Virtual-Cam Hardware Abstraction Layer

The virtual-cam hal is an alternative hardware abstraction layer to the lpc2106-cmucam3 hal that allows for prototyping of cc3 software on a PC. This is very similar to a CMUcam3 simulator. The virtual-cam hal will load frames stored in a directory in the same way that the CMUcam3 normally loads from frames from the camera. All normal cc3 functions are then performed on a virtual fifo. Additional debugging text will be displayed at run time that normally does not appear when the lpc2106-cmucam3 hal is used. This notifies the user of LED states, and low-level cc3 commands that are called. Serial input and output is replaced by the application's standard input and output.

Compiling

To compile code for the virtual-cam, simply type hal=virtual-cam when you run make. For example:

```
projects/hello_world :> make hal=virtual-cam
```

This will generate a hello_world_virtual-cam executable file.

To make clean, you will still need to specify the hal. For example to clean hello_world:

```
projects/hello_world :> make hal=virtual-cam clean
```

Running a virtual-cam Project

The virtual-cam will read images from a directory specified by the CC3_VCAM_PATH environment variable. The path should be relative to the directory where the virtual-cam executable is called. For example in a bash terminal you might run a virtual-cam file like:

```
projects/hello_world :> ls
hello_world_virtual-cam.exe  Makefile main.c my_img_dir

projects/hello_world :> export CC3_VCAM_PATH=my_img_dir

projects/hello_world :> ./hello_world_virtual-cam.exe
```

The virtual-cam directory contains sample images that will be fed into the virtual camera frame buffer. The images need to be in PPM format and numbered "IMGxxxxx.PPM" where the x's represent the frame number. The virtual-cam hal will read in images each time cc3_load_frame() is called in increasing order starting from "IMG00000.PPM". Once no more images are available, the virtual-hal will panic and stop. These images can be created using the ppm-grab project. By default, the **ppm-grab project** stores raw, high resolution, uncompressed images in the correct format to be used by the virtual-cam hal.

Simulation Issues

One main simulation issue is that timing and external input from GPIO or buttons is currently not supported. In many cases, the virtual-cam hal will just always return true or false to allow your application to continue. When compiling with the virtual-cam hal, **VIRTUAL_CAM** will be defined so that your application can switch certain regions of execution based on the current hal. For example to disable a button wait when using the virtual-cam you could use:

```
#ifndef VIRTUAL_CAM
   while(!cc3_read_button());
#endif
```

or if you want to enable more debugging you could write something like:

```
#ifdef VIRTUAL_CAM
   write_debugging_data_to_file();
#endif
```

Sample Images

You can find a zip archive with a few virtual-cam sample images on the Sample Images page. For more images, use the ppm-grab project that comes with the CC3 distribution.