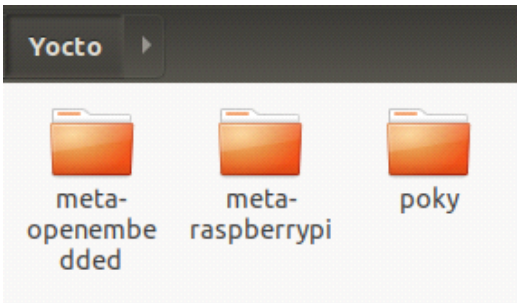
Prototype OpenCV

Thursday, July 16, 2020 8:00 PM

Relevant steps.

1. Cloned meta-openembedded to Yocto folder.



2. Source the oe-init-build-env of your poky folder.
3. Modified bblayers.conf.

```
BBLAYERS ?= " \
  /home/project2/git/Yocto/poky/meta \
  /home/project2/git/Yocto/poky/meta-poky \
  /home/project2/git/Yocto/poky/meta-yocto-bsp \
  /home/project2/git/Yocto/meta-raspberrypi \
  /home/project2/git/Yocto/meta-openembedded/meta-oe \
  "
```

4. Added relevant opencv libraries to local.conf

```
LICENSE_FLAGS_WHITELIST = "commercial"
BB_NUMBER_THREADS = "2"
PARALLEL_MAKE = "-j 2"
SERIAL_CONSOLES = "115200;ttyAMA0"
CORE_IMAGE_EXTRA_INSTALL += "opencv libopencv-core-dev libopencv-imgproc-dev opencv-dev "
```

5. Compile again the file system

   bitbake core-image-base

6. Unpack the just compiled file system into /var/nfs/rootfs.
7. Create script to support cross-compilation.

   bitbake core-image-base -c populate_sdk

8. Run the generated .sh script at build/tmp/deploy/sdk to generate files for cross compilation.

Create code in Eclipse

```
#include <stdio.h>
#include <opencv2/opencv.hpp>

using namespace cv;

int main(int argc, char **argv) {
    Mat rgb_image;
    Mat yuv_image;

    printf("Loading original RGB image ...\n");
    rgb_image = imread("imagejpg.jpg", IMREAD_UNCHANGED);
    // Confirm image was loaded correctly
    if(!rgb_image.data )
    {
        printf("There was a problem loading the RBG image. Aborting!\n");
        return -1;
    }
    int img_height = rgb_image.size[0];
    int img_width = rgb_image.size[1];
    printf("Image loaded. Height %d. Width %d\n", img_height, img_width);

    printf("Converting image to YUV ..\n");
    yuv_image = rgb_image.clone();
    cvtColor(rgb_image, yuv_image, COLOR_BGR2YUV);

    printf("Saving converted image to new file ...\n");
    FILE * output_file = fopen("imageyuv.yuv", "wb");
    size_t bytes_written = fwrite(yuv_image.data, 1, 3*img_height*img_width, output_file);
    printf("File imageyuv.yuv written with %ld bytes\n", bytes_written);

    return 0;
}
```

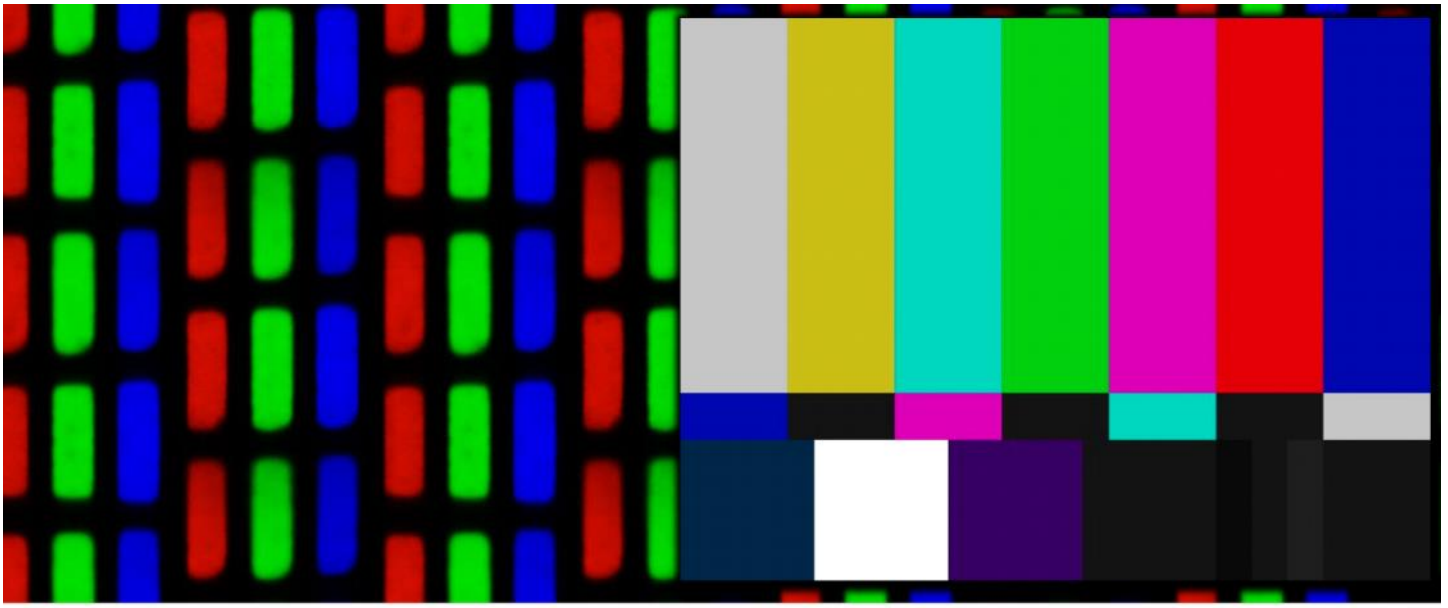
Modified include directories and target link libraries in CMakeLists.txt

```
cmake_minimum_required (VERSION 2.8.1)

######## Project settings ########
PROJECT(rgb_yuv_opencv)
SET(LICENSE "TBD")

######## Build and include settings ########
include_directories(
    inc
    /opt/poky/3.0.3/sysroots/aarch64-poky-linux/usr/include/opencv4
)

link_directories(
    ${LINK_DIRECTORIES}
)


file(GLOB SOURCES
    "src/*.cpp"
)

add_executable(
    rgb_yuv_opencv

    ${SOURCES}
)

TARGET_LINK_LIBRARIES(
    rgb_yuv_opencv
    opencv_core
    opencv_imgcodecs
    opencv_imgproc
    opencv_highgui
)

######## Install targets ########
INSTALL(TARGETS rgb_yuv_opencv
    RUNTIME DESTINATION usr/bin
)
```

Compile, transfer and run from RPI4.

```
root@raspberrypi4-64:~# ./rgb_yuv_opencv
Loading original RGB image ...
Image loaded. Height 480. Width 640
Converting image to YUV ..
Saving converted image to new file ...
File imageyuv.yuv written with 921600 bytes
```

Result file observed from rawpixels.net



Select RAW data: Browse... imageyuv.yuv width: 640 height: 480 offset: 0 flip h: flip v: invert: zoom: 1

Predefined format: YUV444p

Pixel Format: YUV Ignore Alpha: Alpha First:

bpp1: 8 bpp2: 8 bpp3: 8 bpp4: 0 Litle Endian:

Pixel Plane: Packed alignment: 1 subsamplig H: 1 subsamplig V: 1