Ubuntu* with Kernel Overlay on Arrow Lake - U/H for Edge Platforms

Get Started Guide (MR2 Release)

September 2025

Intel Confidential

Document Number: 828853-3.1

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visiting the Intel Resource and Documentation Center.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No product or component can be absolutely secure. Check with your system manufacturer or retailer or learn more at intel.com.

No product or component can be absolutely secure.

Contents

Figures

Tables

Revision History

Date	Revision	Description
September 2025	3.1	Updated case-sensitive issue failed to download kernel overlay from download.01.org in *Section 3.2.3.2 Install Kernel Overlay into Ubuntu* OS*.
September 2025	3.0	MR2 release. • Added *Section 3.2.2 Auto Script Installation*. • Updated *Section 3.2.3 Manual Setup with Step-by-Step Guide*. • Removed *Section 3.3.3 Install Intel® In-band Manageability*.
June 2025	2.0	MR1 release.
April 2025	1.1	• Updated *Section 2.0 Overview and Prerequisites*. • Updated link to download Ubuntu* Desktop in *Section 3.1.1 Download the Ubuntu* Image*. • Updated the content in *Section 3.3.1 Install DRTU Tools for Wi-Fi/Bluetooth®*.
January 2025	1.0	PV release.
September 2024	0.8	Initial release.

§

1.0 Introduction

This document provides instructions on using the Intel-released kernel overlay on the Ubuntu* OS for the Arrow Lake – U/H for Edge Customer Reference Board (CRB). This requires downloading an Ubuntu* image, preparing the bootable image with the USB flash drive to install in the storage on the CRB, and finally switching to the validated kernel version.

You should review the release information before proceeding with this Get Started Guide. For release information, notes, and references, refer to the following documents:

Ubuntu with Kernel Overlay on Arrow Lake – U/H for Edge Platforms - Release Notes (Document Number: 828854)

2.0 Overview and Prerequisites

Kernel overlay is a reference implementation for testing hardware feature enablement. This document covers the step-by-step instructions for switching to the Intel-released kernel overlay on top of the Ubuntu* image, which will be downloaded from the Canonical website.

- Kernel overlay should only be used for platform feature evaluation prior to the platform's launch and NOT for production or deployment with commercial Linux* distribution.
- Kernel updates will be upstreamed to the latest mainline kernel, while user space components will be upstreamed to respective open-source GitHub* projects.
- The version of the kernel source code, binary and the corresponding open-source software components suggested for use in this release are only for reference purposes. If you decide to use them, you must integrate the latest functional and/or security updates when they are available from the open-source community.

For certification and support on production and deployment, engage your respective Linux* distribution vendor.

2.1 Requirement

This section listed the hardware required to get started with Ubuntu with kernel overlay.

- **Host system**: Laptop or PC with Ubuntu* OS. This system will be used to download the Ubuntu* image from the Canonical website, prepare a bootable Ubuntu* image on the USB flash drive, and build the kernel Debian package from the source code.
- **USB flash drive**: 128 GB minimum (to prepare the bootable Ubuntu* image).
- **Arrow Lake - U/H CRB**
  - Contains the latest supported Arrow Lake - U/H silicon
  - Contains a minimum of 500 GB storage
  - Flashed with latest IFWI
- **High-speed network connectivity**
- **USB-to-Ethernet Adapter**: The default Ubuntu* image may not support Arrow Lake - U/H features. Hence, the on-board Ethernet interface may not work. The adapter is required at the initial stage for downloading the Arrow Lake - U/H supported kernel.

2.2 Kernel Source Code Links

Refer to the links below for the kernel source code:

LTS Kernel v6.12.36:

https://github.com/intel/linux-intel-lts/tree/lts-v6.12.36-linux-250711T071314Z

LTS Kernel Overlay v6.12.36:

https://github.com/intel/linux-kernel-overlay/tree/lts-overlay-v6.12.36-ubuntu-250711T071314Z

2.3 Build Kernel Debian Package from Source Code

There is an alternative option to install kernel overlay in the Getting Started with Ubuntu* with Kernel Overlay section if you do not wish to build the kernel Debian package at this moment.

This section describes the steps to build the kernel Debian package from the source code.

2.3.1 Install Dependencies

Follow the following steps to install core dependencies for the first time if they are unavailable in your system.

```
sudo apt install quilt libssl-dev kernel-wedge liblz4-tool libelf-dev
flex bison git
```

2.3.2 Build Kernel Debian Packages

Follow the following steps to build the Debian package for the mainline tracking kernel.

Step 1: Clone the kernel source code and check out to the stated tag.

```
sudo git clone https://github.com/intel/linux-kernel-overlay.git

cd linux-kernel-overlay

sudo git checkout lts-overlay-v6.12.36-ubuntu-250711T071314Z
```

Step 2: Trigger kernel build

```
sudo ./build.sh -r no -t lts-overlay-v6.12.36-ubuntu-250711T071314Z -b
1000 -c lts
```

Once the build is successful, the following Debian packages will be generated under the **linux-kernel-overlay** directory:

- linux-headers-6.12*_amd64.deb
- linux-image-6.12*_amd64.deb
- linux-image-6.12*_dbg_*amd64.deb
- linux-libc-dev_6.12*_amd64.deb

3.0 Getting Started with Ubuntu* with Kernel Overlay

This section describes the steps to obtain the Ubuntu* image from the Canonical website, switch to the kernel overlay (kernel that supports Arrow Lake - U/H for Edge Platforms), and install user space components, tools, and proprietary packages.

Refer to the *Ubuntu with Kernel Overlay on Arrow Lake - U/H for Edge Platforms - Release Notes* (Document Number: 828854) for further information on platform features.

3.1 Getting Started with Ubuntu* Image

This section describes the steps to download ISO image from Ubuntu* website, create bootable disks and install the Ubuntu* image into Arrow Lake - U/H CRB.

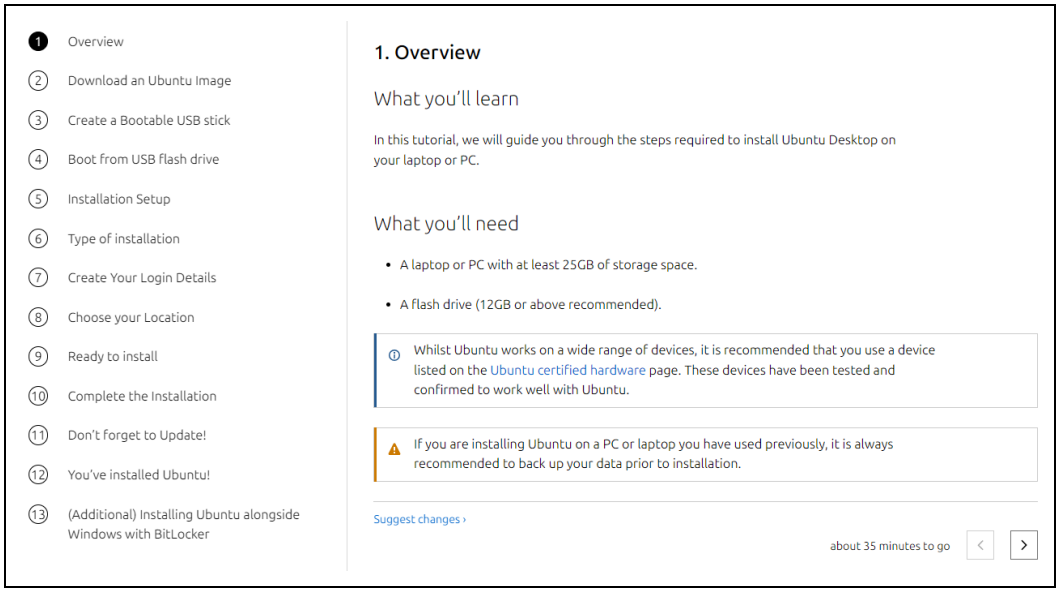
3.1.1 Download the Ubuntu* Image

Download the latest Ubuntu* Desktop 24.04 from the Ubuntu website.

Once the download has been completed, an iso image will be available in the download folder of your host system.

Open the web link by using any web browser, and you will land on a page as shown in Figure 1. The remaining steps in the next few sections are from the webpage.

Figure 1. Web Page to Install Ubuntu* Desktop



3.1.2 Create a Bootable USB Stick

Insert a USB flash drive into your host system. On the web page, click on the 3 Create a Bootable USB Stick and follow the steps on the page to create a bootable USB stick.

Once the installation has been completed, safely remove the USB from the host system.

3.1.3 Boot up CRB from the USB Flash Drive

Insert the USB flash drive into the Arrow Lake - U/H CRB. On the web page, click on the 4 Boot from USB flash drive and follow the steps to boot the system from the USB flash drive.

3.1.4 Complete the Installation Steps

On the webpage, click on the 5 Installation Setup and follow the steps on the page. Follow the remaining steps until 10 Complete the Installation to complete the installation.

3.2 Setting Up User Space and Kernel Overlay

This section describes the steps required for setting up and installing user space and kernel overlay that were validated on Arrow Lake - U/H in this release.

User space overlay and source code are available at 01.org.

3.2.1 Proxy Setup

Internet connection is needed for subsequent sections. On-board ethernet on CRB may fully function until the supported kernel version is installed in the OS. USB-to-Ethernet Adapter is recommended at this step.

If the CRB is connected to a network under a corporate firewall, set up the proxy at this step to get an internet connection. Contact your IT department for the necessary settings.

If the CRB is not connected to a network under a corporate firewall, go to the next section.

3.2.2 Auto Script Installation

This section provides steps to install the user space and kernel by using a script, which automates the installation process and reduces the need for manual intervention. However, if you wish to go through the steps with more clarity and flexibility, you can proceed to the Manual Setup with Step-by-Step Guide section.

Download and unzip the **installer.zip** file from Software Kit: 831484. Then, follow the example below to run the script for the kernel overlay.

```
sudo ./installer.sh UBUNTU_NOBLE ARL lts-overlay-v6.12.36-ubuntu-
250711T071314Z default
```

Once the installation is successful, you are ready to validate the platform. Following the Installing Sample Applications or Proprietary Tools section, you can optionally install sample applications or proprietary tools.

Note: If the auto script fails or errors out, recommend using the original manual method to configure and install the kernel overlay.

3.2.3 Manual Setup with Step-by-Step Guide

This section describes the step-by-step guide to install the user space and kernel overlay with more clarity and flexibility.

3.2.3.1 Configure and Install User Space Overlay

Step 1: Get the latest apt from Ubuntu* archive before setting up PPA from 01.org

```
sudo apt update

sudo apt upgrade
```

Step 2: Create a file at /etc/apt/sources.list.d/intel-arl.list with the content to set PPA from 01.org:

```
sudo vi /etc/apt/sources.list.d/intel-arl.list
```

Add the following content, and then save and exit.

```
deb https://download.01.org/intel-linux-overlay/ubuntu noble main non-
free multimedia kernels
deb-src https://download.01.org/intel-linux-overlay/ubuntu noble main
non-free multimedia kernels
```

Step 3: Download the GPG key to /etc/apt/trusted.gpg.d and rename it to arl.gpg

```
sudo wget https://download.01.org/intel-linux-
overlay/ubuntu/E6FA98203588250569758E97D176E3162086EE4C.gpg -O
/etc/apt/trusted.gpg.d/arl.gpg
```

Step 4: Set the preferred list.

```
sudo vi /etc/apt/preferences.d/intel-arl
```

Add the following content, and then save and exit.

```
Package: *
Pin: release o=intel-iot-linux-overlay-noble
Pin-Priority: 2000
```

Step 5: Download the package information from all configured sources and continue to install the essential tools. Here is a list of recommended tools that may be convenient for your testing later. You may choose to install the tool now or later when needed. You may also exclude tools that may not be required in your application.
sudo apt update

```
export DEBIAN_FRONTEND=noninteractive
sudo apt install vim ocl-icd-libopencl1 curl openssh-server net-tools
gir1.2-gst-plugins-bad-1.0 gir1.2-gst-plugins-base-1.0 gir1.2-
gstreamer-1.0 gir1.2-gst-rtsp-server-1.0 gstreamer1.0-alsa
gstreamer1.0-gl gstreamer1.0-gtk3 gstreamer1.0-opencv gstreamer1.0-
plugins-bad gstreamer1.0-plugins-bad-apps gstreamer1.0-plugins-base
gstreamer1.0-plugins-base-apps gstreamer1.0-plugins-good gstreamer1.0-
plugins-ugly gstreamer1.0-pulseaudio gstreamer1.0-qt5 gstreamer1.0-rtsp
gstreamer1.0-tools gstreamer1.0-x intel-media-va-driver-non-free
libdrm-amdgpu1 libdrm-common libdrm-dev libdrm-intel1 libdrm-nouveau2
libdrm-radeon1 libdrm-tests libdrm2 libgstrtspserver-1.0-dev
libgstrtspserver-1.0-0 libgstreamer-gl1.0-0 libgstreamer-opencv1.0-0
libgstreamer-plugins-bad1.0-0 libgstreamer-plugins-bad1.0-dev
libgstreamer-plugins-base1.0-0 libgstreamer-plugins-base1.0-dev
libgstreamer-plugins-good1.0-0 libgstreamer-plugins-good1.0-dev
libgstreamer1.0-0 libgstreamer1.0-dev libigdgmm-dev libigdgmm12
libigfxcmrt-dev libigfxcmrt7 libmfx-gen1.2 libtpms-dev libtpms0 libva-
dev libva-drm2 libva-glx2 libva-wayland2 libva-x11-2 libva2 libwayland-
bin libwayland-client0 libwayland-cursor0 libwayland-dev libwayland-doc
libwayland-egl-backend-dev libwayland-egl1 libwayland-server0
libxatracker2 linux-firmware mesa-utils mesa-va-drivers mesa-vdpau-
drivers mesa-vulkan-drivers libvpl-dev libmfx-gen-dev onevpl-tools
libvpl-tools ovmf ovmf-ia32 qemu-system-modules-opengl qemu-block-extra
```

```
qemu-guest-agent qemu-system qemu-system-arm qemu-system-common qemu-
system-data qemu-system-gui qemu-system-mips qemu-system-misc qemu-
system-ppc qemu-system-s390x qemu-system-sparc qemu-system-x86 qemu-
user qemu-user-binfmt qemu-utils va-driver-all vainfo weston xserver-
xorg-core xserver-common xnest xserver-xorg-dev xvfb libvirt0 libvirt-
clients libvirt-daemon libvirt-daemon-config-network libvirt-daemon-
config-nwfilter libvirt-daemon-driver-lxc libvirt-daemon-driver-qemu
libvirt-daemon-driver-storage-gluster libvirt-daemon-driver-storage-
iscsi-direct libvirt-daemon-driver-storage-rbd libvirt-daemon-driver-
storage-zfs libvirt-daemon-driver-vbox libvirt-daemon-driver-xen
libvirt-daemon-system libvirt-daemon-system-systemd libvirt-dev
libvirt-doc libvirt-login-shell libvirt-sanlock libvirt-wireshark
libnss-libvirt swtpm swtpm-tools bmap-tools adb autoconf automake
libtool cmake g++ gcc git intel-gpu-tools libssl3 libssl-dev make
mosquitto mosquitto-clients build-essential apt-transport-https
default-jre docker-compose ffmpeg git-lfs gnuplot lbzip2 libglew-dev
libglm-dev libsdl2-dev mc openssl pciutils python3-pandas python3-pip
python3-seaborn terminator vim wmctrl wayland-protocols gdbserver
ethtool iperf3 msr-tools powertop linuxptp lsscsi tpm2-tools tpm2-abrmd
binutils cifs-utils i2c-tools xdotool gnupg lsb-release ethtool
iproute2 socat virt-viewer spice-client-gtk util-linux-extra dbus-x11 -
y --allow-downgrades
```

3.2.3.2 Install Kernel Overlay into Ubuntu* OS

Choose one of the following options to install kernel overlay.

Option 1: Install kernel overlay from 01.org by using the following commands:

```
sudo apt install linux-image-6.12-intel=250711t071314Zz-r2

sudo apt install linux-headers-6.12-intel=250711t071314Zz-r2
```

Note: For Option 1, the kernel Debian package on 01.org will be continuously updated to the latest versions, replacing previous ones. These latest versions may not be validated for this platform in the current release. Therefore, if you encounter an "unable to locate package" issue, we recommend Option 2.

Option 2: Install kernel Debian packages that you built from Build Kernel Debian Packages by using the following commands:

```
sudo dpkg -i linux-image-<version>_amd64.deb

sudo dpkg -i linux-headers-<version>_amd64.deb
```

3.2.3.3 Change Grub Configuration

A few configurations can be changed in /etc/default/grub to add kernel parameters, show the grub menu during boot up or even boot to the desired default kernel.

3.2.3.4 Compulsory Kernel Parameters

Required for Graphics:
The following kernel parameters are required to load the correct GuC FW:

```
sudo vi /etc/default/grub
```

Before Change	After Change
GRUB_CMDLINE_LINUX_DEFAULT ="quiet splash"	GRUB_CMDLINE_LINUX_DEFAULT="quiet splash i915.enable_guc=3 i915.max_vfs=7 i915.force_probe=* udmabuf.list_limit=8192"

3.2.3.5 Use GRUB_DEFAULT To Select A Default Kernel Version

When there are more than one kernels being installed in the system, the highest kernel version will be booted by default. Selecting the kernel version from the grub menu each time after rebooting is tedious. This can be avoided by setting GRUB_DEFAULT. By doing so, the system will be booted with the kernel specified in the GRUB_DEFAULT string.

If the kernel version that you installed is the highest version, you can skip this section and proceed to next.

Note: The string shown in this section is an example output. The string may be different in your system, but the steps to find out the string are common.

Step 1: Find entrance from /boot/grub/grub.cfg

Get the submenu:

```
sudo grep submenu /boot/grub/grub.cfg
```

Example output:

```
submenu 'Advanced options for Ubuntu' $menuentry_id_option ' gnulinux-advanced-5efe8b52-8c51-436a-b80e-5f9f0d7d037c ' {
```

In this case, '**Advanced options for Ubuntu**' is what we need.

Get the menuentry for the specific kernel option that you have installed. In this specific example, assuming kernel version 6.12.36 is being installed:

```
sudo grep 6.12.36 /boot/grub/grub.cfg
```

Example output:

```
menuentry 'Ubuntu, with Linux 6.12.36-nonrt-000' --class ubuntu --class gnu-linux
--class gnu --class os $menuentry_id_option 'gnulinux-6.12.36-nonrt-000-
advanced-5efe8b52-8c51-436a-b80e-5f9f0d7d037c' {

menuentry 'Ubuntu, with Linux 6.12.36-nonrt-000 (recovery mode)' --class ubuntu
--class gnu-linux --class gnu --class os $menuentry_id_option 'gnulinux-6.12.36-
nonrt-000-recovery-5efe8b52-8c51-436a-b80e-5f9f0d7d037c' {
```

In this example, the kernel that is being installed contains '6.12.36', hence the menuentry, 'Ubuntu, with Linux 6.12.36-nonrt-000' is what we need. Do not take the menuentry with recovery mode.

Step 2: Set GRUB_DEFAULT in /etc/default/grub by joining two entry strings (submenu and menuentry) obtained above by '>', set to GRUB_DEFAULT.

```
sudo vi /etc/default/grub
```

Before Change	After Change
GRUB_DEFAULT=0	GRUB_DEFAULT="Advanced options for Ubuntu>Ubuntu, with Linux 6.12.36-nonrt-000"

3.2.3.6 Use GRUB_TIMEOUT to Show Grub Menu Entry

If you would like to show the grub menu, look for GRUB_TIMEOUT_STYLE and GRUB_TIMEOUT and change according to the table below:

```
sudo vi /etc/default/grub
```

Before Change	After Change
GRUB_TIMEOUT_STYLE=hidden	#GRUB_TIMEOUT_STYLE=hidden
GRUB_TIMEOUT=0	GRUB_TIMEOUT=5

3.2.3.7 Grub Update and Reboot System

Once all the configuration in /etc/default/grub is completed, follow the steps below to complete the grub update.

Step 1: Update grub.

```
sudo update-grub
```

Step 2: Reboot the machine.

```
sudo reboot
```

3.2.3.8 Verify System Configuration

Make sure the OS is booted to the correct kernel version.

```
uname -a
```

You should see that the output contains the kernel version that you set as the default kernel.

3.3 Installing Sample Applications or Proprietary Tools

This section describes the steps to install a sample application for other platforms' features and tools.

3.3.1 Install DRTU Tools for Wi-Fi/Bluetooth®

Perform the installation if you need to use the DRTU tool for regulatory testing of Intel wireless device.

The DRTU tool available in the OEM Tools for this version requires additional steps to upgrade the iwlwifi version to support the tool. Follow the instructions to upgrade the driver before installation.

3.3.1.1 Upgrade the Wi-Fi/Bluetooth® Driver

These are the instructions to upgrade the driver:

Step 1: Download the driver.

```
mkdir upgrade-wifi

cd upgrade-wifi

git clone
https://git.kernel.org/pub/scm/linux/kernel/git/iwlwifi/backport-
iwlwifi.git

cd backport-iwlwifi/

sudo git checkout 0c1fc2832de3b58660674beef1ef31ffc0db7672
```

Step 2: Compile the driver and install it to the system.

```
cp defconfig .config

sudo apt update

sudo apt install -y build-essential

sudo chmod -R 777 .

sudo make -j8
```

If you encounter this compilation error:

```
Example of error:
/--------------
| Your kernel headers are incomplete/not installed.
| Please install kernel headers, including a .config
| file or use the KLIB/KLIB_BUILD make variables to
| set the kernel to build against, e.g.
|   make KLIB=/lib/modules/3.1.7/
| to compile/install for the installed kernel 3.1.7
| (that isn't currently running.)
\--
make[1]: *** [Makefile:41: modules] Error 1
make: *** [Makefile:30: default] Error 2
```

Execute the following command:

```
sudo cp /boot/config-$(uname -r) /usr/src/linux-headers-$(uname -
r)/.config
```

and compile again:

```
sudo make -j8
```

Step 3: Install the driver.

```
sudo make install

sudo depmod
```

Note: Press "Enter" to install the default suggested installation

Step 4: Download and copy the firmware to /lib/firmware.

```
cd ~

mkdir wifi-fw

cd wifi-fw

git clone
https://git.kernel.org/pub/scm/linux/kernel/git/firmware/linux-
firmware.git

cd linux-firmware/

sudo cp -r *.ucode /lib/firmware/

sudo reboot
```

Step 5: Insert the iwlxvt module if not yet installed.

```
sudo modprobe iwlxvt
```

Step 6: Check that the correct core driver is loaded.

```
sudo dmesg | grep iwl
```

Output as follows:

```
iwlwifi 0000:aa:00.0: loaded firmware version 96.44729d4e.0 gl-c0-fm-c0-96.ucode op_mode iwlmvm
```

OR

```
sudo lsmod | grep iwl
```

Example of output as follows:

```
lwlxvt         65536   0
iwlmvm        729088   0
mac80211    13217194   1   iwlmvm
iwlwifi       593920   2   iwlmvm,iwlxvt
iwlmei         53248   2   iwlmvm,iwlwifi
cfg80211     1216512   5   iwlmvm,iwlxvt,iwlmei,iwlwifi,mac80211
compat         12288   6   iwlmvm,iwlxvt,iwlmei,iwlwifi,mac80211,cfg80211
```

Step 7: After the system has rebooted, verify the Ethernet IP Address, which is required for the DRTU client connection.

3.3.1.2 Install DRTU Tool

Step 1: Install the library dependencies for the tool.

```
sudo apt install libc++-dev
```

Step 2: Download SBHWFW01964_23.120.0.3Tools_Chrome.zip from Software Kit: 855658 and unzip the package.

```
unzip SBHWFW01964_23.120.0.3Tools_Chrome.zip
```

Step 3: Change directory to OEM\ Tools/DRTU/R120AndAbove and install.

```
cd OEM_Tools/DRTU/R120AndAbove

sudo chmod +x *.sh
```

```
sudo bash ./install.sh -o linux -d typ
```

Note: This error can be ignored as it does not impact the installation: "ln: failed to create symbolic link 'libstdc++.so.6': File exists"

Step 4: Run one of the web-servers for DRTU tool.

```
sudo bash ./runDRTUWeb.sh &
```

Example of the output as follows:

```
United tool, Version DRTU_06642_23.60.0

(C) 2007-2024 Intel corporation / OEM Tools group. All rights reserved.

********* Common::Actor::Connected

---------- United Tool ----------

CITU Not Loaded

NVM Burner Not Loaded

NDT Not Loaded

DRTU Loaded Successfully
```

Note: Only one server can run at a time. Kill the earlier web server before starting a new one.

Step 5: Log in to the tool web page using a Chrome browser in any Windows* host machine, which is in the same subnet as the platform. Replace <IP address> with the IP address of the DUT.

<IP address>:8738

Refer to the *Intel® Wireless Connectivity Product User Guide* (Document Number: 617199) for more information.

3.3.2 Install Intel® SoC Watch

Step 1: Follow the steps in How to Get Intel® SoC Watch NDA Version on how to get the Intel® SoC Watch NDA version.

Step 2: Once it has been downloaded, untar the package.

```
tar zxvf socwatch_linux_NDA_<ver>.tar.gz
```

Step 3: Build the required drivers.

```
cd socwatch_linux_NDA_<ver>

sudo ./build_drivers.sh
```

Step 4: If Step 3 is successful without any error message, socwatch2_16.ko will be created in drivers/.

```
cd drivers/

ls socwatch_linux_NDA_<ver>/drivers
```

Example of output:

```
insmod-socwatch  rmmod-socwatch  socwatch2_16.ko
```

Step 5: Insert the module.

```
sudo ./insmod-socwatch
```

Step 6: Set up the socwatch environment.

```
cd ..

sudo chmod 755 ./setup_socwatch_env.sh

sudo ./setup_socwatch_env.sh
```

Step 7: Check the CPU state with socwatch.

```
sudo ./socwatch -f cpu -t 10 -o testcstate
```

Refer to the user guide in the package for more information.

3.3.3 Install IPU Components for HDMI Capture Features

Step 1: Download ARL-UH_IPU_FW_HDMI-in.zip from Software Kit: 831484 and unzip the package.

```
sudo unzip ARL-UH_IPU_FW_HDMI-in.zip
```

Step 2: Convert all the rpm packages to deb packages and install them.

```
sudo apt install alien
sudo alien *.rpm
sudo apt install alien
sudo alien *.rpm
sudo dpkg -i --force-overwrite aiqb-ipu6epmtl_1.0.0-*.deb
sudo dpkg -i --force-overwrite icamerasrc_1.0.0-*.deb
sudo dpkg -i --force-overwrite libcamhal_1.0.0-*.deb
sudo dpkg -i --force-overwrite libiaaiq-ipu6epmtl_1.0.0-*.deb
sudo dpkg -i --force-overwrite libiacss-ipu6epmtl_1.0.0-*.deb
sudo dpkg -i --force-overwrite ipu6epmtlfw_1.0.0-*.deb
```

Note: Refer to the *HDMI Capture Software Enabling Guide* (Document Number: 789643) for more information.

4.0 Terminology

This section lists the terminology and descriptions that are being used throughout the document.

Table 1. Terminology

Term	Description
DUT	Device Under Test
HD	High-Definition
IBV	Independent BIOS Vendor
OS	Operating System

5.0 Reference Documents

Log in to the Resource and Documentation Center (rdc.intel.com) to search for and download the document numbers listed in the following table. Contact your Intel field representative for access.

Note: Third-party links are provided as a reference only. Intel does not control or audit third-party benchmark data or the websites referenced in this document. You should visit the referenced website and confirm whether the referenced data are accurate.

Table 2. Reference Documents

Document	Document No./Location
Arrow Lake-U/H for Edge Applications - Gold Deck	823305
Meteor Lake-U/H Customer Reference Board (CRB) COM-HPC - User Guide	783466
Ubuntu with Kernel Overlay on Arrow Lake - U/H for Edge Platforms Release Notes	828854
UEFI Reference BIOS on Arrow Lake - U/H for Edge Platforms - Package	830743
UEFI Reference BIOS on Arrow Lake - U/H for Edge Platforms - Release Note	863413
Slim Bootloader on Arrow Lake - U/H for Edge Platforms - Package	844128
Slim Bootloader on Arrow Lake - U/H for Edge Platforms - Release Notes	863414
Intel® In-Band Manageability Framework on Arrow Lake – U/H for Edge Platforms - Package	842787
Intel® In-Band Manageability Framework on Arrow Lake - U/H for Edge Platforms - Release Notes	830711
Arrow Lake - U/H for Edge Platform - Ubuntu with Kernel Overlay Software Packages	831484

6.0 Appendix

6.1 Further References

The following links are some useful references to understand further some terms or Linux* commands that are used in this document:

https://ubuntu.com/tutorials This link provides tutorials and a step-by-step process for performing development and dev-ops activities on Ubuntu* machines, servers, or devices.
https://ubuntu.com/download/iot/intel-iot This link provides the release info for some of the Ubuntu* OS on the Intel IoT Platform.
https://www.tecmint.com/find-usb-device-name-in-linux/ This link provides tips on finding a USB device name in Linux*.
https://www.kernel.org/doc/Documentation/gpio/sysfs.txt This link provides information about the GPIO sysfs Interface for user space.
https://thesofproject.github.io/latest/index.html This link provides information about Sound Open Firmware.

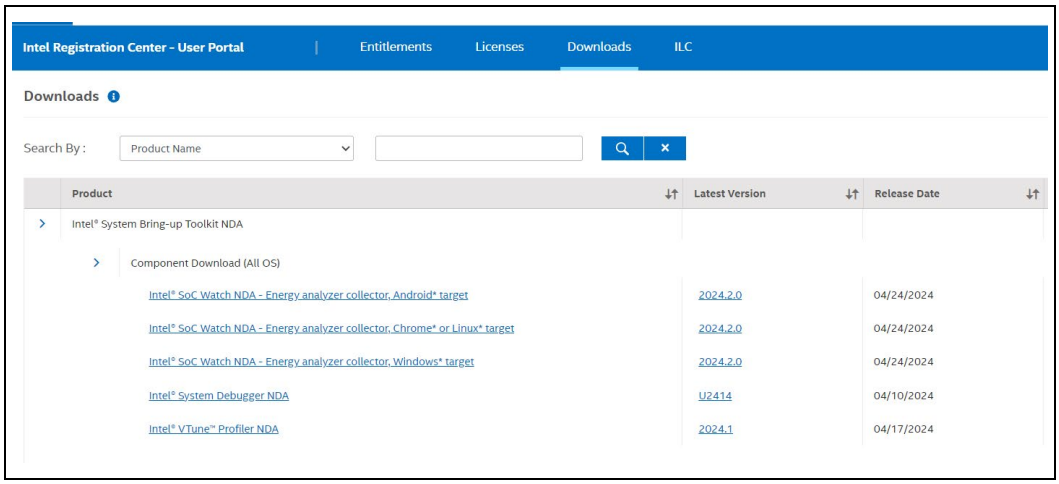
6.2 How to Get Intel® SoC Watch NDA Version

Step 1: Fill in an online form in the Intel Registration Center to request for Intel compiler. Once registered, wait for the notification email with the license key.

Step 2: Sign in to Intel Registration Center: Downloads with your Intel NDA Account (if you do not have an account, create one now by choosing sign-up from the same link). Once logged in, under the Downloads tab, find "Intel® SoC Watch NDA – Energy analyzer collector, Chrome* or Linux* target" under "Intel® System Bring-up Toolkit".

Note: Figure 2 shows an example of the result on the page. It may not reflect the latest version.

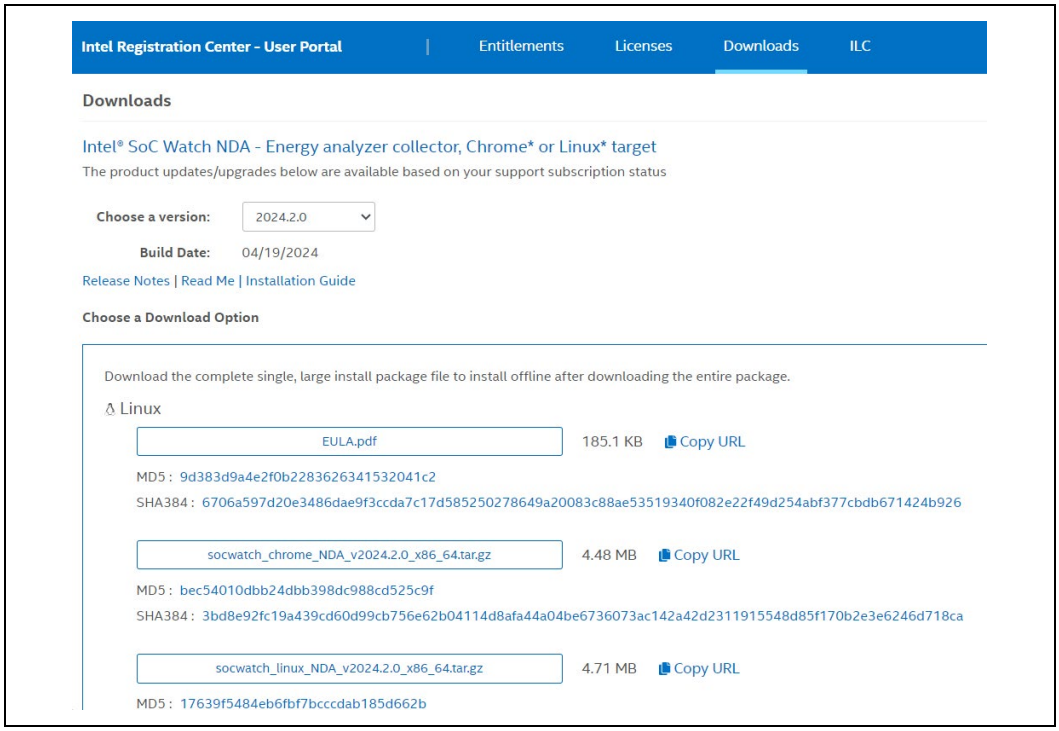
Figure 2. Downloads: Intel® SoC Watch NDA



Step 3: Click "socwatch_linux_NDA_<version>_x86_64.tar.gz" to download the package.

Note: Figure 3 shows an example of the result on the page. It may not reflect the latest version.

Figure 3. Intel® SoC Watch NDA – Energy analyzer collector, Chrome* or Linux* target



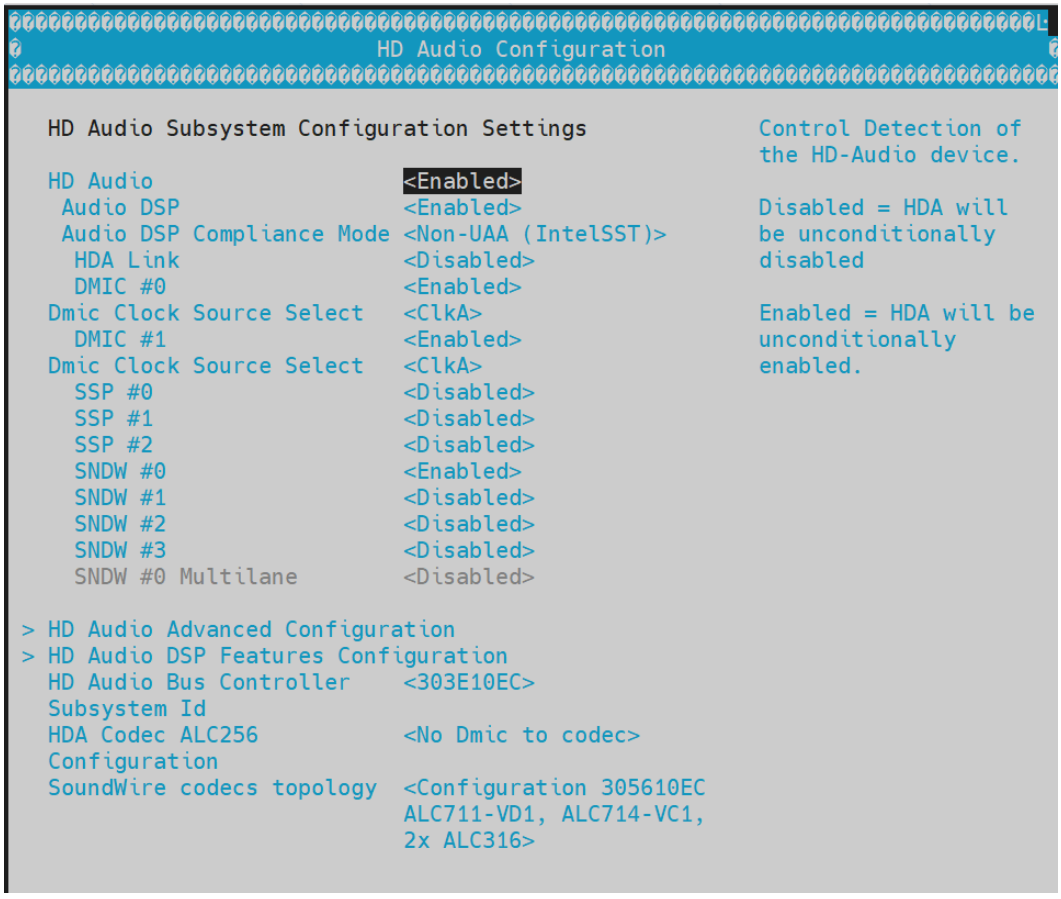
6.3 Audio

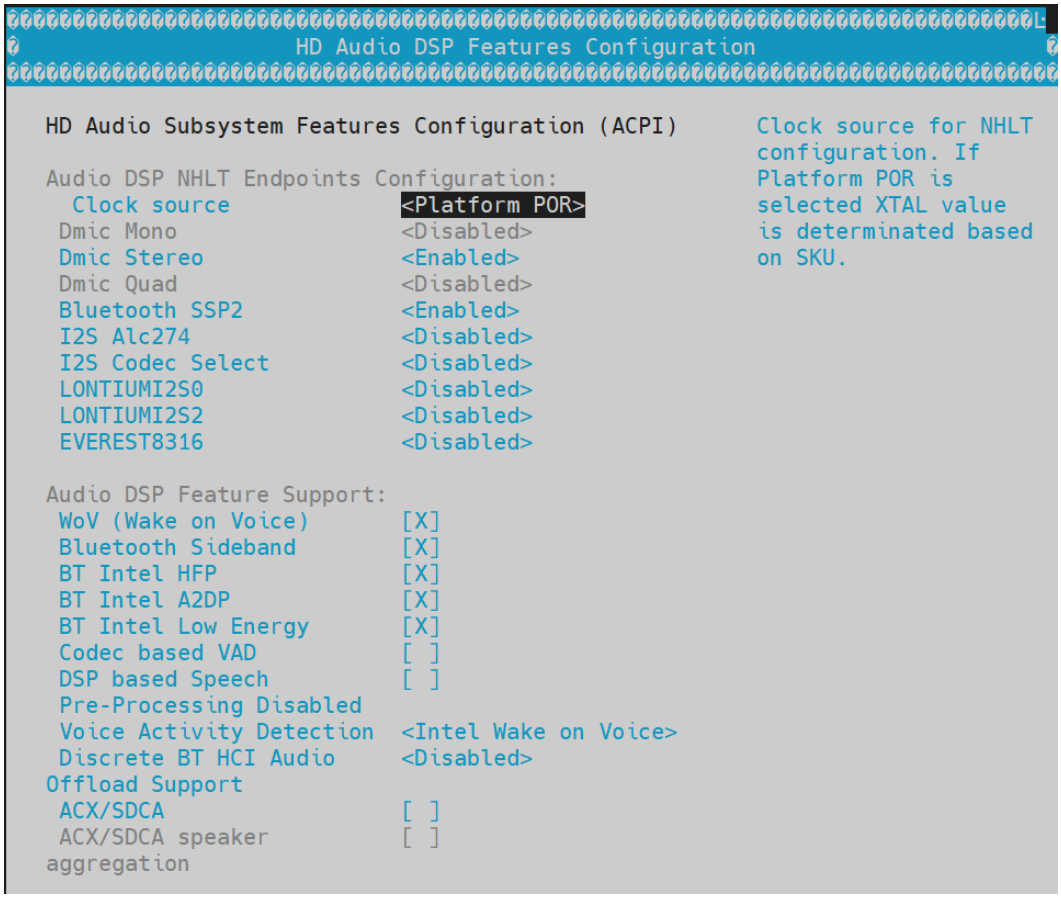
This section describes some of the required settings in BIOS menu and/or other configuration for audio features and it is applicable for Intel released IFWI binary. The name might not be the same as the IFWI released by IBV. In this case, please contact your IBV for more information.

6.3.1 BIOS Settings for Soundwire Audio via DSP

Configure audio setting on BIOS menu based on Figure 4

Figure 4. BIOS Settings for Soundwire Audio





6.3.2 Integrate Audio Firmware and Topology

Once the CRB is booted into Ubuntu* OS,

Step 1: Confirm the existence of the directory with the following commands, if it does not exist, the directory will be created.

```
test -d "/lib/firmware/intel/sof-ipc4/arl" && echo "Directory exists"
|| mkdir /lib/firmware/intel/sof-ipc4/arl

test -d "/lib/firmware/intel/sof-ace-tplg" && echo "Directory exists"
|| mkdir /lib/firmware/intel/sof-ace-tplg
```

Step 2: Download sof-mtl.ri from GitHub, put it into /lib/firmware/intel/sof-ipc4/arl. The following command will rename sof-mtl.ri to sof-arl.ri

```
sudo mv /lib/firmware/intel/sof-ipc4/arl/sof-mtl.ri
/lib/firmware/intel/sof-ipc4/arl/sof-arl.ri
```

Step 3: Download topology file from GitHub, put it into /lib/firmware/intel/sof-ace-tplg/. The following command will rename sof-mtl-rt711-4ch.tplg to sof-arl-rt711-l0-4ch.tplg

```
sudo mv /lib/firmware/intel/sof-ace-tplg/ sof-mtl-rt711-4ch.tplg
/lib/firmware/intel/sof-ace-tplg/ sof-arl-rt711-l0-4ch.tplg
```

Step 4: Create a blacklist_hda.conf into the directory /etc/modprobe.d/

```
sudo vim /etc/modprobe.d/blacklist_hda.conf
#Add the following content
blacklist snd_hda_intel
blacklist snd_hda_core
#save and exit
```

Step 5: Create an alsa.conf into the directory /etc/modprobe.d/alsa.conf

```
sudo vim /etc/modprobe.d/alsa.conf
#Add the following content
options snd_sof_intel_hda_common sof_use_tplg_nhlt=1
options snd_hda_core gpu_bind=0
options snd_soc_sof_sdw quirk=1
#save and exit
```

Step 6: Reboot the system.

```
sudo reboot
```

6.4 S0ix

This section describes some of the required settings in the BIOS menu and/or other configurations for S0ix.

6.4.1 Recommended BIOS Settings

Intel Advanced Menu → System Agent (SA) Configuration → GNA Device → Disabled

Intel Advanced Menu → PCH-IO Configuration → PCH_LAN → Disabled

6.4.2 Query for S0ix Residency Values

The following commands are used to query the S0ix residency values. If the system successfully enters the S0ix state, the value will increase after waking up from it. Perform these steps before and after the system enters the S0ix state.

```
cat /sys/devices/system/cpu/cpuidle/low_power_idle_cpu_residency_us
cat /sys/kernel/debug/pmc_core/slp_s0_residency_usec
cat /sys/kernel/debug/pmc_core/package_cstate_show
cat /sys/kernel/debug/pmc_core/substate_residencies
```

6.4.3 Method to Enter S0ix State

There are two ways to enter the S0ix state:

Opportunistic Idle

Once the required BIOS settings are configured, the system will enter S0ix when there are no activities

Suspend-to-idle

Once the required BIOS settings are configured, the system can be forced to enter S0ix via the following commands:

```
echo freeze > /sys/power/state
```

6.5 OS Configuration for S4, S5, and Warm Reset

S4:

```
sudo nano /etc/default/grub

#add resume=/dev/nvme1n0p3 ignore_loglevel into
GRUB_CMDLINE_LINUX_DEFAULT as below:

GRUB_CMDLINE_LINUX_DEFAULT="quiet splash i915.enable_guc=3
i915.max_vfs=7 i915.force_probe=* udmabuf.list_limit=8192
resume=/dev/nvme1n0p3 ignore_loglevel"
```

S5 and warm reset:

```
sudo nano /etc/systemd/system.conf

#uncomment #DefaultTimeoutStopSec=90s and update it to 5s as below
DefaultTimeoutStopSec=5s

#save and exit

sudo service cups-browsed restart
sudo service cups restart
sudo update-initramfs -u
sudo reboot
```