Building Own Application for Fbdev

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.
   Dolphin 5 (Yocto based SDK)

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

   Allwinner T507 (Buildroot based SDK)

   ```
   export SDKTARGETSYSROOT=/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/aarch64-buildroot-linux-gnu/sysroot
   export PATH=/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/bin:$PATH
   ```

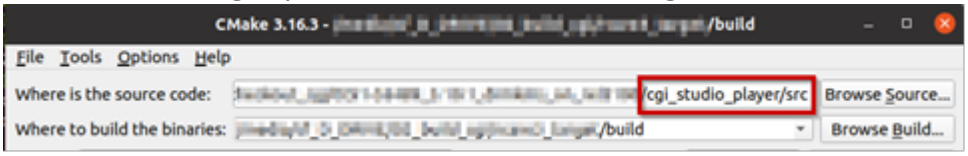
3. Important! From the same console with the set environment start the CMake GUI or run the CMake commandline. We will continue with the CMake GUI in this guide.
   Dolphin 5 (Yocto based SDK)

```
cd <build directory>
cmake-gui .
```

Allwinner T507 (Buildroot based SDK)

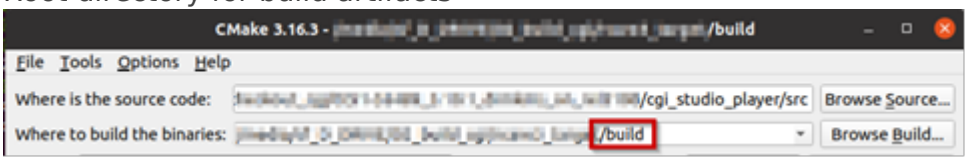
```
cd <build directory>
# Environment variables for sdk
export SDKTARGETSYSROOT="/home/candera/Desktop/OKT507-linux-
sdk20/out/t507/okt507/longan/buildroot/host/aarch64-buildroot-linux-gnu/sysroot"
export PATH="/home/candera/Desktop/OKT507-linux-
sdk20/out/t507/okt507/longan/buildroot/host/bin:$PATH"
export SYSROOT="$SDKTARGETSYSROOT"
export SDKSYSROOT="$SYSROOT"
export PKG_CONFIG="/usr/bin/pkg-config"
export PKG_CONFIG_SYSROOT_DIR="$SYSROOT"
export PKG_CONFIG_PATH="$SYSROOT/usr/lib/pkgconfig"
export CXX="/home/candera/Desktop/OKT507-linux-
sdk20/out/t507/okt507/longan/buildroot/host/bin/aarch64-linux-gnu-g++ --
sysroot=$SYSROOT"
export CC="/home/candera/Desktop/OKT507-linux-
sdk20/out/t507/okt507/longan/buildroot/host/bin/aarch64-linux-gnu-gcc --
sysroot=$SYSROOT"
export LD="/home/candera/Desktop/OKT507-linux-
sdk20/out/t507/okt507/longan/buildroot/host/bin/aarch64-linux-gnu-ld --
sysroot=$SYSROOT"
cmake-gui .
```

4. Point CMake to source directory
   - Folder containing top-level "CMakeLists.txt", e.g. /cmake/Candera/Player

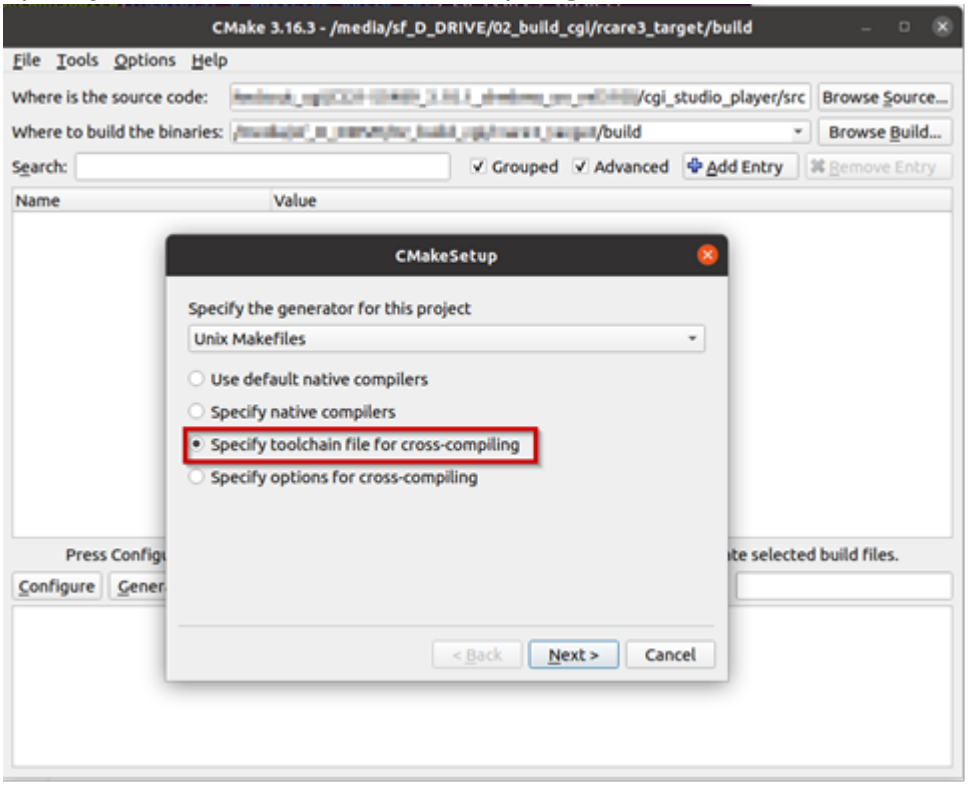


5. Point CMake to binary directory
   - May not exist, create folder with CMake if necessary
   - Root directory for generated makefiles / solutions

- 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/Fbdev/ToolchainFiles/GCC-toolchain-<platform>-Fbdev-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/Buildroot 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.

Dolphin 5 (Yocto based SDK)

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

Allwinner T507 (Buildroot based SDK)

```
# Environment variables for sdk
export SDKTARGETSYSROOT="/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/aarch64-buildroot-linux-gnu/sysroot"
export PATH="/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/bin:$PATH"
export SYSROOT="$SDKTARGETSYSROOT"
export SDKSYSROOT="$SYSROOT"
export PKG_CONFIG="/usr/bin/pkg-config"
export PKG_CONFIG_SYSROOT_DIR="$SYSROOT"
export PKG_CONFIG_PATH="$SYSROOT/usr/lib/pkgconfig"
export CXX="/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/bin/aarch64-linux-gnu-g++ --sysroot=$SYSROOT"
export CC="/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/bin/aarch64-linux-gnu-gcc --sysroot=$SYSROOT"
export LD="/home/candera/Desktop/OKT507-linux-sdk20/out/t507/okt507/longan/buildroot/host/bin/aarch64-linux-gnu-ld --sysroot=$SYSROOT"
```

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.

Revision #8
Created 2025-10-29 07:07:51 UTC by Lernbeiss Alexander
Updated 2025-12-11 05:46:10 UTC by Kanai Tomoaki