holOS system architecture

1. Initialize system (in order)
   1. Export all GPIO’s (\_\_init\_\_.exportGPIO)
   2. Define all GPIO’s (\_\_init\_\_.defineGPIO)
   3. Start moisture sensor monitoring routine
   4. Start battery monitoring routine
   5. Establish and initialize the virtual server that facilitates communication to and from the Arduino
      1. This has been written as a separate subroutine that can be called by all other programs (SERIAL.py)
   6. Once connected to Arduino, flash all LED’s a few times to make sure they work and notify user that startup is underway
   7. Turn the busy LED on to show its still initializing
   8. Try to connect with host computer (connectUDP)
   9. If connected via UDP to host, sync time so all data sets are consistently named
   10. Make sure camera is connected
   11. Ramp up laser
   12. Start laser diode current monitoring routine
   13. Start operation button monitoring routine
   14. Turn the busy LED off and turn on the ready LED
2. Start monitoring the UDP channel for commands
   1. Pump on/off
   2. Valve 1 on/off
   3. Valve 2 on/off
   4. Sample side valves on/off
   5. Valve 3 on/off
   6. Status query
   7. Live video of what the camera is seeing
      1. From what I read on [stack exchange](https://stackoverflow.com/questions/6187456/tcp-vs-udp-on-video-stream), UDP seems the best to ensure that the most current images always make it through. Because TCP is a stream, any lost packets must be retransmitted before the most current ones can show (i.e. high latency potential)
      2. Look at this [gitHub](https://github.com/Maggeych/vimbaGrabber) as a possible vimba SDK to grab videos and prep them for a live stream
   8. Start recording
   9. Stop recording
   10. Auto DAQ sequence
   11. Change fps

Host computer side OS

1. Terminal port UDP communication
2. GUI to display live stream and status information, as well as buttons to send commands

External ADC notes

1. UDOO x86 has 2 UART groups so one can be used for this ADC (8 channel 12-bit)
2. The ADC requires excitation voltage of 2.7 to 5.5 volts
   1. Connect directly to a power pin from the Arduino since its to be on all the time