SSKC GUI Explanation

Samuel Reddekop

July 21, 2024

1 Introduction

This document outlines how to interface with the Nextion GUI designed for the SSKC Race Director Console.
The GUI consists of two "pages". First is the startup splash screen:

Figure 1: Startup Page

The second is the interface page:

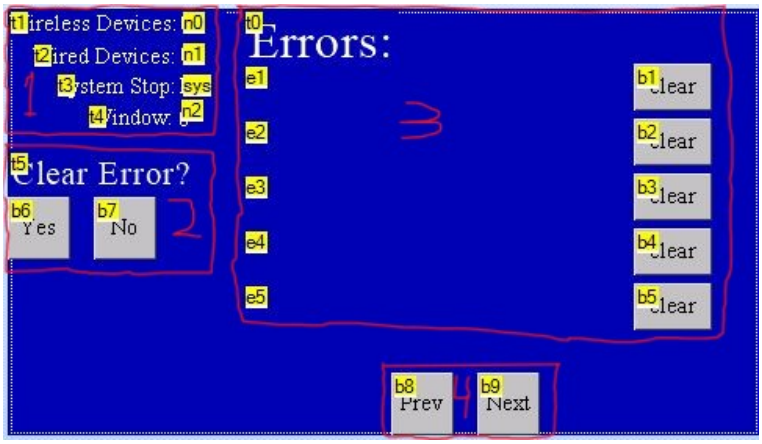

Figure 2: Interface Page In Design Window

The interface page consists of four sections:

1. Globally important info:
   a. Number of wireless devices
   b. Number of Wired Devices
   c. Emergency/System Stop indicator
   d. Window Number Indicator
2. Error Clear popup
3. Error window and corresponding clear buttons
4. Next and Previous buttons to advance error window.

2 Using the interface

On power up, the startup splash page will load. After the master console has fully booted, the interface page will load and will look similar to this:

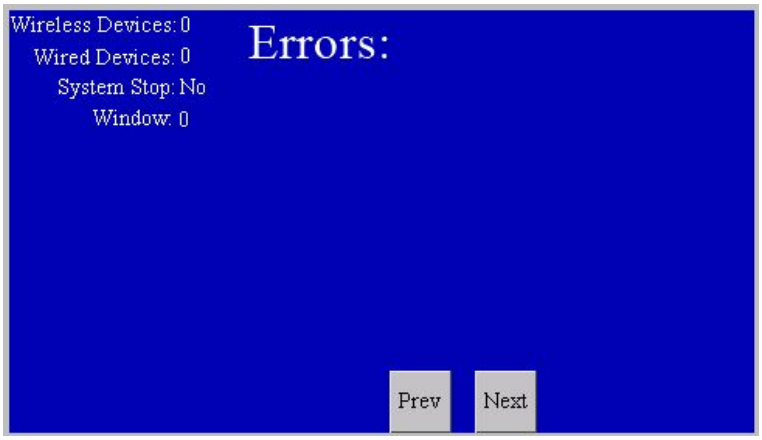

Figure 3: Startup Page

When an error is received, it will be populated in one of the error text boxes with a corresponding clear button appearing as well, similar to below:

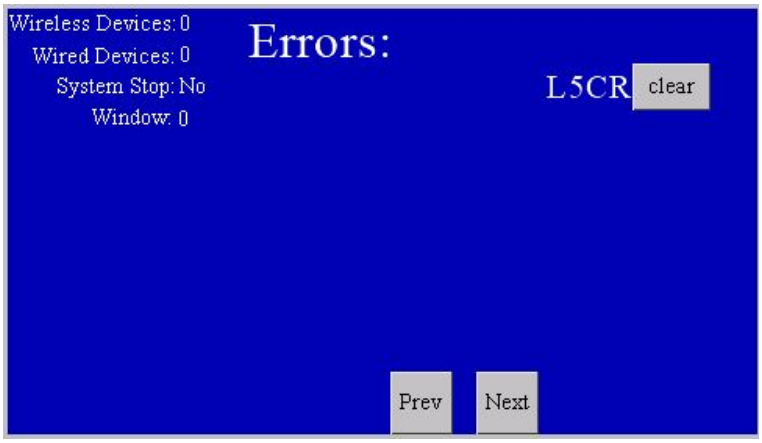

Figure 4: Error received

To clear an error, touch the corresponding clear button. The error will be highlighted and area two of the screen will appear prompting to clear the error.

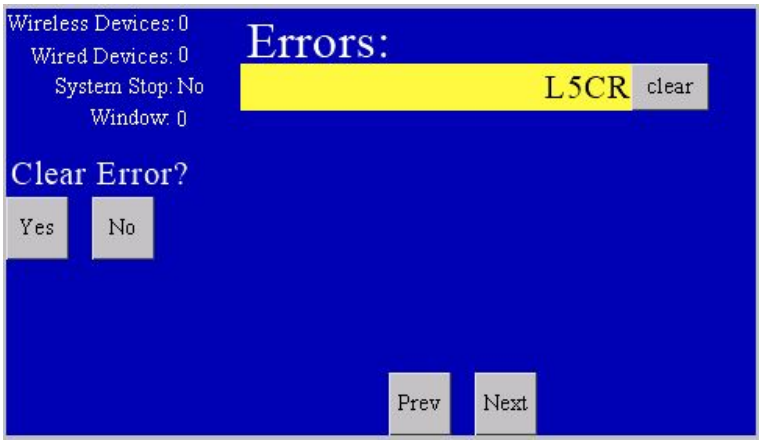

Figure 5: Error Clear Prompt

After yes is selected, that entry will be deleted. If there are other errors on screen below the deleted one, they will be resorted and moved up.

3 Using HMI with a microcontroller

Nextion makes it relatively easy to implement and control a GUI. It is very similar to a Microsoft Visual Basic design environment, where you can place different objects and then control those objects via different parameters like a struct datatype.

The Nextion is controlled through a UART interface with the following specs:
Baud: 115200bps
Parity: None - Cannot be changed unfortunately
StopBits: One

All commands are sent as ASCII strings terminated with three 0xFF characters.

3.1 Sequence of operation

On power up, the first page (page 0) loads. It waits for two seconds, then sends 0x50 0x48 or "PS" to signal that it's on the start splash screen. The micro controller then must issue the command "page 1" (no quotations)(0x70 0x61 0x67 0x65 0x20 0x31 0xff 0xff 0xff) to advance to the interface page.

When the interface page first loads, it will configure and hide any hidden elements, no need to send commands for that. It will send the following: 0x50, 0x48, 0x57, 0x01. This corresponds to "PH" and "W1" to signal that the display is on the "home/interface" page and that the microcontroller should load the first five errors corresponding to Window one.

Each error textbox is named e1-e5. Because these are text objects, you address the .txt parameter of those elements. As an example, say you want to make e1 say "Hello World", you would send e1.txt="Hello World" (include quotations) (0x65 0x31 0x2E 0x74 0x78 0x74 0x3D 0x22 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x22 0xff 0xff 0xff). When adding text to an error textbox you must also make the corresponding clear button appear as I could not make this happen automatically when text is put into the textbox. This is done using the vis command. For example, for e1 you want button b1 to become visible when you add text to e1, so you would send (vis, b1,1) (0x76 0x69 0x73 0x20 0x62 0x31 0x2C 0x31 0xff 0xff 0xff) after updating e1.

When the clear button is pressed for the corresponding error textbox, it will make visible area two of the screen and highlight the corresponding error message to confirm for the operator. When the 'No' button is pressed it simply unhighlights the error textbox and hides area two. No message sent to the microcontroller. When yes is pressed, it will send 'C' followed by a value between one and five corresponding to e1-e5, then the corresponding clear button will hide and the error textbox will be erased. It is then expected that the microcontroller will sort it's error messages to fill in that blank spot.

When the Prev or Next button is pressed an internal variable (va0) is incremented or decremented. Then the message 'W' followed by a value between one and 10 is sent to signal what Window number should be displayed. Note that on the display there is the number n2 for what Window is currently being displayed. This value must be updated by the microcontroller as a way to visually confirm that the UART link is working properly. Being that it is an integer number object you address it's .val parameter. Say you want n2 to display 10. You would send n2.val=10 (0x6E 0x32 0x2E 0x76 0x61 0x6C 0x3D 0x31 0x30 0xff 0xff 0xff), note that each digit must be sent as a separate ascii character.

The Nextion displays also will send responses to various errors and events that happen. These can be found on section seven of the Nextion Instruction Set webpage.