After power-up board (or after reset), firmware flow is this:

1. Initializing the system clock and board [in this routines, program can stuck only if microprocessor is bad/wrong, or external crystal don't oscillate]
2. Initializing application and stopwatch [0% chance to stuck program]
3. Switching ON the Green LED [start of first 0.5sec blink]
4. Initializing of the MCP23008 and I2C [program can stuck, if wrong I2C address are selected by HW (A0, A1)]
5. Waiting for 0.5sec
6. Switching OFF the Green LED [end of first 0.5sec blink]
7. Waiting for 0.5sec
8. Enabling UART interrupt [program can hardly slower if UART communication is active with high data payloads - Master error]
9. Switching ON the Green LED [start of second 0.5sec blink]
10. Waiting for 0.5sec
11. Switching OFF the Green LED [end of second 0.5sec blink]
12. Run to main loop

Based on this flow,

- if I2C communication freeze (bad I2C address, missing jumper J1 or J2 on LPC824 MCU module,...) first blink don't finish, and Green LED still lit On and program is stopped.
- If first blink is gone, second blink can be unfinished only when UART communication is active in high bandwidth (master continuosly send commands into module and don't wait to response).