Using Ice with Yocto

On this page:

Overview of Ice with Yocto

The Yocto Project allows you to create a custom Linux distribution for your embedded product. With ZeroC's meta layer for Yocto, you can easily include Ice for C++, Ice for Python and/or the Glacier2 service in your custom Linux distribution.

Development Boards

Ice was tested with the following boards and images:

Board	Yocto version
Freescale SABRE SD	2.3 (Pyro) with the Freescale Community BSP
BeagleBone Black board (Rev C)	2.3 (Pyro)

Meta Layer

Refer to the README.md file in the zeroc-ice/meta-zeroc repository.

Installing the ZeroC Meta Layer

Simply clone the `meta-zeroc` git repository and add `meta-zeroc` to the BBLAYERS variable in your `bblayers.conf` file:

In /home/user/repos

```
# Clone poky on branch pyro
git clone -b pyro git://git.yoctoproject.org/poky.git

# Clone meta-zeroc on branch pyro
git clone -b pyro git://github.com/zeroc-ice/meta-zeroc.git
```

bblayers.conf

```
BBLAYERS ?= " \
  /home/user/repos/poky/meta \
  /home/user/repos/poky/meta-yocto \
  /home/user/repos/poky/meta-yocto-bsp \
  /home/user/repos/meta-zeroc \
  "
```

The `meta-zeroc` repository has the same branches as the poky repository (jethro, krogoth, morty, pyro etc.)

Using the ZeroC Meta Layer to install Ice

To include Ice in your image, add the package from the `zeroc-ice` recipe you wish to use to your `local.conf` file:

```
# Install Ice for C++ dynamic libraries and Ice for Python to image
IMAGE_INSTALL_append = " zeroc-ice zeroc-ice-python"

# Add the development package to the SDK
TOOLCHAIN_TARGET_TASK_append = " zeroc-ice-dev zeroc-ice-staticdev"

# Add the development package to the Native SDK
TOOLCHAIN_HOST_TASK_append  = " nativesdk-zeroc-ice-dev"
```

To ensure the correct Ice version is used you can set `PREFERRED_VERSION_zeroc-ice` in your local.conf file:

```
PREFERRED_VERSION_zeroc-ice = "3.7.0"
```

Setting up your cross development environment to use Ice

With the variables set in `local.conf` as shown above, you can generate an SDK for your image with the following command:

Create SDK for the core-image-minimal Image

```
bitbake core-image-minimal -c populate_sdk
```

Once complete, this SDK can be found in `tmp/deploy/sdk`. Execute the `.sh` file to install. By default your SDK will be installed into `/opt/poky/<version>/`. You can then source the cross-development environment as follows:

```
source /opt/poky/2.3/environment-setup-cortexa9hf-vfp-neon-poky-linux-gnueabi
```

Using IceSSL with Platform CAs

To use the `IceSSL.UsePlatformCAs` property you will need to install the `ca-certificates` package as part of your image. Since OpenSSL does not have its default `CAfile` (where IceSSL looks for the default platform CAs) set to the certificate bundle installed from `ca-certifcates` you will need to set an additional IceSSL property:

```
IceSSL.CAs=/etc/ssl/certs/ca-certificates.crt
```

Using the Ice Sample Programs

Sample programs are available in the zeroc-ice/ice-demos repository:

```
git clone -b 3.7 https://github.com/zeroc-ice/ice-demos.git
cd ice-demos
```

The Ice demos for C++ and Python are located in the `cpp11`, `cpp98` and `python` directories. Review the build instructions for more information.

Back to Top ^