About

Table of Contents

NuttX RTOS

Last Updated: April 16, 2020

Overview

Goals. NuttX is a real time embedded operating system (RTOS). Its goals are:

Small Footprint

Usable in all but the tightest micro-controller environments, The focus is on the tiny-to-small, deeply embedded environment.

Rich Feature OS Set

The goal is to provide implementations of most standard POSIX OS interfaces to support a rich, multi-threaded development environment for deeply embedded processors.
NON-GOALS: It is not a goal to provide the level of OS features like those provided by Linux. In order to work with smaller MCUs, small footprint must be more important than an extensive feature set. But standard compliance is more important than small footprint. Surely a smaller RTOS could be produced by ignoring standards. Think of NuttX is a tiny Linux work-alike with a much reduced feature set.

Highly Scalable

Fully scalable from tiny (8-bit) to moderate embedded (32-bit). Scalability with rich feature set is accomplished with: Many tiny source files, link from static libraries, highly configurable, use of weak symbols when available.

Standards Compliance

NuttX strives to achieve a high degree of standards compliance. The primary governing standards are POSIX and ANSI standards. Additional standard APIs from Unix and other common RTOS's are adopted for functionality not available under these standards or for functionality that is not appropriate for the deeply-embedded RTOS (such as `fork()`).

Because of this standards conformance, software developed under other standard OSs (such as Linux) should port easily to NuttX.

Real-Time

Fully pre-emptible; fixed priority, round-robin, and "sporadic" scheduling.

Totally Open

Non-restrictive BSD license.

GNU Toolchains

Compatible GNU toolchains based on buildroot available for download to provide a complete development environment for many architectures.

Feature Set. Key features of NuttX include:

Standards Compliant Core Task Management

- Fully pre-emptible.
- Naturally scalable.
- Highly configurable.
- Easily extensible to new processor architectures, SoC architecture, or board architectures. A Porting Guide is available.
- FIFO and round-robin scheduling.
- Realtime, deterministic, with support for priority inheritance
- Tickless Operation

- POSIX/ANSI-like task controls, named message queues, counting semaphores, clocks/timers, signals, pthreads, robust mutexes, cancellation points, environment variables, filesystem.
- Standard default signal actions (optional).
- VxWorks-like task management and watchdog timers.
- BSD socket interface.
- Extensions to manage pre-emption.
- Optional tasks with address environments (*Processes*).
- Loadable kernel modules; lightweight, embedded shared libraries.
- Memory Configurations: (1) Flat embedded build, (2) Protected build with MPU, and (3) Kernel build with MMU.
- Memory Allocators: (1) standard heap memory allocation, (2) granule allocator, (3) shared memory, and (4) dynamically sized, per-process heaps.
- Inheritable "controlling terminals" and I/O re-direction. Pseudo-terminals
- On-demand paging.
- System logging.

May be built either as an open, flat embedded RTOS or as a separately built, secure, monolithic kernel with a system call interface.

Built-in, per-thread CPU load measurements.

Well documented in the NuttX User Guide.

File system

Tiny, in-memory, root pseudo-file-system.

Virtual file system (VFS) supports drivers and mountpoints.

Mount-able volumes. Bind mountpoint, filesystem, and block device driver.

Generic system logging (SYSLOG) support.

FAT12/16/32 filesystem support with optional FAT long file name support[1].

NFS Client. Client side support for a Network File System (NFS, version 3, UDP).

NXFFS. The tiny NuttX wear-leveling FLASH file system.

SMART. FLASH file system from Ken Pettit.

SPIFFS. FLASH file system, originally by Peter Anderson.

LittleFS. FLASH file system from ARM mbed..

ROMFS filesystem support (XIP capable).

CROMFS (Compressed ROMFS) filesystem support.

TMPFS RAM filesystem support.

BINFS pseudo-filesystem support.

HOSTFS filesystem support (simulation only).

Union filesystem - Supports combining and overlaying file systems.

UserFS - User application file system.

`procfs/` pseudo-filesystem support.

A binary loader with support for the following formats:

- Separately linked ELF modules.
- Separately linked NXFLAT modules. NXFLAT is a binary format that can be XIP from a file system.
- "Built-In" applications.

PATH variable support.

File transfers via TFTP and FTP (`get` and `put`), HTML (`wget`), and Zmodem (`sz` and `rz`).
Intel HEX file conversions.

[1] FAT long file name support may be subject to certain Microsoft patent restrictions if enabled. See the top-level `COPYING` file for details.

 **Device Drivers**

Supports character and block drivers as well as specialized driver interfaces.

Full VFS integration.

Asynchronous I/O (AIO)

Network, USB (host), USB (device), serial, I2C, I2S, NAND, CAN, ADC, DAC, PWM, Quadrature Encoder, I/O Expander, Wireless, generic timer, and watchdog timer driver architectures.

RAMDISK, pipes, FIFO, `/dev/null`, `/dev/zero`, `/dev/random`, and loop drivers.

Generic driver for SPI-based or SDIO-based MMC/SD/SDH cards.

Graphics: framebuffer drivers, graphic- and segment-LCD drivers. VNC server.

Audio subsystem: CODECs, audio input and output drivers. Command line and graphic media player applications.

Cryptographic subsystem.

Power Management sub-system.

ModBus support provided by built-in FreeModBus version 1.5.0.

C/C++ Libraries

Standard C Library Fully integrated into the OS.

Includes floating point support via a Standard Math Library.

Add-on uClibc++ module provides Standard C++ Library (LGPL).

Networking

Multiple network interface support; multiple network link layer support.

IPv4, IPv6, TCP/IP, UDP, ICMP, ICMPv6, IGMPv2 and MLDv1/v2 (client) stacks.

IP Forwarding (routing) support.

User space stacks.

Stream and datagram sockets.

Address Families: IPv4/IPv6 (`AF_INET`/`AF_INET6`), Raw socket (`AF_PACKET`), raw IEEE 802.15.4 (`AF_IEEE802154`), raw Bluetooth (`AF_BLUETOOTH`), and local, Unix domain socket support (`AF_LOCAL`).

Special `INET` protocol sockets: Raw ICMP and ICMPv6 protocol ping sockets (`IPPROTO_ICMP`/`IPPROTO_ICMP6`).

Custom user sockets.

IP Forwarding.

DNS name resolution / NetDB

IEEE 802.11 FullMac

Radio Network Drivers: IEEE 802.15.4 MAC, Generic Packet Radio, Bluetooth LE

6LoWPAN for radio network drivers (IEEE 802.15.4 MAC and generic packet radios)

SLIP, TUN/PPP, Local loopback devices

A port cJSON

Small footprint.

BSD compatible socket layer.

Networking utilities (DHCP server and client, SMTP client, Telnet server and client, FTP server and client, TFTP client, HTTP server and client, PPPD, NTP client). Inheritable TELNET server sessions (as "controlling terminal"). VNC server.

ICMPv6 autonomous auto-configuration

NFS Client. Client side support for a Network File System (NFS, version 3, UDP).

A NuttX port of Jeff Poskanzer's THTTPD HTTP server integrated with the NuttX binary loader to provide true, embedded CGI.

PHY Link Status Management.

UDP Network Discovery (Contributed by Richard Cochran).

XML RPC Server (Contributed by Richard Cochran).

Support for networking modules (e.g., ESP8266).

 **FLASH Support**

MTD-inspired interface for *M*emory *T*echnology *D*evices.

NAND support.

FTL. Simple *F*lash *T*ranslation *L*ayer support file systems on FLASH.

Wear-Leveling FLASH File Systems: NXFFS, SmartFS, SPIFFS.

Support for SPI-based FLASH and FRAM devices.

 **USB Host Support**

USB host architecture for USB host controller drivers and device-dependent USB class drivers.

USB host controller drivers available for the Atmel SAMA5Dx, NXP LPC17xx, LPC31xx, and STmicro STM32

Device-dependent USB class drivers available for USB mass storage, CDC/ACM serial, HID keyboard, and HID mouse.

Seam-less support for USB hubs.

 **USB Device Support**

Gadget-like architecture for USB device controller drivers and device-dependent USB class drivers.

USB device controller drivers available for the most MCU architectures includeing PIC32, Atmel AVR, SAM3, SAM4, SAMv7, and SAMA5Dx, NXP /Freescale LPC17xx, LPC214x, LPC313x, LPC43xx, and Kinetis, Silicon Laboraties EFM32, STMicro STM32 F1, F2, F3, F4, and F7, TI DM320, and others.

Device-dependent USB class drivers available for USB serial (CDC/ACM and a PL2303 emulation), for USB mass storage, for USB networking (RNDIS and CDC/ECM), DFU, and for a dynamically configurable, composite USB devices.

Built-in USB device and USB host trace functionality for non-invasive USB debug.

 **Graphics Support**

Framebuffer drivers.

Graphic LCD drivers for both parallel and SPI LCDs and OLEDs.

Segment LCD drivers.

VNC Server.

`mmap`-able, framebuffer character driver.

NX: A graphics library, tiny windowing system and tiny font support that works with either framebuffer or LCD drivers. Documented in the NX Graphics Subsystem manual.

Font management sub-system.

NxWidgets: NXWidgets is library of graphic objects, or "widgets," (labels, buttons, text boxes, images, sliders, progress bars, etc.). NXWidgets is written in C++ and integrates seamlessly with the NuttX NX graphics and font management subsystems.

NxWM: NxWM is the tiny NuttX window manager based on NX and NxWidgets.

 **Input Devices**

Touchscreen, USB keyboard, GPIO-based buttons and keypads.

Analog Devices

Support for Analog-to-Digital conversion (ADC), Digital-to-Analog conversion (DAC), multiplexers, and amplifiers.

Motor Control

Pulse width modulation (PWM) / Pulse count modulation.

NuttX Add-Ons. The following packages are available to extend the basic NuttX feature set:

NuttShell (NSH)

- A small, scalable, bash-like shell for NuttX with rich feature set and small footprint. See the NuttShell User Guide.

BAS 2.4

- Seamless integration of Michael Haardt's BAS 2.4: "Bas is an interpreter for the classic dialect of the programming language BASIC. It is pretty compatible to typical BASIC interpreters of the 1980s, unlike some other UNIX BASIC interpreters, that implement a different syntax, breaking compatibility to existing programs. Bas offers many ANSI BASIC statements for structured programming, such as procedures, local variables and various loop types. Further there are matrix operations, automatic LIST indentation and many statements and functions found in specific classic dialects. Line numbers are not required."

Look at all those files and features... How can it be a tiny OS?. The NuttX feature list (above) is fairly long and if you look at the NuttX source tree, you will see that there are hundreds of source files comprising NuttX. How can NuttX be a tiny OS with all of that?

Lots of Features -- More can be smaller!

The philosophy behind that NuttX is that lots of features are great... *BUT* also that if you don't use those features, then you should not have to pay a penalty for the unused features. And, with NuttX, you don't! If you don't use a feature, it will not be included in the final executable binary. You only have to pay the penalty of increased footprint for the features that you actually use.

Using a variety of technologies, NuttX can scale from the very tiny to the moderate-size system. I have executed NuttX with some simple applications in as little as 32K *total* memory (code and data). On the other hand, typical, richly featured NuttX builds require more like 64K (and if all of the features are used, this can push 100K).

Many, many files -- More really is smaller!

One may be intimidated by the size NuttX source tree. There are hundreds of source files! How can that be a tiny OS? Actually, the large number of files is one of the tricks to keep NuttX small and as scalable as possible. Most files contain only a single function. Sometimes just one tiny function with only a few lines of code. Why?

- **Static Libraries**. Because in the NuttX build processed, objects are compiled and saved into *static libraries* (*archives*). Then, when the file executable is linked, only the object files that are needed are extracted from the archive and added to the final executable. By having many, many tiny source files, you can assure that no code that you do not execute is ever included in the link. And by having many, tiny source files you have better granularity -- if you don't use that tiny function of even just a few lines of code, it will not be included in the binary.

Other Tricks

As mentioned above, the use of many, tiny source files and linking from static libraries keeps the size of NuttX down. Other tricks used in NuttX include:

- **Configuration Files**. Before you build NuttX, you must provide a configuration file that specifies what features you plan to use and which features you do not. This configuration file contains a long list of settings that control what is built into NuttX and what is not. There are hundreds of such settings (see the Configuration Variable Documentation for a partial list that excludes platform specific settings). These many, many configuration options allow NuttX to be highly tuned to meet size requirements. The downside to all of these configuration options is that it greatly complicates the maintenance of NuttX -- but that is my problem, not yours.
- **Weak Symbols** The GNU toolchain supports *weak* symbols and these also help to keep the size of NuttX down. Weak symbols prevent object files from being drawn into the link even if they are accessed from source code. Careful use of weak symbols is another trick for keep unused code out of the final binary.

NuttX Discussion Group

Most NuttX-related discussion occurs on the *Google* NuttX group. You are cordially invited to join. In most cases, I make a special effort to answer any questions and provide any help that I can.

Downloads

Git Repository

The working version of NuttX is available from the Bitbucket GIT repository here. That same page provides the URLs and instructions for *cloning* the GIT repository.

Released Versions

In addition to the ever-changing GIT repository, there are frozen released versions of NuttX available. The current release is NuttX 9.0. NuttX 9.0 is the 134^{r} release of NuttX. It was released on April 30, 2020, and is available for download from the nuttx.apache.org website. Note that the release consists of two tarballs: `apache-nuttx-9.0.x-incubating.tar.gz` and `apache-nuttx-apps-9.0.x-incubating.tar.gz`. Both may be needed (see the top-level `nuttx/README.txt` file for build information).

Release Notes and Change Logs:

- **nuttx**.

  Release notes for all released versions on NuttX are available in GitHub. The ReleaseNotes for the current release is at the bottom of that file. The ChangeLog for all releases of NuttX is available in the ChangeLog file that can viewed in GitHub. The ChangeLog for the current release is at the bottom of that file.

apps.

Release notes for all released versions on NuttX are available in GitHub The ReleaseNotes for the current release is at the bottom of that file. The ChangeLog for all releases of `apps/` is available in the ChangeLog file that can viewed in the GitHub. The ChangeLog for the current release is at the bottom of that file.

NxWidgets.

As of NuttX-7.27, the content of the NxWidgets repository has been included within the `apps/` package and no longer has separate releases.

buildroot.

Release notes for buildroot 1.14 are available here. Release notes for all released versions on buildroot are available in the Bitbucket GIT The ChangeLog for all releases of buildroot is available at the bottom of the ChangeLog file that can viewed in the Bitbucket GIT.

Supported Platforms

Supported Platforms by CPU core. The number of ports to this CPU follow in parentheses. The state of the various ports vary from board-to-board. Follow the links for the details:

- Linux/Cygwin user mode simulation (1)
  - ARM
    - ARM7TDMI (4)
    - ARM920T (1)
    - ARM926EJS (4)
    - Other ARMv4 (1)
    - ARM1176JZ (1)
    - ARM Cortex-A5 (3)
    - ARM Cortex-A8 (2)
    - ARM Cortex-A9 (1)
    - ARM Cortex-R4 (2)
    - ARM Cortex-M0/M0+ (13)
    - ARM Cortex-M3 (39)
    - ARM Cortex-M4 (59)
    - ARM Cortex-M7 (15)
  - Atmel AVR
    - Atmel 8-bit AVR (5)
    - Atmel AVR32 (1)
  - Freescale
    - M68HCS12 (2)
  - Intel
    - Intel 80x86 (2)
  - Microchip
    - PIC32MX (MIPS M4K) (4)
    - PIC32MZEC (MIPS microAptive) (1)
    - PIC32MZEF (MIPS M5150) (1)
  - Misoc
    - LM32 (1)
    - Minerva (1)
  - OpenRISC
    - mor1kx (1)
  - Renesas/Hitachi:
    - Renesas/Hitachi SuperH (1/2)
    - Renesas M16C/26 (1/2)
    - Renesas RX65N (2)
  - RISC-V (2)
    - NEXT RISC-V NR5Mxx (1)
    - GreenWaves GAP8 (1)
    - Kendryte K210 (1)
    - Litex (1)
  - Xtensa LX6:
    - ESP32 (1)
  - ZiLOG
    - ZiLOG ZNEO Z16F (2)
    - ZiLOG eZ80 Acclaim! (4)
    - ZiLOG Z8Encore! (2)
    - ZiLOG Z180 (1)
    - ZiLOG Z80 (2)

Supported Platforms by Manufacturer/MCU Family. CPU core type follows in parentheses. The state of the various ports vary from MCU to MCU. Follow the links for the details:

- Allwinner
  - A10 (Cortex-A8)
- Broadcom
  - BCM2708 (ARM1176JZ)
- Espressif
  - ESP32 (Dual Xtensa LX6)
- GreenWaves
  - GAP8 (RISC-V RV32IM)
- Host PC based simulations
  - Linux/Cygwin user mode simulation
- Infineon
  - Infineon XMC45xx
- Intel
  - Intel 80x86
- Maxim Integrated
  - MAX32660 (ARM Cortex-M3)
- Microchip
  - PIC32MX2xx Family (MIPS32 M4K)
  - PIC32MX4xx Family (MIPS32 M4K)
  - PIC32MX7xx Family (MIPS32 M4K)
  - PIC32MZEC Family (MIPS32 microAptiv)
  - PIC32MZEF Family (MIPS32 M5150)
- Microchip (Formerly Atmel)
  - AVR ATMega128 (8-bit AVR)
  - AVR ATMega1284p (8-bit AVR)
  - AVR ATMega2560 (8-bit AVR)
  - AVR AT90USB64x and AT90USB6128x (8-bit AVR)
  - AVR32 AT32UC3BXXX (32-bit AVR32)
  - Atmel SAMD20 (ARM Cortex-M0+)
  - Atmel SAMD21 (ARM Cortex-M0+)
  - Atmel SAML21 (ARM Cortex-M0+)
  - Atmel SAM3U (ARM Cortex-M3)
  - Atmel SAM3X (ARM Cortex-M3)
  - Atmel SAM4C (ARM Cortex-M4)
  - Atmel SAM4E (ARM Cortex-M4)
  - Atmel SAM4L (ARM Cortex-M4)
  - Atmel SAM4S (ARM Cortex-M4)
  - Atmel SAMD5x/E5x (ARM Cortex-M4)
  - Atmel SAME70 (ARM Cortex-M7)
  - Atmel SAMV71 (ARM Cortex-M7)
  - Atmel SAMA5D2 (ARM Cortex-A5)
  - Atmel SAMA5D3 (ARM Cortex-A5)
  - Atmel SAMA5D4 (ARM Cortex-A5)
- Moxa
  - Moxa NP51x0 (ARMv4)
- nuvoTon
  - nuvoTon NUC120 (ARM Cortex-M0)
- Nordic Semiconductor
  - NRF52xxx (ARM Cortex-M4)
- NXP/Freescale
  - M68HCS12
  - NXP/Freescale i.MX1 (ARM920-T)
  - NXP/Freescale i.MX6 (ARM Cortex-A9)
  - NXP/Freescale i.MX RT (ARM Cortex-M7)
  - NXP/FreeScale KL25Z (ARM Cortex-M0+)
  - NXP/FreeScale KL26Z (ARM Cortex-M0+)
- NXP/Freescale (Continued)
  - NXP/FreeScale Kinetis K20 (ARM Cortex-M4)
  - NXP/FreeScale Kinetis K28 (ARM Cortex-M4)
  - NXP/FreeScale Kinetis K40 (ARM Cortex-M4)
  - NXP/FreeScale Kinetis K60 (ARM Cortex-M4)
  - NXP/FreeScale Kinetis K64 (ARM Cortex-M4)
  - NXP/FreeScale Kinetis K66 (ARM Cortex-M4)
  - NXP LPC11xx (Cortex-M0)
  - NXP LPC214x (ARM7TDMI)
  - NXP LPC2378 (ARM7TDMI)
  - NXP LPC3131 (ARM9E6JS)
  - NXP LPC315x (ARM9E6JS)
  - NXP LPC176x (ARM Cortex-M3)
  - NXP LPC178x (ARM Cortex-M3)
  - NXP LPC40xx (ARM Cortex-M4)
  - NXP LPC43xx (ARM Cortex-M4)
  - NXP LPC54xx (ARM Cortex-M4)
  - NXP S32K11x (Cortex-M0+)
  - NXP S32K14x (Cortex-M4F)
- ON Semiconductor:
  - LC823450 (Dual core ARM Cortex-M3)
- Renesas/Hitachi:
  - Renesas/Hitachi SuperH
  - Renesas M16C/26
  - Renesas RX65N
- Silicon Laboratories, Inc.
  - EFM32 Gecko (ARM Cortex-M3)
  - EFM32 Giant Gecko (ARM Cortex-M3)
- Sony.
  - CXD56*xx* (6 x ARM Cortex-M4)
- STMicroelectronics
  - STMicro STR71x (ARM7TDMI)
  - STMicro STM32F0xx (STM32 F0, ARM Cortex-M0)
  - STMicro STM32L0xx (STM32 L0, ARM Cortex-M0)
  - STMicro STM32G0xx (STM32 G0 ARM Cortex-M0+)
  - STMicro STM32L152 (STM32 L1 "EnergyLite" Line, ARM Cortex-M3)
  - STMicro STM32L162 (STM32 L1 "EnergyLite" Medium+ Density, ARM Cortex-M3)
  - STMicro STM32F100x (STM32 F1 "Value Line" Family, ARM Cortex-M3)
  - STMicro STM32F102x (STM32 F1 Family, ARM Cortex-M3)
  - STMicro STM32F103C4/C8 (STM32 F1 "Low- and Medium-Density Line" Family, ARM Cortex-M3)
  - STMicro STM32F103x (STM32 F1 Family, ARM Cortex-M3)
  - STMicro STM32F105x (ARM Cortex-M3)
  - STMicro STM32F107x (STM32 F1 "Connectivity Line" family, ARM Cortex-M3)
  - STMicro STM32F205x (STM32 F2 family, ARM Cortex-M3)
  - STMicro STM32F207x (STM32 F2 family, ARM Cortex-M3)
  - STMicro STM32F302x (STM32 F3 family, ARM Cortex-M4)
  - STMicro STM32F303x (STM32 F3 family, ARM Cortex-M4)
  - STMicro STM32F334 (STM32 F3 family, ARM Cortex-M4)
- STMicroelectronics (Continued)
  - STMicro STM32 F372/F373 (ARM Cortex-M4)
  - STMicro STM32F4x1 (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32F410 (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32F405x/407x (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32 F427/F437 (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32 F429 (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32 F433 (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32 F446 (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32 F46xx (STM32 F4 family, ARM Cortex-M4)
  - STMicro STM32 L4x2 (STM32 L4 family, ARM Cortex-M4)
  - STMicro STM32 L475 (STM32 L4 family, ARM Cortex-M4)
  - STMicro STM32 L476 (STM32 L4 family, ARM Cortex-M4)
  - STMicro STM32 L496 (STM32 L4 family, ARM Cortex-M4)
  - STMicro STM32 L4Rx (STM32 L4 family, ARM Cortex-M4)
  - STMicro STM32 F72x/F73x (STM32 F7 family, ARM Cortex-M7)
  - STMicro STM32 F745/F746 (STM32 F7 family, ARM Cortex-M7)
  - STMicro STM32 F756 (STM32 F7 family, ARM Cortex-M7)
  - STMicro STM32 F76xx/F77xx (STM32 F7 family, ARM Cortex-M7)
  - STMicro STM32 H7x3 (STM32 H7 family, ARM Cortex-M7)
- Texas Instruments (some formerly Luminary)
  - TI TMS320-C5471 (ARM7TDMI)
  - TI TMS320-DM320 (ARM9E6JS)
  - TI/Stellaris LM3S6432 (ARM Cortex-M3)
  - TI/Stellaris LM3S6432S2E (ARM Cortex-M3)
  - TI/Stellaris LM3S6918 (ARM Cortex-M3)
  - TI/Stellaris LM3S6965 (ARM Cortex-M3)
  - TI/Stellaris LM3S8962 (ARM Cortex-M3)
  - TI/Stellaris LM3S9B92 (ARM Cortex-M3)
  - TI/Stellaris LM3S9B96 (ARM Cortex-M3)
  - TI/SimpleLink CC13x0 (ARM Cortex-M3)
  - TI/Stellaris LM4F120x (ARM Cortex-M4)
  - TI/Tiva TM4C123G (ARM Cortex-M4)
  - TI/Tiva TM4C1294 (ARM Cortex-M4)
  - TI/Tiva TM4C129X (ARM Cortex-M4)
  - TI/SimpleLink CC13x2 (ARM Cortex-M4)
  - TI/Hercules TMS570LS04xx (ARM Cortex-R4)
  - TI/Hercules TMS570LS31xx (ARM Cortex-R4)
  - TI/Sitara AM335x (Cortex-A8)
- ZiLOG
  - ZiLOG ZNEO Z16F
  - ZiLOG eZ80 Acclaim!
  - ZiLOG Z8Encore!
  - ZiLOG Z180
  - ZiLOG Z80

Details. The details, caveats and fine print follow. For even more information see the *README* files that can be found here.

Linux User Mode.

A user-mode port of NuttX to the x86 Linux/Cygwin platform is available. The purpose of this port is primarily to support OS feature development.

STATUS: Does not support interrupts but is otherwise fully functional. Refer to the NuttX README file for further information.

ARM7TDMI.

TI TMS320C5471 (also called **C5471** or **TMS320DA180** or **DA180**). NuttX operates on the ARM7 of this dual core processor. This port uses the Spectrum Digital evaluation board with a GNU arm-nuttx-elf toolchain* under Linux or Cygwin.

STATUS: This port is complete, verified, and included in the initial NuttX release. Refer to the NuttX board README file for further information.

NXP LPC214x. Support is provided for the NXP LPC214x family of processors. In particular, support is provided for (1) the mcu123.com lpc214x evaluation board (LPC2148) and (1) the The0.net ZPA213X/4XPA development board (with the The0.net UG-2864AMBAG01 OLED) This port also used the GNU arm-nuttx-elf toolchain* under Linux or Cygwin.

STATUS: This port boots and passes the OS test (apps/examples/ostest). The port is complete and verified. As of NuttX 0.3.17, the port includes: timer interrupts, serial console, USB driver, and SPI-based MMC/SD card support. A verified NuttShell (NSH) configuration is also available. Refer to the NuttX board README files for the mcu123.com and for the ZPA213X/4XPA boards for further information.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain (CodeSourcery or devkitARM), or 4) Native Windows. A DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package.

NXP LPC2378. Support is provided for the NXP LPC2378 MCU. In particular, support is provided for the Olimex-LPC2378 development board. This port was contributed by Rommel Marcelo is was first released in NuttX-5.3. This port also used the GNU arm-nuttx-elf toolchain* under Linux or Cygwin.

STATUS: This port boots and passes the OS test (apps/examples/ostest) and includes a working implementation of the NuttShell (NSH). The port is complete and verified. As of NuttX 5.3, the port included only basic timer interrupts and serial console support. In NuttX 7.1, Lizhuoyi contributed additional I2C and SPI drivers. Refer to the NuttX board README file for further information.

Development Environments: (Same as for the NXP LPC214x).

STMicro STR71x. Support is provided for the STMicro STR71x family of processors. In particular, support is provided for the Olimex STR-P711 evaluation board. This port also used the GNU arm-nuttx-elf toolchain* under Linux or Cygwin.

STATUS: Integration is complete on the basic port (boot logic, system time, serial console). Two configurations have been verified: (1) The board boots and passes the OS test with console output visible on UART0, and the NuttShell (NSH) is fully functional with interrupt driven serial console. An SPI driver is available but only partially tested. Additional features are needed: USB driver, MMC integration, to name two (the slot on the board appears to accept on MMC card dimensions; I have only SD cards). An SPI-based ENC28J60 Ethernet driver for add-on hardware is available and but has not been fully verified on the Olimex board (due to issues powering the ENC28J60 add-on board). Refer to the NuttX board README file for further information.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain (CodeSourcery or devkitARM), or 4) Native Windows. A DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package.

ARM920T.

Freescale MC9328MX1 or **i.MX1**. This port uses the Freescale MX1ADS development board with a GNU arm-nuttx-elf toolchain* under either Linux or Cygwin.

STATUS: This port has stalled due to development tool issues. Coding is complete on the basic port (timer, serial console, SPI). Refer to the NuttX board README file for further information.

NOTE: This port has been obsoleted. I know longer have the hardware and the likelihood that the port would ever be completed is infitesmal. The unfinished board support is still available in the `Obsoleted` repository if anyone would ever like to resurrect it.

ARM926EJS.

TI TMS320DM320 (also called **DM320**). NuttX operates on the ARM9 of this dual core processor. This port uses the Neuros OSD with a GNU arm-nuttx-elf toolchain* under Linux or Cygwin. The port was performed using the OSD v1.0, development board.

STATUS: The basic port (timer interrupts, serial ports, network, framebuffer, etc.) is complete. All implemented features have been verified with the exception of the USB device-side driver; that implementation is complete but untested. Refer to the NuttX board README file for further information.

NXP LPC3131. Two boards based on the NXP LPC3131 are supported:

- First, a port for the NXP LPC3131 on the Embedded Artists EA3131 development board was first released in NuttX-5.1 (but was not functional until NuttX-5.2).

STATUS: The basic EA3131 port is complete and verified in NuttX-5.2. This basic port includes basic boot-up, serial console, and timer interrupts. This port was extended in NuttX 5.3 with a USB high speed driver contributed by David Hewson. David also contributed I2C and SPI drivers plus several important LPC313x USB bug fixes that appear in the NuttX 5.6 release. This port has been verified using the NuttX OS test, USB serial and mass storage tests and includes a working implementation of the NuttShell (NSH).

Support for on-demand paging has been developed for the EA3131. That support would all execute of a program in SPI FLASH by paging code sections out of SPI flash as needed. However, as of this writing, I have not had the opportunity to verify this new feature.

Refer to the Embedded Artists EA3131 board README file for further information.

- A second port to the NXP LPC3131 on the Olimex LPC-H3131 development board was added in NuttX-6.32.

STATUS: The basic H3131 port is complete and verified in NuttX-6.3. It is similar to the EA3131 port except: (1) I have not yet gotten the SDRAM to work, and (2) this board was used to develop and verify the USB 2.0, low-/full-/high-speed EHCI host controller driver. NOTE: That driver should work on the EA3131 as well. However, the EA3131 uses a PCA9532 PWM part to controller the port power so the it would not quite be a simple drop-in.

Refer to the Olimex LPC-H3131 board README file for further information.

NXP LPC315x. Support for the NXP LPC315x family has been incorporated into the code base as of NuttX-6.4. Support was added for the Embedded Artists EA3152 board in NuttX-6.11.

STATUS: Basic support is in place for both the LPC3152 MCU and the EA3152 board. Verification of the port was deferred due to tool issues However, because of the high degree of compatibility between the LPC313x and LPC315x family, it is very likely that the support is in place (or at least very close). At this point, verification of the EA3152 port has been overcome by events and may never happen. However, the port is available for anyone who may want to use it. Refer to the NuttX board README file for further information.

Other ARMv4.

MoxaRT A port to the Moxa NP51x0 series of 2-port advanced RS-232/422/485 serial device servers was contributed by Anton D. Kachalov in NuttX-7.11. This port includes a NuttShell (NSH) configuration with support for the Faraday FTMAC100 Ethernet MAC Driver.

ARM1176JZ.

Broadcom BCM2708. Very basic support for the Broadcom BCM2708 was released with NuttX-7.23.

Raspberry Pi Zero. This support was provided for the Raspberry Pi Zero which is based on the BCM2835. Basic logic is in place but the port is incomplete and completely untested as of the NuttX-7.23 released. Refer to the NuttX board README file for further information.

Obsoleted:: Support for the Raspberry Pi Zero was never completed. The incomplete port along with all support for the BCM2708 was removed from the repository with the NuttX-7.28 release but can still be be found in the *Obsoleted* repository.

ARM Cortex-A5.

Atmel SAMA5D2.

- **Atmel SAMA5D2 Xplained Ultra development board**. This is the port of NuttX to the Atmel SAMA5D2 Xplained Ultra development board. This board features the Atmel SAMA5D27 microprocessor.

STATUS. Initial support for the SAMA5D2 was released in NuttX-7.12. This port is code complete but, however, still a work in progress and has not been verified in this this initial release.

Atmel SAMA5D3. There are ports to two Atmel SAMA5D3 boards:

- **Atmel SAMA5D3*x*-EK development boards**. This is the port of NuttX to the Atmel SAMA5D3*x*-EK development boards (where *x*=1,3,4, or 5). These boards feature the Atmel SAMA5D3*x* microprocessors. Four different SAMA5D3*x*-EK kits are available

  - SAMA5D31-EK with the ATSAMA5D31
  - SAMA5D33-EK with the ATSAMA5D33
  - SAMA5D34-EK with the ATSAMA5D34
  - SAMA5D35-EK with the ATSAMA5D35

  The each kit consist of an identical base board with different plug-in modules for each CPU. All four boards are supported by NuttX with a simple reconfiguration of the processor type.

  STATUS. Initial support for the SAMA5D3x-EK was released in NuttX-6.29. That initial support was minimal: There are simple test configurations that run out of internal SRAM and extended configurations that run out of the on-board NOR FLASH:

  - A barebones NuttShell (NSH) configuration that can be used as the basis for further application development.
  - A full-loaded NuttShell (NSH) configuration that demonstrates all of the SAMA5D3x features.

  The following support was added in Nuttx 6.30:

  - DMA support, and
  - PIO interrupts,

  And drivers for

  - SPI (with DMA support),
  - AT25 Serial Flash,
  - Two Wire Interface (TWI), and
  - HSMCI memory cards.

  NuttX-6.30 also introduces full USB support:

  - High speed device controller driver,
  - OHCI (low- and full-speed) and
  - EHCI (high-speed) host controller driver support.

  With NuttX-6.31, these additional drivers were added:

  - A 10/100Base-T Ethernet (EMAC) driver,
  - A 1000Base-T Ethernet (GMAC) driver,
  - A Real Time Clock (RTC) driver and integrated with the NuttX system time logic
  - `/dev/random` using the SAMA5D3x True Random Number Generator (TRNG),
  - A Watchdog Timer (WDT) driver,
  - A Timer/Counter (TC) library with interface that make be used by other drivers that need timer support,
  - An ADC driver that can collect multiple samples using the sequencer, can be trigger by a timer/counter, and supports DMA data transfers,
  - A touchscreen driver based on the special features of the SAMA5D3 ADC peripheral, An LCD controller (LCDC) frame buffer driver, and
  - A CAN driver (Testing of the CAN has been delayed because of cabling issues).

  Additional board configurations were added to test and demonstrate these new drivers including new graphics and NxWM configurations.

  These drivers were added in NuttX-6.32:

  - A PWM driver with DMA support
  - An SSC-based I2S driver
  - Support for Programmable clock outputs
  - NAND support including support for the PMECC hardware ECC and for DMA transfers.

  DBGU support was added in NuttX-7.2 (primarily for the SAMA5D3 Xplained board).

  NuttX-7.4 added support for the on-board WM8904 CODEC chip and for *Tickless* operation.

  Refer to the NuttX board README file for further information.
- **Atmel SAMA5D3 Xplained development board** This is the port of NuttX to the Atmel SAMA5D3 Xplained development board. The board features the Atmel SAMA5D36 microprocessor. See the Atmel Website for additional information about this board.

  STATUS. This port is complete as of this writing and ready for general use. The basic port is expected to be simple because of the similarity to the SAMAD3*x*-EK boards and is available in the NuttX 7.2 release.

  Most of the drivers and capabilities of the SAMA5D3x-EK boards can be used with the SAMA5D3 Xplained board. The primary difference between the ports is that the SAMA5D3x-EK supports NOR FLASH and NuttX can be configured to boot directly from NOR FLASH. The SAMA5D3 Xplained board does not have NOR FLASH and, as a consequence NuttX must boot into SDRAM with the help of U-Boot.

  Refer to the NuttX board README file for further information.

Atmel SAMA5D4. There is a port in progress on one Atmel SAMA5D4 board:

- **Atmel SAMA5D4-EK/MB development boards** This is the port of NuttX to the Atmel SAMA5D4-MB Rev C. development board (which should be compatible with the SAMA5D4-EK). These boards feature the Atmel SAMA5D44 microprocessors with compatibility with most of the SAMA5D3 peripherals.

STATUS. At the time of the release of NuttX-7.3, the basic port for the SAMA5D4-MB was complete. The board had basic functionality. But full functionality was not available until NuttX-7.4. In NuttX-7.4 support was added for the L2 cache, many security features, XDMAC, HSMCI and Ethernet integrated with XDMAC, the LCDC, TWI, SSC, and most of the existing SAMA5 drivers. Timers were added to support *Tickless* operation. The TM7000 LCDC with the maXTouch multi-touch controller are also fully support in a special NxWM configuration for that larger display. Support for a graphics media player is included (although there were issues with the WM8904 audio CODEC on my board). An SRAM bootloader was also included. Refer to the NuttX board README file for current status.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain, or 4) Native Windows. All testing has been performed with the CodeSourcery toolchain (GCC version 4.7.3) in the Cygwin environment under Windows.

ARM Cortex-A8.

Allwinner A10. These following boards are based on the Allwinner A10 have are supported by NuttX:

- **pcDuino v1**. A port of NuttX to the pcDuino v1 board was first released in NuttX-6.33. See http://www.pcduino.com/ for information about pcDuino Lite, v1, and v2 boards. These boards are based around the Allwinner A10 Cortex-A8 CPU. This port was developed on the v1 board, but the others may be compatible:

Refer to the NuttX board README file for further information.

STATUS. This port was an experiment was was not completely developed. This configuration builds and runs an NuttShell (NSH), but only if a patch to work around some issues is applied. While not ready for "prime time", the pcDuino port is functional and could the basis for a more extensive development. There is, at present, no work in progress to extend this port, however.

TI/Sitara AM335x. These following boards are based on the TI/Sitara AM335x are supported by NuttX:

- **Beaglebone Black**. A port of NuttX to the Beaglebone Black board was first released in NuttX-7.28. This port was contributed by Petro Karashchenko. This board is based on the TI/Sitara AM3358 Cortex-A8 CPU running 1GHz.

  - **NuttX-7.28**. This initial port in NuttX-7.28 is very sparse. While not ready for prodcution use, the Beaglebone Black port is functional and will be the basis for a more extensive development. Additional work in progress to extend this port and more capable is anticipated in NuttX-7.29.
    - **NuttX-9.0** CAN support was added. Clock Configuration was added.
      - **NuttX-7.31**. An LCD driver was added in NuttX-7.31.

Refer to the Beaglebone Black board README file for further, up-to-date information.

ARM Cortex-A9.

NXP/Freescale i.MX6. The basic port has been completed for the following i.MX6 board

- **Sabre-6Quad**. This is a port to the NXP/Freescale Sabre-6Quad board. Refer to the NuttX board README file for further information.

STATUS: The basic, minimal port is code complete and introduced in NuttX-7.15, but had not yet been tested at that time due to the inavailability of hardware. This basic port was verified in the NuttX-7.16 release, however. The port is still minimal and more device drivers are needed to make the port usable.

Basic support of NuttX running in SMP mode on the i.MX6Q was also accomplished in NuttX-7.16. However, there are still known issues with SMP support on this platform as described in the README file for the board.

ARM Cortex-R4.

TI/Hercules TMS570LS04xx. A port is available for the Texas Instruments Hercules TMS570LS04x/03x LaunchPad Evaluation Kit (*LAUNCHXL-TMS57004*) featuring the Hercules TMS570LS0432PZ chip.

STATUS. My porting efforts were stalled due to tool-related issues. Refer to the NuttX board README file for further information. However, my understanding is that other people have successfully completed the port and submitted changes to the NuttX repository.

Toolchain: The TMS570 is a big-endian ARM platform and requires a big-endian ARM toolchain. All testing has been performed using a big-endian NuttX buildroot toolchain. Instructions for building this toolchain are included in the board README file.

TI/Hercules TMS570LS31xx. Architecture support for the TMS570LS3137ZWT part was added in NuttX 7.25 by Ivan Ucherdzhiev. Ivan also added support for the TI Hercules TMS570LS31x USB Kit.

STATUS. Refer to the NuttX board README file for further information.

Toolchain: See discussion related to the TI/Hercules TMS570LS04xx above.

ARM Cortex-M0/M0+.

nuvoTon NUC120. This is a port of NuttX to the nuvoTon NuTiny-SDK-NUC120 that features the NUC120LE3AN MCU.

STATUS. Initial support for the NUC120 was released in NuttX-6.26. This initial support is very minimal: There is a NuttShell (NSH) configuration that might be the basis for an application development. As of this writing, more device drivers are needed to make this a more complete port. Refer to the NuttX board README file for further information.

Memory Usage. For a full-featured RTOS such as NuttX, providing support in a usable and meaningful way within the tiny memories of the NUC120 demonstrates the scalability of NuttX. The NUC120LE2AN comes in a 48-pin package and has 128KB FLASH and 16KB of SRAM. When running the NSH configuration (itself a full up application), there is still more than 90KB of FLASH and 10KB or SRAM available for further application development).

Static memory usage can be shown with `size` command:

```
$ size nuttx
   text    data     bss     dec     hex filename
  35037     106    1092   36235    8d8b nuttx
```

NuttX, the NSH application, and GCC libraries use 34.2KB of FLASH leaving 93.8KB of FLASH (72%) free from additional application development. Static SRAM usage is about 1.2KB (<4%) and leaves 14.8KB (86%) available for heap at runtime. SRAM usage at run-time can be shown with the NSH `free` command:

```
NuttShell (NSH) NuttX-6.26
nsh> free
             total       used       free    largest
Mem:         14160       3944      10216       10216
nsh>
```

You can see that 10.0KB (62%) is available for further application development.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain, or 4) Native Windows. A DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package.

FreeScale KL25Z. There are two board ports for the KL25Z parts:

Freedom KL25Z. This is a port of NuttX to the Freedom KL25Z board that features the MKL25Z128 Cortex-M0+ MCU, 128KB of FLASH and 16KB of SRAM. See the Freescale website for further information about this board.

STATUS. This is the work of Alan Carvalho de Assis. Verified, initial, minimal support for the Freedom KL25Z is in place in NuttX 6.27 and 6.28: There is a working NuttShell (NSH) configuration that might be the basis for an application development. As of NuttX-6.28 more device driver development would be needed to make this a complete port, particularly to support USB OTG. A TSI and a SPI driver were added in NuttX-6.29. Alan contributed a PWM driver in NuttX-6.32. Refer to the Freedom KL25Z board README file for further information.

PJRC Teensy-LC. This is a port of NuttX to the PJRC Teensy-LC board that features the MKL25Z64 Cortex-M0+ MCU, 64KB of FLASH and 8KB of SRAM. The Teensy LC is a DIP style breakout board for the MKL25Z64 and comes with a USB based bootloader. See the Freescale website for further information about this board.

STATUS. This is the work of Michael Hope. Verified, initial support for the Teensy-LC first appeared in NuttX-7.10. Refer to the Teensy-LC board README file for further information.

FreeScale Freedom KL26Z. This is a port of NuttX to the Freedom KL25Z board that features the MK26Z128VLH4 Cortex-M0+ MCU, 128KB of FLASH and 16KB of SRAM. See the Freescale website for further information about this board.

STATUS. This work was contributed in NuttX 7.8 by Derek B. Noonburg. The board support is very similar to the Freedom-KL25Z. It was decided to support this a a separate board, however, due to some small board-level differences. Refer to the Freedom KL26Z board README file for further information.

Atmel SAMD20. The port of NuttX to the Atmel SAMD20-Xplained Pro development board. This board features the ATSAMD20J18A MCU (Cortex-M0+ with 256KB of FLASH and 32KB of SRAM).

STATUS. The initial SAMD20 Xplained Pro release (NuttX 7.1) included a functional NuttShell (NSH) configuration. An SPI driver was also included to support the OLED1 and I/O1 modules. That SPI driver, however, was not completely verified due to higher priority tasks that came up (I hope to get back to this later). Refer to the SAMD20 Explained Pro board README file for further information.

The fully verified SPI driver appeared in the NuttX-7.22 release.

Atmel SAMD21. There two boards supported for the SAMD21:

1. The port of NuttX to the Atmel SAMD21-Xplained Pro development board added in NuttX-7.11, and
2. The port of NuttX to the Arduino-M0 contributed by Alan Carvalho de Assis in NuttX-8.2. The initial release included *nsh* and *usbnsh* configurations.

STATUS. Refer to the board README files for the SAMD21-Xplained and the Arduino-M0 for further information.

Atmel SAML21. The port of NuttX to the Atmel SAML21-Xplained Pro development board. This board features the ATSAML21J18A MCU (Cortex-M0+ with 256KB of FLASH and 32KB of SRAM).

STATUS. Initial support for the SAML21 Xplained Pro was release in the NuttX 7.10. This initial support included a basic configuration for the NuttShell (NSH) (see the NSH User Guide). Refer to the SAML21 Explained Pro board README file for further information.

The fully verified SPI driver appeared in the NuttX-7.22 release along with an I2C and USB device driver.

NXP LPC11xx. Support is provided for the NXP LPC11xx family of processors. In particular, support is provided for LPCXpresso LPC1115 board. This port was contributed by Alan Carvalho de Assis.

STATUS: The first released version was provided in NuttX 7.10. Refer to the board README.txt file for further information.

Obsoleted: Support for the LPCXpresso-LPC1115 and for the LPC1115 architecture in general was removed after NuttX-7.30. The LPC11 port was never really used (to my knowledge) and was no longer supported. A snapshot of the port is still available in the Obsoleted repository. It can be brought back into the main repository at any time if anyone is willing to provide support for the architecture.

NXP S32K11x. Support is provided for the NXP S32K11x family of processors and, in particular, the S32K118EVB development board.

STATUS: The first released version was provided in NuttX 8.1. The S32K118EVB port port provides a minimal NSH configuration. Refer to the S32K118EVB board README.txt file for further information.

ARM Cortex-M3.

TI/Stellaris LM3S6432. This is a port of NuttX to the Stellaris RDK-S2E Reference Design Kit and the MDL-S2E Ethernet to Serial module (contributed by Mike Smith).

TI/Stellaris LM3S6432S2E. This port uses Serial-to-Ethernet Reference Design Kit (RDK-S2E) and has similar support as for the other Stellaris family members. A configuration is available for the NuttShell (NSH) (see the NSH User Guide). The NSH configuration including networking support with a Telnet NSH console. This port was contributed by Mike Smith.

STATUS: This port was was released in NuttX 6.14. Refer to the NuttX board README file for further information.

TI/Stellaris LM3S6918. This port uses the Micromint Eagle-100 development board with a GNU arm-nuttx-elf toolchain* under either Linux or Cygwin.

STATUS: The initial, release of this port was included in NuttX version 0.4.6. The current port includes timer, serial console, Ethernet, SSI, and microSD support. There are working configurations to run the NuttShell (NSH), the NuttX networking test, and the uIP web server. Refer to the NuttX board README file for further information.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain (CodeSourcery or devkitARM), or 4) Native Windows. A DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package.

TI/Stellaris LM3S6965. This port uses the Stellaris LM3S6965 Ethernet Evaluation Kit with a GNU arm-nuttx-elf toolchain* under either Linux or Cygwin.

STATUS: This port was released in NuttX 5.5. Features are the same as with the Eagle-100 LM3S6918 described above. The apps/examples /ostest configuration has been successfully verified and an NSH configuration with Telnet support is available. MMC/SD and Networking support was not been thoroughly verified: Current development efforts are focused on porting the NuttX window system (NX) to work with the Evaluation Kits OLED display.

NuttX-9.0 added protected build support to the LM3S6965-ek.

NOTE: As it is configured now, you MUST have a network connected. Otherwise, the NSH prompt will not come up because the Ethernet driver is waiting for the network to come up. Refer to the NuttX board README file for further information.

Development Environments: See the Eagle-100 LM3S6918 above.

TI/Stellaris LM3S8962. This port uses the Stellaris EKC-LM3S8962 Ethernet+CAN Evaluation Kit with a GNU arm-nuttx-elf toolchain* under either Linux or Cygwin. Contributed by Larry Arnold.

STATUS: This port was released in NuttX 5.10. Features are the same as with the Eagle-100 LM3S6918 described above. Refer to the NuttX board README file for further information.

TI/Stellaris LM3S9B92. Architectural support for the LM3S9B92 was contributed by Lwazi Dube in NuttX 7.28. No board support for boards using the LM3S9B92 are currently available.

TI/Stellaris LM3S9B96. Header file support was contributed by Tiago Maluta for this part. Jose Pablo Rojas V. is used those header file changes to port NuttX to the TI/Stellaris EKK-LM3S9B96. That port was available in the NuttX-6.20 release. Refer to the NuttX board README file for further information.

TI/SimpleLink CC13x0. Basic, unverified architectural support for the CC13x0 was added in NuttX-7.28. This is a work in progress and, with any luck, a fully verified port will be available in NuttX-7.29.

SiLabs EFM32 Gecko. This is a port for the Silicon Laboratories' EFM32 *Gecko* family. Board support is available for the following:

1. **SiLabs EFM32 Gecko Starter Kit (EFM32-G8XX-STK)**. The Gecko Starter Kit features:

- EFM32G890F128 MCU with 128 kB flash and 16 kB RAM
- 32.768 kHz crystal (LXFO) and 32 MHz crystal (HXFO)
- Advanced Energy Monitoring
- Touch slider
- 4x40 LCD
- 4 User LEDs
- 2 pushbutton switches
- Reset button and a switch to disconnect the battery.
- On-board SEGGER J-Link USB emulator
- ARM 20 pin JTAG/SWD standard Debug in/out connector

STATUS. The basic port is verified and available now. This includes on-board LED and button support and a serial console available on LEUART0. A single configuration is available using the NuttShell NSH and the LEUART0 serial console. DMA and USART-based SPI supported are included, but not fully tested.

Refer to the EFM32 Gecko Starter Kit *README.txt* file for further information.

2. **Olimex EFM32G880F120-STK**. This board features:

- EFM32G880F128 with 128 kB flash and 16 kB RAM
- 32.768 kHz crystal (LXFO) and 32 MHz crystal (HXFO)
- LCD custom display
- DEBUG connector with ARM 2x10 pin layout for programming/debugging with ARM-JTAG-EW
- UEXT connector
- EXT extension connector
- RS232 connector and driver
- Four user buttons
- Buzzer

STATUS. The board support is complete but untested because of tool-related issues. An OpenOCD compatible, SWD debugger would be required to make further progress in testing.

Refer to the Olimex EFM32G880F120-STK *README.txt* for further information.

SiLabs EFM32 Giant Gecko. This is a port for the Silicon Laboratories' EFM32 *Giant Gecko* family. This board features the EFM32GG990F1024 MCU with 1 MB flash and 128 kB RAM.

Board support is available for the following:

- **SiLabs EFM32 Giant Gecko Starter Kit t (EFM32GG-STK3700)**. The Gecko Starter Kit features:

  - EFM32GG990F1024 MCU with 1 MB flash and 128 kB RAM
  - 32.768 kHz crystal (LXFO) and 48 MHz crystal (HXFO)
  - 32 MB NAND flash
  - Advanced Energy Monitoring
  - Touch slider
  - 8x20 LCD
  - 2 user LEDs
  - 2 user buttons
  - USB interface for Host/Device/OTG
  - Ambient light sensor and inductive-capacitive metal sensor
  - EFM32 OPAMP footprint
  - 20 pin expansion header
  - Breakout pads for easy access to I/O pins
  - Power sources (USB and CR2032 battery)
  - Backup Capacitor for RTC mode
  - Integrated Segger J-Link USB debugger/emulator

STATUS.

- The basic board support for the *Giant Gecko* was introduced int the NuttX source tree in NuttX-7.6. A verified configuration was available for the basic NuttShell (NSH) using LEUART0 for the serial console.
- Development of USB support is in started, but never completed.
- Reset Management Unit (RMU) was added Pierre-noel Bouteville in NuttX-7.7.

STMicro STM32L152 (STM32L "EnergyLite" Line). Two boards are supported:

- STM32L-Discovery. This is a port of NuttX to the STMicro STM32L-Discovery development board. The STM32L-Discovery board is based on the STM32L152RBT6 MCU (128KB FLASH and 16KB of SRAM).

  The STM32L-Discovery and STM32L152C DISCOVERY kits are functionally equivalent. The difference is the internal Flash memory size (STM32L152RBT6 with 128 Kbytes or STM32L152RCT6 with 256 Kbytes). Both boards feature:

  - An ST-LINK/V2 embedded debug tool interface,
  - LCD (24 segments, 4 commons),
  - LEDs,
  - Pushbuttons,
  - A linear touch sensor, and
  - Four touchkeys.
- Nucleo-L152RE. Board support for the Nucleo-L152RE was contributed by Mateusz Szafoni in NuttX-7.28. Available configurations include NSH, ADC, and PWM.

STATUS. Initial support for the STM32L-Discovery was released in NuttX-6.28. Addition (architecture-only) support for the STM32L152xC family was added in NuttX-7.21. Support for the Nucleo-L152RE was added in NuttX-7.28.

That initial STM32L-Discovery support included a configuration using the NuttShell (NSH) that might be the basis for an application development. A driver for the on-board segment LCD is included as well as an option to drive the segment LCD from an NSH "built-in" command. Refer to the STM32L-Discovery board README file for further information.

Memory Usage.

> *REVISIT: These numbers are out of date. Current NuttX sizing might be somewhat larger.*

For a full-featured RTOS such as NuttX, providing support in a usable and meaningful way within the tiny memories of the STM32L152RBT6 demonstrates the scalability of NuttX. The STM32L152RBT6 comes in a 64-pin package and has 128KB FLASH and 16KB of SRAM.

Static memory usage can be shown with `size` command:

```
$ size nuttx
   text    data     bss     dec     hex filename
  39664     132    1124   40920    9fd8 nuttx
```

NuttX, the NSH application, and GCC libraries use 38.7KB of FLASH leaving 89.3B of FLASH (70%) free from additional application development. Static SRAM usage is about 1.2KB (<4%) and leaves 14.8KB (86%) available for heap at runtime.

SRAM usage at run-time can be shown with the NSH `free` command:

```
NuttShell (NSH) NuttX-6.27
nsh> free
             total       used       free    largest
Mem:         14096       3928      10168      10168
nsh>
```

You can see that 9.9KB (62%) of SRAM heap is still available for further application development while NSH is running.

STMicro STM32L152x/162x (STM32 L1 "EnergyLite" Medium+ Density Family). Support for the STM32L152 and STM32L162 Medium+ density parts from Jussi Kivilinna and Sami Pelkonen was included in NuttX-7.3, extending the basic STM32L152x support. This is *architecture-only* support, meaning that support for the boards with these chips is available, but no support for any publicly available boards is included.

STMicro STM32F0xx (STM32 F0, ARM Cortex-M0). Support for the STM32 F0 family was contributed by Alan Carvalho de Assis in NuttX-7.21. There are ports to three different boards in this repository:

- **STM32F0-Discovery** This board features the STM32 2F051R8 and was used by Alan to produce the initial STM32 F0 port. However, its very limited 8KB SRAM makes this port unsuitable for for usages. Contributed by Alan Carvalho de Assis in NuttX-7.21.
- **Nucleo-F072RB** With 16KB of SRAM the STM32 F072RB makes a much more usable platform.
- **Nucleo-F091RC** With 32KB of SRAM the STM32 F091RC this board is a great match for NuttX. Contributed by Juha Niskanen in NuttX-7.21.

STMicro STM32L0xx (STM32 L0, ARM Cortex-M0). Support for the STM32 FL family was contributed by Mateusz Sfafoni in NuttX-7.28. There are ports to two different STM32 L0 boards in the repository:

- **B-L072Z-LRWAN1** Contributed byMateusz Sfafoni in NuttX-7.28.
- **Nucleo-L073RZ** Contributed byMateusz Sfafoni in NuttX-7.28.

STMicro STM32G0xx (STM32 G0, ARM Cortex-M0+). Support for the STM32 FL family was contributed by Mateusz Sfafoni in NuttX-7.28. There are ports to two different STM32 L0 boards in the repository:

- **Nucleo-G071RB** Initial support for Nucleo-G071RB was contributed by Mateusz Szafoni in NuttX-7.31. Refer to the board README file for further information.
- **Nucleo-G070RB** Contributed by Daniel Pereira Volpato. in NuttX-8.2. Refer to the board README file for further information.

STATUS: Status for the STM32F0xx, STM32L0xx, and STM32G0xx is shown together since these parts share many drivers in common.

- **NuttX-7.21**. In this initial release, the level of support for the STM32 F0 family is minimal. Certainly enough is in place to support a robust NSH configuration. There are also unverified I2C and USB device drivers available in NuttX-7.21.
- **NuttX-7.28** Added support for GPIO EXTI. From Mateusz Sfafoni.
- **NuttX-7.29** Added an SPI driver. From Mateusz Sfafoni.
- **NuttX-7.30** Added ADC and I2C drivers. From Mateusz Szafoni. Add AES and RND drivers for the L0. From Mateusz Szafoni. Add support for HS148 for L0. From Mateusz Szafoni.
- **NuttX-8.2** Added PWM and TIM drivers for the G0. From Daniel Pereira Volpato.
- **NuttX-9.0** Added I2C support for F0, L0 and G0.

STMicro STM32F100x (STM32 F1 "Value Line"Family).

- **Proprietary Boards** Chip support for these STM32 "Value Line" family was contributed by Mike Smith and users have reported that they have successful brought up NuttX on their proprietary boards using this logic.
- **STM32VL-Discovery**. In NuttX-6.33, support for the STMicro STM32VL-Discovery board was contributed by Alan Carvalho de Assis. The STM32VL-Discovery board features an STM32F100RB MCU. Refer to the NuttX board README file for further information.

STMicro STM32F102. Architecture support (only) for the STM32 F102 family was contributed by the PX4 team in NuttX-7.7.

STATUS: Architecture support only is provided. No specific STM32 F102 boards are supported.

STMicro STM32F103C4/8 (STM32 F1 Low- and Medium-Density Family). There are two ports available for this family:

- One port is for "STM32 Tiny" development board. This board is available from several vendors on the net, and may be sold under different names. It is based on a STM32 F103C8T6 MCU, and is bundled with a nRF24L01 wireless communication module.
- The other port is for a generic minimal STM32F103CBT6 "blue" board contributed by Alan Carvalho de Assis. Alan added support for numerous sensors, tone generators, user LEDs, and LCD support in NuttX 7.18.

STATUS:

The basic STM32F103C8 port was released in NuttX version 6.28. This work was contributed by Laurent Latil. Refer to the NuttX board README file for further information.

STMicro STM32F103x (STM32 F1 Family). Support for five board configurations are available. MCU support includes all of the high density and connectivity line families. Board supported is available specifically for: STM32F103ZET6, STM32F103RET6, STM32F103VCT, STM32F103VET6, STM32F103RBT6, and STM32103CBT6. Boards supported include:

1. **STM3210E-EVAL**. A port for the STMicro STM3210E-EVAL development board that features the STM32F103ZET6 MCU. Refer to the NuttX board README file for further information.
2. **HY-Mini STM32v board**. This board is based on the STM32F103VCT chip. Port contributed by Laurent Latil. Refer to the NuttX board README file.
3. **The M3 Wildfire development board (STM32F103VET6), version 2**. See http://firestm32.taobao.com (the current board is version 3). Refer to the NuttX board README file for further information.
4. **LeafLab's Maple and Maple Mini boards**. These boards are based on the STM32F103RBT6 chip for the standard version and on the STM32F103CBT6 for the mini version. See the LeafLabs web site for hardware information; see the NuttX board README file for further information about the NuttX port.
5. **Olimexino-STM32**. This port uses the Olimexino STM32 board (STM32F103RBT6). See the http://www.olimex.com for further information. Contributed by David Sidrane.
6. **Nucleo-STM32F103RB**. This port uses the STM32F103RBT6. It was contributed by Mateusz Szafoni in NuttX-7.28,

These ports uses a GNU arm-nuttx-elf toolchain* under either Linux or Cygwin (with native Windows GNU tools or Cygwin-based GNU tools).

STATUS:

- **Basic Support/Drivers**. The basic STM32 port was released in NuttX version 0.4.12. The basic port includes boot-up logic, interrupt driven serial console, and system timer interrupts. The 0.4.13 release added support for SPI, serial FLASH, and USB device.; The 4.14 release added support for buttons and SDIO-based MMC/SD and verified DMA support. Verified configurations are available for the NuttShell (NSH) example, the USB serial device class, and the USB mass storage device class example.
- **Additional Drivers**. Additional drivers and configurations were added in NuttX 6.13 and later releases for the STM32 F1 and F4. F1 compatible drivers include an Ethernet driver, ADC driver, DAC driver, PWM driver, IWDG, WWDG, and CAN drivers.
- **M3 Wildfire**. Support for the Wildfire board was included in version 6.22 of NuttX. The board port is basically functional. Not all features have been verified. Support for FAT file system on an an SD card had been verified. The ENC28J60 network is functional (but required lifting the chip select pin on the W25x16 part). Customizations for the v3 version of the Wildfire board are selectable (but untested).
- **Maple**. Support for the Maple boards was contributed by Yiran Liao and first appear in NuttX 6-30.
- **Olimexino-STM32**. Contributed by David Sidrane and introduced with NuttX 7.9. Configurations are included for the NuttShell (NSH), a tiny version of the NuttShell, USB composite CDC/ACM + MSC, CAN support, and two tiny, small-footprint NSH configurations.
- **Nucleo-STM32F103RB**. Contributed by Mateusz Szafoni and introduced with NuttX 7.28. Configurations are included for the NuttShell (NSH), ADC, and PWM.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain (RIDE7, CodeSourcery or devkitARM), or 4) Native Windows. A DIY toolchain or Linux or Cygwin is provided by the NuttX buildroot package.

STMicro STM32F105x. Architecture support (only) for the STM32 F105R was contribed in NuttX-7.17 by Konstantin Berezenko. There is currently no support for boards using any STM32F105x parts in the source tree.

STMicro STM32F107x (STM32 F1 "Connectivity Line" family). Chip support for the STM32 F1 "Connectivity Line" family has been present in NuttX for some time and users have reported that they have successful brought up NuttX on their proprietary boards using this logic.

- **Olimex STM32-P107** Support for the Olimex STM32-P107 was contributed by Max Holtzberg and first appeared in NuttX-6.21. That port features the STMicro STM32F107VC MCU.

STATUS: A configuration for the NuttShell (NSH) is available and verified. Networking is functional. Support for an external ENCX24J600 network was added in NuttX 6.30.

- **Shenzhou IV** A port of NuttX to the Shenzhou IV development board (See www.armjishu.com) featuring the STMicro STM32F107VCT MCU was added in NuttX-6.22.

STATUS: In progress. The following have been verified: (1) Basic Cortex-M3 port, (2) Ethernet, (3) On-board LEDs. Refer to the NuttX board README file for further information.

- **ViewTool STM32F103/F107** Support for the Viewtool STM32F103/F107 board was added in NuttX-6.32. That board features the STMicro STM32F107VCT6 MCU. Networking, LCD, and touchscreen support were added in NuttX-6.33.

Three configurations are available:

1. A standard NuttShell (NSH) configuration that will work with either the STM32F103 or STM32F107 part.
2. A network-enabled NuttShell (NSH) configuration that will work only with the STM32F107 part.
3. The configuration that was used to verify the Nuttx high-priority, nested interrupt feature.

STATUS: Networking and touchscreen support are well test. But, at present, neither USB nor LCD functionality have been verified. Refer to the Viewtool STM32F103/F107 README file for further information.

- **Kamami STM32 Butterfly 2** Support for the Kamami STM32 Butterfly 2 was contributed by Micha yszczek in NuttX-7.18. That port features the STMicro STM32F107VC MCU.

STATUS: A configuration for the NuttShell (NSH), NSH with networking, and NSH with USB host are available and verified.

STMicro STM32F205 (STM32 F2 family). Architecture only support for the STM32F205RG was contributed as an anonymous contribution in NuttX-7.10.

Particle.io Phone. Support for the Particle.io Photon board was contributed by Simon Pirious in NuttX-7.21. The Photon board features the STM32F205RG MCU. The STM32F205RG is a 120 MHz Cortex-M3 operation with 1Mbit Flash memory and 128kbytes. The board port includes support for the on-board Broadcom BCM43362 WiFi and fully usable FullMac network support.

STATUS: In addition to the above-mention WiFI support, the Photon board support includes buttons, LEDS, IWDG, USB OTG HS, and procfs support. Configurations available for nsh, usbnsh, and wlan configurations. See the Photon README file for additional information.

STMicro STM32F207 (STM32 F2 family).

- Support for the STMicro STM3220G-EVAL development board was contributed by Gary Teravskis and first released in NuttX-6.16. This board uses the STM32F207IG.
- Martin Lederhilger contributed support for the Olimex STM32 P207 board using the STM32F207ZE MCU.
- Board support for the Nucleo-L152RE was contributed by Mateusz Szafoni in NuttX-7.28. Available configurations include NSH, ADC, and PWM.

STATUS: The peripherals of the STM32 F2 family are compatible with the STM32 F4 family. See discussion of the STM3240G-EVAL board below for further information. Refer also to the NuttX board README file for further information.

Support for both the IAR and uVision GCC IDEs added for the STM3220G-EVAL board in NuttX 7.16. From Kha Vo.

Atmel SAM3U. This port uses the Atmel SAM3U-EK development board that features the SAM3U4E MCU. This port uses a GNU arm-nuttx-elf or arm-nuttx-eabi toolchain* under either Linux or Cygwin (with native Windows GNU tools or Cygwin-based GNU tools).

STATUS: The basic SAM3U-EK port was released in NuttX version 5.1. The basic port includes boot-up logic, interrupt driven serial console, and system timer interrupts. That release passes the NuttX OS test and is proven to have a valid OS implementation. A configuration to support the NuttShell is also included. NuttX version 5.4 adds support for the HX8347 LCD on the SAM3U-EK board. This LCD support includes an example using the NX graphics system. NuttX version 6.10 adds SPI support. Touchscreen support was added in NuttX-6.29.

Subsequent NuttX releases will extend this port and add support for the SDIO-based SD cards and USB device. Refer to the NuttX board README file for further information about this port.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain (CodeSourcery or devkitARM), or 4) Native Windows. A DIY toolchain for inux or Cygwin is provided by the NuttX buildroot package.

Atmel SAM3X. There are two SAM3X boards supported:

1. The Arduino Due development board that features the ATSAM3X8E MCU running at 84MHz. See the Arduino Due page for more information.
2. The Mikroelektronika Flip&Click SAM3X development board. This board is an Arduino Due *work-alike* with additional support for 4 mikroBUS Click boards.

STATUS: As of this writing, the basic Arduino Due port is code complete and a fully verified configuration exists for the NuttShell NSH). The first fully functional Arduino Due port was released in NuttX-6.29. Refer to the NuttX board README file for further information.

Support for the Flip&Click SAM3X was added in NuttX-7.24.

Development Environments: See the Atmel SAM3U discussion above.

NXP LPC1766, LPC1768, and LPC1769. Drivers are available for CAN, DAC, Ethernet, GPIO, GPIO interrupts, I2C, UARTs, SPI, SSP, USB host, and USB device. Additional drivers for the RTC, ADC, DAC, Timers, PWM and MCPWM were contributed by Max (himax) in NuttX-7.3. Verified LPC17xx configurations are available for these boards:

- The Nucleus 2G board from 2G Engineering (LPC1768),
- The mbed board from mbed.org (LPC1768, Contributed by Dave Marples), and
- The LPC1766-STK board from Olimex (LPC1766).
- The Embedded Artists base board with NXP LPCXpresso LPC1768.
- Zilogic's ZKIT-ARM-1769 board.
- The Micromint Lincoln60 board with an NXP LPC1769.
- A version of the LPCXPresso LPC1768 board with special support for the U-Blox model evaluation board.
- Support for the Keil MCB1700 was contributed by Alan Carvalho de Assis in NuttX-7.23.
- Support for the NXP Semiconductors' PN5180 NFC Frontend Development Kit was contributed by Michael Jung in NuttX-7.1. This board is based on the NXP LPC1769 MCU.

The Nucleus 2G board, the mbed board, the LPCXpresso, and the MCB1700 all feature the NXP LPC1768 MCU; the Olimex LPC1766-STK board features an LPC1766. All use a GNU arm-nuttx-elf or arm-eabi toolchain* under either Linux or Cygwin (with native Windows GNU tools or Cygwin-based GNU tools).

STATUS: The following summarizes the features that has been developed and verified on individual LPC17xx-based boards. These features should, however, be common and available for all LPC17xx-based boards.

1. **Nucleus2G LPC1768**

- Some initial files for the LPC17xx family were released in NuttX 5.6, but
- The first functional release for the NXP LPC1768/Nucleus2G occurred with NuttX 5.7 with Some additional enhancements through NuttX-5.9. Refer to the NuttX board README file for further information.

That initial, 5.6, basic release included *timer* interrupts and a *serial console* and was verified using the NuttX OS test (`apps/examples/ostest`). Configurations available include include a verified NuttShell (NSH) configuration (see the NSH User Guide). The NSH configuration supports the Nucleus2G's microSD slot and additional configurations are available to exercise the USB serial and USB mass storage devices. However, due to some technical reasons, neither the SPI nor the USB device drivers are fully verified. (Although they have since been verified on other platforms; this needs to be revisited on the Nucleus2G).

Obsoleted. Support for the Nucleus2G board was terminated on 2016-04-12. There has not been any activity with the commercial board in a few years and it no longer appears to be available from the 2g-eng.com website. Since the board is commercial and no longer publicly available, it no longer qualifies for inclusion in the open source repositories. A snapshot of the code is still available in the Obsoleted repository and can easily be *reconstitued* if needed.

2. **mbed LPC1768**

- Support for the mbed board was contributed by Dave Marples and released in NuttX-5.11. Refer to the NuttX board README file for further information.

3. **Olimex LPC1766-STK**

- Support for that Olimex-LPC1766-STK board was added to NuttX 5.13.
- The NuttX-5.14 release extended that support with an *Ethernet driver*.
- The NuttX-5.15 release further extended the support with a functional *USB device driver* and *SPI-based micro-SD*.
- The NuttX-5.16 release added a functional *USB host controller driver* and *USB host mass storage class driver*.
- The NuttX-5.17 released added support for low-speed USB devices, interrupt endpoints, and a *USB host HID keyboard class driver*.
- Refer to the NuttX board README file for further information.

Verified configurations are now available for the NuttShell with networking and microSD support(NSH, see the NSH User Guide), for the NuttX network test, for the THTTPD webserver, for USB serial deive and USB storage devices examples, and for the USB host HID keyboard driver. Support for the USB host mass storage device can optionally be configured for the NSH example. A driver for the *Nokia 6100 LCD* and an NX graphics configuration for the Olimex LPC1766-STK have been added. However, neither the LCD driver nor the NX configuration have been verified as of the NuttX-5.17 release.

4. **Embedded Artists base board with NXP LPCXpresso LPC1768**

An fully verified board configuration is included in NuttX-6.2. The Code Red toolchain is supported under either Linux or Windows. Verified configurations include DHCPD, the NuttShell (NSH), NuttX graphis (NX), THTTPD, and USB mass storage device. Refer to the NuttX board README file for further information.

5. **Zilogic's ZKIT-ARM-1769 board**

Zilogic System's ARM development Kit, ZKIT-ARM-1769. This board is based on the NXP LPC1769. The initial release was included NuttX-6.26. The Nuttx Buildroot toolchain is used by default. Verifed configurations include the "Hello, World!" example application and a THTTPD demonstration. Refer to the NuttX board README file for further information.

6. **Micromint Lincoln60 board with an NXP LPC1769**

This board configuration was contributed and made available in NuttX-6.20. As contributed board support, I am unsure of what all has been verfied and what has not. See the Micromint website Lincoln60 board and the NuttX board README file for further information about the Lincoln board.

7. **U-Blox Modem Evaluation (LPCXpresso LPC1768)**

This board configuration was contributed by Vladimir Komendantskiy and made available in NuttX-7.15. This is a variant of the LPCXpresso LPC1768 board support with special provisions for the U-Blox Model Evaluation board. See the NuttX board README file for further information about this port.

8. **Keil MCB1700 (LPC1768)**

This board configuration was contributed by Alan Carvalho de Assis in NuttX-7.23.

9. **PN5180 NFC Frontend Development Kit**

This board configuration was contributed by Michael Jung in NuttX-7.31.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain (CodeSourcery devkitARM or Code Red), or 4) Native Windows. A DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package.

NXP LPC1788. The port of NuttX to the WaveShare Open1788 is a collaborative effort between Rommel Marcelo and myself (with Rommel being the leading contributor and I claiming only a support role). You can get more information at the Open1788 board from the WaveShare website.

STATUS: Initial Open1788 support appeared in NuttX-6.26 with the first verified configurations in NuttX-6.27. In NuttX-6.27 there is a working basic port with OS verification, Nuttshell (NSH) configurations, and a graphics test configuration. SDRAM and GPDMA are working. The NSH configuration includes verified support for a DMA-based SD card interface. The frame-buffer LCD driver is functional and uses the SDRAM for frame-buffer memory. A touchscreen interface has been developed but there appears to be a hardware issue with the WaveShare implementation of the XPT2046 touchscreen controller. Refer to the NuttX board README file for further information.

NuttX-7.29 Configurations were added to verify the use of *NxTerms* in the `PROTECTED` build most as was as graphic examples using the "Per-Window Framebuffer" support which was also introduced in NuttX-7.29.

ON Semiconductor LC823450 (Dual core ARM Cortex-M3). In NuttX-7.22, Masayuki Ishikawa contributed support for both the LC823450 architecture and for ON Semiconductor's **LC823450XGEVK board**:

> *The LC823450XGEVK is an audio processing system Evaluation Board Kit used to demonstrate the LC823450. This part can record and playback, and offers High-Resolution 32-bit & 192 kHz audio processing capability. It is possible to cover most of the functions necessary for a portable audio with only this LSI as follows. It has Dual CPU and DSP with High processing capability, and internal 1656K-Byte SRAM, which make it possible to implement large scale program. And it has integrated analog functions (low-power Class D HP amplifier, PLL, ADC etc.) so that PCB space and cost is reduced, and it has various interface (USB, SD, SPI, UART, etc.) to make extensibility high. Also it is provided with various function including SBC/AAC codec by DSP and UART and ASRC (Asynchronous Sample Rate Converter) for Bluetooth® audio. It is very small chip size in spite of the multi-funciton as described above and it realizes the low power consumption. Therefore, it is applicable to portable audio markets such as Wireless headsets and will show high performance.*

Further information about the LC823450XGEVK is available on from the the ON Semiconductor website as are LC823450 related technical documents. Refer to the NuttX board README file for details of the NuttX port.

This port is intended to test LC823450 features including SMP. Supported peripherals include UART, TIMER, RTC, GPIO, DMA, I2C, SPI, LCD, eMMC, and USB device. ADC, Watchdog, IPC2, and I2S support was added by Masayuki Ishikawa in NuttX-7.23. Bluetooth, SPI, and *PROTECTED* build support were added by Masayuki Ishikawa in NuttX-7.26. Support for for SPI flash boot was added in NuttX-7.28.

Maxim Integrated MAX36600. Architectural upport for the MAX32660 was added (along with partial support for other members of the MAX326xx family) in NuttX 7.28.

MAX32660-EVSYS. Basic support for the Maxim Integrated MAC3X660 EVSYS was included in the NuttX-7.28 release. A basic NSH configuration is available and is fully functional. Includes unverified support for an SPI0-based SD card.

STATUS:

NuttX-7.29. The initial release included: Clock configuration, timer, GPIO pin configuration, ICC, and UART. Additional untested drivers are complete and ready for testing: DMA, GPIO interrupts, SPI0 Master, TC, WDT. The following drivers are not yet implemented: I2C and I2S.

README File. Refer to the MAX32660-EVSYS README file for further information.

(ARM Cortex-M3)

ARM Cortex-M4.

Infineon XMC45xx. An initial but still incomplete port to the XMC4500 Relax board was released with NuttX-7.21 (although it was not really ready for prime time). Basic NSH functionality was a serial console was added by Alan Carvahlo de Assis in NuttX-7.23. Alan also added an SPI driver in NuttX-7.25.

This initial porting effort uses the Infineon XMC4500 Relax v1 board as described on the manufacturer's website. The current status of the board is available in the board README file

---

Nordic Semiconductor/NRF52xxx. Initial architecture support of the NRF52 including UART, Timer, and GPIOs was contributed by Janne Rosberg in NuttX-7.25. Janne also contributed board support for the NRF52-PCA10040 development board at that time.

The NRF52 was generalized by Hanya Zou in NuttX-7.28 for any similar board based on the NRF52832 MCU. Support was specifically included for the Adafruit NRF52 Feather board.

Available drivers include:

- **NuttX-7.25**. UART, Timer, and GPIOs from Janne Rosberg and a watchdog timer driver was added by Levin Li.
- **NuttX-7.25**. Flash PROGMEM support was added by Alan Carvalho de Assis.
- **NuttX-7.29**. Support for the 52804 family and an RNG drivers was added by Levin Li.

STATUS: Refer to the generic NRF52 board README file for further information.

NXP/FreeScale Kinetis K20/Teensy-3.x. Architecture support (only) was added in NuttX-7.10. This support was taken from PX4 and is the work of Jakob Odersky. Support was added for the PJRC Teensy-3.1 board in NuttX-7.11. Backward compatible support for the Teensy-3.0 is included.

STATUS: Refer to the Teensy-3.1 board README file for further information.

NXP/FreeScale Kinetis K28F/Freedom-K28F. Architecture support for the Kinetis K28F along with board support for the Freedom-K28F was added in NuttX-7.15. The Freedom-K28F board is based on the Kinetis MK28FN2M0VMI15 MCU (ARM Cortex-M4 at150 MHz, 1 MB SRAM, 2 MB flash, HS and FS USB, 169 MAPBGA package). More information is available from the NXP website.

STATUS: Refer to the Freedom-K28F board README file for further information.

NXP/FreeScale Kinetis K40. This port uses the Freescale Kinetis KwikStik K40. Refer to the Freescale web site for further information about this board. The Kwikstik is used with the FreeScale Tower System (mostly just to provide a simple UART connection)

STATUS: The unverified KwikStik K40 first appeared in NuttX-6.8 As of this writing, the basic port is complete but I accidentally locked my board during the initial bringup. Further development is stalled unless I learn how to unlock the device (or until I get another K40). Additional work remaining includes, among other things: (1) complete the basic bring-up, (2) bring up the NuttShell NSH, (3) develop support for the SDHC-based SD card, (4) develop support for USB host and device, and (2) develop an LCD driver. NOTE: Some of these remaining tasks are shared with the K60 work described below. Refer to the NuttX board README file for further information.

NXP/FreeScale Kinetis K60. This port uses the **Freescale Kinetis TWR-K60N512** tower system. Refer to the Freescale web site for further information about this board. The TWR-K60N51 includes with the FreeScale Tower System which provides (among other things) a DBP UART connection.

STATUS: As of this writing, the basic port is complete and passes the NuttX OS test. An additional, validated configuration exists for the NuttShell (NSH, see the NSH User Guide). This basic TWR-K60N512 first appeared in NuttX-6.8. Refer to the NuttX board README file for further information.

MK60N512VLL100. Architecture support for the MK60N512VLL100 was contributed by Andrew Webster in NuttX-7.14.

NXP/FreeScale Kinetis K64. Support for the Kinetis K64 family and specifically for the **NXP/Freescale Freedom K64F** board was added in NuttX 7.17. Initial release includes two NSH configurations with support for on-board LEDs, buttons, and Ethernet with the on-board KSZ8081 PHY. SDHC supported has been integrated, but not verified. Refer to the NuttX board README file for further information.

MK64FN1M0VMD12. Architecture support for the _MK64FN1M0VMD12 was contributed by Maciej Skrzypek in NuttX-7.20.

NXP/Freescale Kinetis TWR-K64F120M. Support for the Freescale Kinetis TWR-K64F120M was contributed in NuttX-7.20 by Maciej Skrzypek. Refer to the Freescale web site for further information about this board. The board may be complemented by TWR-SER which includes (among other things), an RS232 and Ethernet connections. Refer to the NuttX board README file for further information.

Driver Status.

- **NuttX-6.8**. Ethernet and SD card (SDHC) drivers also exist: The SDHC driver is partially integrated in to the NSH configuration but has some outstanding issues. Additional work remaining includes: (1) integrate th SDHC drivers, and (2) develop support for USB host and device. NOTE: Most of these remaining tasks are the same as the pending K40 tasks described above.
- **NuttX-7.14**. The Ethernet driver became stable in NuttX-7.14 thanks to the efforts of Andrew Webster.
- **NuttX-7.17**. Ethernet support was extended and verified on the Freedom K64F. A Kinetis USB device controller driver and PWM support was contributed by kfazz.

NXP/FreeScale Kinetis K66. Support for the Kinetis K64 family and specifically for the **NXP/Freescale Freedom K66F** board was contributed by David Sidrane in NuttX 7.20. Refer to the NuttX board README file for further information.

Driver Status.

- Most K6x drivers are compatible with the K66.
- **NuttX-7.20**. David Sidrane also contributed support for a serial driver on the K66's LPUART.
- **NuttX-7.22**. David Sidrane contributed improvements to the USB and I2C device drivers, RTC alarm functionality, and new SPI driver.
- **NuttX-7.26**. David Sidrane contributed DMA support to the Kinetis K6x family.

Sony CXD56*xx* (6 x ARM Cortex-M4). Support for the CXD56*xx* was introduced by Nobuto Kobayashi in NuttX-7.30.

Sony Spresence. Spresense is a compact development board based on Sony's power-efficient multicore microcontroller CXD5602. Basic support for the Sony Spresense board was included in the contribution of Nobuto Kobayashi in NuttX-7.30. *NOTE*: That was an initial, bare bones basic Spresense port sufficient for running a NuttShell (NSH) and should not be confused with the full Spresence SDK offered from Sony. Since then there has been much development of the NuttX CXD56xx port.

Features:

- Integrated GPS: Embedded GNSS with support for GPS, QZSS.
- Hi-res audio output and multi mic inputs" Advanced 192kHz/24 bit audio codec and amplifier for audio output, and support for up to 8 mic input channels.
- Multicore microcontroller: Spresense is powered by Sony's CXD5602 microcontroller (ARM® Cortex®-M4F × 6 cores), with a clock speed of 156 MHz.

Driver Status:

- **NuttX-3.31**. In this release, many new architectural features, peripheral drivers, and board configurations were contributed primarily through the work of Alin Jerpelea. These new architectural features include: Inter-core communications, power management, and clock management. New drivers include: GPIO, PMIC, USB, SDHC, SPI, I2C, DMA, RTC, PWM, Timers, Watchdog Timer, UID, SCU, ADC, eMMC, Camera CISIF, GNSS, and others.

NuttX-8.1. Alin Jerpelea brought in ten (external) sensor drivers that integrate through the CXD56xx's SCU.

NuttX-8.2. Masayuki Ishikawa implemented SMP operation of the CX56Dxx parts. Alin Jerpelea: Added support for the Altair LTE modem support, enabled support for accelerated format converter, rotation and so on using the CXD5602 image processing accelerator, added ISX012 camera support, added audio and board audio control implementation, added an audio_tone_generator, added optional initialization of GNSS and GEOFENCE at boot if the drivers are enabled, added an lcd examples configuration.

STMicro STM32 F302 (STM32 F3 family). Architecture (only) support for the STM32 F302 was contributed in NuttX-7.10 by Ben Dyer (via the PX4 team and David Sidrane).

Support for the Nucleo-F302R8 board was added by raiden00pl in NuttX-7.27. Refer to the NuttX board README file for further information.

STMicro STM32 F303 (STM32 F3 family).

- **STM32F3-Discovery**. This port uses the STMicro STM32F3-Discovery board featuring the STM32F303VCT6 MCU (STM32 F3 family). Refer to the STMicro web site for further information about this board.

  STATUS: The basic port for the STM32F3-Discover was first released in NuttX-6.26. Many of the drivers previously released for the STM32 F1, Value Line, and F2 and F4 may be usable on this platform as well. New drivers will be required for ADC and I2C which are very different on this platform. Refer to the NuttX board README file for further information.
- **STMicro ST Nucleo F303RE board**. The basic port for the Nucleo F303RE was contributed by Paul Alexander Patience and first released in NuttX-7.12. Refer to the NuttX board README file for further information.
- **STMicro ST Nucleo F303ZE board**. Support for the Nucleo-F303ZE board was added by Mateusz Szafoni in NuttX-7.27. Refer to the NuttX board README file for further information.

STMicro STM32 F334 (STM32 F3 family, ARM Cortex-M4).

Support for the STMicro **STM32F334-Disco** board was contributed by Mateusz Szafoni in NuttX-7.22 and for the **Nucleo-STM32F334R8** was contributed in an earlier release.

STMicro STM32 F372/F373 (ARM Cortex-M4).

Basic architecture support for the STM32F372/F373 was contributed by Marten Svanfeldt in NuttX 7.9. There are no STM32F*72 boards currently supported, however.

STMicro STM324x1 (STM32 F4 family).

Nucleo F401RE. This port uses the STMicro Nucleo F401RE board featuring the STM32F401RE MCU. Refer to the STMicro web site for further information about this board.

Nucleo F411RE. This port uses the STMicro Nucleo F411RE board featuring the STM32F411RE MCU. Refer to the STMicro web site for further information about this board.

STATUS:

- **NuttX-7.2** The basic port for STMicro Nucleo F401RE board was contributed by Frank Bennett.
- **NuttX-7.6** The basic port for STMicro Nucleo F401RE board was added by Serg Podtynnyi.
- **NuttX-7.25** Architecture support (only) for STMicro STM32F401xB and STM32F401xC pars was added.
- Refer to the NuttX board README file for further information.

STMicro STM32410 (STM32 F4 family).

Architecture-only support was contributed to NuttX-7.21 by Gwenhael Goavec-Merou.

STMicro STM32405x/407x (STM32 F4 family).

STMicro STM3240G-EVAL. This port uses the STMicro STM3240G-EVAL board featuring the STM32F407IGH6 MCU. Refer to the STMicro web site for further information about this board.

STATUS:

- **NuttX-6.12** The basic port is complete and first appeared in NuttX-6.12. The initial port passes the NuttX OS test and includes a validated configuration for the NuttShell (NSH, see the NSH User Guide) as well as several other configurations.
- **NuttX-6.13-6.16** Additional drivers and configurations were added in NuttX 6.13-6.16. Drivers include an Ethernet driver, ADC driver, DAC driver, PWM driver, CAN driver, F4 RTC driver, Quadrature Encoder, DMA, SDIO with DMA (these should all be compatible with the STM32 F2 family and many should also be compatible with the STM32 F1 family as well).
- **NuttX-6.16** The NuttX 6.16 release also includes and logic for saving/restoring F4 FPU registers in context switches. Networking intensions include support for Telnet NSH sessions and new configurations for DHPCD and the networking test (nettest).
- **NuttX-6.17** The USB OTG device controller driver, and LCD driver and a function I2C driver were added in NuttX 6.17.
- **NuttX-6.18** STM32 IWDG and WWDG watchdog timer drivers were added in NuttX 6.18 (should be compatible with F1 and F2). An LCD driver and a touchscreen driver for the STM3240G-EVAL based on the STMPE811 I/O expander were also added in NuttX 6.18.
- **NuttX-6.21** A USB OTG host controller driver was added in NuttX 6.21.
- **NuttX-7.3** Support for the Olimex STM32 H405 board was added in NuttX-7.3.
- **NuttX-7.14** Support for the Olimex STM32 H407 board was added in NuttX-7.14.
- **NuttX-7.17** Support for the Olimex STM32 E407 board was added in NuttX-7.17.
- **NuttX-7.19** Support for the Olimex STM32 P407 board was added in NuttX-7.19.
- **NuttX-7.21** Support for the MikroElektronika Clicker2 for STM32 (STM32 P405) board was added in NuttX-7.21.
- **NuttX-7.29** Support for the OmnibusF4 architecture (STM32 P405) board was added in NuttX-7.29.

Refer to the STM3240G-EVAL board README file for further information.

STMicro STM32F4-Discovery. This port uses the STMicro STM32F4-Discovery board featuring the STM32F407VGT6 MCU. The STM32F407VGT6 is a 168MHz Cortex-M4 operation with 1Mbit Flash memory and 128kbytes. The board features:

- On-board ST-LINK/V2 for programming and debugging,
- LIS302DL, ST MEMS motion sensor, 3-axis digital output accelerometer,
- MP45DT02, ST MEMS audio sensor, omni-directional digital microphone,
- CS43L22, audio DAC with integrated class D speaker driver,
- Eight LEDs and two push-buttons,
- USB OTG FS with micro-AB connector, and
- Easy access to most MCU pins.

Support for the STM3F4DIS-BB base board was added in NuttX-7.5. This includes support for the serial communications via the on-board DB-9 connector, Networking, and the microSD card slot.

Refer to the STMicro web site for further information about this board and to

STATUS: The basic port for the STM32F4-Discovery was contributed by Mike Smith and was first released in NuttX-6.14. All drivers listed for the STM3240G-EVAL are usable on this platform as well. Refer to the NuttX board README file for further information.

MikroElektronika Mikromedia for STM32F4. This is another board supported by NuttX that uses the same STM32F407VGT6 MCU as does the STM32F4-Discovery board. This board, however, has very different on-board peripherals than does the STM32F4-Discovery:

- TFT display with touch panel,
- VS1053 stereo audio codec with headphone jack,
- SD card slot,
- Serial FLASH memory,
- USB OTG FS with micro-AB connector, and
- Battery connect and batter charger circuit.

See the Mikroelektronika website for more information about this board and the NuttX board README file for further information about the NuttX port.

STATUS: The basic port for the Mikromedia STM32 M4 was contributed by Ken Petit and was first released in NuttX-6.128. All drivers for the STM32 F4 family may be used with this board as well.

Olimex STM32 H405. Support for the Olimex STM32 H405 development board was contributed by Martin Lederhilger and appeared in NuttX-7.3. See the NuttX board README file for further information about the NuttX port.

Olimex STM32 H407. Support for the Olimex STM32 H407 development board was contributed by Neil Hancock and appeared in NuttX-7.14. See the NuttX board README file for further information about the NuttX port.

Olimex STM32 E407. Support for the Olimex STM32 E407 development board was contributed by Mateusz Szafoni and appeared in NuttX-7.17. Networking configurations were added in NuttX-7.18. See the NuttX board README file for further information about the NuttX port.

Olimex STM32 P407. Support for the Olimex STM32 P407 development board appeared in NuttX-7.19. See the NuttX board README file for further information about the NuttX port.

MikroElektronika Clicker2 for STM32. This is yet another board supported by NuttX that uses the same STM32F407VGT6 MCU as does the STM32F4-Discovery board. This board has been used primarily with the MRF24J40 Click board for the development of IEEE 802.15.4 MAC and 6LoWPAN support.

See the Mikroelektronika website for more information about this board and the NuttX board README file for further information about the NuttX port.

STATUS: The basic port for the Clicker2 STM32 was contributed by Anthony Merlino and was first released in NuttX-7.21. All compatible drivers for the STM32 F4 family may be used with this board as well.

OmnibusF4. Initial support for the OmnibusF4 family of flight management units was contributed by Bill Gatliff in NuttX-7.29. "OmnibusF4" is not a product name *per se*, but rather a design specification that many product vendors adhere to. The specification defines the major components, and how those components are wired into the microcontroller. *Airbot* is one such vendor. They publish a schematic. Other software that supports the OmnibusF4 family include Betaflight, iNAV, and many others. PX4 recently added support as well, also based on NuttX. No code from those resources is included in this port. The OmnibusF4 specification mandates the InvenSense MPU6000 which is included in NuttX-7.29 along with a driver for the MAX7546 OSD.

STMicro STM32 F427/437. General architectural support was provided for the F427/437 family in NuttX 6.27. Specific support includes the STM32F427I, STM32F427Z, and STM32F427V chips. This is *architecture-only* support, meaning that support for the boards with these chips is available, but not support for any publicly available boards is included. This support was contributed by Mike Smith.

The F427/437 port adds (1) additional SPI ports, (2) additional UART ports, (3) analog and digital noise filters on the I2C ports, (4) up to 2MB of flash, (5) an additional lower-power mode for the internal voltage regulator, (6) a new prescaling option for timer clock, (7) a larger FSMSC write FIFO, and (8) additional crypto modes (F437 only).

Axlotoi. In NuttX-7.31, Jason Harris contributed support for the Axloti board. That is the board for the Axoloti open source synthesizer board featuring the STM32F427IGH6 MCU The STM32F427IGH6 has a 180MHz Cortex-M4 core with 1MiB Flash memory and 256KiB of SRAM The Axloti board features:

- ADAU1961 24-bit 96 kHz stereo CODEC
- 1/4" in/out jacks for analog audio signals
- 3.5 mm jack for analog audio signals
- 8 MiB of SDRAM (Alliance Memory AS4C4M16SA)
- Serial MIDI in/out ports
- SD Card slot
- Two user LEDs and one (GPIO) push-button
- USB OTG FS with Micro-AB connector (USB device mode operation)
- USB OTG HS with Type-A connector (USB host mode operation)
- Easy access to most IO pins

Refer to Axloti website for further information about this board.

STMicro STM32 F429. Support for STMicro STM32F429I-Discovery development board featuring the STM32F429ZIT6 MCU was contributed in NuttX-6.32 by Ken Pettit. The STM32F429ZIT6 is a 180MHz Cortex-M4 operation with 2Mbit Flash memory and 256kbytes.

STATUS:

- The initial release included support from either OTG FS or OTG HS in FS mode.
- The F429 port adds support for the STM32F439 LCD and OTG HS (in FS mode).
- In Nutt-7.6, Brennan Ashton added support for concurrent OTG FS and OTG HS (still in FS mode) and Marco Krahl added support for an SPI-based LCD .
- In Nutt-7.7, Marco Krahl included support for a framebuffer based driver using the LTDC and DMA2D. Marcos's implementation included extensions to support more advance LTDC functions through an auxiliary interface.
- Support for the uVision GCC IDE added for theSTM32F429I-Discovery board in NuttX 7.16. From Kha Vo.

Refer to the STM32F429I-Discovery board README file for further information.

STMicro STM32 F433. Architecture-only support for the STM32 F433 family was contributed by Alan Carvalho de Assis in NuttX-7.22 (meaning that the parts are supported, but there is no example board supported in the system). This support was contributed by David Sidrane and made available in NuttX-7.11.

STMicro STM32 F446. Architecture-only support is available for the STM32 F446 family (meaning that the parts are supported, but there is no example board supported in the system). This support was contributed by David Sidrane and made available in NuttX-7.11.

STMicro STM32 F46xx. Architecture-only support is available for the STM32 F46xx family (meaning that the parts are supported, but there is no example board supported in the system). This support was contributed by Paul Alexander Patienc and made available in NuttX-7.15.

STMicro STM32 L475. One board in supported in this family:

- **B-L475E-IOT01A Discovery Kit**. Board support for the STMicro B-L475E-IOT01A board from ST Micro was contributed by Simon Piriou in NuttX-7.22. See the STMicro website and the board README file for further information.

This board STMicro is powered by STM32L475VG Cortex-M4 and targets IoT nodes with a choice of connectivity options including WiFi, Bluetooth LE, NFC, and sub-GHZ RF at 868 or 915 MHz, as well as a long list of various environmental sensors.

Status:

NuttX-7.22. The initial board support was released. Since this board is highly compatible with the related, more mature STM32 L4 parts, it is expected that there is a high degree of compatibility and with those part.

This board has been used extensive to develop NuttX PktRadio support for the onboard Spirit1 radio (SPSGRF-915) radio. 6LoWPAN radio communications are fully supported in point-to-point and in star topologies.

Simon Pirou also contributed support for the on-board Macronix QuadSPI FLASH in NuttX 7.22.

STMicro STM32 L476. Three boards are supported in this family:

- **Nucleo-L476RG**. Board support for the STMicro NucleoL476RG board from ST Micro was contributed by Sebastien Lorquet in NuttX-7.15. See the STMicro website and the board README file for further information.
- **STM32L476VG Discovery**. Board support for the STMicro STM32L476VG Discovery board from ST Micro was contributed by Dave in NuttX-7.15. See the STMicro website and the board README file for further information.
- **STM32L476 MDK**. Very basic support for NuttX on the Motorola Moto Z MDK was contributed by Jim Wylder in NuttX 7.18. A simple NSH configuration is available for the STM32L476 chip. See the Moto Mods Development Kit and the board README file for further information.

Status:

NuttX-7.15. Only the first initial release of support for this family is present. It includes these basics:

- RCC, clocking, Interrupts, System timer
- UART, USART, Serial Console
- GPIO, DMA, I2C, RNG, SPI

NuttX-7.16. Additional drivers were contributed:

- QSPI with DMA and memory mapped support. From Dave (ziggurat29).
- CAN contributed by Sebastien Lorquet.
- I2C made functional by Dave (ziggurat29).

NuttX-7.17. Additional drivers/features were contributed:

- Support for tickless mode.
- CAN driver enhancements.

NuttX-7.18. Additional drivers were contributed:

- Oneshot timer driver.
- Quadrature encode contributed by Sebastien Lorquet.

NuttX-7.20. Additional drivers were added:

- Serial Audio Interface (SAI).
- Power Management.
- LPTIM.
- Comparator (COMP).

NuttX-7.21. Additional drivers were added:

- Internal Watchdog (IWDG).

NuttX-7.22.

- DAC and ADC drivers were contributed by Juha Niskanen.

NuttX-7.30.

- Added USB FS device driver, CRS and HSI38 support from Juha Niskanen.

NuttX-8.2.

- Add DMA support for STM32L4+ series. From Jussi Kivilinna.
- Add support for LPTIM timers on the STM32L4 as PWM outputs. From Matias N.

Enable OTGFS for STM32L4+ series. The OTGFS peripheral on stm32l4x6 and stm32l4rxxx reference manual is exactly the same. From Jussi Kivilinna.

STMicro STM32 L4x2. Architecture support for STM32 L4x2 family was contributed by Juha Niskanen in NuttX-7.21. Support was extended for the STM32L412 and STM32L422 chips in NuttX-7.27. Two boards are currently supported.

- **Nucleo-L432KC**. Board support for the STMicro Nucleo-L432KC board from ST Micro was contributed by JSebastien Lorquet in NuttX-7.21. See the STMicro website and the board README file for further information.
- **Nucleo-L452RE**. Board support for the STMicro Nucleo-L452RE board from ST Micro was contributed by Juha Niskanen in NuttX-7.21. See the ST Micro website and the board README file for further information.

See also the status above for the STM32 L476 most of which also applies to these parts.

STMicro STM32 L496. Architecture support for STM32 L496 was contributed by Juha Niskanen along with board support for the Nucleo-L496ZG in NuttX-7.21:

- **Nucleo-L496ZG**. Board support for the STMicro Nucleo-L496ZG board from ST Micro was contributed by Juha Niskanen in NuttX-7.21. See the ST Micro website and the board README file for further information. See also the status above for the STM32 L476 most of which also applies to this part.

STMicro STM32 L4Rx. Architecture support for STM32 L4+ family was contributed by Juha Niskanen along with board support for the STM32L4R9I-Discovery in NuttX-7.26. Additional support for the STM32L4R5ZI part was added by Jussi in NuttX-8.2.

- **STM32L4R9I-Discovery**. Board support for the STMicro STM32L4R9I-Discovery board from ST Micro was contributed by Juha Niskanen in NuttX-7.26. That development board uses the STM32L4R9AI part. See the STMicro website and the board README file for further information. See also the status above for the opther STM32 L4 parts, most of which also applies to this part.

NCP LPC40xx. The LPC40xx family is very similar to the LPC17xx family except that it features a Cortex-M4F versus the LPC17xx's Cortex-M3. Architectural support for the LPC40xx family was built on top of the existing LPC17xx by jjlange in NuttX-7.31. With that architectural support came support for two boards also contributed by jjlange:

Embedded Artists LPC4088 Developer's kit. See the Embedded Artists website for further information about this board.

Embedded Artists LPC4088 Developer's kit. See the Embedded Artists website for further information about this board.

LX CPU. Pavel Pisa add support for the PiKRON LX CPU board. This board may be configured to use either the LPC4088 or the LPC1788.

Driver Status.

NuttX-7.31. No new unique drivers for the LPC40xx family are needed. Most (if not all) LPC17xx drivers should be simply used with the LPC40xx. That is an unverified assertion, however, not a proven fact.

NCP LPC43xx. Several board ports are available for this higher end, NXP Cortex-M4F part:

NXG Technologies LPC4330-Xplorer. This NuttX port is for the LPC4330-Xplorer board from NGX Technologies featuring the NXP LPC4330FET100 MCU. See the NXG website for further information about this board.

- **STATUS:** Refer to the NuttX board README file for more detailed information about this port.
- **NuttX-6.20** The basic LPC4330-Xplorer port is complete. The basic NuttShell (NSH) configuration is present and fully verified. This includes verified support for: SYSTICK system time, pin and GPIO configuration, and a serial console.

NXP/Embest LPC4357-EVB. This NuttX port is for the LPC4357-EVB from NXP/Embest featuring the NXP LPC4357FET256 MCU. The LPC4357 differs from the LPC4330 primarily in that it includes 1024KiB of on-chip NOR FLASH. See the NXP website for more detailed information about the LPC4357 and the LPC4357-EVB.

- **STATUS:** Refer to the NuttX board README file for more detailed information about this port.
- **NuttX-7.6**. The basic port is was contributed by Toby Duckworth. This port leverages from the LPC4330-Xplorer port (and, as of this writing, still requires some clean up of the technical discussion in some files). The basic NuttShell (NSH) configuration is present and has been verified. Support is generally the same as for the LPC4330-Xplorer as discussed above.

NXP LPC4370-Link2. This is the NuttX port to the NXP LPC4370-Link2 development board featuring the NXP LPC4370FET100 MCU.

- **STATUS:** Refer to the NuttX board README file for more detailed information about this port.
- **NuttX-7.12** The NXP LPC4370-Link2 port is was contributed by Lok Tep and first released in NuttX-7.12.

WaveShare LPC4337-WS. This is the NuttX port to the WaveShare LPC4337-WS development board featuring the NXP LPC4337JBD144 MCU.

- **STATUS:** Refer to the NuttX board README file for more detailed information about this port.
- **NuttX-7.14** The NXP WaveShare LPC4337-WS port is was contributed by Lok Tep and first released in NuttX-7.14.
- **NuttX-7.16** Support for the LPC4337JET100 chip was contribed by Alexander Vasiljev. Alexander also contributed an LPC43xx AES driver available in NuttX-7.16.

Driver Status.

- **NuttX-6.20** Several drivers have been copied from the related LPC17xx port but require integration into the LPC43xx: ADC, DAC, GPDMA, I2C, SPI, and SSP. The registers for these blocks are the same in both the LPC43xx and the LPC17xx and they should integrate into the LPC43xx very easily by simply adapting the clocking and pin configuration logic.

Other LPC17xx drivers were not brought into the LPC43xx port because these peripherals have been completely redesigned: CAN, Ethernet, USB device, and USB host.

So then there is no support for the following LPC43xx peripherals: SD/MMC, EMC, USB0,USB1, Ethernet, LCD, SCT, Timers 0-3, MCPWM, QEI, Alarm timer, WWDT, RTC, Event monitor, and CAN.

Some of these can be leveraged from other MCUs that appear to support the same peripheral IP:

  - The LPC43xx USB0 peripheral appears to be the same as the USB OTG peripheral for the LPC31xx. The LPC31xx USB0 device-side driver has been copied from the LPC31xx port but also integration into the LPC43xx (clocking and pin configuration). It should be possible to complete porting of this LPC31xx driver with a small porting effort.
  - The Ethernet block looks to be based on the same IP as the STM32 Ethernet and, as a result, it should be possible to leverage the NuttX STM32 Ethernet driver with a little more effort.
- **NuttX-6.21** Added support for a SPIFI block driver and for RS-485 option to the serial driver.
- **NuttX-7.17** EMC support was extended to include support SDRAM by Vytautas Lukenska.
- **NuttX-7.23** A CAN driver was contributed by Alexander Vasiljev in NuttX-7.23.
- **NuttX-7.24** RTC and Windowed Watchdog Timer (WWDT) drivers were leveraged from the LPC17 and contributed by Gintaras Drukteinis. Leveraged the LPC54xx SD/MMC to the LPC43xx. There are still remaining issues with the SD/MMC driver and it is not yet functional.

NCP LPC54xx. A port to the LPCXpresso-LPC54628 was added in NuttX-7.24. Initial configurations include: A basic NSH configuration (nsh), a networking configuration (netnsh), and three graphics configurations (nxwm, fb, and lvgl).

LPC4508. The port was verified on an LPC5408 by a NuttX user with relevant changes incorporated in NuttX-7.26.

Driver Status.

- **NuttX-7.24** The initial release for the LPC54xx in NuttX included the following drivers: UARTs, SysTick, SD/MMC, DMA, GPIO, GPIO interrupts, LEDs and buttons, LCD, WWDT, RTC, RNG, Ethernet, and SPI. The SPI driver is untested and there are known issues with the SD/MMC driver, however.
- **NuttX-7.29** Configurations were added to verify the "Per-Window Framebuffer" feature also added in NuttX-7.29.

Refer to the LPCXpresso-LPC54628 board README file for more detailed information about this port.

NXP S32K14x. Support for the S32K14x family was added in NuttX-8.1. Two boards are supported

- **S32K146EVB**. A port to the S32K146EVB was included in NuttX-8.1. The initial release supports two full-featured NSH configurations. Refer to the S32K146EVB board README file for more detailed information about this port.
- **S32K148EVB**. A port to the S32K148EVB was also provided in NuttX-8.1. The initial release supports two full-featured NSH configurations. Refer to the S32K148EVB board README file for more detailed information about this port.

Both boards featured two NSH configurations: One for execution out of FLASH and a *safe* version that executes out of SRAM and, hence, cannot lock up the system because of a bad FLASH image.

Driver Status.

- **NuttX-8.1** The initial release for the S32K14x boards in NuttX included the following verfied drivers: Basic boot up logic, clock configuration, LPUART console, Systick timer, and GPIO controls. Additional complete-but-unverified drivers were also included: GPIO interrupts, eDMA, LPSPI, LPI2C, and Ethernet (S32K148 only).

TI Stellaris LM4F120. This port uses the TI Stellaris LM4F120 LaunchPad. Jose Pablo Carballo and I are doing this port.

STATUS: As of this writing, the basic port is code complete and a fully verified configuration exists for the NuttShell NSH). The first fully functional LM4F120 LaunchPad port was released in NuttX-6.27.

TI Tiva TM4C123G. This port uses the Tiva C Series TM4C123G LaunchPad Evaluation Kit (EK-TM4C123GXL).

Refer to the EK-TM4C123GXL board README file for more detailed information about this port.

TI Tiva TM4C123H. Architectural support for the Tiva TM4C123AH6PM was contributed in NuttX-8.1 by Nathan Hartman.

STATUS:

- **NuttX-7.1**. Initial architectural support for the EK-TM4C123GXL was implemented and was released in NuttX 7.1. Basic board support the EK-TM4C123GXL was also included in that release but was not fully tested. This basic board support included a configuration for the NuttShell NSH).
- **NuttX-7.2**. The fully verified port to the EK-TM4C123GXL was provided in NuttX-7.2.
- **NuttX-7.7**. An I2C driver was added in NuttX-7.7.
- **NuttX-8.1**. Along with TM4C123AH6PM support, Nathan Hartman also reinstated and extended the Tiva Quadrature Encoder driver.

TI Tiva TM4C1294. This port uses the TI Tiva C Series TM4C1294 Connected LaunchPad (EK-TM4C1294XL).

STATUS:

- Support for the EK-TM4C1294XL was contributed by Frank Sautter and was released in NuttX 7.9. This basic board support included a configuration for the NuttShell NSH) and a configuration for testing IPv6. See drivers for the TI Tiva TM4C129X.
- FLASH and EEPROM drivers from Shirshak Sengupta were included in NuttX-7.25.

Refer to the EK-TM4C1294XL board README file for more detailed information about this port.

TI Tiva TM4C129X. This port uses the TI Tiva C Series TM4C129X Connected Development Kit (DK-TM4C129X).

STATUS:

- A mature port to the DK-TM4C129X was implemented and was released in NuttX 7.7.
- At the initial release, verified drivers were available for Ethernet interface, I2C, and timers as well as board LEDs and push buttons. Other Tiva /Stellaris drivers should port to the TM4C129X without major difficulty.
- This board supports included two configurations for the NuttShell (NSH). Both are networked enabled: One configured to support IPv4 and one configured to supported IPv6. Instructions are included in the board README file for configuring both IPv4 and IPv6 simultaneously.
- Tiva PWM and Quadrature Encoder drivers were contributed to NuttX in 7.18 by Young.

Refer to the DK-TM4C129X board README file for more detailed information about this port.

TI/SimpleLink CC13x2. Basic, unverified architectural support for the CC13x2 was added in NuttX-7.28. Fragmentary support for very similar CC26x2 family is included. This is a work in progress and, with any luck, a fully verified port will be available in NuttX-7.29. It is currently code complete (minus some ROM *DriverLib* hooks) but untested.

TI LaunchXL-CC1312R1. Basic board support for the TI LaunchXL-CC1312R1 board is in place. Board bring-up, however, cannot be done until the the basic CC13x2 architecture support is complete, hopefully in NuttX-7.29.

STATUS: The basic port is nearly code complete but unverified in NuttX-7.28; The NSH configuration was fully functional in NuttX-7.29. This effort is stalled on further radio development. The plan was to integrate the IEEE 802.15.4 radio provided by the co-resident Cortex-M0. The Cortex-M0 interface, however, is not available with an open license that would permit redistribution. Refer to the NuttX board README file for further information.

Atmel SAM4L. This port uses the Atmel SAM4L Xplained Pro development board. This board features the ATSAM4LC4C MCU running at 48MHz with 256KB of FLASH and 32KB of internal SRAM.

STATUS: As of this writing, the basic port is code complete and a fully verified configuration exists for the NuttShell NSH). The first fully functional SAM4L Xplained Pro port was released in NuttX-6.28. Support for the SAM4L Xplained modules was added in NuttX-6.29:

- Support for the SPI-based SD card on the I/O1 module.
- Driver for the LED1 segment LCD module.
- Support for the UG-2832HSWEG04 OLED on the SAM4L Xplained Pro's OLED1 module

Refer to the NuttX board README file for further information.

Memory Usage. The ATSAM4LC4C comes in a 100-pin package and has 256KB FLASH and 32KB of SRAM. Below is the current memory usage for the NSH configuration (June 9, 2013). This is *not* a minimal implementation, but a full-featured NSH configuration.

Static memory usage can be shown with `size` command:

```
$ size nuttx
   text    data     bss     dec     hex filename
  43572     122    2380   46074    b3fa nuttx
```

NuttX, the NSH application, and GCC libraries use 42.6KB of FLASH leaving 213.4B of FLASH (83.4%) free from additional application development. Static SRAM usage is about 2.3KB (<7%) and leaves 29.7KB (92.7%) available for heap at runtime.

SRAM usage at run-time can be shown with the NSH `free` command. This runtime memory usage includes the static memory usage *plus* all dynamic memory allocation for things like stacks and I/O buffers:

```
NuttShell (NSH) NuttX-6.28
nsh> free
             total       used       free    largest
Mem:         29232       5920      23312      23312
```

You can see that 22.8KB (71.1%) of the SRAM heap is still available for further application development while NSH is running.

Atmel SAM4CM. General architectural support was provided for SAM4CM family in NuttX 7.3 This was *architecture-only* support, meaning that support for the boards with these chips is available, but no support for any publicly available boards was included. The SAM4CM port should be compatible with most of the SAM3/4 drivers (like HSMCI, DMAC, etc.) but those have not be verified on hardware as of this writing. This support was contributed in part by Max Neklyudov.

Atmel SAM4CMP-DB. Support for the SAM4CMP-DB board was contributed to NuttX by Masayuki Ishikawa in NuttX-7.19. The SAM4CM is a dual-CPU part and SMP was included for the ARMv7-M and SAM3/4 families. The SAM4CMP-DB board support includes an NSH configuration that operates in an SMP configuration. Refer to the NuttX board README file for further information.

Atmel SAM4E. General architectural support was provided for the SAM4E family in NuttX 6.32. This was *architecture-only* support, meaning that support for the boards with these chips is available, but no support for any publicly available boards was included. This support was contributed in part by Mitko.

Atmel SAM4E-EK. Board support was added for the SAM4E-EK development board in NuttX 7.1. A fully functional NuttShell (NSH) configuration is available (see the NSH User Guide). That NSH configuration includes networking support and support for an AT25 Serial FLASH file system.

STATUS. A new Ethernet MAC driver has been developed and is functional in the NSH configuration. A DMA-base SPI driver is supported and has been verified with the AT25 Serial FLASH. Touchscreen and LCD support was added in NuttX-7.3, but has not been fully integrated as of this writing. The SAM4E-EK should be compatible with most of the other SAM3/4 drivers (like HSMCI, DMAC, etc.) but those have not be verified on the SAM4E-EK as of this writing. Refer to the NuttX board README file for further information.

Atmel SAM4S. There are ports to two Atmel SAM4S board:

- There is a port the Atmel SAM4S Xplained development board. This board features the ATSAM4S16 MCU running at 120MHz with 1MB of FLASH and 128KB of internal SRAM.

  STATUS: As of this writing, the basic port is code complete and a fully verified configuration exists for the NuttShell NSH). The first fully functional SAM4S Xplained port was released in NuttX-6.28. Support for the on-board 1MB SRAM was added in NuttX-6.29. An RTT driver was Bob Doiron in NuttX-7.3. Bob also added an high resolution RTC emulation using the RTT for the sub-second counter. Refer to the NuttX board README file for further information.
- There is also a port to the Atmel SAM4S Xplained *Pro* development board. This board features the ATSAM4S32C MCU running at 120MHz with 2MB of FLASH and 160KB of internal SRAM.

  STATUS: As of this writing, the basic port is code complete and a fully verified configuration exists for the NuttShell NSH). The first fully functional SAM4S Xplained Pro port was released in NuttX-7.2. This supported also added HSMCI, RTC, and watchdog and verified support for USB device. Refer to the NuttX board README file for further information.

Atmel SAM4E. General architectural support was provided for the SAM4E family in NuttX 6.32. This was *architecture-only* support, meaning that support for the boards with these chips is available, but no support for any publicly available boards was included. This support was contributed in part by Mitko.

Atmel SAM4E-EK. Board support was added for the SAM4E-EK development board in NuttX 7.1. A fully functional NuttShell (NSH) configuration is available (see the NSH User Guide). That NSH configuration includes networking support and support for an AT25 Serial FLASH file system.

STATUS. This is very much a work in progress. A new Ethernet MAC driver has been developed and is functional in the NSH configuration. A DMA-base SPI driver is supported and has been verified with the AT25 Serial FLASH. The SAM4E-EK should be compatible with most of the other SAM3/4 drivers (like HSMCI, DMAC, etc.) but those have not be verified on the SAM4E-EK as of this writing. Refer to the NuttX board README file for further information.

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU Cortex-M3 or 4 toolchain, 3) Cygwin /MSYS with Windows native GNU Cortex-M3 or M4 toolchain (CodeSourcery or devkitARM), or 4) Native Windows. A DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package.

ARM Cortex-M7.

Atmel SAMV71. This port uses Atmel SAM V71 Xplained Ultra Evaluation Kit (SAMV71-XULT). This board features the ATSAMV71Q21 Cortex-M7 microcontroller. Refer to the Atmel web site for further information about this board.

STATUS: The basic port is complete and there are several different, verified configurations available. All configurations use the NuttShell (NSH) and a serial console. The first release of the SAMV71-XULT port was available in NuttX-7.9. Support for the connect maXTouch Xplained Pro LCD as added in NuttX-7.10.

Additional drivers, with status as of 2015-04-03, include:

- PIO configuration, including PIO interrupts,
- On-board LEDs and buttons,
- DMA,
- SDRAM (not yet functional),
- UART/USART-based serial drivers, including the NuttShell serial console,
- High Speed Memory Card Interface (HSMCI) with support for the on board SD card slot,
- SPI (not fully tested),
- TWIHS/I2C, with the support for the on-board serial EEPROM,
- SSC/I2S (not fully tested),
- Ethernet MAC,
- USB device controller driver (complete, partially functional, but not well tested).
- On-board AT24 I2C EEPROM.
- On-board WM8904 Audio CODEC with CS2100-CP Fractional-N Multiplier (not yet tested).
- Support for the (optional) maXTouch Xplained Pro LCD module.

Additional Drivers added in NuttX-7.11 include:

- MCAN CAN device driver (fully verified in loopback mode only).
- SPI slave driver.

Additional Drivers added in NuttX-7.13 include:

- MPU and protected build mode support.
- Timer/Counter driver, one-shot timer, free-running timer support.
- *Tickless* mode of operation.
- QuadSPI driver.
- Support for programming on-chip FLASH.

And in NuttX-7.14:

- TRNG driver,
- WDT driver, and
- RSWDT driver.

Refer to the NuttX board README file for further information.

Atmel SAME70. This port uses Atmel SAM E70 Xplained Evaluation Kit (ATSAME70-XPLD). This board is essentially a lower cost version of the SAMV71-XULT board featuring the ATSAME70Q21 Cortex-M7 microcontroller. See the Atmel SAMV71 for supported features. Also refer to the NuttX board README file for further information.

Atmel SAMD5x/E5x. The port of NuttX to Adafruit Metro M4 development board was released with NuttX-7/26. This board is essentially a advanced version of the Adafruit Metro board based on the SAMD21, but upgraded to the SAMD51, specifically the SAMD51J19. See the Adafruit web page for additional information about the board.

A fully-function, basic NuttShell (NSH) configuration was was available in this initial NuttX-7.26 release. That initial port verifies clock configuration boot-up logic, SysTick timer, and SERCOM USART for the serial console. The NSH configuration also includes use of the Cortex-M Cache Controller (CMCC) which give the SAMD51's Cortex-M4 a performance boost.

Because of the similarity in peripherals, several drivers were brought in from the SAML21 port. Most have not been verified as of the NuttX-7.26 release. These unverfied drivers include: SPI, I2C, DMA, USB. Also refer to the NuttX board README file for further information about the current state of the port.

Nuttx-9.0 added basic support for Microchip SAME54 Xplained Pro board. An ethernet driver was also added to the SAME5x familly.

STMicro STM32 F72x/F73x. Support for the F72x/F73x family was provided by Bob Feretich in NuttX-7.23. A single board is supported in this family:

Nucleo F722ZE. This is a member of the common board support for the common Nucleo-144 boards, this one featuring the STM32F722ZE. This port was also provided by Bob Feretich in NuttX-7.23. See the board README.txt file for further information.

STATUS: See below for STM32 F7 driver availability.

STMicro STM32 F745/F746. Three boards are supported for this MCU:

1. **STM32F746G Discovery**. One port uses the STMicro STM32F746G-DISCO development board featuring the STM32F746NGH6 MCU. The STM32F746NGH6 is a 216MHz Cortex-M7 operation with 1024Kb Flash. The first release of the STM32F746G_DISCO port was available in NuttX-7.11. Refer to the STMicro web site for further information about this board.
2. **Nucleo-144 board with STM32F746ZG**. A basic port for the Nucleo-144 board with the STM32F746ZG MCU was contributed in NuttX-7.16 by Kconstantin Berezenko.

STATUS: Refer to the NuttX board README file for further information.

STM32 F7 Driver Status:

- **NuttX-7.11**. Serial driver and Ethernet driver support, along with DMA support, were available in this initial release. The STM32 F7 peripherals are very similar to some members of the STM32 F4 and additional drivers can easily be ported the F7 as discussed in this Wiki page: Porting Drivers to the STM32 F7
- **NuttX-7.17**. David Sidrane contributed PWR, RTC, BBSRAM, and DBGMCU support. Lok Tep contribed SPI, I2c, ADC, SDMMC, and USB device driver support.
- **NuttX-7.22**. Titus von Boxberg also contributed LTDC support for the onboard LCD in NuttX-7.22.
- **NuttX-7.29**. In NuttX-7.29, Valmantas Paliksa added a timer lowerhalf driver for STM32F7, ITM syslog support, a CAN driver with support for three bxCAN interfaces, and STM32F7 Quad SPI support. Support for DMA and USB OTG was added by Mateusz Szafoni in NuttX-7.29.
- **NuttX-7.30**. From Eduard Niesner contributed a PWM driver. Added UID access from Valmantas Paliksa. USB High speed driver was added for STM32F7 series by Ramtin Amin.
- **NuttX-9.0**. Added serial DMA support.

STMicro STM32 F756. Architecture-only support is available for the STM32 F756 family (meaning that the parts are supported, but there is no example board supported in the system). This support was made available in NuttX-7.11. See above for STM32 F7 driver availability.

STMicro STM32 F76xx/F77xx. Architecture support for the STM32 F76xx and F77xx families was contributed by David Sidrane in NuttX 7.17. Support is available for two boards from this family:

- **Nucleo-F767ZI**. This is a member of the Nucleo-144 board family. Support for this board was also contributed by David Sidrane in NuttX-7.17. See the board README.txt file for further information.
- **STM32F76I-DISCO**. Support for the STM32F76I-DISCO was contributed by Titus von Boxberg in NuttX-7.22. The STMicro STM32F769I-DISCO development board features the STM32F769NIH6 MCU. The STM32F769NIH6 is a 216MHz Cortex-M7 operating with 2048K Flash memory and 512Kb SRAM. The board features:

  - On-board ST-LINK/V2 for programming and debugging,
  - Mbed-enabled (mbed.org)
  - 4-inch 800x472 color LCD-TFT with capacitive touch screen
  - SAI audio codec
  - Audio line in and line out jack
  - Two ST MEMS microphones
  - SPDIF RCA input connector
  - Two pushbuttons (user and reset)
  - 512-Mbit Quad-SPI Flash memory
  - 128-Mbit SDRAM
  - Connector for microSD card
  - RF-EEPROM daughterboard connector
  - USB OTG HS with Micro-AB connectors
  - Ethernet connector compliant with IEEE-802.3-2002 and PoE

  Refer to the http://www.st.com website for further information about this board (search keyword: stm32f769i-disco). See also the board README.txt file for further information.

  STATUS: See above for STM32 F7 driver availability.

STMicro STM32 H7x3 Architecture support for the STM32 H7x3 was added through efforts of several people in NuttX-7.26. Support is available for one board from this family:

- **Nucleo-H743ZI**. This is a member of the Nucleo-144 board family. Support for this board was added in NuttX-7.26. See the board README.txt file for further information.

The basic NSH configuration is fully, thanks to the bring-up efforts of Mateusz Szafoni. This port is still a work in progress and additional drivers are being ported from the F7 family.

- **STMicro STM32H747I-DISCO**. Support for this board was added in NuttX-9.0. See the board README.txt file for further information.

This port is still a work in progress.

Driver Availability:

- **NuttX-7.27**. Add I2C and SPI support for the STM32H7. From Mateusz Szafoni.
- **NuttX-7.30**. Added support for Ethernet, SDMMC, and Timer drivers. All from Jukka Laitinen.
- **NuttX-8.1**. Added support for BBSRAM, DTCM, RTC, and UID. All from David Sidrane.
- **NuttX-8.2**. Added support for SDMMC and FLASH progmem. From David Sidrane.
- **NuttX-9.0**. Added QSPI support for the STM32H7.

NXP/Freescale i.MX RT. The initial port to the IMXRT1050-EVKB featuring the MIMXRT1052DVL6A *Crossover* MCU was included initially in NuttX-7.25. The initial port was the joint effort of Janne Rosberg, Ivan Ucherdzhiev, and myself. Ivan gets credit for the bulk of the bring-up work and for the Hyper FLASH boot logic.

Another port, this one for the IMXRT1060-EVKB featuring the MIMXRT1062DVL6A *Crossover* MCU, was added by David Sidrane in NuttX-7.27.

STATUS:

- The basic IMXRT1050-EVK port is complete and verified configurations are available. Refer to the NuttX board README file for further information.
- The basic IMXRT1060-EVK port was complete but un-verified as of NuttX-7.27 but has been fully verified since NuttX-7.27 Refer to the NuttX board README file for more current status information.
- Architecture-only support for the IMXRT1020 family was contributed in NuttX-7.30 by Dave Marples.
- The basic IMXRT1020-EVK port was complete with verified configurations in NuttX-8.2. This is again the work of Dave Marples. The initial release includes *nsh*, *netnsh*, and *usdhc* configurations. Refer to the NuttX board README file for further information.

i.MX RT Driver Status:

- **NuttX-7.25**. The initial release in NuttX-7.25 includes UART, Timer, GPIO, DMA, and Ethernet support (Ethernet support was contributed by Jake Choy).
- **NuttX-7.26**. NuttX-7.26 added RTC, SNVS, and Serial TERMIOS support.
- **NuttX-7.27**. NuttX-7.27 added LPI2C (from Ivan Ucherdzhiev) and SD card support via USDHC (from Dave Marples).
- **NuttX-7.28**. GPIO support Input daisy selection was added in NuttX-7.28 by David Sidrane
- **NuttX-7.29**. XBAR and OCOTP support was added in NuttX-7.28 by David Sidrane. LCD Framebuffer support was added by Johannes.
- **NuttX-7.31**. USB EHCI Host and USDHC drivers were added in NuttX-7.31 by Dave Marples.
- **NuttX-8.2**. An LCD drivers was added in NuttX-8.2 by Fabio Balzano.

NuttX-9.0. Added USB Device support.

Development Environments: The same basic development environment is recommended for the Cortex-M7 as for the Cortex-M4. It would be wise to use the latest GNU toolchains for this part because as of this writing (2015-02-09), support for the Cortex-M7 is a very new GCC feature.

Atmel AVR.

AVR ATMega.

SoC Robotics ATMega128. This port of NuttX to the Amber Web Server from SoC Robotics is partially completed. The Amber Web Server is based on an Atmel ATMega128.

STATUS: Work on this port has stalled due to toolchain issues. Complete, but untested code for this port appears in the NuttX 6.5 release. Refer to the NuttX board README file for further information.

LowPowerLab MoteinoMEGA. This port of NuttX to the MoteinoMEGA from LowPowerLab. The MoteinoMEGA is based on an Atmel ATMega1284P. See the LowPowerlab website and the board README file for further information.

STATUS: The basic function port support the NuttShell (NSH) was contribute by Jedi Tek'Enum and first appeared in the NuttX 7.8 release.

Arduino MEGA2560. Extension of the AVR architecture to support the ATMega2560 and specifi support for the Arduion MEGA2560 board were contributed by Dimitry Kloper and first released in NuttX-7.14.

STATUS: The basic port was released in NuttX-7.14 including a simple "Hello, World!" and OS test configurations. Extensive effort was made to the use the special capabilities of the Atmel Studio AVR compiler to retain strings in FLASH memory and so keep the SRAM memory usage to a minimum. Refer to the NuttX board README file for further information.

AVR AT90USB64x and **AT90USB6128x**.

Micropendous 3 AT90USB64x and **AT90USB6128x**. This port of NuttX to the Opendous Micropendous 3 board. The Micropendous3 is may be populated with an AT90USB646, 647, 1286, or 1287. I have only the AT90USB647 version for testing. This version have very limited memory resources: 64K of FLASH and 4K of SRAM.

STATUS: The basic port was released in NuttX-6.5. This basic port consists only of a "Hello, World!!" example that demonstrates initialization of the OS, creation of a simple task, and serial console output. Refer to the NuttX board README file for further information.

PJRC Teensy++ 2.0 AT90USB1286. This is a port of NuttX to the PJRC Teensy++ 2.0 board. This board was developed by PJRC. The Teensy++ 2.0 is based on an Atmel AT90USB1286 MCU.

STATUS: The basic port was released in NuttX-6.5. This basic port consists of a "Hello, World!!" example that demonstrates initialization of the OS, creation of a simple task, and serial console output as well as a somewhat simplified NuttShell (NSH) configuration (see the NSH User Guide).

An SPI driver and a USB device driver exist for the AT90USB as well as a USB mass storage configuration. However, this configuration is not fully debugged as of the NuttX-6.5 release. Refer to the NuttX board README file for further information.

AVR-Specific Issues. The basic AVR port is solid. The biggest issue for using AVR is its tiny SRAM memory and its Harvard architecture. Because of the Harvard architecture, constant data that resides to flash is inaccessible using "normal" memory reads and writes (only SRAM data can be accessed "normally"). Special AVR instructions are available for accessing data in FLASH, but these have not been integrated into the normal, general purpose OS.

Most NuttX test applications are console-oriented with lots of strings used for `printf()` and debug output. These strings are all stored in SRAM now due to these data accessing issues and even the smallest console-oriented applications can quickly fill a 4-8K memory. So, in order for the AVR port to be useful, one of two things would need to be done:

1. Don't use console applications that required lots of strings. The basic AVR port is solid and your typical deeply embedded application should work fine. Or,
2. Create a special version of printf that knows how to access strings that reside in FLASH (or EEPROM).

Development Environments: 1) Linux with native Linux GNU toolchain, 2) Cygwin/MSYS with Cygwin GNU toolchain, 3) Cygwin/MSYS with Windows native toolchain, or 4) Native Windows. All testing, however, has been performed using the NuttX DIY toolchain for Linux or Cygwin is provided by the NuttX buildroot package. As a result, that toolchain is recommended.

Atmel AVR32.

AV32DEV1. This port uses the www.mcuzone.com AVRDEV1 board based on the Atmel AT32UC3B0256 MCU. This port requires a special GNU avr32 toolchain available from atmel.com website. This is a windows native toolchain and so can be used only under Cygwin on Windows.

STATUS: This port is has completed all basic development, but there is more that needs to be done. All code is complete for the basic NuttX port including header files for all AT32UC3* peripherals. The untested AVR32 code was present in the 5.12 release of NuttX. Since then, the basic RTOS port has solidified:

- The port successfully passes the NuttX OS test (apps/examples/ostest).
- A NuttShell (NSH) configuration is in place (see the NSH User Guide). Testing of that configuration has been postponed (because it got bumped by the Olimex LPC1766-STK port). Current Status: I think I have a hardware problem with my serial port setup. There is a good chance that the NSH port is complete and functional, but I am not yet able to demonstrate that. At present, I get nothing coming in the serial RXD line (probably because the pins are configured wrong or I have the MAX232 connected wrong).

The basic, port was be released in NuttX-5.13. A complete port will include drivers for additional AVR32 UC3 devices -- like SPI and USB --- and will be available in a later release, time permitting. Refer to the NuttX board README file for further information.

Misoc.

Misoc LM32 Architectural Support. Architectural support for the Misoc LM32 was contributed by Ramtin Amin in NuttX 7.19

Minerva. Architectural support for the Misoc Minoerva was contributed by Ramtin Amin in NuttX 7.29.

Drivers. Driver support is basic in these initial releases: Serial, Timer, and Ethernet. "Board" support is a available for developing with Misoc LM32 under Qemu or on your custom FPGA.

OpenRISC mor1kx.

OpenRISC mor1kx Architectural Support. Architectural support for the OpenRISC mor1kx was developed by Matt Thompson Amin and released in NuttX 7.25. Currently only an mor1kx Qemu simulation is available for testing.

Freescale M68HCS12.

MC9S12NE64. Support for the MC9S12NE64 MCU and two boards are included:

- The Freescale DEMO9S12NE64 Evaluation Board, and
- The Future Electronics Group NE64 /PoE Badge board.

Both use a GNU arm-nuttx-elf toolchain* under Linux or Cygwin. The NuttX buildroot provides a properly patched GCC 3.4.4 toolchain that is highly optimized for the m9s12x family.

STATUS: Coding is complete for the MC9S12NE64 and for the NE64 Badge board. However, testing has not yet begun due to issues with BDMs, Code Warrior, and the paging in the build process. Progress is slow, but I hope to see a fully verified MC9S12NE64 port in the near future. Refer to the NuttX board README files for DEMO9S12NE64 and for the NE64 /PoE Badge for further information.

Intel 80x86.

QEMU/Bifferboard i486. This port uses the QEMU i486 and the native Linux, Cygwin, MinGW the GCC toolchain under Linux or Cygwin.

STATUS: The basic port was code-complete in NuttX-5.19 and verified in NuttX-6.0. The port was verified using the OS and NuttShell (NSH) examples under QEMU. The port is reported to be functional on the Bifferboard as well. In NuttX 7.1, Lizhuoyi contributed additional keyboard and VGA drivers. This is a great, stable starting point for anyone interested in fleshing out the x86 port! Refer to the NuttX README file for further information.

QEMU/Intel64 An x86_64 flat address port was ported in NuttX-9.0. It consists of the following feautres:

- - Runs in x86_64 long mode.
- - Configurable SSE/AVX support.
- - IRQs are managed by LAPIC(X2APIC) and IOAPIC.
- - Used TSC_DEADLINE or APIC timer for systick.
- - Pages are now maps the kernel at 4GB~, but changeable.

This kernel with ostest have been tested with

- Qemu/KVM on a Xeon 2630v4 machine.
- Bochs with broadwell_ult emulation.

Microchip PIC32MX (MIPS M4K).

PIC32MX250F128D. A port is in progress from the DTX1-4000L "Mirtoo" module from Dimitech. This module uses Microchip PIC32MX250F128D and the Dimitech DTX1-4000L EV-kit1 V2. See the Dimitech website for further information.

STATUS: The basic port is code complete. The OS test configuration is fully functional and proves that we have a basically healthy NuttX port to the Mirtoo. A configuration is available for the NuttShell (NSH). The NSH configuration includes support for a serial console and for the SST25 serial FLASH and the PGA117 amplifier/multiplexer on board the module. The NSH configuration is set up to use the NuttX wear-leveling FLASH file system (NXFFS). The PGA117, however, is not yet fully integrated to support ADC sampling. See the NSH User Guide for further information about NSH. The first verified port to the Mirtoo module was available with the NuttX 6.20 release. Refer to the NuttX board README file for further information.

PIC32MX4xx Family.

PIC32MX440F512H. This port uses the "Advanced USB Storage Demo Board," Model DB-DP11215, from Sure Electronics. This board features the Microchip PIC32MX440F512H. See the Sure website for further information about the DB-DP11215 board. (I believe that that the DB-DP11215 may be obsoleted now but replaced with the very similar, DB-DP11212. The DB-DP11212 board differs, I believe, only in its serial port configuration.)

STATUS: This NuttX port is code complete and has considerable test testing. The port for this board was completed in NuttX 6.11, but still required a few bug fixes before it will be ready for prime time. The fully verified port first appeared in NuttX 6.13. Available configurations include the NuttShell (NSH - see the NSH User Guide). An untested USB device-side driver is available in the source tree. A more complete port would include support of the USB OTG port and of the LCD display on this board. Those drivers are not yet available as of this writing. Refer to the NuttX board README file for further information.

PIC32MX460F512L. There one two board ports using this chip:

- **PIC32MX Board from PCB Logic Design Co**. This port is for the PIC32MX board from PCB Logic Design Co. and used the PIC32MX460F512L. The board is a very simple -- little more than a carrier for the PIC32 MCU plus voltage regulation, debug interface, and an OTG connector.

STATUS: The basic port is code complete and fully verified in NuttX 6.13. Available configurations include the NuttShell (NSH - see the NSH User Guide). Refer to the NuttX board README file for further information.

- **UBW32 Board from Sparkfun** This is the port to the Sparkfun UBW32 board. This port uses the original v2.5 board which is based on the Microchip PIC32MX460F512L. This older version has been replaced with this newer board. See also the UBW32 web site.

STATUS: The basic port is code complete and fully verified in NuttX 6.18. Available configurations include the NuttShell (NSH - see the NSH User Guide). USB has not yet been fully tested but on first pass appears to be functional. Refer to the NuttX board README file for further information.

PIC32MX795F512L. There one two board ports using this chip:

- **Microchip PIC32 Ethernet Starter Kit**. This port uses the Microchip PIC32 Ethernet Starter Kit (DM320004) with the Expansion I/O board. See the Microchip website for further information.

STATUS: This port was started and then shelved for some time until I received the Expansion I/O board. The basic Starter Kit (even with the Multimedia Expansion Board, MEB, DM320005)) has no serial port and most NuttX test configurations depend heavily on console output.

A verified configuration is available for the NuttShel (NSH) appeared in NuttX-6.16. Board support includes a verified USB (device-side) driver. Also included are a a verified Ethernet driver, a partially verified USB device controller driver, and an unverifed SPI driver. Refer to the NuttX board README file for further information.

- **Mikroelektronika PIC32MX7 Mulitmedia Board (MMB)**. A port has been completed for the Mikroelektronika PIC32MX7 Multimedia Board (MMB). See http://www.mikroe.com/ for further information about this board.

STATUS: A verified configuration is available for an extensive NuttShell (NSH) configuration. The NSH configuration includes: (1) Full network support, (2) Verified SPI driver, (3) SPI-based SD Card support, (4) USB device support (including configuration options for the USB mass storage device and the CDC/ACM serial class), and (5) Support for the MIO283QT2 LCD on the PIC32MX7 MMB. (6) Support for the MIO283QT9A LCD used on later boards (NuttX 7.1).

The PIC32MX7 MMB's touchscreen is connected directly to the MCU via ADC pins. A touchscreen driver has been developed using the PIC32's ADC capabilities and can be enabled in the NSH configuration. However, additional verification and tuning of this driver is required. Further display /touchscreen verification would require C++ support (for NxWidgets and NxWM). Since I there is no PIC32 C++ is the free version of the MPLAB C32 toolchain, further graphics development is stalled. Refer to the NuttX board README file for further information.

Development Environment: These ports uses either:

1. The *LITE* version of the PIC32MX toolchain available for download from the Microchip website, or
2. The Pinguino MIPS ELF toolchain available from the Pinguino website.
3. The MIPS SDE toolchain available from the Mentor Graphics website.

Microchip PIC32MZ.

PIC32MZEC Family (MIPS microAptiv). A port is in available for the PIC32MZ Embedded Connectivity (EC) Starter Kit. There are two configurations of the Microchip PIC32MZ EC Starter Kit:

1. The PIC32MZ Embedded Connectivity Starter Kit based on the PIC32MZ2048ECH144-I/PH chip (DM320006), and
2. The PIC32MZ Embedded Connectivity Starter Kit based on the PIC32MZ2048ECM144-I/PH w/Crypto Engine (DM320006-C).

See the Microchip website for further information.

This was a collaborative effort between Kristopher Tate, David Sidrane and myself. The basic port is functional and a NuttShell (NSH) configuration is available.

PIC32MZEF Family (MIPS M5150). A port is in available for the MikroElectronika Flip&Click PIC32MZ development board based on the PIC32MZ2048EFH100 MCU. This board configuration was added in NuttX-7.24 and is, for the most part, compatible with the PIC32MZEC family.

STATUS:

NuttX-7.9. The first official release was in NuttX-7.9. Many drivers port simply from the PIC32MX; others require more extensive efforts. Driver status as of (2015-03-29) is provided below:

- I/O ports include I/O port interrupts
- UART serial driver that provides the NSH console,
- Timer,
- I2C (untested),
- SPI (untested),
- On-board buttons and LEDs,
- Ethernet (code complete, but not yet functional),

NuttX-7.29. Abdelatif Guettouche contributed additional timer support including: Timer lower half driver, free-running, and one-shot timers.

NuttX-7.31. Abdelatif Guettouche contributed DMA support.

NuttX-9.0. Cache operations were implemented.

Refer to the NuttX board README file for further information.

Development Environment: Same as for the PIC32MZ.

Renesas/Hitachi SuperH.

SH-1 SH7032. This port uses the Hitachi SH-1 Low-Cost Evaluation Board (SH1_LCEVB1), US7032EVB, with a GNU ELF toolchain* under Linux or Cygwin.

STATUS: This port is available as of release 0.3.18 of NuttX. The port is basically complete and many examples run correctly. However, there are remaining instabilities that make the port un-usable. The nature of these is not understood; the behavior is that certain SH-1 instructions stop working as advertised. This could be a silicon problem, some pipeline issue that is not handled properly by the gcc 3.4.5 toolchain (which has very limit SH-1 support to begin with), or perhaps with the CMON debugger. At any rate, I have exhausted all of the energy that I am willing to put into this cool old processor for the time being. Refer to the NuttX board README file for further information.

Renesas M16C/26.

Renesas M16C/26 Microcontroller. This port uses the Renesas SKP16C26 Starter kit and the GNU M32C toolchain. The development environment is either Linux or Cygwin under WinXP.

STATUS: Initial source files released in nuttx-0.4.2. At this point, the port has not been integrated; the target cannot be built because the GNU `m16c-nuttx-elf-ld` link fails with the following message:

```
m32c-nuttx-elf-ld: BFD (GNU Binutils) 2.19 assertion fail /home/Owner/projects/nuttx/buildroot
/toolchain_build_m32c/binutils-2.19/bfd/elf32-m32c.c:482
```

Where the reference line is:

```
/* If the symbol is out of range for a 16-bit address,
   we must have allocated a plt entry.  */
BFD_ASSERT (*plt_offset != (bfd_vma) -1);
```

No workaround is known at this time. This is a show stopper for M16C. Refer to the NuttX board README file for further information.

Renesas RX65N.

Support for the Renesas RX65N family was released in NuttX with a contribution from Anjana. Two boards are supported in this initial release:

- **RSK RX65N-2MB**.
This board features the R5F565NEHDFC (176pin). Refer to the boar README file for further information.
- **GR-Rose**.
The GR-ROSE board was produced by Gadget Renesas. This board features the R5F565NEHDFP (100pin QFP). Refer to the board README file for further information.

STATUS

- **NuttX-8.2**
Basic support for the RX65N family was released by Anjana with support for two boards: The RSK RX65N-2MB and the GR-Rose.
- **NuttX-9.0** RTC driver for the RX65N was added.

RISC-V.

RISC-V Architectural Support. Basic support for the RISC-V architecture was contributed by Ken Pettit in NuttX-7.19. This support is for a custom NEXT RISC-V NR5Mxx (RV32IM). The initial release is *thin* but a great starting point for anyone interested in RISC-V development with NuttX.

GreenWaves GAP8 (RV32IM). Basic support GreenWaves GAP8 *gapuino* board was added by hhuysqt in NuttX-7.27. The GAP8 is a 1+8-core DSP-like RISC-V MCU. The GAP8 features a RI5CY core called Fabric Controller(FC), and a cluster of 8 RI5CY cores that runs at a bit slower speed. The GAP8 is an implementation of the opensource PULP platform, a Parallel-Ultra-Low-Power design.

- Sipeed Maix bit
Initial support for the Sipeed Maix bit board was added in Nuttx-9.0.

Litex ARTY_A7. Support for the Digilent ARTY_A7 board along with CPU VexRiscV SOC were added in NuttX-9.0.

ESP32 (Dual Xtensa LX6).

Xtensa LX6 ESP32 Architectural Support. Basic architectural support for Xtensa LX6 processors and the port for the Espressif ESP32 were added in NuttX-7.19. The basic ESP32 port is function in both single CPU and dual CPU SMP configurations.

Espressif ESP32 Core v2 Board The NuttX release includes support for Espressif ESP32 Core v2 board. There is an NSH configuration for each CPU configuration and an OS test configuration for verificatin of the port.

STATUS. ESP32 support in NuttX-7.19 is functional, but very preliminary. There is little yet in the way of device driver support. Outstanding issues include missing clock configuration logic, missing partition tables to support correct configuration from FLASH, and some serial driver pin configuration issues. The configuration is usable despite these limitations. Refer to the NuttX board README file for further information.

Zilog ZNEO Z16F.

- **Zilog z16f2800100zcog development kit**. This port use the Zilog z16f2800100zcog development kit and the Zilog ZDS-II Windows command line tools. The development environment is either Windows native or Cygwin under Windows.

STATUS: The initial release of support for the z16f was made available in NuttX version 0.3.7. A working NuttShell (NSH) configuration as added in NuttX-6.33 (although a patch is required to work around an issue with a ZDS-II 5.0.1 tool problem). An ESPI driver was added in NuttX-7.2. Refer to the NuttX board README file for further information.

Zilog eZ80 Acclaim!.

Zilog eZ80Acclaim! Microcontroller. There are four eZ80Acclaim! ports:

- The ZiLOG ez80f0910200kitg development kit.
- The ZiLOG ez80f0910200zcog-d development kit.
- The MakerLisp CPU board.
- The Z20x DIY computing system.

All three boards are based on the eZ80F091 part and all use the Zilog ZDS-II Windows command line tools. The development environment is either Windows native or Cygwin or MSYS2 under Windows.

NuttX-0.4.x: Integration and testing of NuttX on the ZiLOG ez80f0910200zcog-d was completed. The first integrated version was released in NuttX version 0.4.2 (with important early bugfixes in 0.4.3 and 0.4.4). As of this writing, that port provides basic board support with a serial console, SPI, and eZ80F91 EMAC driver. Refer to the NuttX board README files for the ez80f0910200kitg and ez80f910200zcofile for further information.

NuttX-7.31: Support for the MakerLisp board as well as an RTC driver and an improved SPI driver were included. The MakerLisp machine is a portable, modular computer system, designed to recapture the feel of classic computing, with modern hardware.

The machine centers on a 2" x 3.5" business card-sized CPU, which can be used stand-alone, or plugged in to a 2" x 8" main board, for expansion into a full computer system. A laser-cut wood enclosure holds a small keyboard, an LCD monitor, the circuit boards, and a prototyping area with a breadboard for electronics experimentation and development.

The eZ80 running at 50 MHz, with up to 16 Mb of zero-wait state RAM. A VGA display adapter provides an IBM PC-like color text-mode display; A USB keyboard adapter provides for standard keyboard input. Data storage and interchange is accomplished by a SPI-based micro-SD card.

NuttX-9.0: Support for the Z20X board was added. The port includes: - Initial support to have a functional nsh. - Support for SPI and W25 FLASH. - A bootloader capable of writing code to the W25 FLASH and load it to the SRAM to be executed.

Zilog Z8Encore!.

Zilog Z8Encore! Microcontroller. This port uses the either:

- Zilog z8encore000zco development kit, Z8F6403 part, or
- Zilog z8f64200100kit development kit, Z8F6423 part

and the Zilog ZDS-II Windows command line tools. The development environment is either Windows native or Cygwin under Windows.

STATUS: This release has been verified only on the ZiLOG ZDS-II Z8Encore! chip simulation as of nuttx-0.3.9. Refer to the NuttX board README files for the z8encore000zco and for thez8f64200100kit for further information.

Zilog Z180.

P112. The P112 is a hobbyist single board computer based on a 16MHz Z80182 with up to 1MB of memory, serial, parallel and diskette IO, and realtime clock, in a 3.5-inch drive form factor. The P112 computer originated as a commercial product of "D-X Designs Pty Ltd"[of Australia.

Dave Brooks was successfully funded through Kickstarter for and another run of P112 boards in November of 2012. In addition Terry Gulczynski makes additional P112 derivative hobbyist home brew computers.

STATUS: Most of the NuttX is in port for both the Z80182 and for the P112 board. Boards from Kickstarter project will not be available, however, until the third quarter of 2013. So it will be some time before this port is verified on hardware. Refer to the NuttX board README file for further information.

Zilog Z80.

Z80 Instruction Set Simulator. This port uses the SDCC toolchain under Linux or Cygwin (verified using version 2.6.0). This port has been verified using only a Z80 instruction simulator called z80sim.

STATUS: This port is complete and stable to the extent that it can be tested using an instruction set simulator. Refer to the NuttX board README file for further information.

XTRS: TRS-80 Model I/III/4/4P Emulator for Unix. A very similar Z80 port is available for XTRS, the TRS-80 Model I/III/4/4P Emulator for Unix. That port also uses the SDCC toolchain under Linux or Cygwin (verified using version 2.6.0).

STATUS: Basically the same as for the Z80 instruction set simulator. This port was contributed by Jacques Pelletier. Refer to the NuttX board README file for further information.

NOTE: This port was removed from the NuttX source tree on 2017-11-24. It was removed because (1) it is unfinished, unverified, and unsupported, and (2) the TRS-80 simulation is a sub-optimal platform.i That platform includes a 16-bit ROM image and only a 48Kb RAM space for NuttX. The removed board support is still available in the `Obsoleted` repository if anyone would ever like to resurrect it.

> ** A highly modified buildroot is available that may be used to build a NuttX-compatible ELF toolchain under Linux or Cygwin. Configurations are available in that buildroot to support ARM, Cortex-M3, avr, m68k, m68hc11, m68hc12, m9s12, blackfin, m32c, h8, and SuperH ports.*

Development Environments

Linux + GNU `make` + GCC/binutils for Linux

The is the most natural development environment for NuttX. Any version of the GCC/binutils toolchain may be used. There is a highly modified buildroot available for download from the NuttX Bitbucket.org page. This download may be used to build a NuttX-compatible ELF toolchain under Linux or Cygwin. That toolchain will support ARM, m68k, m68hc11, m68hc12, and SuperH ports. The buildroot GIT may be accessed in the NuttX buildroot GIT.

Linux + GNU `make` + SDCC for Linux

Also very usable is the Linux environment using the SDCC compiler. The SDCC compiler provides support for the 8051/2, z80, hc08, and other microcontrollers. The SDCC-based logic is less well exercised and you will likely find some compilation issues if you use parts of NuttX with SDCC that have not been well-tested.

Windows with Cygwin + GNU `make` + GCC/binutils (custom built under Cygwin)

This combination works well too. It works just as well as the native Linux environment except that compilation and build times are a little longer. The custom NuttX buildroot referenced above may be build in the Cygwin environment as well.

Windows with Cygwin + GNU `make` + SDCC (custom built under Cygwin)

I have never tried this combination, but it would probably work just fine.

Windows with Cygwin + GNU `make` + Windows Native Toolchain

This is a tougher environment. In this case, the Windows native toolchain is unaware of the Cygwin *sandbox* and, instead, operates in the native Windows environment. The primary difficulties with this are:

- **Paths**. Full paths for the native toolchain must follow Windows standards. For example, the path `/home/my\ name/nuttx/include` my have to be converted to something like `'C:\cygwin\home\my name\nuttx\include'` to be usable by the toolchain.

  Fortunately, this conversion is done simply using the `cygpath` utility.

- **Symbolic Links** NuttX depends on symbolic links to install platform-specific directories in the build system. On Linux, true symbolic links are used. On Cygwin, emulated symbolic links are used. Unfortunately, for native Windows applications that operate outside of the Cygwin *sandbox*, these symbolic links cannot be used.

  The NuttX make system works around this limitation by copying the platform specific directories in place. These copied directories make work a little more complex, but otherwise work well.

  NOTE: In this environment, it should be possible to use the NTFS `mklink` command to create links. This should only require a minor modification to the build scripts (see `tools/copydir.sh` script).

- **Dependencies** NuttX uses the GCC compiler's `-M` option to generate make dependencies. These dependencies are retained in files called `Make.deps` throughout the system. For compilers other than GCC, there is no support for making dependencies in this way.

  NOTE: dependencies may be suppressed by setting the make variable `MKDEPS` to point to the do-nothing dependency script, `tools/mknulldeps.sh`.

Supported Windows Native Toolchains. At present, the following Windows native toolchains are in use:

1. GCC built for Windows (such as CodeSourcery, Atollic, devkitARM, etc.),
2. SDCC built for Windows,
3. the ZiLOG XDS-II toolchain for Z16F, z8Encore, and eZ80Acclaim parts.

Windows Native (`CMD.exe`) + GNUWin32 (including GNU `make`) + MinGW Host GCC compiler + Windows Native Toolchain

Build support has been added to support building natively in a Windows console rather than in a POSIX-like environment.

This build:

1. Uses all Windows style paths
2. Uses primarily Windows batch commands from cmd.exe, with
3. A few extensions from GNUWin32

This capability first appeared in NuttX-6.24 and should still be considered a work in progress because: (1) it has not been verfied on all targets and tools, and (2) still lacks some of the creature-comforts of the more mature environments. The windows native build logic initiated if `CONFIG_WINDOWS_NATIVE=y` is defined in the NuttX configuration file:

At present, this build environment also requires:

- **Windows Console**. The build must be performed in a Windows console window. This may be using the standard `CMD.exe` terminal that comes with Windows. I prefer the ConEmu terminal which can be downloaded from: http://code.google.com/p/conemu-maximus5/
- **GNUWin32**. The build still relies on some Unix-like commands. I usethe GNUWin32 tools that can be downloaded from http://gnuwin32.sourceforge.net/. See the top-level `nuttx/README.txt` file for some download, build, and installation notes.

- **MinGW-GCC**. MinGW-GCC is used to compiler the C tools in the `nuttx/tools` directory that are needed by the build. MinGW-GCC can be downloaded from http://www.mingw.org/. If you are using GNUWin32, then it is recommended that you not install the optional MSYS components as there may be conflicts.

Wine + GNU make + Windows Native Toolchain

I've never tried this one, but I off the following reported by an ez80 user using the ZiLOG ZDS-II Windows-native toolchain:

> *"I've installed ZDS-II 5.1.1 (IDE for ez80-based boards) on wine (windows emulator for UNIX) and to my surprise, not many changes were needed to make GIT snapshot of NuttX buildable... I've tried nsh profile and build process completed successfully. One remark is necessary: NuttX makefiles for ez80 are referencing `cygpath` utility. Wine provides similar thing called `winepath` which is compatible and offers compatible syntax. To use that, `winepath` (which itself is a shell script) has to be copied as `cygpath` somewhere in `$PATH`, and edited as in following patch:*
>
> ```
> # diff -u `which winepath` `which cygpath`
> --- /usr/bin/winepath 2011-05-02 16:00:40.000000000 +0200
> +++ /usr/bin/cygpath 2011-06-22 20:57:27.199351255 +0200
> @@ -20,7 +20,7 @@
> #
>
> # determine the app Winelib library name
> -appname=`basename "$0" .exe`.exe
> +appname=winepath.exe
>
> # first try explicit WINELOADER
> if [ -x "$WINELOADER" ]; then exec "$WINELOADER" "$appname" "$@"; fi
> ```
>
> *"Better solution would be replacing all `cygpath` references in `Makefiles` with `$(CONVPATH)` (or `${CONVPATH}` in shell scripts) and setting `CONVPATH` to `cygpath` or `winepath` regarding to currently used environment.*

Other Environments?

Environment Dependencies. The primary environmental dependency of NuttX are (1) GNU make, (2) bash scripting, and (3) Linux utilities (such as cat, sed, etc.). If you have other platforms that support GNU make or make utilities that are compatible with GNU make, then it is very likely that NuttX would work in that environment as well (with some porting effort). If GNU make is not supported, then some significant modification of the Make system would be required.

MSYS. I have not used MSYS but what I gather from talking with NuttX users is that MSYS can be used as an alternative to Cygwin in any of the above Cygwin environments. This is not surprising since MSYS is based on an older version of Cygwin (cygwin-1.3). MSYS has been modified, however, to interoperate in the Windows environment better than Cygwin and that may be of value to some users.

MSYS, however, cannot be used with the native Windows NuttX build because it will invoke the MSYS bash shell instead of the `CMD.exe` shell. Use GNUWin32 in the native Windows build environment.

Licensing

NuttX is available under the highly permissive BSD license. Other than some fine print that you agree to respect the copyright you should feel absolutely free to use NuttX in any environment and without any concern for jeopardizing any proprietary software that you may link with it.

Bugs, Issues, *Things-To-Do*

The current list of NuttX *Things-To-Do* in GIT here.

Other Documentation

 Getting Started

 User Guide

 Porting Guide

 Configuration Variables[1]

 NuttShell (NSH)

 NuttX Binary Loader

 NXFLAT Binary Format

 NX Graphics Subsystem

 NxWidgets

 Demand Paging

 NuttX README Files

 NuttX To-Do List

 USB Device Driver Tracing

[1] *This configuration variable document is auto-generated using the kconfig2html tool That tool analyzes the NuttX `Kconfig` files and generates the HTML document. As a consequence, this file may not be present at any given time but can be regenerated following the instructions in `tools` directory README file.*

Trademarks

- NuttX is a registered trademark of Gregory Nutt.
- ARM, ARM7 ARM7TDMI, ARM9, ARM920T, ARM926EJS, Cortex-M3 are trademarks of Advanced RISC Machines, Limited.
- Beaglebone is a trademark of GHI.
- BSD is a trademark of the University of California, Berkeley, USA.
- Cygwin is a trademark of Red Hat, Incorporated.
- Linux is a registered trademark of Linus Torvalds.
- Eagle-100 is a trademark of Micromint USA, LLC.
- EnergyLite is a trademark of STMicroelectronics.
- EFM32 is a trademark of Silicon Laboratories, Inc.
- LPC2148 is a trademark of NXP Semiconductors.
- POSIX is a trademark of the Institute of Electrical and Electronic Engineers, Inc.
- RISC-V is a registration pending trade mark of the RISC-V Foundation.
- Sitara is a trademark of Texas Instruments Incorporated.
- TI is a tradename of Texas Instruments Incorporated.
- Tiva is a trademark of Texas Instruments Incorporated.
- UNIX is a registered trademark of The Open Group.
- VxWorks is a registered trademark of Wind River Systems, Incorporated.
- ZDS, ZNEO, Z16F, Z80, and Zilog are a registered trademark of Zilog, Inc.

NOTE: NuttX is *not* licensed to use the POSIX trademark. NuttX uses the POSIX standard as a development guideline only.