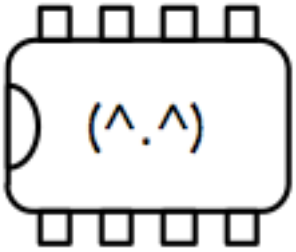
ZenAI Community of Robots ch3



www.ZENMCU.com

Draft 1 2023-01-19

zenai-0.0a

Yes, this is yet another brief article in this seemingly never ending list of experiments and wacky ideas. Since you have been eagerly reading my previous articles you are aware that most of my experiments have been at best only partial successes. Briefly I will summarize my current options:

To enable robots to locate each other:

- I can continue to use I2c driver and sensor chips, hopefully finding some in packages that I can actually solder (or get a reflow hot-plate to solder them).
- I can go back to using an MCU as an I2C slave, directly driving an RGB LED and using it as a very crude color sensor (assuming I can improve my implementation to the point of being reliable).

To communicate between robots:

- I can use the I/R UART, which is known to work.
- I can continue to develop a low bandwidth LED to LED communications link.
- I can use some form of RF data link (which dramatically increases the BOM and complexity).

Obviously the original idea of an MCU driving an RGB LED, sensing color with it, and using it for communications is very appealing because it provides three different capabilities in a simple, inexpensive package.

Of course, the TCS34725 was very usable as a color sensor (though it is end of life / obsolete / unavailable). And the VEML6040 is reasonably capable as well (and is just at the limit of my soldering capabilities). So the color sensors seem attractive for their (apparent) accuracy and sensitivity.

The KTD2026 is still an unknown. I have soldered exactly one of them, at the cost of 3 or four likely ruined. I still need to test that it actually drives the RGB LED as anticipated. (The code is written, just need time to sit down and execute it).

Update - I tested the LED driver, and while I was able to communicate with it seemingly successfully, I was never able to light the LEDs. I went through the data sheet extensively, but it is unclear regarding bit 2 of reg 0 (en/rst), so I am unsure about it. Regardless, I tried a variety of tests and was not able to light any LED. Yes, I checked that the LEDs were mounted the right way around, thank you. I guess am going to need to need to use my new i2cdriver with bus capture thingie to verify the i2c traffic.

So here is a probably circular rethink to try to help determine the next thing to try.

Try again to do it all with an MCU dedicated and an RGB LED, where the MCU is dedicated solely to that purpose? From a hardware perspective this is inexpensive and simple. Two components; the MCU and the LED (Ok and a decoupling cap. And maybe a QWIIC connector if it gets fancy.)

Sensing incident illumination is effectively charging the junction like a cap, then timing how long it takes for the charge to drain down to a specific voltage. Using the digital 1/0

detection of an I/O pin is a simple and basically free way to do it.

Timing via loop counting is trivially simple, but needs to account for the fCPU (or only use a fixed fCPU). Timing via a timer is fCPU insensitive, but is arguably a little more work to set up the timer.

One of the difficulties I had when doing this previously was trying to capture to memory, perform averaging, and look in the debugger at a breakpoint. This is obviously not a very efficient way to glean the information that I need. When I did the color sensors I used the hif and printed a continuous list of all colors, and even color coded them (when using the UART hif ... picocom interprets ansi escapes, zenterm doesn't yet, so they look awful over USB hif in zentem).

The plan in x4th has been to watch memory locations and graph their values. I need to implement a simple variant of that. Just a ddd app that reads fixed addresses over USB and draws the graph, Sysview style. But I need to get a Sysview style graph working on ddd without any use of RayLib. I've been putting that off...

End