Model Railroad System

2.2.1

Generated by Doxygen 1.8.17

Chapter 1

Example Siding CTC Panels

This folder contains a collection of example siding CTC panels

The Azatrax_Siding.tcl file contains a simple passing siding using Azatrax devices for block occupancy detection, turnout activation, and turnout point state detection. It uses code from MRD2_Block.tcl and SR4_MRD2_Switch.tcl.

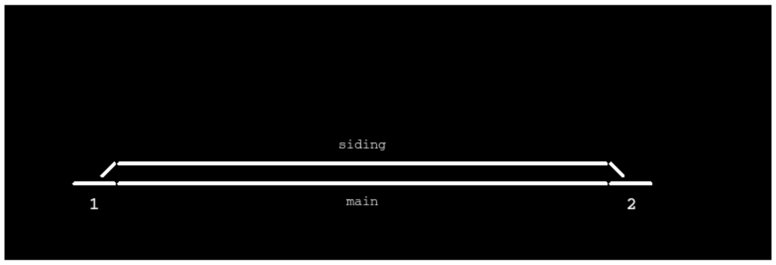

Figure 1.1 Schematic of the track work

This is the Schematic of the track work, which is a simple siding with a passing siding.

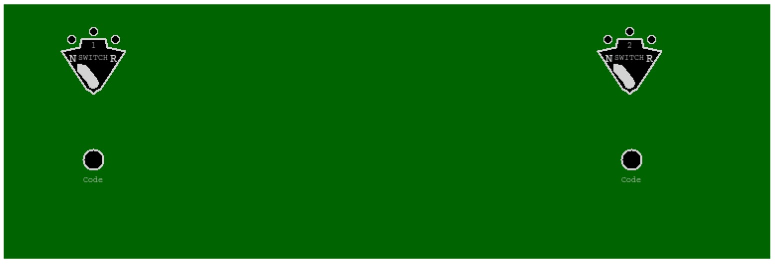

Figure 1.2 CTC Panel for the siding

This is the CTC Panel for the siding, which is simply a switch plate and code button for each of the turnouts.

Using the Abstract Data Types (Classes) MRD2_Block and SR4_MRD2_Switch almost all of the code is embeded in these Abstract Data Types, which makes the code here very simple. Specificly, we create a MRD2_Block object for the main and siding tracks:

```
# Two straight blocks, with MRD2s
MRD2_Block main   -sensorsn 0200001234
MRD2_Block siding -sensorsn 0200001235
```

Then create a connection to the SR4 for use by the two turnouts:

```
# One SR4 for turnout control
SR4 turnoutControl1 -this [Azatrax_OpenDevice 0400001234 \
                           $::Azatrax_idSR4Product]
```

Then we create a SR4_MRD2_Switch object for each turnout, linking them to the main and siding tracks.

```
# Two turnouts, one at each end.
SR4_MRD2_Switch switch1 -motorobj turnoutControl1 -motorhalf lower \
      -pointsenseobj turnoutControl1 -pointsensehalf lower \
      -ossensorsn 0200001236 -nextmainblock main -nextdivergentblock siding \
      -plate SwitchPlate1
SR4_MRD2_Switch switch2 -motorobj turnoutControl1 -motorhalf upper \
      -pointsenseobj turnoutControl1 -pointsensehalf upper \
      -ossensorsn 0200001237 -forwarddirection reverse  \
      -nextmainblock main -nextdivergentblock siding \
      -plate SwitchPlate2
```

Then connect the main and siding MRD2_Block objects to the SR4_MRD2_Switch objects.

```
# Connect the siding and main to the switch frogs
main configure -previousblock switch1 -nextblock switch2
siding configure -previousblock switch1 -nextblock switch2
```

Finally, we create a pair of CodeButton objects for the code buttons.

```
# Two Code buttons
ControlPoint code1 -cpname CP1
ControlPoint code2 -cpname CP2
```

When we assembled the track work and control panel, we set the scripts to run the various callback methods: For the Main straight block, the "Occupied Script" would be "main occupiedp". For the Siding it would be "siding occupiedp". For Switches, "State Script" would be set to "switchN pointstate" and "Occupied Script" would be set to "switchN occupiedp". For the switch plates, the "Normal Script" would be "switchN motor normal" and the "Reverse Script" would be "switchN motor reverse". Finally the code buttons would have an "Action Script" of "codeN code". Thus everything is tied together. The main loop 'invokes' the track work elements, which runs the occupency methods.

```
# Main Loop Start
while {true} {
  # Read all AZATRAX state data
  # Invoke all trackwork and get occupicency
  MainWindow ctcpanel invoke Switch2
  MainWindow ctcpanel invoke Main
  MainWindow ctcpanel invoke Siding
  MainWindow ctcpanel invoke Switch1
  update;# Update display
}
# Main Loop End
```

The code buttons will run the switch plate functions which in turn will activate the switch machines.