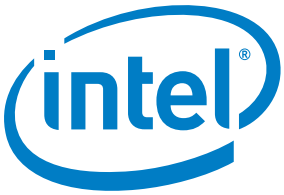


Intel® Unified 3D Library for Intel® Atom™ E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200 for Yocto Project*

Release Notes

August 2017

MR3.1 Release

Document Number: 335619-003

Contents

1.0 Introduction 5

1.1 Terminology 5

2.0 Driver Information 6

3.0 System Requirements 7

3.1 Prerequisites and Constraints 7

4.0 Using Intel® Unified 3D Library 9

4.1 Integrating the Intel® Unified 3D Library Driver in the Yocto* Build 9

4.1.1 Verifying the Installation of the Intel® Unified 3D Library Driver 12

5.0 Enabled Features 13

6.0 Disabled Features 14

7.0 Fixed Issues 15

7.1 Fixed Issues 15

8.0 Known Issues and Limitations 16

8.1 Known Issues 16

8.2 Limitations 16

Figures

Figure 1. Choosing the Custom Build 9

Figure 2. Adding the meta-intel-ufo Path in bblayers.conf 10

Figure 3. Adding the RPM Binary and Appending ufo-dri in local.conf 11

Figure 4. Appending ufo-opencl in local.conf 11

Tables

Table 1. Terminology 5

Revision History

Date	Revision	Description
February 2017	001	MR2 release
May 2017	002	MR3 realease
August 2017	003	MR3.1 release

§

1.0 Introduction

This document describes the typical hardware system requirements, features, testing, and use of the driver for Intel® Unified 3D Library.

The Intel® Unified 3D Library driver is available for 64-bit systems for the Intel® Atom™ E3900 SoC Family, Intel® Celeron® Processor N3350, and Intel® Pentium® Processor N4200 for Yocto Project*.

1.1 Terminology

Table 1. Terminology

Term	Description
BSP	Board Support Package
DRI	Direct Rendering Infrastructure
IOTG	Internet of Things Group
OS	Operating System
RPM	RPM Package Manager
SoC	System-on-a-Chip

§

2.0 Driver Information

The Intel® Unified 3D Library driver is available for 64-bit systems and is part of the release package. The details of the Linux* drivers are as follows:

Intel® Unified 3D Library Driver: intel-linux-ufo-yocto_<version>-64bit.tar.gz

§

3.0 System Requirements

The Intel® Unified 3D Library driver supports the following platforms:

- Intel® Atom™ E3900 SoC Family
- Intel® Celeron® Processor N3350
- Intel® Pentium® Processor N4200

3.1 Prerequisites and Constraints

The Intel® Unified 3D Library driver is part of Intel proprietary software.

To integrate the driver, you must obtain the board support package (BSP) and add the meta-intel-proprietary recipe along with the RPM binary.

1. Download the BSP for Yocto Project* from GitHub to your host machine
   - HTTPS directly from https://github.com/01org/iotg-yocto-bsp-public/tree/e3900/master by selecting the appropriate tag version, for example, E3900-MR2, from the top left menu or
   - SSH using the following command:

     ```
     git clone https://github.com/01org/iotg-yocto-bsp-public.git -b e3900/master
     ```

     This git tree is maintained as single product branch. To get code base from the previous release, for example the PV release, check out to the specific tag.
   - For PV release: git checkout E3900-PV
   - For Maintenance Release Version 1: git checkout E3900-MR1
   - For Maintenance Release Version 2: git checkout E3900-MR2
   - For Maintenance Release Version 3: git checkout E3900-MR3
   - For Maintenance Release Version 3.1: git checkout E3900-MR3.1 (use this tag for this release)

     ```
     $cd /home/user/development
     ```

Note: The directory path mentioned here is an example for demonstration purposes. You can use any convenient directory location for the build.

2. Extract the Intel® Unified 3D Library DRI proprietary meta layer.

   ```
   $ tar xvf meta-intel-proprietary-<component>.tar.bz2
   ```

3. Download the Intel® Unified 3D Library RPM compressed tar ball and put the content in one location. The Intel® Unified 3D Library DRI RPM binary is provided with this package.

```
Example Download path: /home/user/rpm-binaries

$cd /home/user/rpm-binaries
```

Refer to Section 4.0 for instructions on integrating and building with Intel® Unified 3D Library.

§

4.0 Using Intel® Unified 3D Library

This section describes how to integrate, build, and verify the Intel® Unified 3D Library driver.

4.1 Integrating the Intel® Unified 3D Library Driver in the Yocto* Build

The Intel® Unified 3D Library driver is available with the release package provided.

As mentioned in the previous section, the proprietary meta layer and the Intel® Unified 3D Library DRI RPM packages must be extracted to a particular location.

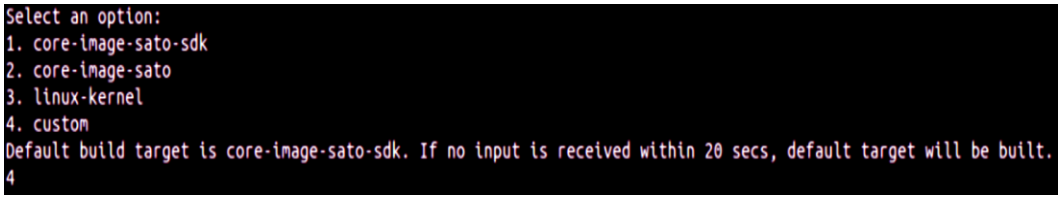
1. Go to the directory location where the Yocto BSP meta layer is cloned in the host (Ubuntu*) machine.

   `bsp-apollolake-i`

   Refer to the Yocto BSP Gold release for more details on Yocto BSP cloning and building.
2. Run the `setup.sh` script to initiate the build.

   `$./setup.sh`
3. When prompted for the image type, choose option 4 to build a custom image.

Figure 1. Choosing the Custom Build



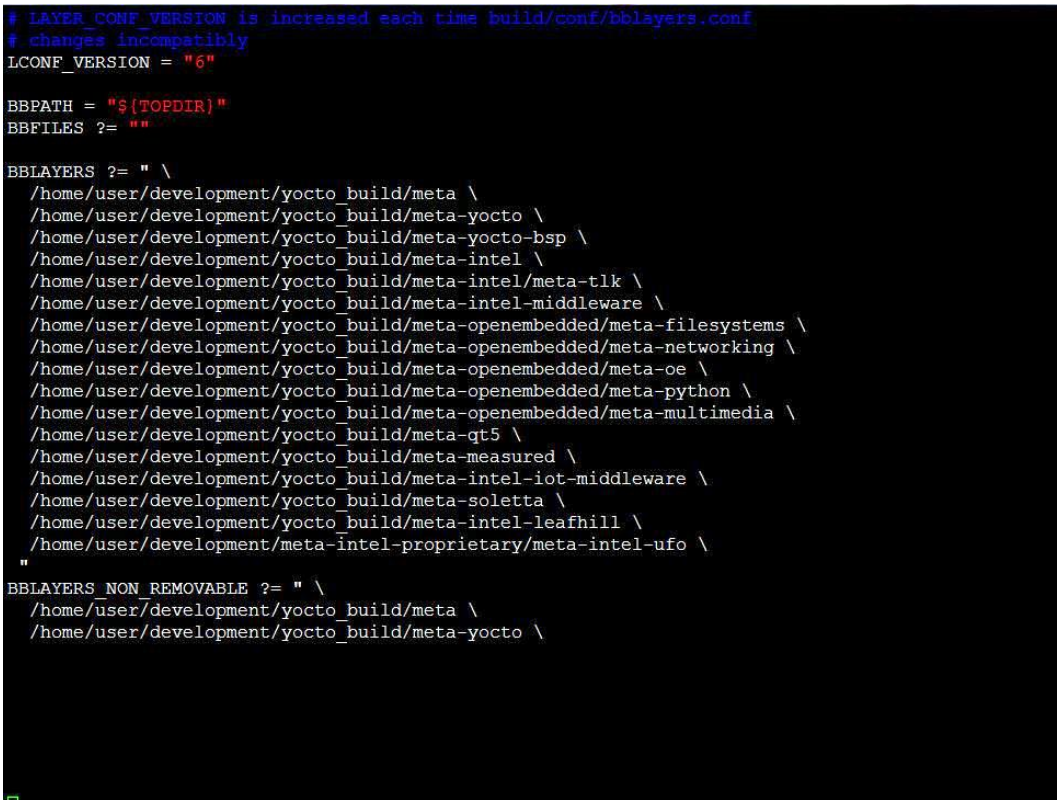
   When the `setup.sh` script has completed, a new folder named `yocto_build` appears under `/home/user/development/`.
4. Change the directory to `yocto_build/build/conf`.

   `$cd /home/user/development/yocto_build/build/conf`
5. Edit `bblayers.conf` to include the meta-intel-proprietary layer. This layer was previously downloaded.

   `$ vim bblayers.conf`
6. Add the path `meta-intel-proprietary/meta-intel-ufo` as shown in Figure 2. Save and exit.

Note: The path is an example for demonstration purposes. Your path might differ based on the location of the meta layers.

Figure 2. Adding the meta-intel-ufo Path in bblayers.conf



7. Edit `local.conf` located in the same directory to include the path where the downloaded RPM binaries are located.

   ```
   $vim local.conf
   ```

   Add the following line at the top of the `local.conf` file.

   ```
   export RPM_PATH=''<full_path_to_the_downloaded_rpm>''
   ```

Example: If the RPM binaries are in `/home/user/rpm-binary/`, insert the following command in `local.conf`:

```
export RPM_PATH=/home/user/rpm-binary''
```

8. Further edit `local.conf` to let the compiler know to install the RPM binaries during the build.

   Add the following line:

   ```
   IMAGE_INSTALL_append = `` ufo-dri''
   ```

   Figure 3 shows the RPM path and `ufo-dri` appended in `local.conf`.

Note: There is a <space> in front of `ufo-dri` as shown in Figure 3.

Figure 3. Adding the RPM Binary and Appending ufo-dri in local.conf

```
export RPM_PATH="/home/user/rpm-binary"

IMAGE_INSTALL_append = " ufo-dri"
```

9. OpenCL and ufo-nano are also provided with the UFO RPM release package. Follow these steps to install OpenCL and ufo-nano as part of the binary package:
   a. Copy the `ufo-openCL` and `ufo-nano` RPMs provided with the UFO binaries into the RPM-binary directory in the build system.
   b. Append the following line in `local.conf` to add `ufo-opencl`

      ```
      IMAGE_INSTALL_append = " ufo-opencl"
      ```

Figure 4. Appending ufo-opencl in local.conf

```
export RPM_PATH="/home/user/rpm-binary"

IMAGE_INSTALL_append = " ufo-dri"
IMAGE_INSTALL_append = " ufo-opencl"
```

10. If ufo-nano is needed, append the following line in `local.conf` to add `ufo-nano`

    ```
    IMAGE_INSTALL_append = " ufo-nano"
    ```

11. Go up one level to `yocto_build`.

    ```
    $ cd ../..
    ```

12. Prepare the environment to run the bitbake command.

```
$ source oe-init-build-env
$ cd build
```

13. Start the image compilation.

```
$ bitbake core-image-sato-sdk
```

Note: It is recommended to start with a new image build. However, if the previous build is being used, execute the following commands before image compilation.

```
$ bitbake linux-firmware -c cleanall
$ bitbake libva -c cleanall && bb virtual/libva -c cleanall
$ bitbake libva -c cleansstate && bb virtual/libva -c
cleansstate
$ bitbake core-image-sato-sdk
```

Note: Refer to *Intel® Atom™ E3900 SoC Family Yocto BSP Media-SDK Getting Started Guide* provided with the IOTG Yocto BKC document for the procedure to integrate UFO DRI with closed Source Media Stack.

4.1.1 Verifying the Installation of the Intel® Unified 3D Library Driver

To verify that the Intel® Unified 3D Library driver is installed and being used:

1. If ufo-nano is not installed, execute the following command in the terminal.

```
$ glxinfo | grep -i OpenGL
```

The output should contain the following library description.

```
'OpenGL renderer string: Intel(R) Unified 3D Library'
```

2. If ufo-nano is installed, execute the following command in the terminal.

```
$ es2_info
```

The output should contain the following library description.

```
'GL_VERSION: OpenGL ES 3.1 - Build <version>-nano-production'
'GL_RENDERER: Intel(R) Unified 3D Library '
```

§

5.0 Enabled Features

This section is not applicable to the current release.

§

6.0 Disabled Features

This section is not applicable to the current release.

§

7.0 Fixed Issues

This section describes the fixed issues that affect the Intel® Unified 3D Library driver.

7.1 Fixed Issues

Refer to the graphics section in *Intel® Atom™ E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200 Board Support Package for Yocto Project* Release Notes* for details.

§

8.0 Known Issues and Limitations

This section describes the known issues and limitations that affect the Intel® Unified 3D Library driver.

8.1 Known Issues

Refer to the graphics section in *Intel® Atom™ E3900 SoC Family/Intel® Celeron® Processor N3350/Intel® Pentium® Processor N4200 Board Support Package for Yocto Project* Release Notes* for details.

8.2 Limitations

None

§