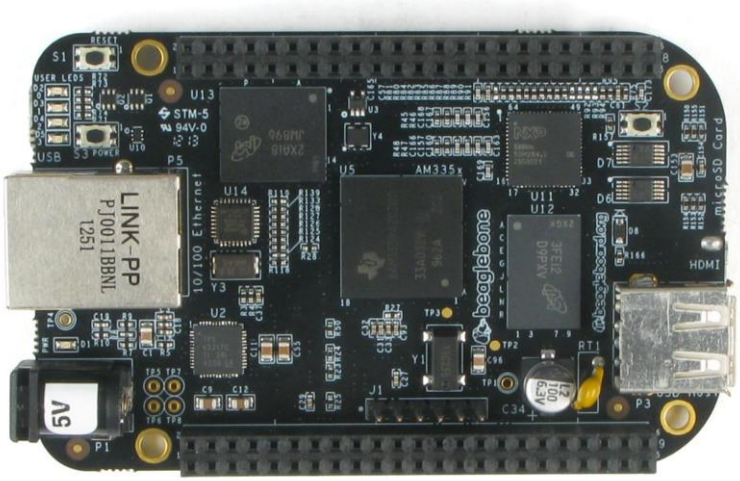
Recording Webcam Videos with the BeagleBone Black



by Josh Vazquez, Matthew Fraser, Cody Rossiter, and Duji Tufail

1. Plug your webcam directly into the BeagleBone. This may seem obvious, but some may forget about the USB port on the BeagleBone and instead plug it in directly into the host machine.

   The BeagleBone likely cannot supply enough power to your USB webcam by itself, so you should plug in AC power to the board. Without it, you may encounter issues such as unreliable capture, video corruption, or low framerate.

2. Make sure your webcam is showing up with the `lsusb` command, and that when you type `ls /dev/video*` from the root directory that `/dev/video0` shows up.

   Good output:

```
root@dtufail-beagle:/# lsusb
Bus 001 Device 002: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
root@dtufail-beagle:/#
root@dtufail-beagle:/#
root@dtufail-beagle:/#
root@dtufail-beagle:/#
root@dtufail-beagle:/# ls /dev/video0
/dev/video0
root@dtufail-beagle:/#
```

   If not, then your webcam probably isn't being detected. Try 1) Plugging in the AC power, 2) Replugging the webcam, 3) Rebooting the BBB. 4) If it's still not working, try plugging another peripheral device into the BBB and run `lsusb` to ensure the USB port is working properly. Another possibility is the webcam may not be supported.

3. For this tutorial, the following packages need to be directly installed onto your BeagleBone (`sudo apt-get install` on the BBB):
   - `libv4l-dev`: for use with Derek Molloy's boneCV program
   - `v4l-utils`: for checking camera capabilities and configuring the camera
   - `ffmpeg`: for converting RAW video files into usable formats

4. Download Derek Molloy's boneCV and place the folder in a directory of your choice on your BBB:
   https://github.com/derekmolloy/boneCV

5. Enter the folder and run `./build` to prepare boneCV for use.

6. Ensure boneCV is set to the correct settings to match your webcam.
   a) Type the following command in to view your webcam's specification:
   `v4l2-ctl --all | less`

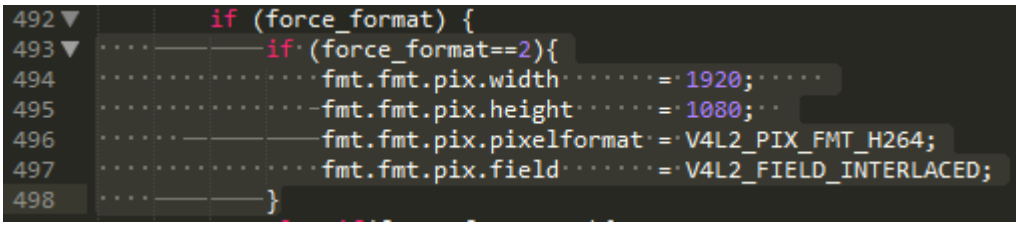
   The output looks like this for the Logitech C270:

```
Driver Info (not using libv4l2):
        Driver name   : uvcvideo
        Card type     : UVC Camera (046d:0825)
        Bus info      : usb-musb-hdrc.1.auto-1
        Driver version: 3.8.13
        Capabilities  : 0x84000001
                Video Capture
                Streaming
Format Video Capture:
        Width/Height  : 640/480
        Pixel Format  : 'YUYV'
        Field         : None
        Bytes per Line: 1280
        Size Image    : 614400
        Colorspace    : SRGB
Crop Capability Video Capture:
        Bounds      : Left 0, Top 0, Width 640, Height 480
        Default     : Left 0, Top 0, Width 640, Height 480
        Pixel Aspect: 1/1
Video input : 0 (Camera 1: ok)
Streaming Parameters Video Capture:
        Capabilities     : timeperframe
        Frames per second: 30.000 (30/1)
:
```

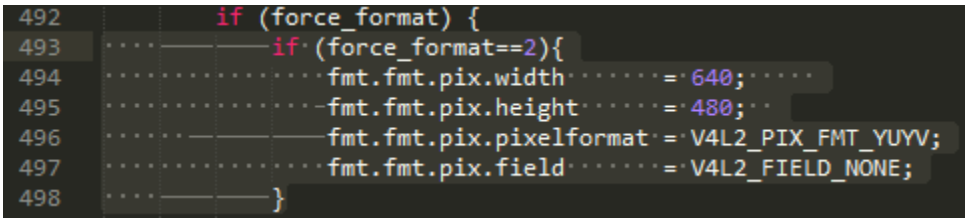
   b) Edit the capture.c file (specifically the code highlighted in the figure below) to match your webcam's settings. After modifying it, run `./build` to produce a new executable.
   Refer to http://linuxtv.org/downloads/v4l-dvb-apis/pixfmt.html for a full list of options for the `field` and `pixelformat` variables.

For the Logitech C270, one would edit the file in the following way:

Before:



After:



7. To record video using the settings that you specified in capture.c, for 10 seconds with a camera that records at 30 frames per second, all into a file named output.raw, the following command can be used:

   ```
   ./capture -F -c 300 -o > output.raw
   ```

   Your camera may light up at this point if it so happens to have an indicator. If you are getting a timeout error, go back into capture.c and reduce the resolution.

   If it worked, you should end up with a .raw video file that has a size greater than zero. If not, then you may have to experiment more with the pixel format settings.

8. Your RAW file needs to be converted to a usable video format. This is where ffmpeg comes in.

BoneCV conveniently includes an ffmpeg script that converts a RAW encoded in H264 to an MP4 file titled `output.mp4`, which can be run by typing in `./raw2mpg4`

It is by default:

```
ffmpeg -f h264 -i output.raw -vcodec copy output.mp4
```

You may need to tinker around with this script to find the right video settings to match your settings (i.e. changing H264 to another format).

9. Your video file outputted by ffmpeg should be viewable if all worked out correctly! Remember, a lot of this depends on having the right settings in `capture.c` and your ffmpeg script.

10. Now that you can record from the terminal, it's easy to add this functionality to your C or C++ program. For example:

```
char *args[] = {"boneCV/capture", "-d", "/dev/video0", "-F", "-c",
    numFrames, "-o", (char *) 0 };
execvp(args[0], args);
```

Since the capture program sends frames to standard output, you can redirect this to a file by placing the following before the exec statement:

```
int fd = open(myOutputFilePath.c_str(), O_RDWR | O_CREAT, S_IRUSR | S_IWUSR);
dup2(fd, 1);
close(fd);
```

Similarly, using execvp() you can craft an ffmpeg command.

References:

Molloy, D. [DerekMolloyDCU]. (2013, May, 25). Beaglebone: Video Capture and Image Processing on Embedded Linux using OpenCV [Video file]. Retrieved from http://www.youtube.com/watch?v=8QouvYMfmQo