Communication Protocol Boogie Bungalow

**Master Node -> PC**

Packet format:

Array of 1x29 values, being each value **unsigned long.** Content:

Position 1: MasterNodeUpTime

Positions 2 to 29 (4 values per node): Node\_N\_ID, Node\_N\_UpTime, Node\_N\_Solved, Node\_N\_State,

Codes embedded in the first value of the packets arriving to the pult. These notify important events happening at the master node.

|  |  |
| --- | --- |
| Code | Notification |
|  |  |
| 70 - 120 | The network channel has been set to [code] |
| 1 | The board is asking for a Sync |
| 2 | The board found a communication problem with the antenna so it has been reset to try to fix it |
| 3 | The board found that the network is up but is unstable. The pult should ask if it is a good moment to reset network |
| 4 | A network reset has been requested by gamemaster |
| 5 | The board found that a Node is not updated to the its latest state, so it re-sent the last command for it |
| 6 | Sync successful (init) |
| 7 | Sync failed (timeout). Setting it to New Game |
| 8 | An Electronics reset has been requested by gamemaster |
|  |  |
| 11-17 | The board found the node [code] has a worrying delay, so it made it restart |
|  |  |
|  |  |
|  |  |

**PC -> Master node**

Packet format:

Array of 1x3 values, being each value **char.** Content:

NodeID, NodeSolve, NodeState

The NodeID can be: 1,2,3,4,5,13,21

The NodeSolve can be 1 or 0 if the node must solve the riddle or not.

The NodeState can be any char, it depends on the node itself and how many nodes. The Separee, for example, has 0=LedsOff, 1=R, 2=G, 3=B, 4=StartSequence (see next point for detail on each node)

If the node id is 0, then the message is for the Master node (typically to answer to a notification by the Master node). When the message is for the master node, we have:

|  |  |
| --- | --- |
| NodeState | Request |
| 1 | The GM forced an Electronics soft reset |
| 2 | The GM forced an Network soft reset |

**Nodes Modes**

Here you will find what code you have to send to each node, to launch each action. The modes are set as numbers (0-9).

*NOTE: If at some point more than 10 states on a single node are needed, I recommend changing the modes of that node to letters, so that you can have +20 modes still with one single character (and you do not need to re-program the COM protocol).*

The codes sent **by the PC** have the format **NodeID, Solved, State**, where:

* NodeID can be: 01,02,03,04,05,13,21
* Solved can be: 00,01 (NotSolved / Solved)
* State: a letter; it depends to each node… See table below.

The codes sent by the PC are then read by the Master node, that reads to which node it goes (NodeID), and forwards the message to that node in the format **Solved, State** (the destination is enclosed in the header of the message, so no need to send the NodeID (like the address in a letter))

**Note: When a riddle is solved (so it goes NodeID,01,XX), the state is ignored because it launches the “solve procedure”. So I will put XX to let that clear**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Node** | **PC sends** | **Master sends** | **Action** | |
| Separee (01) | 01,00,00 | 0,0 | Not solved, LEDs off (Riddle off) | |
| 01,00,01 | 0,1 | Not solved, LEDs Red | |
| 01,00,02 | 0,2 | Not solved, LEDs Green | |
| 01,00,03 | 0,3 | Not Solved, LEDs Blue | |
| 01,00,04 | 0,4 | Not Solved, Init sequence | |
| 01,01,XX | 1,0 | Launches Solving sequence. No come back! | |
| Dance/DontDance (02) | 02,00,00 | 0,0 | Not solved, Bulbs off (Riddle off) | |
| 02,00,01 | 0,1 | Not solved, Dance ON (Green) | |
| 02,00,02 | 0,2 | Not solved, Dance OFF (Red) | |
| 02,00,03 | 0,3 | Not Solved, Init sequence | |
| 02,01,XX | 1,0 | Launches Solving sequence. No come back! | |
| Sparkastchen (03) | 03,00,00 | 0,0 | Not solved, LEDs off (Riddle off) | |
| 03,00,01 | 0,1 | Not solved, Riddle on | |
| 03,00,02 | 0,2 | Not solved, Init sequence | |
| 03,01,XX | 1,0 | Launches Solving sequence. No come back! | |
| 4Drinks (04) | 04,01,00 | 1,0 | Launches Solving sequence. No come back! | |
| DancefloorCtrl (05) | 05,00,00 | 0,0 | DanceFloor OFF | |
| 05,00,01 | 0,1 | DanceFloor ON | |
| Jukebox  (015 = 13DEC) | 13,00,00 | 0,0 | Not solved, LEDs off (Riddle off) | |
| 13,00,01 | 0,1 | Not solved, solve Song 1 | |
| 13,00,02 | 0,2 | Not solved, solve Song 2 | |
| 13,00,03 | 0,3 | Not Solved, solve Song 3 | |
| 13,00,04 | 0,4 | Not Solved, solve Song 4 | |
| 13,00,05 | 0,5 | Not Solved, Init sequence | |
| 13,01,XX | 1,0 | Launches Solving sequence. No come back! | |
| WasserHahn / WorkSchedules  (025 = 21DEC) | 21,00,00 | 0,0 | Not solved, Block Tap | WasserHahn Riddle |
| 21,00,01 | 0,1 | Not solved, Enable Tap |
| 21,00,02 | 0,2 | Not solved, Solve Tap Riddle |
| 21,01,XX | 1,0 | Launches Solving sequence\*. No come back! (both riddles) | Working Schedules Riddle |

\* The WasserHahn riddle is solved when the state 2 is received. From then onwards, the working schedule is to be done, and the door of the WasserHahn riddle will open itself if the players close it for any reason. When the solve code is received, the WasserHahn stays solved and now the working schedules actions also happen. Since both riddles have to be solved sequentially, there is no need to keep states of both simultaneously 😊