A picture containing text, sign, clipart

Description automatically generated

DCA Testing

Hardware Events

2023-03-22

Table of Contents

1. Purpose……………………………………………………………………………………………………………………………… 2
2. Overview……………………………………………………………………………………………………………………………. 2
3. Scope…………………………………………………………………………………………………………………………………. x
   1. Startup……………………………………………………………………………………………………… x
   2. Time Sync…………………………………………………………………………………………………. x
   3. Reset Cause ………………………………………………………………………………………….…. x
   4. Errors …………………………………………………………………………………………………….... x
4. Results………………………………………………………………………………………………………………………………. x
5. Analysis……………………………………………………………………………………………………………………………… x
6. **Purpose**

Event handling inside the hardware is important to ensure expected results are occurring. Without the logging of events, there would be no way for an Admin, Dev, User or the server from knowing what the device is doing without shelling in. Without events, there’s only guess and hoping that expected results are occurring.

1. **Overview**

The event handling inside the hardware is modular and allows easy adding and removing of new and old events. While more events will be added in the future such as getvar, setvar etc.. only a basic skeleton currently is available. Current available events are Startup, Time Sync, Reset Cause, and Errors.

* Screenshots/pictures

1. **Scope**

The entirety of the event class will be tested. Each event will be massaged to run so the logging of the event occurs.

3.1 Startup

This event occurs anytime the device powers up or is restarted either forcible or accidentally.

To cause this event to occur, the hardware module will be manually unplugged, and re plugged to power. This will be done at the wall, and not the hardware module itself to prevent exhaustion of the DC Barrel.

3.2 Time Sync

This event occurs ONLY after another event is already sent to the server. A timestamp is always returned from a CmdSend to the server. On recv, if the time differs more than 3 seconds, the hardware will updated it’s internal clock, and send a Time Sync event.

To have this occur, a delay can be injected into each main loop iteration to cross the 3sec resync delay. Another method to make this occur is to unplug the device for a set period of time, once reconnected a startup event will occur, and a Time Sync event will piggyback.

3.3 Reset Cause

The current reset causes available are WDT reset, Brown out 1.8V reset, and Brown out 1.2V Brown out reset.

A method to for this to occur is to make the main loop delay longer than the WDT timeout (8sec). Another method is to temp comment out the WDT Feed function.

3.4 Error

This event occurs when any unexpected errors occur. This can appear as an initialization issue, a return issue, a type cast issue etc.

In an ideal world, this never happens. However, the event is captured and transmitted to the device log to help debug issues. To manually cause an error to occur, I can transmit an unexpected response from the server to cause the device to log an error.

1. **Results**

|  |  |  |  |
| --- | --- | --- | --- |
| Number of Test Cases Planned | Number of Test Cases Executed | Number of Tests Cases Passed | Number of Test Cases Failed |
| 18 | 18 | 18 | 0 |

Tests were carried out for each scope one at a time, three times each, for a total of 18 test cases.

1. **Analysis**

All test cases were passed without issue. Every endpoint returned an expected value and was repeatable. This ensures the user’s experience is as expected every time.

\*There’s an intermitted issue where the memory maps on the embedded system are not initialized as expected. If a large number of events occur, the EventLogBuffer in memory is stored in the wrong location and is sometimes smaller than expected, or overridden. This isn’t an issue in an environment with a solid internet connection, if the connection starts to slow down and the buffer increases in size without sending logs, It acts as a FIFO and first items will be lost.