Target setup for DRM/KMS

This document describes the setup required to build and run CGI Studio applications with DRM/KMS device package.
It is explained how to run the DRM/KMS target demo application part of the release package, how to setup the build environment and how to build own applications for DRM/KMS.

Table of Contents

- DRM/KMS BSP
  - DRM/KMS: Prerequisites
  - Running the CGI Player Demo Application on DRM/KMS
  - Building Own Application for DRM/KMS

DRM/KMS BSP

DRM/KMS: Prerequisites

Hardware: Raspberry Pi 4

CGI Studio has been tested on **Raspberry Pi 4** 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)

- Create an SD card image with Raspberry Pi Imager - Raspberry Pi OS 64 Legacy, 64-bit (Debian Bullseye). Make sure to activate ssh access. This can be done with the Raspberry Pi Imager advanced options (strg+shift+x).
- Start the target
- On Target install the *-dev packages needed for your application plus

```
apt-get -yqq update && \
    apt-get -yqq install libinput-dev libudev-dev libgles2-mesa-dev libdrm-dev libgbm-dev l
```

- If an image with display server is used please make sure to boot to cli.
- On host install a VM / WSL with Ubuntu 20.04 LTS (Focal Fossa)

```
apt-get -yqq update && \
    apt-get -yqq install python3 build-essential crossbuild-essential-armhf crossbuild-esse
```

  Install CMake
  Copy /usr/lib and /usr/include from target SD Card to /opt/raspberry/bullseye/sysroot or /opt/raspberry/bullseye/sysroot64 if you use the 64 bit version.

- Compiler/s used for this release is GCC (GNU 9.4.0).

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

Hardware: iMX8QM

CGI Studio has been tested on Toradex Apalis **iMX8QM** 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)

- Yocto 6.6.0 built at Candera according to Toradex manuals.
- Image and SDK were prepared by the Candera.
- The GCC compiler is not part of the delivery. It is contained in the SDK that has to be provided by the customer.
- Compiler/s used for this release is GCC (GNU 11.4.0).

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

Hardware: R-Car E3

CGI Studio has been tested on **R-Car E3** System Evaluation 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)

- The SDK used for this release is based on Yocto Poky 2.4.3.
- SD card image and SDK were prepared by according to the RCar-E3 user guide.
- The GCC compiler is not part of the delivery. It is contained in the SDK that has to be provided by the customer.
  Compiler/s used for this release is
- GCC (GNU 7.3.0)

>

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

Hardware: R-Car M3N

CGI Studio has been tested on **R-Car M3N** Customer 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)

- Yocto Poky 3.1.11 provided by the customer was used to test this release .
- Image and SDK were prepared by the customer.
- The GCC compiler is not part of the delivery. It is contained in the SDK that has to be provided by the customer.
- Compiler/s used for this release is GCC (GNU 9.3.0).

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

Running the CGI Player Demo Application on DRM/KMS

Copy the application and the asset to the target. This can be done by copying the filed to the SD card containing the system, via ssh or with an USB flash drive.

Loading and running the application

Note that no other application that uses graphics hardware can be running when using the Player, the CourierSampleApp or the CitDemoApp.
If your Yocto image is built with Weston or X you must shut these down before running the application.

Player can be run by using the command:

```
$ /PathToPlayer/Player /PathToAsset/AssetName.bin
```

> If touch input handling is enabled, touch devices must be assigned to seat so that display identifier matches seat number. For example, display-0 must be assigned to seat0. Refer to udev documentation for more information on how to assign devices to seats.

Building Own Application for DRM/KMS

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.

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

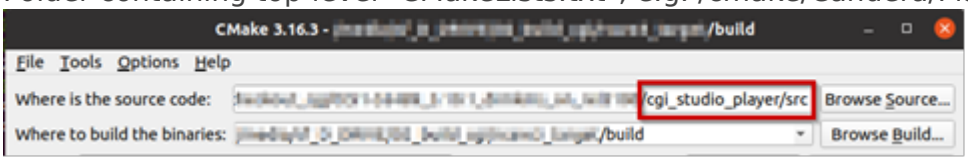
To configure and generate makefiles for compiling the CGI Player, follow below steps:

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

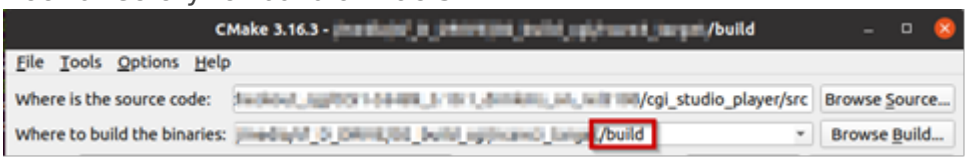
1. Open a terminal.
2. Set the environment. This step is mandatory. Omitting this step will cause building with the host systems compiler instead of cross compiling.

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

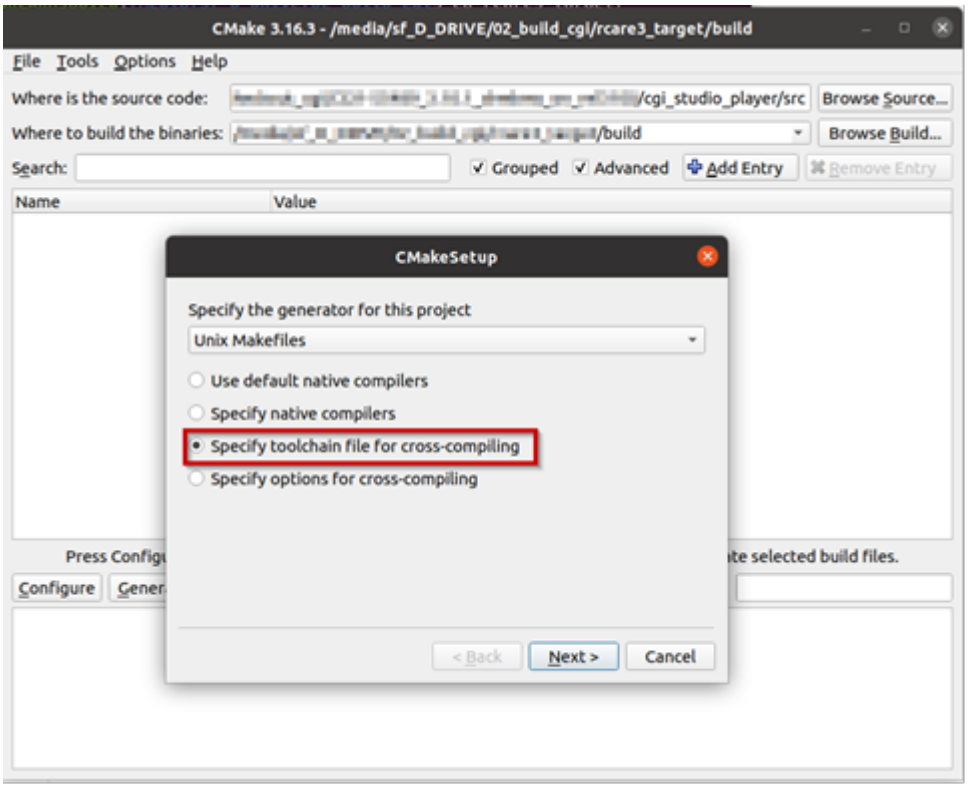
3. Important! From the same console with the set environment start the CMake GUI or run the CMake commandline. We will continue with the CMake GUI in this guide.
4. Point CMake to source directory
   - Folder containing top-level "CMakeLists.txt", e.g. /cmake/Candera/Player



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/DrmKms/ToolchainFiles/GCC-toolchain-<platform>-DrmKms-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.