

NuttX RTOS



ARM

Development

Gregory Nutt



Overview

- Getting the Source Code
- Configuring NuttX
- The NuttX Simulator
- The NuttShell
- ARM Build Tools
- ARM Cross Development
- Build Modes
- JTAG
- Debugging the ARM Target



Getting the Source Code Frozen Versions

The top screenshot shows the Bitbucket interface for the repository `nuttx/apps`. The URL `https://bitbucket.org/nuttx/apps/downloads` is highlighted in a yellow box.

The bottom screenshot shows the Bitbucket interface for the repository `patacongo/nuttx`. The URL `https://bitbucket.org/patacongo/nuttx/downloads` is highlighted in a yellow box. Below the URL, there is a "Downloads" section with a "Choose files to upload" button and a table of files.

Name	Size	Uploaded by	Downloads	Date	
Download repository	59.3 MB				
nuttx-7_11-ReleaseNotes.txt	16.0 KB	patacongo	59	2015-08-16	Delete
nuttx-7.11.tar.gz	13.4 MB	patacongo	71	2015-08-13	Delete
nuttx-7.10.tar.gz	12.8 MB	patacongo	169	2015-06-28	Delete
nuttx_7_10-README.mkd	19.3 KB	patacongo	98	2015-06-28	Delete
nuttx-7.10-patches.zip	5.6 KB	patacongo	72	2015-06-28	Delete

- NuttX Package (Core OS) +
- (optional) Application (`apps/`) package



Getting the Source Code

Frozen Versions (Cont'd)

[Home](#) / [Browse](#) / [System Administration](#) / [Operating System Kernels](#) / [NuttX](#) / [Files](#)



NuttX

Real-Time Embedded Operating System

Brought to you by: [patacongo](#)

<http://sourceforge.net/projects/nuttX/files/nuttX>

[Summary](#) | [Files](#) | [Reviews](#) | [Support](#) | [Mailing Lists](#)

Looking for the latest version? [Download nuttx-7.11.tar.gz \(14.0 MB\)](#)

[Home](#) / [nuttx](#)

Name ↕	Modified ↕	Size ↕	Downloads / Week ↕
↑ Parent folder			
nuttx-7.11	2015-08-13		143
nuttx-7.10	2015-06-25		6
nuttx-7.9	2015-04-19		2
nuttx-7.7	2015-02-11		3
nuttx-7.8	2015-02-11		2

Totals: 5 Items

- NuttX Package (Core OS)
+
- (optional) Application
(apps /) package



Getting the Source Code

Other Frozen Packages

<http://sourceforge.net/projects/nuttX/files>

[Home](#) / [Browse](#) / [System Administration](#) / [Operating System Kernels](#) / [NuttX](#) / [Files](#)



NuttX











Real-Time Embedded Operating System

Brought to you by: [patacongo](#)

[Summary](#) | [Files](#) | [Reviews](#) | [Support](#) | [Mailing Lists](#)

Looking for the latest version? [Download nuttx-7.11.tar.gz \(14.0 MB\)](#)

Home

Name ↕	Modified ↕	Size ↕	Downloads / Week ↕
 nuttx	2015-08-13		156 
 NxWidgets	2015-04-14		11 
 buildroot	2014-03-15		8 
 uClibc++	2012-11-05		15 
 pascal	2011-05-15		1 

Totals: 5 Items

- NxWidgets:
 - Graphics Package
- Buildroot:
 - NuttX customized toolchain
- Uclibc++:
 - Standard C++ Library
- Pascal:
 - Pascal p-code compiler



Getting the Source Code

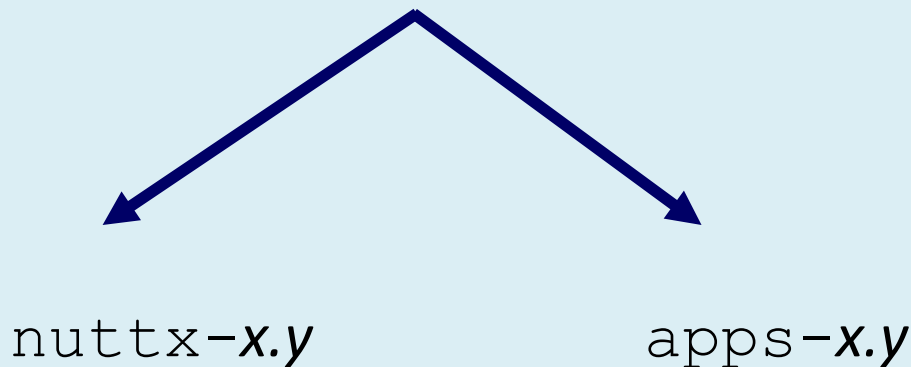
Creating a Build Directory

- Download `nutt-x.y.tar.gz` and `apps-x.y.tar.gz`

- Unpack

```
tar zxf nuttx-x.y.tar.gz
```

```
tar zxf apps-x.y.tar.gz
```



Where `x.y` is a frozen version number like 7.11



Getting the Source Code GIT Repositories

The screenshot displays the Bitbucket web interface. At the top, a navigation bar includes 'Dashboard', 'Teams', 'Repositories', 'Snippets', and a 'Create' button. The main content area shows the 'Overview' page for a repository. A yellow highlight box contains the URL <https://bitbucket.org/nuttx/apps>. Below this, another yellow highlight box shows the URL <https://bitbucket.org/patacongo/nuttx>. The repository overview includes a table with the following data:

Last updated	2015-09-20	1	99+
Website	http://www.nuttx.org/	Branch	Tags
Language	C	17	13
Access level	Admin	Forks	Watchers

Below the table, there is an 'Edit README' button and a section for the README file content, which includes the text 'README', 'README', and '^^^^^^'. A light blue box at the bottom left contains the following URLs:


- <https://bitbucket.org/patacongo/nuttx.git>
- <https://bitbucket.org/nuttx/apps.git>

At the bottom of the page, there is a status bar that says 'Downloading from Repositories' and 'in the Path'.



Getting the Source Code Other Repositories

Atlassian
Bitbucket Features Pricing Find a repo



NuttX (nuttx)
Team since June 2015

<https://bitbucket.org/nuttx>

Overview Snippets Followers 15 Members 5

Language ▾ Find repositories

- arch
- boards
- apps
- Documentation

NuttX Submodules


- arch
- boards
- documentation



Getting the Source Code







Other Repositories (Cont'd)

Atlassian
Bitbucket Features Pricing



NuttX (nuttx)
Team since June 2015

<https://bitbucket.org/nuttx>

-  tools
-  drivers
-  NxWidgets
-  uClibc++
-  buildroot
-  Pascal

- NxWidgets:
 - Graphics Package
- Buildroot:
 - NuttX customized toolchain
- Uclibc++:
 - Standard C++ Library
- Pascal:
 - Pascal p-code compiler

- tools:
 - Tools for use with NuttX
- drivers:
 - GPL drivers

Updated 2015-09-06

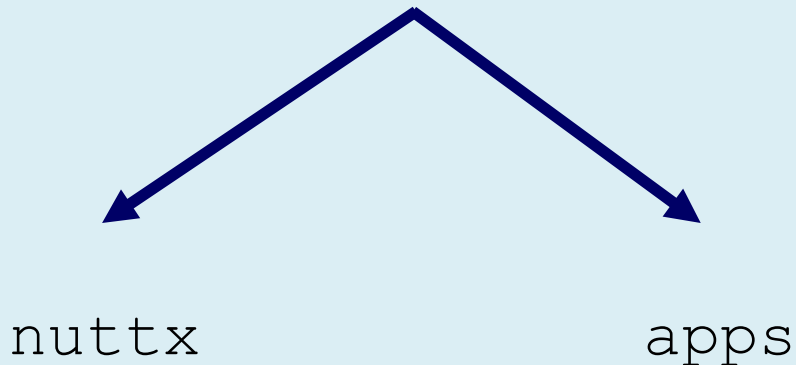


Getting the Source Code

Cloning a Build Directory

Clone Repositories

```
git clone https://bitbucket.org/patacongo/nutttx.git nuttx
git clone https://bitbucket.org/nutttx/apps.git apps
```



Initialize GIT Sub-modules

```
cd nuttx
git submodule init
git submodule update
```

More Info:

<http://www.nutttx.org/doku.php?id=downloads>



Configuring NuttX

Building kconfig-frontends

```
./configure --enable-mconf --disable-nconf --disable-gconf  
--disable-qconf
```

```
LD_RUN_PATH=/usr/local/lib
```

```
make
```

```
make install
```

May need root privileges to install

Graphical configuration Tools

Qt:

```
./configure ... --enable-qconf
```

GTK:

```
./configure ... --enable-gconf ...
```



Configuring NuttX

Board Support Logic in sub-directories of configs/

Form: configs/<board-name>

Example: configs/stm32f4discovery

```
cd tools/
```

```
./configure.sh <code-name>/<configuration>
```

Example: ./configure.sh stm32f4discovery/nsh

Equivalent Manual Configuration:

```
cp stm32f4discovery/nsh/defconfig .config
```

```
cp stm32f4discovery/nsh/Make.defs Make.defs
```

```
cp stm32f4discovery/nsh/setenv.sh setenv.sh
```



Additional Pre-Build Steps

Modify configuration for build environment:

```
make menuconfig
```

- Build Host: Linux, Windows, OSX, etc.
- Windows Build Framework: Cygwin, MSYS, Native
- Toolchain

Make sure configuration is up to date:

```
make oldconfig
```

Establish PATH to build tool binaries:

```
. ./setenv.sh
```

Optional
Needs to be customized



Building Executables

Linux, FreeBSD, OSX,
or Windows+Cygwin

Command Line Shell:

```
$ make
```

GNU Make

Compiler: `gcc`

Archiver: `ar`
(aka "Librarian")

Linker: `ld`

Hard Drive

Makefiles

Source files:

`*.c *.cxx *.S`

Object Files: `*.o`

Archives: `*.a`
(aka "static libraries")

Executables:

`nuttX` and `nuttX.exe`



Building the NuttX Simulator

Building the NuttX Simulator

```
$ cd nuttx/tools
$ ./configure.sh sim/nsh
$ cd ..
$ make menuconfig
$ . ./setenv.sh
$ make
```

Refresh Configuration
Configuration build environment

Optional
Need path to build tool binaries

Example

Linux->Windows

Cygwin

32/64 bit

Microsoft ABI

Running the NuttX Simulator

```
$ ./nuttx.exe
```



The NuttShell (NSH)

A “thin” program to interface with the OS
Many commands similar to the bash shell

```
nsh> help
```



Debugging the Simulator

1 Reconfigure

- Enable debug symbols in the configuration
- Disable optimization (optional)

2 Rebuild

3 `gdb nuttx.exe`

Using `ddd` graphical front end:

- `export DISPLAY=:0`
- `ddd nuttx.exe &`



ARM Build Tools

Linux / OSX

- *GCC Toolchain:* arm-none-eabi-gcc
 - <https://launchpad.net/gcc-arm-embedded>
- *Configuration Tool:* kconfig-frontends
Same configuration tool used with Linux kernel
 - <http://ymorin.is-a-geek.org/projects/kconfig-frontends>
 - <https://bitbucket.org/nuttx/tools>
 - <https://bitbucket.org/nuttx/buildroot>
- *Other Tools:* genromfs
 - <http://romfs.sourceforge.net/>
 - <https://bitbucket.org/nuttx/tools>



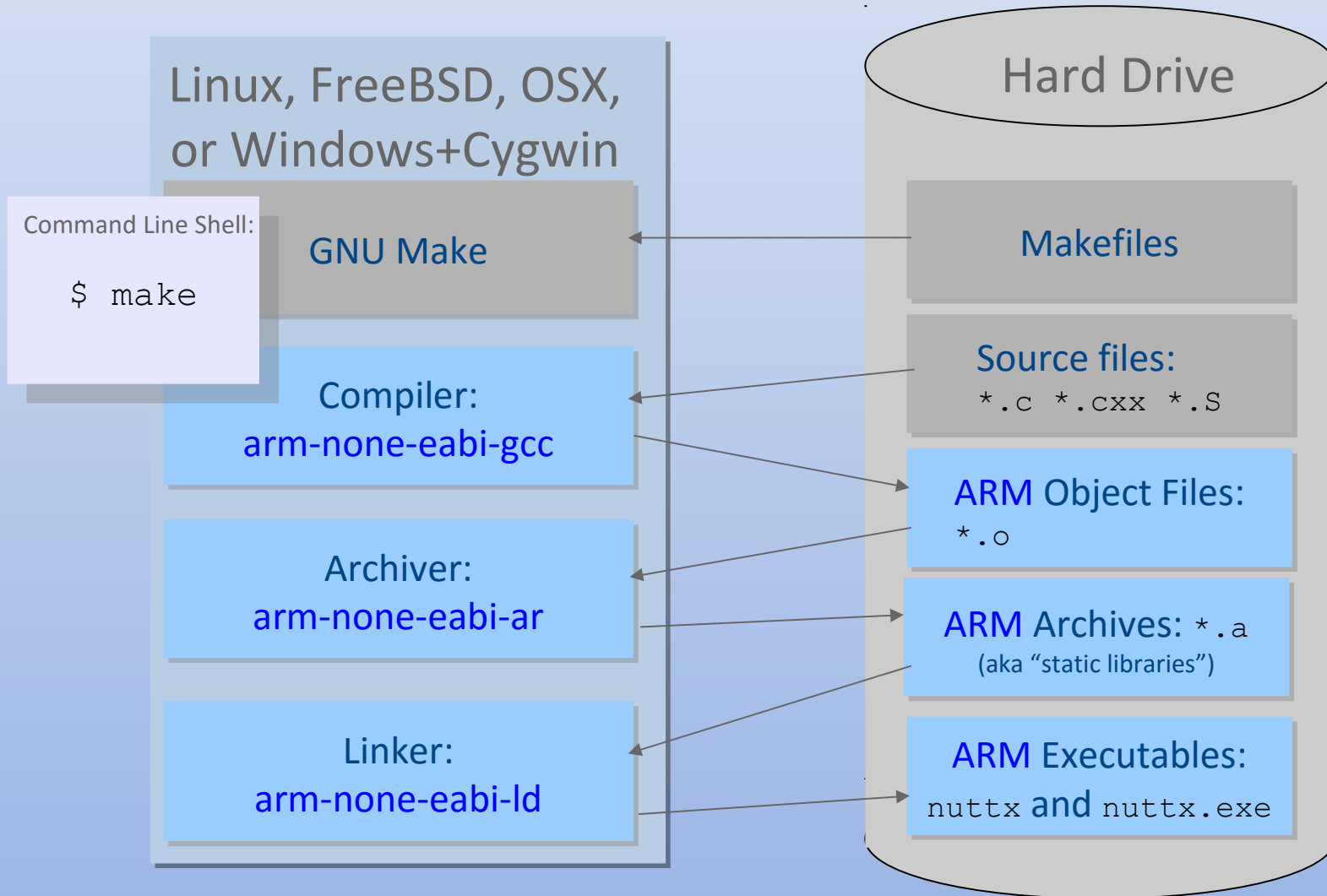
ARM Build Tools

Windows

- Cygwin
 - <http://cygwin.com/>
 - Creates POSIX development environment on Windows
 - Use same Linux tools (built for Cygwin)
 - Can also integrate Native Windows Tools
- MSYS is another option
- Native Windows Build is also possible



ARM Cross Development



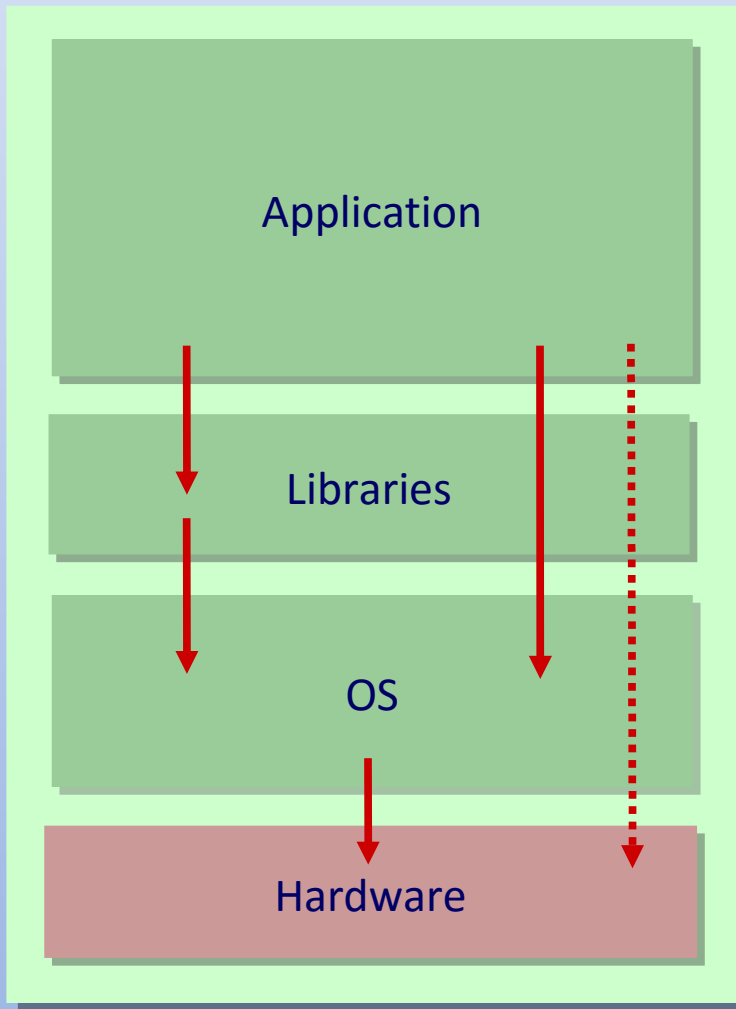
make export

Will extract all start-up files, libraries, and header files into a package that can be then be used in another build environment.
Example, with an IDE

```
make export
```



Linked Binary (Flat Build Mode)

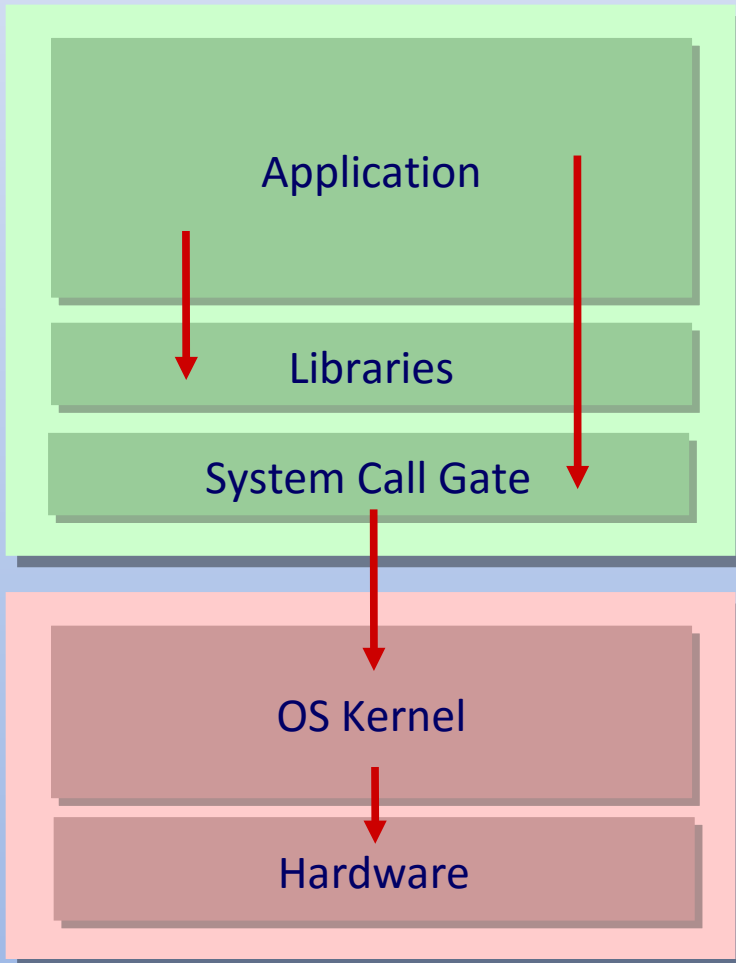


- *Flat* address space: All memory accessible by the application.
- Nothing is protected.
- Application can directly access OS/hardware resources
- No special hardware required
- `CONFIG_BUILD_FLAT=y`



Linked Binaries

(Protected Build Mode)

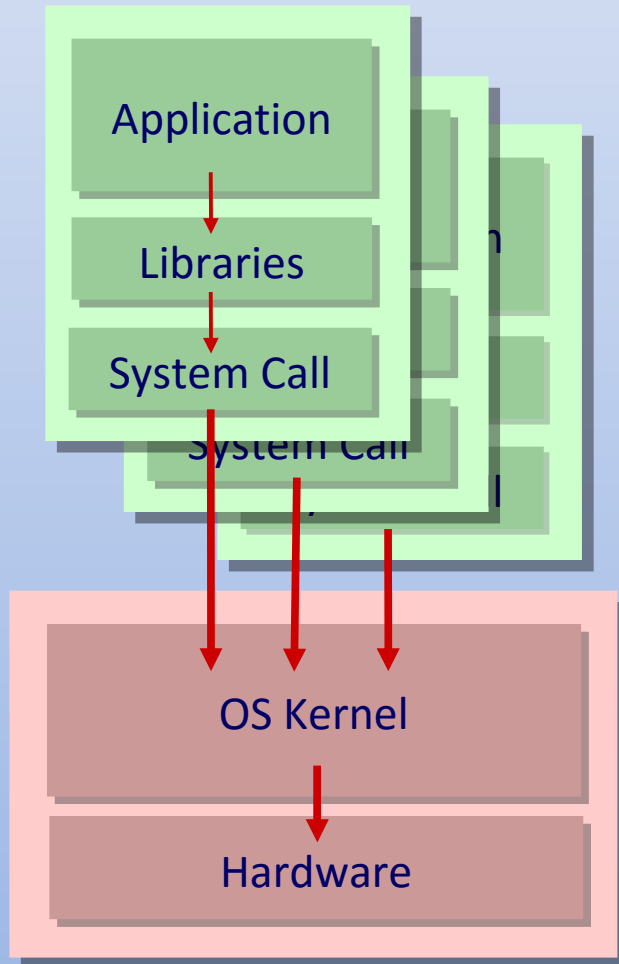


- OS resources protected from application.
- Application accesses OS resources indirectly through a *call gate*
- Require Memory Protection Unit (MPU)
- Example, Cortex-M
- `CONFIG_BUILD_PROTECTED=y`



Linked Binaries

(Kernel Build Mode)



- *Processes*: Each application protected in a separate address space.
- Require Memory Management Unit (MMU)
- Example, Cortex-A
- `CONFIG_BUILD_KERNEL=y`



JTAG/SWD

IC Debug Ports

Debuggers communicate through JTAG or SWD

Load code into FLASH

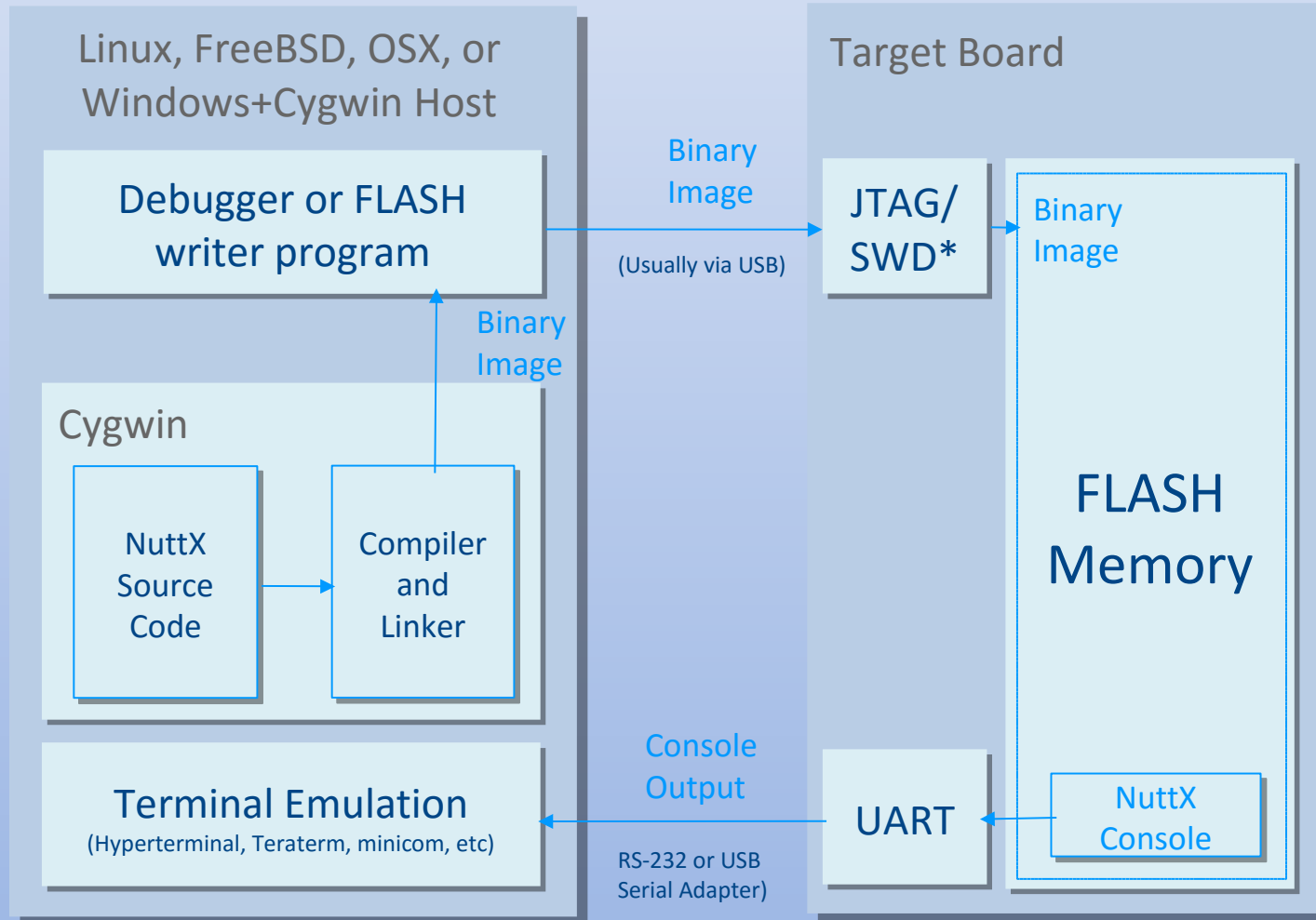
Debug code (breakpoints, single stepping, etc.)

Joint Test Action Group (JTAG) – IEEE 1149.1

Serial Wire Debug (SWD) – ARM



Hardware Connections



*Not usually on the board.



Debugging the ARM Target

1 Reconfigure

- Enable debug symbols in the configuration
- Disable optimization (optional)

2 Rebuild

3 Start the GDB server (OpenOCD, Segger J-Link, others)

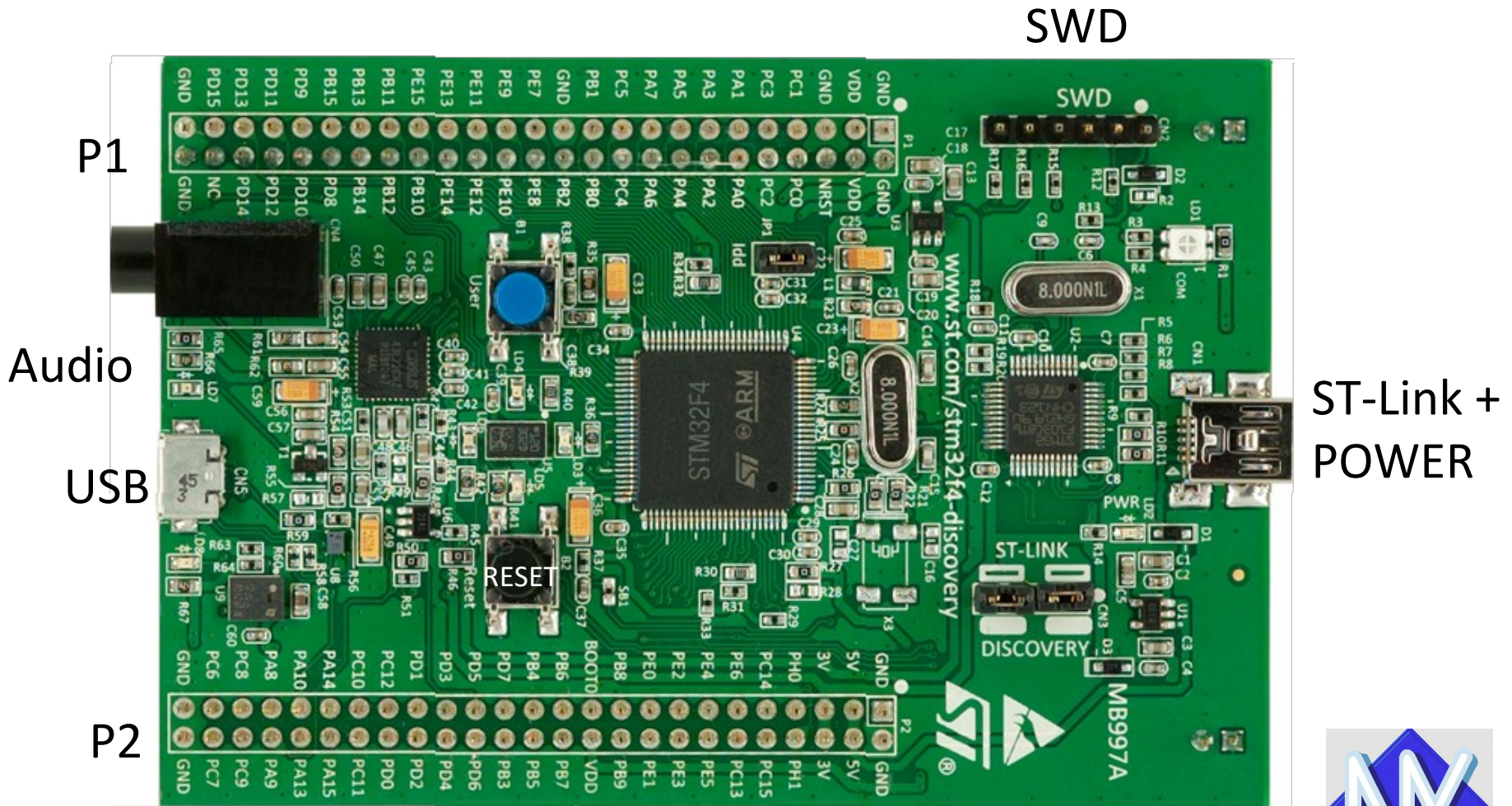
4 `arm-none-eabi-gdb nuttx.exe`

Using `ddd` graphical front end:

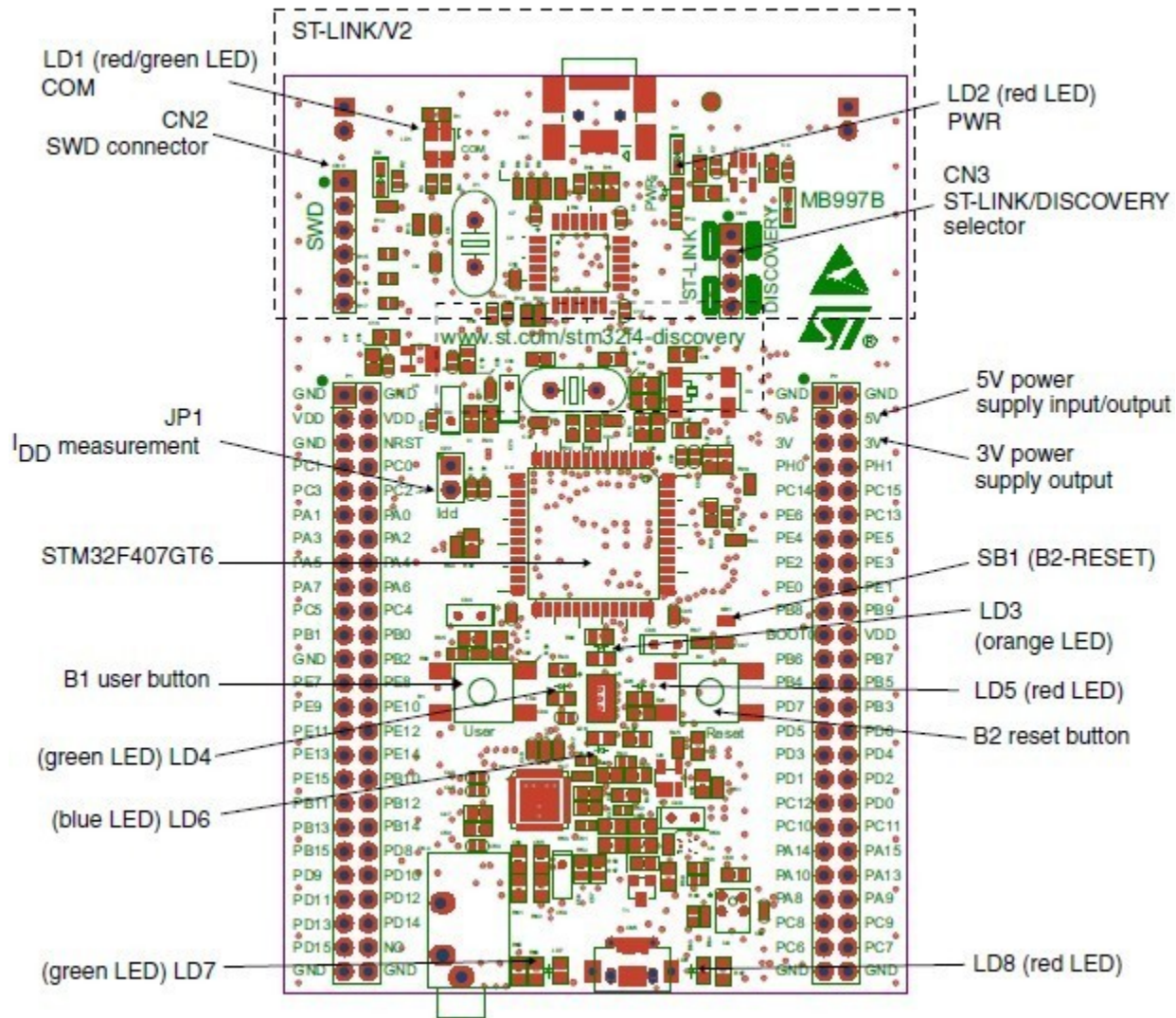
- `export DISPLAY=:0`
- `ddd --debugger arm-none-eabi-gdb nuttx &`



The STM32F4Discovery



The STMicro STM32F4-Discovery



Atmel SAMV71 Xplained Ultra

