Rebuilding The NDK Core Using Gmake

Contents

Note: These instructions apply to NDK version 2.21 (and higher) only. For instructions on how to (re)build NDK 2.20x, please refer to http://processors.wiki.ti.com/index.php/Rebuilding_the_NDK_Core

Overview

The Network Developer's Kit (NDK) version 2.21.00.32 includes source files and a makefile that allow you to modify the stack sources and rebuild its libraries. You can rebuild the NDK core in order to modify, update, or add functionality. If you edit the NDK source code, an NDK project file, and/or an NDK RTSC build script, you must also rebuild the NDK core in order to create new libraries containing these modifications.

The instructions on this page are intended for rebuilding the NDK on Microsoft Windows or Linux using Gmake.

Warning: This appendix provides details about rebuilding the NDK stack source code. We strongly recommend that you copy the NDK installation to a directory with a different name and rebuild that copy, rather than rebuilding the original installation.

Prerequisites

In order to rebuild the NDK, the SYS/BIOS and XDCtools products must both be installed. It is important to build SYS/BIOS with a compatible version of XDCtools. To find out which versions are compatible, see the “Compatibility Information” section of the Release Notes in the top-level directory of your SYS/BIOS installation.

Note: You should generally avoid installing the various Texas Instruments tools and source distributions in directories that have spaces in their paths.

Rebuilding The NDK Core Using The ndk.mak Makefile

Rebuilding the NDK itself from the provided source files is straightforward using the provided makefile ndk.mak.

The NDK ships with a ndk.mak file in the top-level installation directory. This makefile enables you to easily (re)build the NDK core using your choice of compilers and desired "targets". A target incorporates a particular ISA and a runtime model; for example, cortex-M3 and the TI compiler with specific options.

The instructions in this section can be used to (re)build core libraries on Windows or Linux. If you are using a Windows machine, you can use the regular DOS command shell provided with Windows. However, you may want to install a Unix-like shell, such as Cygwin. For Windows users, the XDCtools top-level installation directory contains gmake.exe, which is used in the commands that follow to run the Makefile. The gmake utility is a Windows version of the standard GNU "make" utility provided with Linux.

If you are using Linux, change the "gmake" command to "make" in the commands that follow.

For these instructions, suppose you have the following directories:

- $BASE/ndk/ndk_2_21_00_32 — The location where you installed NDK.
- $BASE/ndk/copy-ndk_2_21_00_32 — The location of a copy of the NDK installation.
- $BASE/sysbios/bios_6_32_03_43 — The location where you installed SYS/BIOS.
- $BASE/xdctools_3_22_03_41 — The location where you installed XDCtools.

The following steps refer to the top-level directory of the XDCtools installation as <xdc_install_dir>. They refer to the top-level directory of the copy of the NDK installation as <ndkcopy_install_dir>.

Follow these steps to rebuild the NDK:

1. If you have not already done so, install the NDK, XDCtools and SYS/BIOS.
2. Make a copy of the NDK installation that you will use when rebuilding. This leaves you with an unmodified installation as a backup. For example, use commands similar to the following on Windows:

```
mkdir c:\ndk\copy-ndk_2_21_00_32
```

copy c:\ndk\ndk_2_21_00_32 c:\ndk\copy-ndk_2_21_00_32

Or, use the a command similar to the following on Linux:

```
cp -r $BASE/ndk/ndk_2_21_00_32/* $BASE/ndk/copy-ndk_2_21_00_32
```

1. Make sure you have access to compilers for any targets for which you want to be able be able to build applications using the rebuilt NDK. Note the path to the directory containing the executable for each compiler. These compilers can include Texas Instruments compilers as well as any other command-line compilers for any targets supported by NDK.
2. If you are using Windows and the gmake utility provided in the top-level directory of the XDCtools installation, you should add the <xdc_install_dir> to your PATH environment variable so that the gmake executable can be found.
3. You may remove the top-level doc directory located in <ndkcopy_install_dir>/docs if you need to save disk space.
4. At this point, you may want to add the remaining files in the NDK installation tree to your Software Configuration Management (SCM) system.

5. Open the <ndkcopy_install_dir>/ndk.mak file with a text editor, and make the following changes for any options you want to hard code in the file. (You can also set these options on the command line if you want to override the settings in the ndk.mak file.) Ignore the following lines near the beginning of the file. These definitions are used internally, but few users will have a need to change them.

   ```
   #
   ```

# Where to install/stage the packages # Typically this would point to the devkit location # DESTDIR ?= <UNDEFINED> prefix ?= / docdir ?= /docs/bios packagesdir ?= /packages

Specify the location of XDCtools. For example:

```
XDC_INSTALL_DIR ?= $(BASE)/xdctools_3_22_03_41
```

— Specify the location of the compiler executable for all targets you want to be able to build NDK for. Use only the directory path; do not include the name of the executable file. Any targets for which you do not specify a compiler location will be skipped during the build. For example, on Linux you might specify the following:

```
ti.targets.C674 ?= /opt/ti/ccsv5/tools/compiler/c6000

ti.targets.arm.elf.M3 ?= /opt/ti/ccsv5/tools/compiler/tms470
```

— Similarly, on Windows you might specify the following compiler locations:

```
ti.targets.C674 ?= c:/ti/ccsv5/tools/compiler/c6000

ti.targets.arm.elf.M3 ?= c:/ti/ccsv5/tools/compiler/tms470
```

— If you need to add any repositories to your XDCPATH (for example, to reference the packages directory of another component), you should edit the XDCPATH definition.

— You can uncomment the line that sets XDCOPTIONS to “v” if you want more information output during the build.

1. Clean the NDK installation with the following commands. (If you are running the build on Linux, change all "gmake" commands to "make".)

   ```
   cd <ndkcopy_install_dir>
   ```

gmake -f ndk.mak clean

1. Run the ndk.mak file to build SYS/BIOS as follows. (Remember, if you are running the build on Linux, change all "gmake" commands to "make".)
   gmake -f ndk.mak
2. If you want to build NDK in debug mode, you have to uncomment a couple of lines in ndk.bld

   /* Uncomment the following lines to build libraries for debug mode: */
   // Pkg.attrs.profile = "debug";
   // c6xOpts += " -g -o0 ";
   // armOpts += " -g -o0 ";
   // gnuOpts += " -g ";
3. If you want to specify options on the command line to override the settings in ndk.mak, use a command similar to the following.

   ```
   gmake -f ndk.mak XDC_INSTALL_DIR=<xdc_install_dir> ti.targets.C674=<compiler_path>
   ```

{{ 1. switchcategory:MultiCore= ▪ For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum ▪ For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum Please post only comments related to the article **Rebuilding The NDK Core Using Gmake** here.	Keystone= ▪ For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum ▪ For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum Please post only comments related to the article **Rebuilding The NDK Core Using Gmake** here.	C2000=*For technical support on the C2000 please post your questions on The C2000 Forum. Please post only comments about the article* ***Rebuilding The NDK Core Using Gmake*** *here.*	DaVinci=*For technical support on DaVincoplease post your questions on The DaVinci Forum. Please post only comments about the article* ***Rebuilding The NDK Core Using Gmake*** *here.*	MSP430=*For technical support on MSP430 please post your questions on The MSP430 Forum. Please post only comments about the article* ***Rebuilding The NDK Core Using Gmake*** *here.*	OMAP35x=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article* ***Rebuilding The NDK Core Using Gmake*** *here.*	OMAPL1=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article* ***Rebuilding The NDK Core Using Gmake*** *here.*	MAVRK=*For technical support on MAVRK please post your questions on The MAVRK Toolbox Forum. Please post only comments about the article* ***Rebuilding The NDK Core Using Gmake*** *here.*	*For technical s please post yo questions at http://e2e.ti.cor Please post on comments abo article* ***Rebuild The NDK Core Gmake*** *here.* }}

Links

Amplifiers & Linear
Audio
Broadband RF/IF & Digital Radio
Clocks & Timers
Data Converters

DLP & MEMS
High-Reliability
Interface
Logic
Power Management

Processors

- ARM Processors
- Digital Signal Processors (DSP)
- Microcontrollers (MCU)
- OMAP Applications Processors

Switches & Multiplexers
Temperature Sensors & Control ICs
Wireless Connectivity