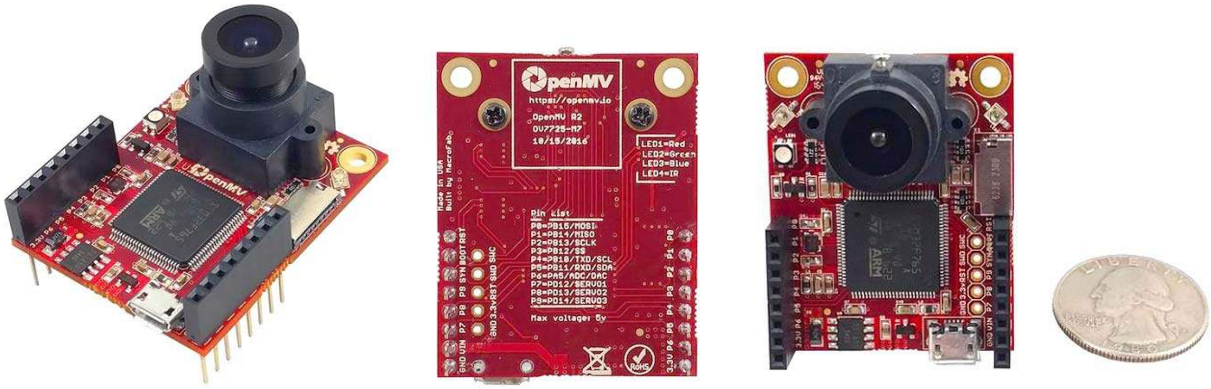
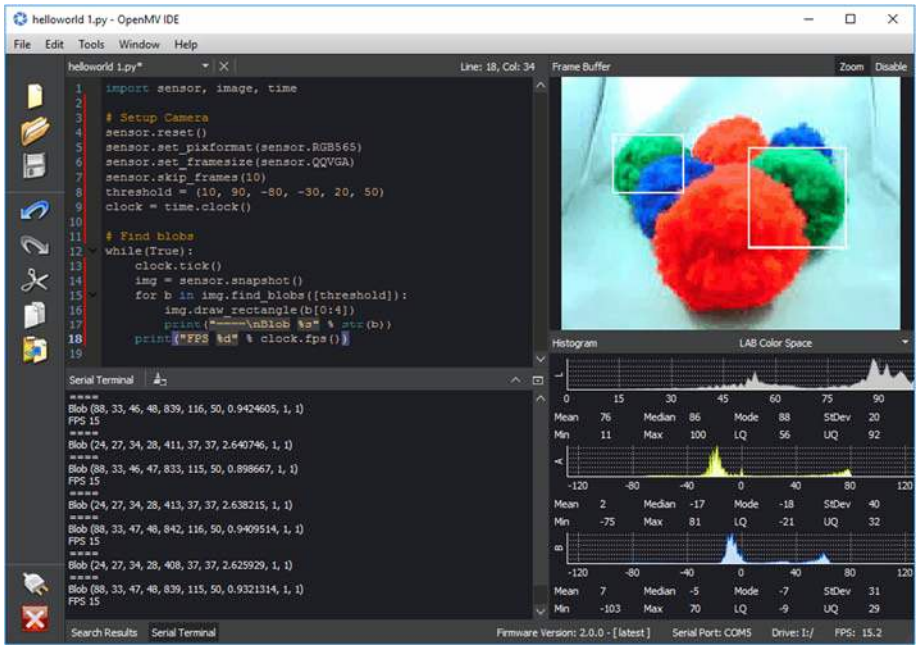
OpenMV Cam M7

SKU 102990763



Description

The OpenMV Cam is a small, low power, microcontroller board which allows you to easily implement applications using machine vision in the real-world. You program the OpenMV Cam in high level Python scripts (courtesy of the MicroPython Operating System) instead of C/C++. This makes it easier to deal with the complex outputs of machine vision algorithms and working with high level data structures. But, you still have total control over your OpenMV Cam and its I/O pins in Python. You can easily trigger taking pictures and video on external events or execute machine vision algorithms to figure out how to control your I/O pins.



Applications

The OpenMV Cam can be used for the following things currently (more in the future):

•Frame Differencing

You can use Frame Differencing on your OpenMV Cam to detect motion in a scene by looking at what's changed. Frame Differencing allows you to use your OpenMV Cam for security applications.

•Color Tracking

You can use your OpenMV Cam to detect up to 32 colors at a time in an image (realistically you'd never want to find more than 4) and each color can have any number of distinct blobs. Your OpenMV Cam will then tell you the position, size, centroid, and orientation of each blob. Using color tracking your OpenMV Cam can be programmed to do things like tracking the sun, line following, target tracking, and much, much, more. Video demo here.

•Marker Tracking

You can use your OpenMV Cam to detect groups of colors instead of independent colors. This allows you to create color makers (2 or more color tags) which can be put on objects allowing your OpenMV Cam to understand what the tagged objects are. Video demo here.

•Face Detection

You can detect Faces with your OpenMV Cam (or any generic object). Your OpenMV Cam can process Haar Cascades to do generic object detection and comes with a built-in Frontal Face Cascade and Eye Haar Cascade to detect faces and eyes.

•Eye Tracking

You can use Eye Tracking with your OpenMV Cam to detect someone's gaze. You can then, for example, use that to control a robot. Eye Tracking detects where the pupil is looking versus detecting if there's an eye in the image.

•Optical Flow

You can use Optical Flow to detect translation of what your OpenMV Cam is looking at. For example, you can use Optical Flow on a quad-copter to determine how stable it is in the air.

•QR Code Detection/Decoding

You can use the OpenMV Cam to read QR Codes in it's field of view. With QR Code Detection/Decoding you can make smart robots which can read labels in the environment. You can see our video on this feature here.

•AprilTag Tracking

Even better than QR Codes above, the OpenMV Cam M7 can also track AprilTags at 160x120 at up to about 12 FPS. AprilTags are rotation, scale, shear, and lighting invariant state-of-the-art fidicual markers. We have a video on this feature here.

•Edge/Line Detection

You can preform edge detection via either the Canny Edge Detector algorithm or simple high-pass filtering followed by thresholding. After you have a binary image you can then use the Hough Detector to find all the lines in the image. With edge/line detection you can use your OpenMV Cam to easily detect the orientation of objects.

•Template Matching

You can use template matching with your OpenMV Cam to detect when a translated pre-saved image is in view. For example, template matching can be used to find fiducials on a PCB or read known digits on a display.

•Image Capture

You can use the OpenMV Cam to capture up to 320x240 RGB565 (or 640x480 Grayscale) BMP/JPG/PPM/PGM images. You directly control how images are captured in your Python script. Best of all, you can preform machine vision functions and/or draw on frames before saving them.

•Video Recording

You can use the OpenMV Cam to record up to 320x240 RGB565 (or 640x480 Grayscale) MJPEG video or GIF images. You directly control how each frame of video is recorded in your Python script and have total control on how video recording starts and finishes. And, like capturing images, you can preform machine vision functions and/or draw on video frames before saving them.

Finally, all the above features can be mixed and matched in your own custom application along with I/O pin control to talk to the real world.

Specification

Processor	ARM®32-bit Cortex®-M7 CPU w/ Double Precision FPU 216 MHz (462 DMIPS) Core Mark Score: 1082 (compare w/ Raspberry Pi Zero: 2060)
RAM Layout	128KB .DATA/.BSS/Heap/Stack 384KB Frame Buffer/Stack (512KB Total)
Flash Layout	32KB Bootloader 96KB Embedded Flash Drive 1920KB Firmware

	(2MB Total)
Supported Image Formats	Grayscale RGB565 JPEG
Maximum Supported Resolutions	Grayscale: 640x480 and under RGB565: 320x240 and under Grayscale JPEG: 640x480 and under RGB565 JPEG: 640x480 and under
Lens Info	IR Cut Filter: 650nm(removable)
Electrical Info	All pins are 5V tolerant with 3.3V output. All pins can sink or source up to 25mA. P6 is not 5V tolerant in ADC or DAC mode. Up to 120mA may be sinked or sourced in total between all pins. VIN may be between 3.6V and 5V. Do not draw more than 250mA from your OpenMV Cam's 3.3V rail.
Dimensions	45(L) x 36(W) x 30(H) mm
Weight	16g

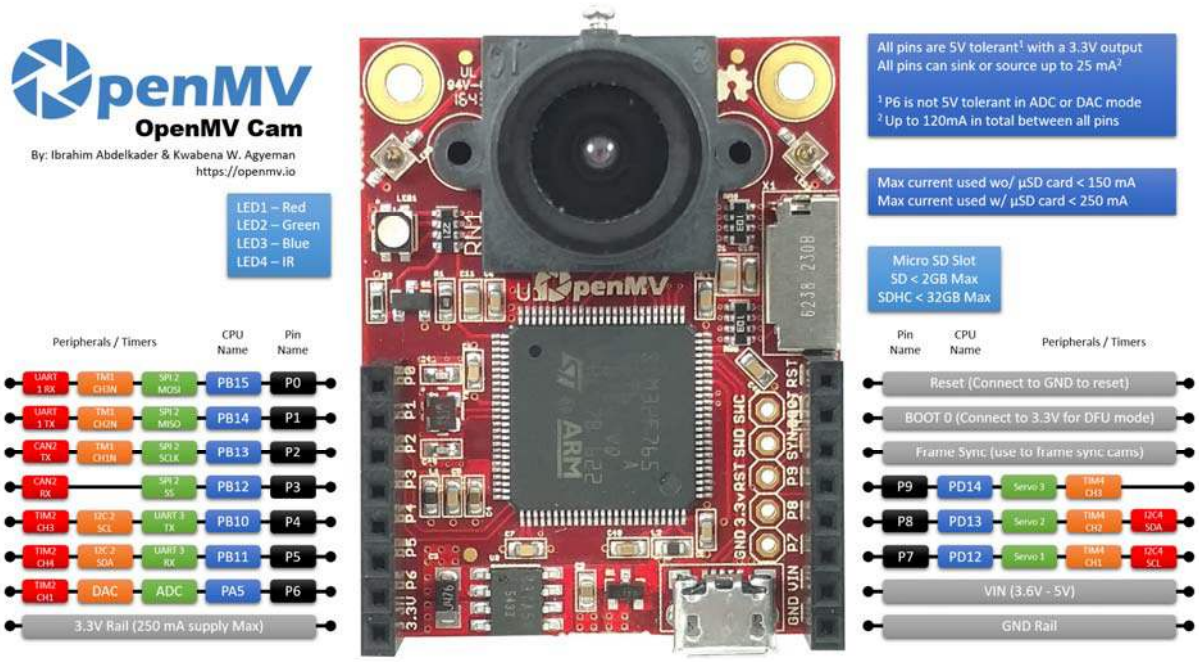
Power Consumption

Idle - No µSD Card	110mA @ 3.3V
Idle - µSD Card	110mA @ 3.3V
Active - No µSD Card	190mA @ 3.3V
Active - µSD Card	200mA @ 3.3V

Temperature Range

Storage	-40°C to 125°C
Operating	-20°C to 70°C

Pinout



Note: For this product, we provide customized service for building boards without headers.