Core Memory Interactive Kit - Core64 – Basic User Manual

This description parallels the naming convention and comments in firmware source code.

Power switch ON connects to battery (3 to 7V). Power switch OFF uses USB 5V if connected.

0:

Heartbeat LED ON and begin blinking.

Serial Port is started with default hard-coded debug level verbose (so things like voltages are sent, and all attempts to connect to subsequent devices and results).

Initialize LED Matrix and clear them.

Check battery, 5V, 3V3 rails. Stored as initial start-up voltages. No load.

1:

Begin alternating Display of "Core" & "64" (assumes stock configuration, middle brightness, orange)

Fill in LEDs on bottom row, bits 0-7, binary # corresponds to Mode #. Continue to update as each step of the start-up sequence is run. If the number gets stuck, that is a great error indication. Use RED LED mode.

2:

Check EEPROM settings, configuration, and user stuff

3:

Initialize alternate LED Matrix or Neon Pixel configurations, brightness, default start color per EEPROM settings. Serial Log attempt/success/fail.

Initialize optional/alternate Display Screens per EEPROM settings.

Display S/N ID in binary, briefly on all available screens.

4:

Initialize buttons, Ambient Light Sensor, per EEPROM settings.

POWER ON BUTTON FUNCTIONS - if buttons are active during power up:

"-" dimmest setting

"+" brightest setting

5:

Initialize optional accessories (SD Card, RTC, NFMC...) per EEPROM settings.

6:

Self-Test

Stabilize. Show live voltage levels on LED Matrix for a few seconds. Should all be up at mid-level scale. Report in debug port. Expected battery level range per EEPROM battery type setting. Color of battery voltage column corresponds to battery type. (EEPROM settable duration)

Will need no-load and load numbers to get the best chance at an estimate. Could show a battery scale, marked 3 to 7.5V, no load, steady mid-load, steady full-load, and then what the real-time value is. Could be an overlay screen.

Touch any key to skip/exit this display mode.

AS03: Self-test mode, full, with no user interaction required (but can be optional).

If EEPROM indicate full test completed >5 times, skip this test, go to next APP.

Core 0 drive voltages profiles: Battery, RPP P-FET, 5V0, 3V3, Top Transistor, Bottom Transistor, ENABLE

Voltage/Current monitor and self-test

Need a way to facilitate hardware debug of the matrix/cores. Idea: monitor 3.3V rail to analog input to confirm the matrix is drawing current when commanded, and the right mount. Or more directly, measure voltage drop across the enable FET, to correlate to current flow? Can it discern between no wires, one wire, two wires, and more than two wires in the matrix being enabled? This would give feedback to me and kit builders about what is happening.

WILL THIS BE HELPFUL FOR SELF TEST MODE AT START-UP?

A0 (spare 3) 5V0\_MON. Monitor nominal 5V rail changes relative to loading of LEDS, LCD, NEON PIXELS. What can be inferred and tested?

The 5 more analog channels to Column 0 monitor for learning and self-test. What can be inferred about the other matrix wires by measuring the common 3V3 rail?

AS04: Self-test mode, partial/quick. Swipe left-to-right on Halls to jump to USR\_APP 1. Auto jump to next APP.

7:

Show the real state of the cores on the LED Matrix for a few seconds. (EEPROM settable duration)

Touch any key to skip/exit this display mode.

8:

End of the start-up sequence. Jump to the first demo mode, scrolling.

9:

First Demo mode, repeating sequence of demos, each runs for several seconds.

"M" DGAUSS menu

"-" previous demo

"+" next demo

"S" settings

AU01: Scrolling text. Add owners name to EEPROM to use in "\_\_\_\_ [Heart] Core Memory!" as default scroll. Default name "I". Adjustable color, store to EEPROM to persist. Stop/start scrolling with S button, Change speed with +/-. Include multi-color symbols in the scroll. If no core memory change has occurred in 10 seconds, keep scrolling text only, and check the Core Memory every 3 seconds until a change is detected to reduce power consumption when it's just scrolling and not being used.

AU02: Animation of how core memory works,