



## How to build Linux kernel user space tools



---

## Contents

---

---



---

A quality version of this page, approved on 22 April 2020, was based off this revision.

## Contents

1 Article purpose .....	4
2 Introduction .....	5
3 Installing the trace and debug tool on your target board .....	6
3.1 Using the STM32MPU Embedded Software distribution .....	6
3.1.1 Developer Package .....	6
3.1.2 Distribution Package .....	7
4 References .....	8



## 1 Article purpose

---

This article provides the basic information needed to build the user space tools available on the Linux<sup>®</sup> kernel.



## 2 Introduction

---

The Linux kernel provides some user-space tools that are available in the tools directory <sup>[1]</sup> of the source tree.

These tools are not compiled by default when compiling the Linux kernel for the target board. They can be compiled independently, depending on the user's needs.



## 3 Installing the trace and debug tool on your target board

### 3.1 Using the STM32MPU Embedded Software distribution

#### 3.1.1 Developer Package

Prerequisites, please ensure:

- the SDK is installed
- the SDK is started up
- the Linux kernel is installed

The available user space tools can be listed by using the following commands in the Linux kernel source root path:

```
PC $> cd <Linux_kernel_source_path>
PC $> make tools/help O="<Linux_kernel_build_dir>" (optional)
Possible targets:

acpi           - ACPI tools
cgroup         - cgroup tools
cpupower       - a tool for all things x86 CPU power
firewire       - the userspace part of nosy, an IEEE-1394 traffic sniffer
freefall       - laptop accelerometer program for disk protection
gpio           - GPIO tools
hv             - tools used when in Hyper-V clients
iio            - IIO tools
kvm_stat       - top-like utility for displaying kvm statistics
leds           - LEDs tools
libblockdep    - user-space wrapper for kernel locking-validator
bpf            - misc BPF tools
perf           - Linux performance measurement and analysis tool
selftests     - various kernel selftests
spi            - spi tools
objtool        - an ELF object analysis tool
tmon           - thermal monitoring and tuning tool
turbostat      - Intel CPU idle stats and freq reporting tool
usb            - USB testing tools
virtio         - vhost test module
vm             - misc vm tools
wmi            - WMI interface examples
x86_energy_perf_policy - Intel energy policy tool
```

*Note: some tools are made for specific platforms (ARM, x86, RISC, and so on), so cannot be used on STM32MPU systems*

The following basic steps must be done :

- Compiling the application:
  - Refer to <Linux kernel installation directory>/README.HOW\_TO.txt helper file to know how to compile (the latest version of this helper file is also available in GitHub: README.HOW\_TO.txt ).
  - Ensure at least that the kernel configuration file is generated (.config) (information available in README.HOW\_TO file previously mentioned)
  - Compile the expected **tool (i.e. iio, spi...)**

```
PC $> cd <Linux_kernel_source_path>/tools
PC $> make <tool> [O=<Linux_kernel_build_dir>]
```



---

Note: The 'O' option can be used to specify the output directory

- Deploying the application on a board:
  - The binary is generated in the directory path `<Linux_kernel_build_dir>/<tool>`
  - Push it onto the board.

```
PC $> scp <tool_binary> root@<board_ip_address>: /<dest_path>
```

PS: please ensure that the `<dest_path>` is known in the `$PATH` to execute the tool binary from anywhere on the target board.

### 3.1.2 Distribution Package

There is currently no recipe to build the Linux kernel user space tools, so the [Developer package](#) has to be used.



---

## 4 References

---

- /tools

Linux® is a registered trademark of Linus Torvalds.

Central processing unit

General-Purpose Input/Output (A realization of open ended transmission between devices on an embedded level. These pins available on a processor can be programmed to be used to either accept input or provide output to external devices depending on user desires and applications requirements.)

Industrial I/O Linux® subsystem

Executable and linkable file