## A. Introduction

1. This build system can be run on Windows and Linux hosts. Usually the hosts is chosen according to the tools and utilities that are used in specific project.
2. This build system is based on Makefiles, so shell(Linux) or command line(Windows) is natural environment to run it. You can use any IDE that support custom build command. In custom build command you just need to call 'make' utility with proper target. (will be explained in details in one of following sections)

## B. Mandatory requirements.

1. Make utility

Build system is based on GNU Make utility. The required version is at least 4.1.

i) Windows:

Make utility that built in Windows OS is very limited and so not supported!
Usually, the Make utility for windows is located in:
$(YOUR_PATH)/uCWorkspace/uCProjects/windows/make/make4.1

ii) Linux:

Usually make utility is sytem wide. Only make sure that version is as required.

You can change the default location of Make utility by editing the file
$(YOUR_PATH)/uCWorkspace/uCProjects/workspace_config.mk.
Uncomment/add/change variable REDEFINE_MAKE_PROGRAM_DIR and assigning to it the path of Make utility folder.
For example: REDEFINE_MAKE_PROGRAM_DIR = c:\make4.1 .
Make sure that this folder contains 'bin' sub-folder with 'make' executable.

2. GIT

GIT is critical component. It's used not only for source control but for synchronization of repositories.
Two options available:

i) Install system wide GIT.

a) Install from official online source.
b) After installing check that installation succeed: open 'cmd' or 'shell' window and run 'git'command. You should see git help/usage output. If command not found then system PATH needs to be fixed.

ii) Use GIT located in particular folder.

a) Edit file $(YOUR_PATH)/uCWorkspace/uCProjects/workspace_config.mk by uncomment/add variable REDEFINE_GIT_ROOT_DIR and assigning to it the path of GIT folder. For example: REDEFINE_GIT_ROOT_DIR = c:\my_git_dir

## C. Optional requirements.

1. KConfig utility

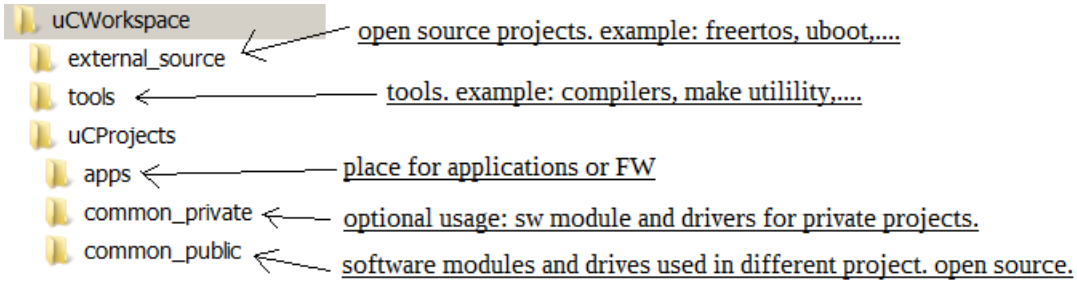
Project configuration is based on .config files.
You can find more about KConfig language in https://www.kernel.org/doc/html/latest/kbuild/kconfig-language.html

i) Windows:
$(YOUR_PATH)/uCWorkspace/uCProjects/windows/kconfig/kconfig3.12.0
ii) Linux:
Usually kconfig is installed system wide during installation of build-essential for kernel. Check according to you distribution.

You can change the default location of KConfig utility by editing the file $(YOUR_PATH)/uCWorkspace/uCProjects/workspace_config.mk.
Uncomment/add/change variable REDEFINE_KCONFIG_DIR and assigning to it the path of KConfig utility folder.
For example: REDEFINE_MAKE_PROGRAM_DIR = c:\kconfig .
Make sure that this folder contains 'kconfig-mconf' executable.

D. Workspace folder structure.

i. Note for Windows: Don't put uCWorkspace folder in too deep path because Windows has limitation for length of path name.

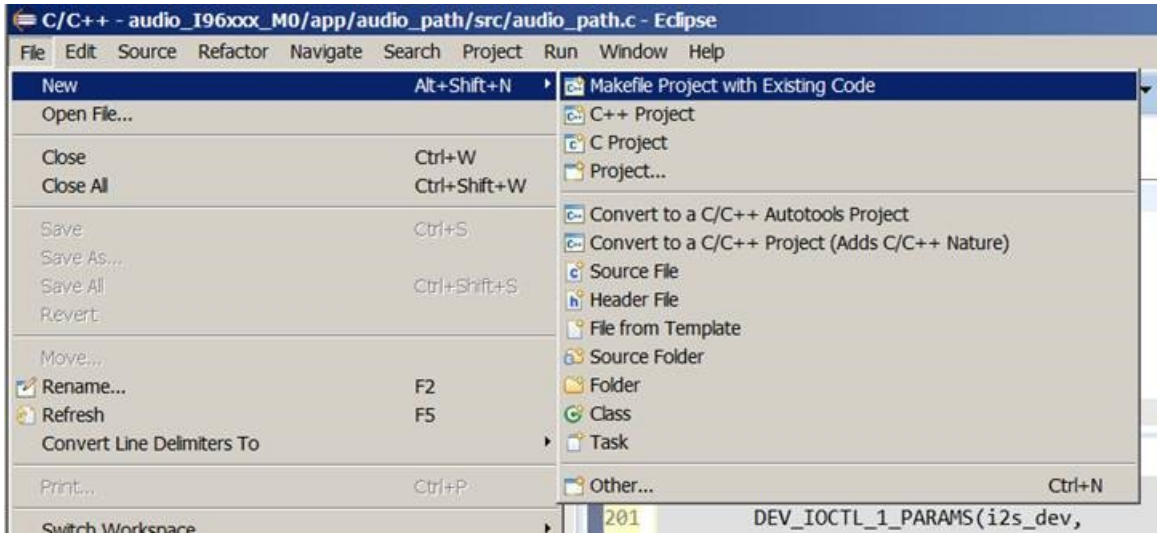



E. Compilation of project.

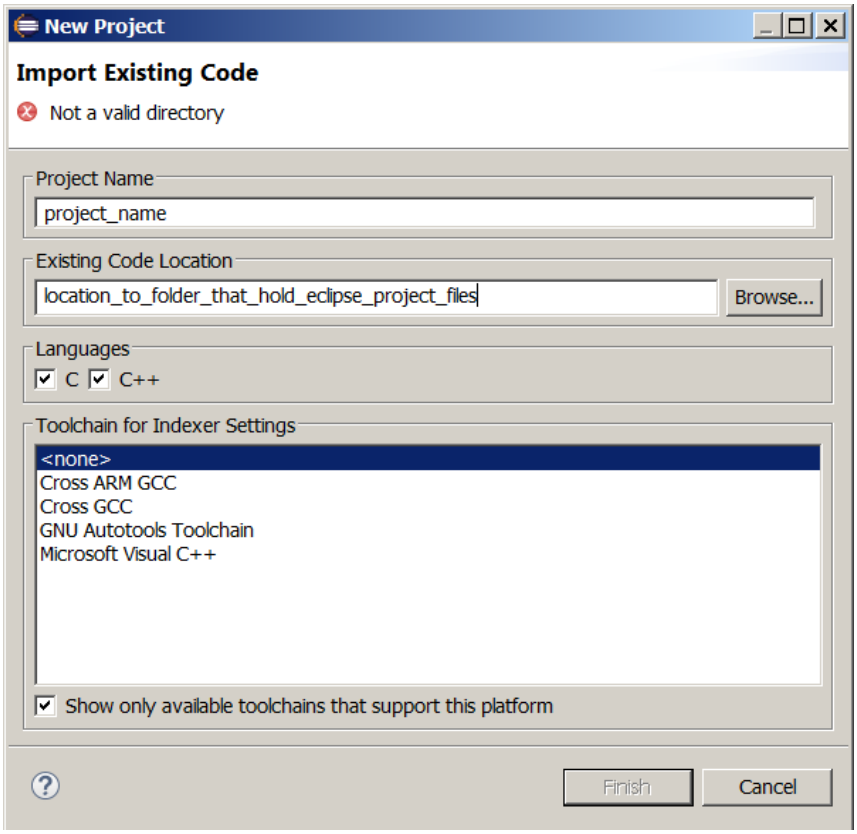
1. From command line or shell.

i. Compiling from command line is highly recommended during bring-up of workspace or if build process from IDE is not performed as expected.
ii. Go to $(YOUR_PATH)/uCWorkspace/uCProjects/apps/your_app
iii. Run 'make all'.
Note: For Windows you cannot run 'make all' because Windows limited built-in of make.exe will run, so you need to run GNU make.exe version:
$(YOUR_PATH)/uCWorkspace/uCProjects/windows/make/make4.1/bin/make.exe all

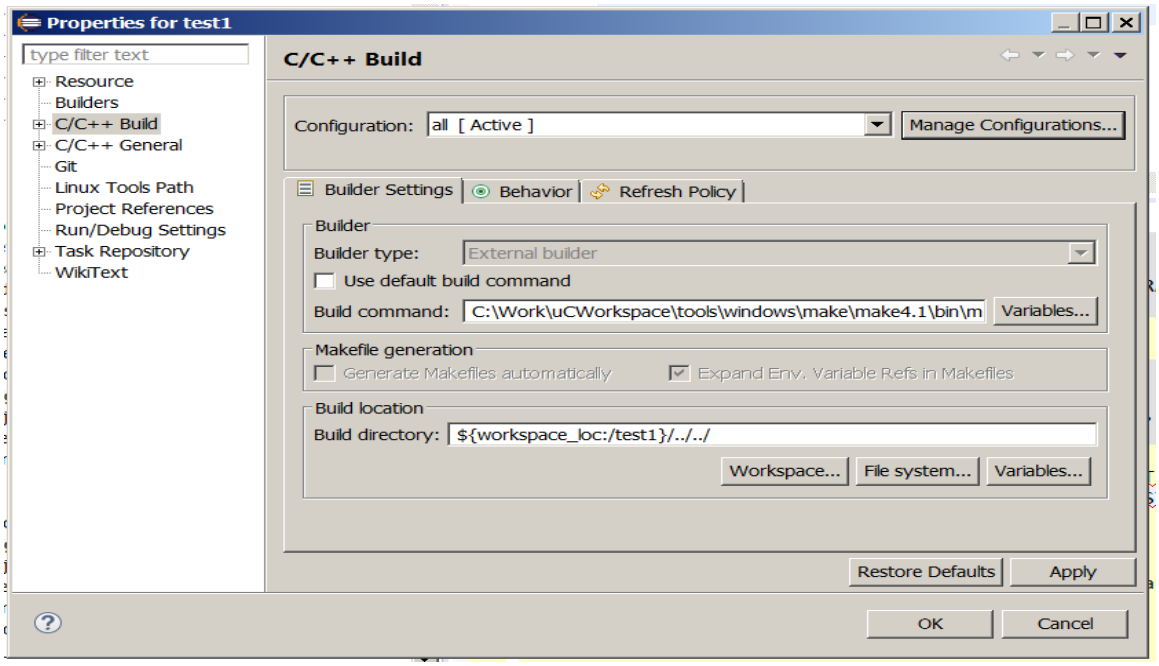
2. From eclipse.
   i. Adding project to Eclipse.
      a. Go to menu -> File -> ‘Makefile Project with Existing Code’




      b. Fill ‘Project Name’ and ‘Existing Code Location’.
      The location should point to folder that will contain eclipse project files, **NOT** application folder.
      The recommended location is in app_path/zIDE/ because zIDE is mentioned in .gitignore file.
      Sometimes the proper project already exists in zIDE folder, then you can try to use it.



ii. Right click on project and select ‘Properties’. In properties window go to ‘C/C++ Build’



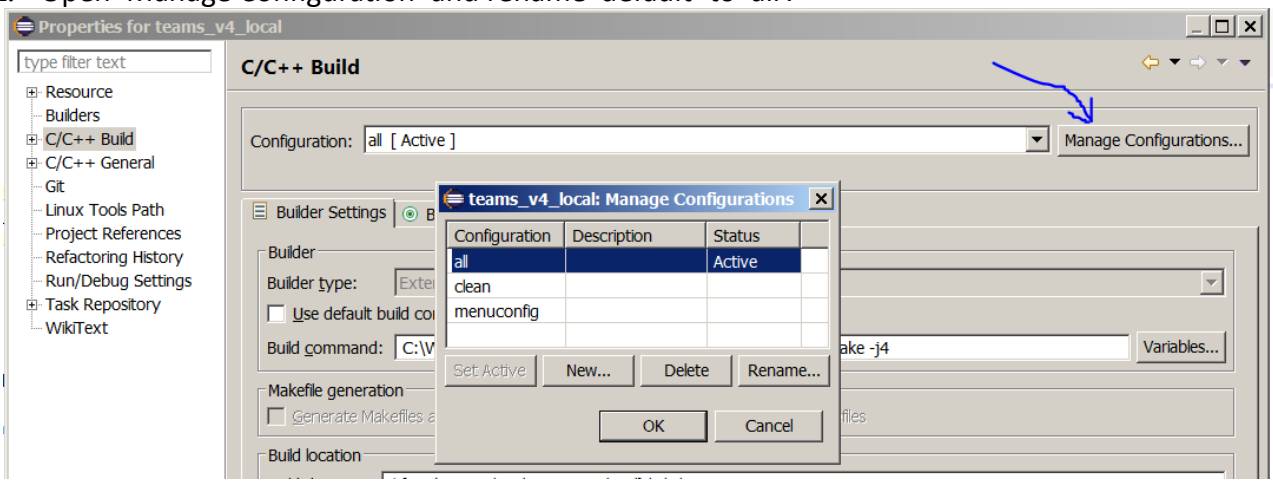

a. Unselect ‘Use default build command’. In ‘Build command:’ put ‘make’ in linux or ‘$(YOUR_PATH)/uCWorkspace/uCProjects/windows/make/make4.1/bin/make.exe’ in Windows. To accelerate build process you can add ‘-j4’ flag.

b. Build location should point to application root folder (the one that contains Makefile). It can be relative to eclipse project folder (for example ‘${workspace_loc:/test1}/../../’) or absolute path (for example C:\work\uCWorkspace\uCProjects\apps\test1).

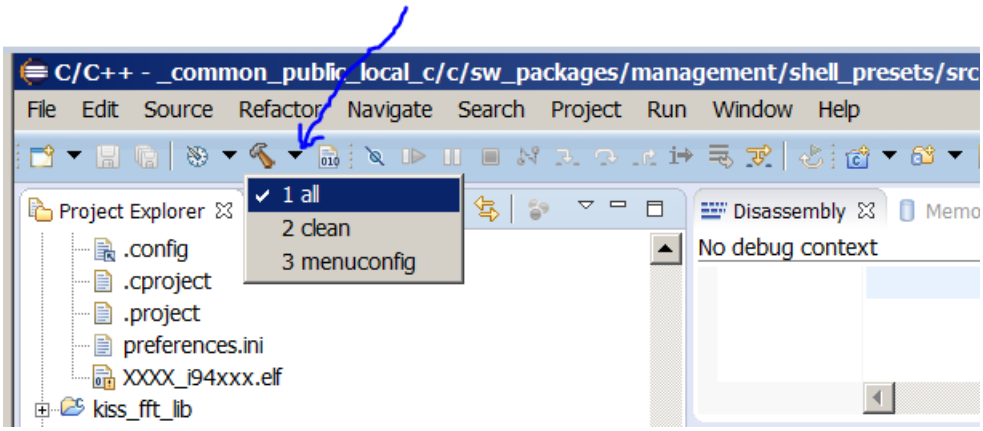
Note: if you got following error pattern when trying to build, then try to use absolute path:

*13:11:52 **** Build of configuration all for project test1 *****
*"D:\\make\\make4.1\\bin\\make" -j4 all*
*make: *** No rule to make target 'all'. Stop.*

iii. Open ‘Manage Configuration’ and rename ‘default’ to ‘all’.



Now you can compile the project by selecting ‘build all’ in eclipse menu:




F. Cleaning of project.

1. Go to $(YOUR_PATH)/uCWorkspace/uCProjects/apps/your_app
2. Run make clean.

   For Windows you cannot run ‘make clean’ because Windows limited built-in of make.exe will run, so you need to run GNU make.exe version:

   $(YOUR_PATH)/uCWorkspace/uCProjects/windows/make/make4.1/bin/make.exe clean