Wayland Linux BSP

- Wayland Linux BSP: Prerequisites
- Running the CGI Player Demo Application on Linux with a Wayland Display Server
- Building Own Application for Linux with a Wayland Display Server

Wayland Linux BSP: Prerequisites

Hardware: Raspberry Pi 5 Board

CGI Studio has been tested on the **Raspberry Pi 5** Board.

CMake

In order to build Candera, the Meta build system CMake needs to be installed with at least version mentioned here.
CMake is able to create the cross compilation make files that can be used with GCC (cross-compile) compiler.

Linux Toolchain (BSP)

- SD card image was prepared with Raspberry Pi Imager.
- Raspberry Pi OS based on Debian Trixie was tested with a Debian Trixie build machine.
  - Install crossbuild-essential-arm64

```
su
apt-get -y update && \
apt-get -y install crossbuild-essential-arm64
```

  - Make the SD card accessible in your build machine.
    Copy the /usr/lib and /usr/Include folders from the SD card to your build machine.
    This can also be done with e.g. scp.

```
#find the mount point of the SD card created with Raspberry Pi Imager
su
mkdir /opt/raspberry
mkdir /opt/raspberry/sysroot64
mkdir /opt/raspberry/sysroot64/usr
cd /opt/raspberry/sysroot64/usr
```

```
cp -ar <mount point of the sd card>/usr/lib  <mount point of the sd
card>/usr/include .
```

> SD Card image, SDK and cross-compiler/s are not part of the delivery and have to be provided by the customer.

Hardware: NXP i.MX8QMMEK Board

CGI Studio has been tested on **NXP i.MX8QMMEK** board.

CMake

In order to build Candera, the Meta build system CMake needs to be installed with at least version mentioned here.
CMake is able to create the cross compilation make files that can be used with GCC (cross-compile) compiler.

Linux Toolchain (BSP)

- Linux version 5.4.3-lts-lf-5.4 (GCC version 9.2.0), Poky 5.4.3 (Zeus) was used to test this release with Weston 7.0.0 (Wayland 1.17.0).
- SD card image and SDK was prepared following the NXP’s Linux user guide.
- Graphic driver: Vivante (OpenGL ES 2.0)

> SD Card image, SDK and cross-compiler/s are not part of the delivery and have to be provided by the customer.

Known Limitations

In case open source graphic driver is used (e.g. Etnaviv) following limitations could occur due to driver limitations:

- issue with shader/s due to limited number of vectors
- MSAA was not supported

Hardware: STM32MP157C-DK2 board

CGI Studio has been tested on STM32MP157C-DK2 board.

CMake

In order to build Candera, the Meta build system CMake needs to be installed with at least version mentioned here.
CMake is able to create the cross compilation make files that can be used with GCC (cross-compile) compiler.

Linux Toolchain (BSP)

- Linux version 5.10.10 (GCC version 9.3.0), ST OpenSTLinux - Weston - (A Yocto Project Based Distro) 3.1-openstlinux-5.10-dunfell-mp1-21-03-31 (dunfell) was used to test this release with Weston 8.0.0 (Wayland 1.18.0).
- SD card image and SDK was prepared following the STM's Linux user guide.
- Graphic driver: Vivante (OpenGL ES 2.0)

> SD Card image, SDK and cross-compiler/s are not part of the delivery and have to be provided by the customer.

Known Limitations

In case open source graphic driver is used (e.g. Etnaviv) following limitations could occur due to driver limitations:

- issue with shader/s due to limited number of vectors
- MSAA was not supported

Hardware: NXP i.MX 8MN board

CGI Studio has been tested on NXP i.MX 8MN board.

CMake

In order to build Candera, the Meta build system CMake needs to be installed with at least version mentioned here.
CMake is able to create the cross compilation make files that can be used with GCC (cross-compile) compiler.

Linux Toolchain (BSP)

- Linux version 5.4.70 (GCC version 9.2.0), Poky 5.4 (Zeus) was used to test this release with Weston 9.0.0 (Wayland 1.18.0).
- SD card image and SDK was prepared following the boards user guide.
- Graphic driver: Etnaviv (OpenGL ES 3.0)

> SD Card image, SDK and cross-compiler/s are not part of the delivery and have to be provided by the customer.

Known Limitations

In case open source graphic driver is used (e.g. Etnaviv) following limitations could occur due to driver limitations:

- issue with shader/s due to limited number of vectors
- MSAA was not supported

Hardware: NXP i.MX 8MM board

CGI Studio has been tested on NXP i.MX 8MM board.

CMake

In order to build Candera, the Meta build system CMake needs to be installed with at least version mentioned here.
CMake is able to create the cross compilation make files that can be used with GCC (cross-compile) compiler.

Linux Toolchain (BSP)

- Linux version 5.4.70 (GCC version 9.2.0), Poky 5.4 (Zeus) was used to test this release with Weston 9.0.0 (Wayland 1.18.0).
- SD card image and SDK was prepared following the boards user guide.
- Graphic driver: Etnaviv (OpenGL ES 2.0)

>

> SD Card image, SDK and cross-compiler/s are not part of the delivery and have to be provided by the customer.

Known Limitations

In case open source graphic driver is used (e.g. Etnaviv) following limitations could occur due to driver limitations:

- issue with shader/s due to limited number of vectors
- MSAA was not supported

Running the CGI Player Demo Application on Linux with a Wayland Display Server

Copy the application and the asset to the target in /home/root folder. This can be done by copying the filed to the SD card containing the file system or via ssh (scp).

Logging Output and Key Input

Use TeraTerm or another terminal software to connect to the target via micro-USB (debugger) to receive logging output and use key input.
BaudRate needs to be setup to 115200 8/n/1/n.

Loading and Running the Application

On target via TeraTerm, navigate (change directory) into the /home/root folder and run the applications with following command:

```
./<app-name> <asset-name>.bin
```

Building Own Application for Linux with a Wayland Display Server

Generate Makefiles with CMake

For general information about CGI Studio Build System and CMake properties available to configure Candera applications refer to CGI Studio Build System

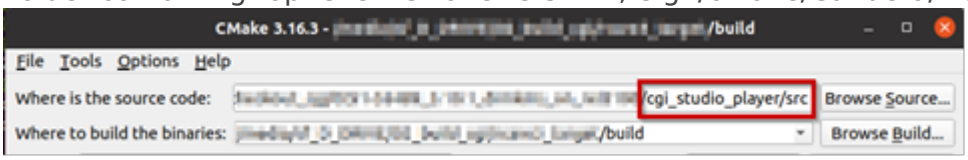
To configure and generate makefiles for compiling the CGI Player, follow below steps:

The first step for cross compiling is to make sure all the environment variables for the Yocto SDK are set!

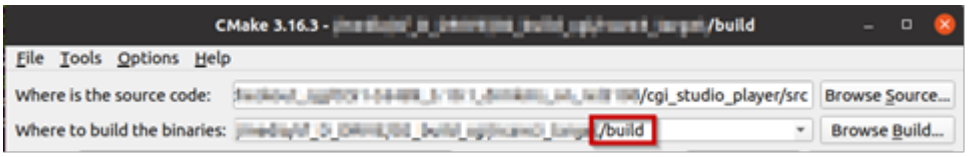
1. Open a terminal.
2. Set the environment. This step is mandatory. Omitting this step will cause building with the host systems compiler instead of cross compiling.

```
$source /<path to the Yocto SDK installation>/environment-setup-<platform specific suffix e.g. aarch64-poky-linux>
```

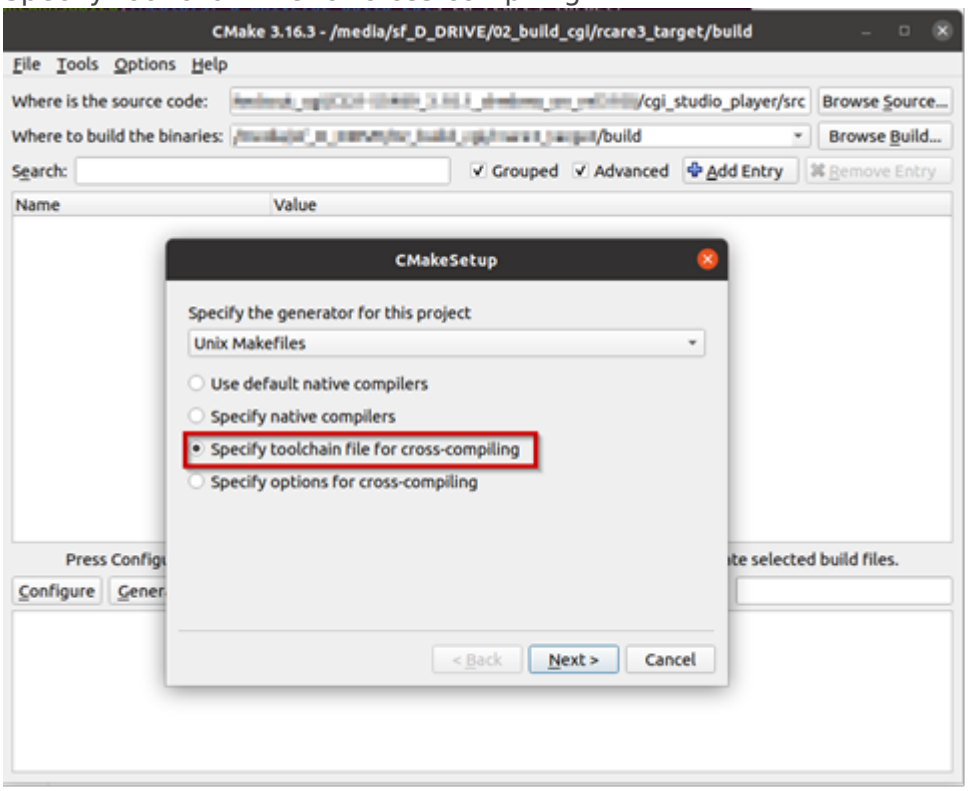
3. Important! From the same console with the set environment start the CMake GUI or run the CMake commandline. We will continue with the CMake GUI in this guide.
4. Point CMake to source directory
   - Folder containing top-level "CMakeLists.txt", e.g. /cmake/Candera/Player



5. Point CMake to binary directory
   - May not exist, create folder with CMake if necessary
   - Root directory for generated makefiles / solutions
   - Root directory for build artifacts



6. Configure
   - Specify Unix Makefiles as generator for the project.
   - Specify tool-chain file for cross-compiling.



7. Select the toolchain-file from the delivery (adapt for your toolchain installation and application specific needs)

```
/cgi_studio_devices/src/Wayland/ToolchainFiles/GCC-toolchain-Wayland-Linux.txt
```

   If Vulkan support is required, use the Vulkan-specific toolchain file.

```
/cgi_studio_devices/src/Wayland/ToolchainFiles/GCC-toolchain-Wayland-Vulkan-Linux.txt
```

8. Adapt the CMake settings, if necessary, e.g.:
   - CMAKE_BUILD_TYPE-> Debug or Release.
   - Enable/disable features.
   - Press Configure.
9. Press Generate.

Building Application

> Make sure that the environment variables for the Yocto SDK are set in the terminal! Either reuse the terminal used to start CMake for building the Player or call the environment setup script again.

```
$source /<path to the Yocto SDK installation>/environment-setup-<platform specific suffix e.g. aarch64-poky-linux>
```

1. Open CMake GUI and configure the project (e.g Player, CourierSampleApp, CitDemoApp) according to Generate Makefiles with CMake.
2. After generating your cmake files, navigate to the specified build directory (binary directory created in CMake GUI) and type "make" to start the build or “make -j” to build with multithreading.
   With some hypervisors, the Linux VM may get unresponsive when using the –j option. Reducing the number of used threads to the number of available cores improves the responsiveness of the VM during the build.

> For CourierSampleApp and CitDemoApp asset file must be created with a matching SCHost.dll as the SCHost.dll for the CourierSampleApp and CitDemoApp contain specialized Widgets.

Vulkan Packaging Workflow

After a successful Vulkan build:

1. Build and copy ANGLE Vulkan Libraries:
   Build your own ANGLE Vulkan libraries for your target platform:
   - libEGL.so
   - libGLESv2.so

   These libraries must match your target architecture.
   Copy into the same directory where the Player executable is located (the build output directory).

2. Run Packaging Script
   From the build directory, execute the `package_player.sh` script. After execution, a folder named **vulkan** will be created.
   Inside this folder you will find **Player.run** this is the only required output package.

3. Run the packaged Player using `./Player.run Your_Asset.bin`