_${NAME}_INCLUDE_DIR})
LIST(APPEND OpenCV_LIBARIES
${OpenCV_${NAME}_LIBRARY})

SET(OPENCV_INCLUDE_DIR ${OpenCV_INCLUDE_DIRS})
SET(OPENCV_LIBRARIES ${OpenCV_LIBRARIES})

- Add libs to EXTERN_LIBS_BIAS !!

3. Setting Variables 1

- Variables are used to tell in BIAS and extern Projects about BIAS
- We use BIAS_HAVE_XXX !
- Define BIAS_HAVE_XXX in BIASConfig.cmake.in

Example: SET(BIAS_HAVE_OPENCV "@BIAS_HAVE_OPENCV@")

- Set the variable in BIAS/CMakeLists.txt

Example: Set(BIAS_HAVE_OPENCV ON) or
Set(BIAS_HAVE_OPENCV TRUE) or
Set(BIAS_HAVE_OPENCV 1)

- CMake parses BIASConfig.cmake.in and generates BIASConfig.cmake automatically which is used by external projects

3. Setting Variables 2

- To use BIAS_HAVE_XXX Flags in code we use bias_config.h
- bias_config.h is generated from bias_config.h.in by BIAS/GenerateConfigHeader.cmake
- If you want a BIAS_HAVE_XXX - Flag in bias_config.h put it in bias_config.h.in
- Has to have exactly the same name!!

Example: // compiled with OpenCV usage ?
#cmakedefine BIAS_HAVE_OPENCV

- Can also be used for other systemwide includes, but with care!!!

4. Debug/Release Builds

- We support many debug/optimized build settings
- Generally 2 are needed: Debug & Release
- We add a “D” at the end of our debug libraries to distinguish from release
- Done automatically by SET(CMAKE_DEBUG_POSTFIX D)
- This works only (!) if CMAKE_BUILD_TYPE is set correctly (Debug/Release)
- CMake automatically selects the correct libraries!

5. Libraries

- We collect all BIAS libraries in a variable BIAS_ALL_LIBS in BIAS/CMakeLists.txt
- If you add a library, add it to the list!
- A list BIAS_ALL_LIBS_REL_DBG is generated holding all debug and optimized libraries.
- This looks like this:
  …optimized BIASBaseImage debug BIASBaseImageD …
- We add all external libraries to BIAS_LIBRARIES, because dependencies to external libs are lost in debug case otherwise!
- In BIASConfig.cmake.in the BIAS_LIBRARIES variable is set to BIAS_ALL_LIBS_REL_DBG
  (SET(BIAS_LIBRARIES @BIAS_ALL_LIBS_REL_DBG@)
- This is transferred to BIASConfig.cmake by CMake
- In BIAS link to the single libraries (e.g. BIASImage, BIASGeometry)
- Outside of BIAS link all applications to BIAS_LIBRARIES.
  The necessary libs are selected automatically

6. Testing

- Testing is an important feature!
- Executed nightly by CMake Nightlybuild
- Add test with :
  ADD_TEST(NAME testName
  COMMAND ${EXECUTABLE_OUTPUT_PATH}/Name [arg1 [arg2 ...]])
- Test data can be included in BIAS/Tests/data
- Variabel BIAS_TESTS_DATA points to that directory (in bias_config.h)

Example : Load image
string filename(BIAS_TESTS_DATA "r4.jpg");

- If an executable returns 0, the test is passed, otherwise it is marked as failed. If a program chrashes it is also failed.
- If a test should fail you can use: e.g.
  ADD_TESTS_EXPECTED_COMPILER_ERRORS(TestSharedPtr.cpp TEST_COMPILE_TIME_ERROR_)
- Tests can be executed locally by enabling Tests in CCMake and typing: make test

7. Using BIAS in other Projects 1

- Set CMakeModule Path:
  SET(CMAKE_MODULE_PATH $ENV{CMAKE_MODULE_PATH})
- Search for BIAS with FIND_PACKAGE
  FIND_PACKAGE(BIAS) // executes FindBIAS.cmake
  IF(BIAS_FOUND)
    INCLUDE(${BIAS_USE_FILE}) #loads BIASConfig.cmake
    SEPARATE_ARGUMENTS(BIAS_WXWIDGETS_DEFINITIONS)
    ADD_DEFINITIONS(${BIAS_WXWIDGETS_DEFINITIONS})
  ENDIF(BIAS_FOUND)
- Set include directories:
  INCLUDE_DIRECTORIES(${BIAS_INCLUDE_DIR}
  ${BIAS_INCLUDE_DIR}/BIAS
  ${WXWIDGETS_INCLUDE_DIR})
- Set Link directories:
  LINK_DIRECTORIES(${BIAS_LINK_DIRECTORIES})

7. Using BIAS in other Projects 2

- Set CXX and C Flags:

```
SET(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} $
{BIAS_CXX_FLAGS}")

SET(CMAKE_C_FLAGS "${CMAKE_C_FLAGS}
${BIAS_C_FLAGS}")
```

- Set build-type as in BIAS:
  SET (CMAKE_BUILD_TYPE ${BIAS_BUILD_TYPE} CACHE STRING "build type for My Project" FORCE)

- Same for other projects you want to add!