```
/**
 * @file MainLine.jsx
 * @author Joey Damico
 * @date September 25, 2019
 * @summary React JSX Component Class that is for the entire Pannel
 *
 * @description Extends the React Component Class and is the UI of the
entrie Pannel, this component
 * contains all the other components, and holds the functions that
allows each component to
 * change the back end class for each enterlocking
 */

// Import React Component
import React, { Component } from 'react';
// Import My Own Clock Class which takes care of trains running
import Clock from '../../scripts/Trains/clock.js';
// To Control All The Trains
import MaineLine_CTC from '../../scripts/CTC/mainLine_ctc.js';
// Import My Train Class
import Train from '../../scripts/Trains/train.js';

// Import the Main Line Components
import MainLineTracks from '../Panel/Main_Line/MainLineTracks.jsx';
import Hilburn from '../Panel/Main_Line/Hilburn.jsx';
import SF from '../Panel/Main_Line/SF.jsx';
import WC from '../Panel/Main_Line/WC.jsx';
import RidgewoodJunction from '../Panel/Main_Line/
RidgewoodJunction.jsx';
import Suscon from '../Panel/Main_Line/Suscon.jsx';
import Mill from '../Panel/Main_Line/Mill.jsx';
import WestSecaucus from '../Panel/Main_Line/WestSecaucus.jsx';
import Laurel from '../Panel/Main_Line/Laurel.jsx';

// Import the Bergen County Line Components
import BergenTracks from '../Panel/Bergen_County_Line/
BergenTracks.jsx';
import BT from '../Panel/Bergen_County_Line/BT.jsx';
import PascackJunction from '../Panel/Bergen_County_Line/
PascackJct.jsx';
import HX from '../Panel/Bergen_County_Line/HX.jsx';

// Import the Southern Tier Line Components
import SouthernTierTracks from '../Panel/Southern_Tier_Line/
SouthernTierTracks.jsx';
import Sparrow from '../Panel/Southern_Tier_Line/Sparrow.jsx';
import PA from '../Panel/Southern_Tier_Line/PA.jsx';
import Port from '../Panel/Southern_Tier_Line/Port.jsx';
import BC from '../Panel/Southern_Tier_Line/BC.jsx';
import OV from '../Panel/Southern_Tier_Line/OV.jsx';
```

```
import Howells from '../Panel/Southern_Tier_Line/Howells.jsx';
import Hall from '../Panel/Southern_Tier_Line/Hall.jsx';
import HudsonJunction from '../Panel/Southern_Tier_Line/
HudsonJunction.jsx';
import CentralValley from '../Panel/Southern_Tier_Line/
CentralValley.jsx';
import Harriman from '../Panel/Southern_Tier_Line/Harriman.jsx';
import Sterling from '../Panel/Southern_Tier_Line/Sterling.jsx';


// Create A new Clock for the Game
var clock = new Clock();
// Create the CTC controler for the game, passing it the clock we
created above
var ctc = new MaineLine_CTC(clock);

// Initialize the clock
clock.startClock;


setTimeout(function(){
    ctc.add_train(new Train("[E] 49", "3_yardEast_port", "EAST", 10));
    ctc.add_train(new Train("3", "3_laurel_westEnd", "WEST", 10));
    ctc.add_train(new Train("1", "1_laurel_westEnd", "WEST", 10));
    ctc.add_train(new Train("2", "2_laurel_westEnd", "WEST", 10));
    ctc.add_train(new Train("4", "4_laurel_westEnd", "WEST", 10));
    ctc.add_train(new Train("50", "3_yardHilburn_sf", "EAST", 10));
    ctc.add_train(new Train("[E] SU100", "1_bingo_sparrow", "EAST",
10));
    ctc.occupy_blocks();
}, 1500);


/**
 * The React JSX Component Class for the entire Maine Line Dispatcher
Panel This class is a JSX React Component for the Maine Line Dispatch
Panel,
 * this will control all the other components that make up the pannel.
This also controls the functions that allow each component to change
their respected
 * back end functions.
 */
class MainLine extends Component {

    /**
     * constructor()
     * @summary The Constructor for the MainLine JSX class.
     *
     * All this does is set that state for every thing getting the
information fro the CTC controller, the state here
```

```
     * is used to send to the child components so they can render the
correct information
     *
     * @param props, Required as park of ReactJS, but is not used here
     */
    constructor(props) {
        super(props);
        /**
         * State
         * @summary Object that holds the state or status information
for the component
         *
         * This object holds all the information for everything on the
pannel that is required to display the routes
         * correctly
         */
        this.state = {
            // Southern Tier Interlockings Status
            status_sparrow:
ctc.get_sparrow().get_interlocking_status(),
            status_pa: ctc.get_pa().get_interlocking_status(),
            status_port: ctc.get_port().get_interlocking_status(),
            status_bc: ctc.get_bc().get_interlocking_status(),
            status_ov: ctc.get_ov().get_interlocking_status(),
            status_howells:
ctc.get_howells().get_interlocking_status(),
            status_hall: ctc.get_hall().get_interlocking_status(),
            status_hudson: ctc.get_hudson().get_interlocking_status(),
            status_valley: ctc.get_valley().get_interlocking_status(),
            status_harriman:
ctc.get_harriman().get_interlocking_status(),
            status_sterling:
ctc.get_sterling().get_interlocking_status(),

            // Main Line Interlockings Status
            status_hilburn:
ctc.get_hilburn().get_interlocking_status(),
            status_sf: ctc.get_sf().get_interlocking_status(),
            status_wc: ctc.get_wc().get_interlocking_status(),
            status_ridgewood:
ctc.get_ridgewood().get_interlocking_status(),
            status_suscon: ctc.get_suscon().get_interlocking_status(),
            status_mill: ctc.get_mill().get_interlocking_status(),
            status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status(),
            status_laurel: ctc.get_laurel().get_interlocking_status(),

            // Bergen County Interlocking Status
            status_bt: ctc.get_bt().get_interlocking_status(),
            status_pascack:
```

```
ctc.get_pascack().get_interlocking_status(),
            status_hx: ctc.get_hx().get_interlocking_status(),

            // Main Line Tracks & Symbols
            status_mainLine: ctc.get_mainLine_blocks_status(),
            symbols_mailLine: ctc.get_mainLine_symbols(),
            // Bergen County Track & Symbols
            status_bergenLine: ctc.get_bergen_blocks_status(),
            symbols_bergenLine: ctc.get_bergen_symbols(),
            // Southern Tier Tracks & Symbols
            status_tier: ctc.get_tier_block_status(),
            symbols_tier: ctc.get_tier_symbols()
        };
    }

    /**
     * update_blocks()
     * @summary This function is called every 0.5 Seconds and updates
all the tracks blocks
     *
     * @description When this function is called it call 2 functions
in the CTC controler class.
     * The first one will check find all the routes at each
interlocking and set the correct
     * next block to routed, so the route can be displayed on the
pannel
     * The second will get all the trains current locations and make
those blocks as occupied,
     * to show the correct location of each train on the pannel
     */
    update_blocks = () => {
        // Update All The Routes
        ctc.update_route_blocks();
        // Update All The Trains
        ctc.occupy_blocks();
        // Set the Component State
        this.setState({
            // Main Line Tracks & Symbols
            status_mainLine: ctc.get_mainLine_blocks_status(),
            symbols_mailLine: ctc.get_mainLine_symbols(),
            // Bergen County Tracks & Symbols
            status_bergenLine: ctc.get_bergen_blocks_status(),
            symbols_bergenLine: ctc.get_bergen_symbols(),
            // Southern Tier Tracks & Symbols
            status_tier: ctc.get_tier_block_status(),
            symbols_tier: ctc.get_tier_symbols()
        });
    }
    // ---- END update_blocks() ----
```

```
    /**
     * update_trains()
     * @summary This function is called every 2 Seconds and updates
all the Trains locations
     *
     * @description When this function is called it will call 2
functions in the CTC controler
     * The first function updates the trains allowing them to move to
the next location if the
     * correct time has be spend in their current block
     * The second function updates the interlockings showing if they
are occupied or cleared if the
     * correct time has passed
     */
    update_trains = () => {
        // Allow trains to update their location if possible
        ctc.update_trains();
        // Update the interlockings
        ctc.update_interlockings();
        // Set The State of the Component
        this.setState({
            // Main Line Tracks & Symbols
            status_mainLine: ctc.get_mainLine_blocks_status(),
            symbols_mailLine: ctc.get_mainLine_symbols(),
            // Bergen County Tracks & Symbols
            status_bergenLine: ctc.get_bergen_blocks_status(),
            symbols_bergenLine: ctc.get_bergen_symbols(),
            // Southern Tier Tracks & Symbols
            status_tier: ctc.get_tier_block_status(),
            symbols_tier: ctc.get_tier_symbols(),

            // Southern Tier Interlockings
            status_sparrow:
ctc.get_sparrow().get_interlocking_status(),
            status_pa: ctc.get_pa().get_interlocking_status(),
            status_port: ctc.get_port().get_interlocking_status(),
            status_bc: ctc.get_bc().get_interlocking_status(),
            status_ov: ctc.get_ov().get_interlocking_status(),
            status_howells:
ctc.get_howells().get_interlocking_status(),
            status_hall: ctc.get_hall().get_interlocking_status(),
            status_hudson: ctc.get_hudson().get_interlocking_status(),
            status_valley: ctc.get_valley().get_interlocking_status(),
            status_harriman:
ctc.get_harriman().get_interlocking_status(),
            status_sterling:
ctc.get_sterling().get_interlocking_status(),

            // Main Line Interlockings
            status_hilburn:
```

```
ctc.get_hilburn().get_interlocking_status(),
            status_sf: ctc.get_sf().get_interlocking_status(),
            status_wc: ctc.get_wc().get_interlocking_status(),
            status_ridgewood:
ctc.get_ridgewood().get_interlocking_status(),
            status_suscon: ctc.get_suscon().get_interlocking_status(),
            status_mill: ctc.get_mill().get_interlocking_status(),
            status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status(),
            status_laurel: ctc.get_laurel().get_interlocking_status(),

            // Bergen County Interlockings
            status_bt: ctc.get_bt().get_interlocking_status(),
            status_pascack:
ctc.get_pascack().get_interlocking_status(),
            status_hx: ctc.get_hx().get_interlocking_status(),
        });
    }

    /**
     * componentDidMount()
     * @summary ReactJS function that allows you do set the intervals
for when certin functions are called
     *
     * @description This function sets the intervals for each function
that is called repeadely after a amount of time
     * Will call the update_blocks() function every 0.5 Seconds
     * Will call the update_trains() function every 2 Seconds
     */
    componentDidMount() {
        // update_blocks() Interval [0.5 Seconds]
        this.interval_update_blocks = setInterval(() =>
this.update_blocks(), 500);
        // update_trains() Interval [2 Seconds]
        this.interval_update_trains = setInterval(() =>
this.update_trains(), 2000);
    }
    // ---- END componentDidMount()

    /**
     * componentWillUnmount()
     * @summary ReactJS function that removes the intervals, this is
never called in this program
     *
     * @description This function deletes the intervals that are used
to update the blocks & trains
     * This is never called in this program
     */
    componentWillUnmount() {
        clearInterval(this.interval_update_blocks);
```

```
        clearInterval(this.interval_update_trains);
    }
    // ---- END componentWillUnmount() ----

    /**
     * render()
     * @summary standard React function that draws all the other
interlockings and track components to the screen
     *
     * @description This will draw all the components to the screen to
assemble the pannel, it also passes all the function
     * and information to each components through their properties or
(props)
     */
    render() {
        // Returns the HTML to draw the interlocking and it's current
state to the screen
        return (
            <div>
                {/* SOUTHERN TIER SECTION */}
                {/* Tracks */}
                <SouthernTierTracks
                    blocks={this.state.status_tier}
                    symbols={this.state.symbols_tier}
                />
                {/* Interlockings */}
                <Sparrow
                    status={this.state.status_sparrow}
                    click_sig_2w_1={this.sparrow_click_sig_2w_1}
                    click_sig_2w_2={this.sparrow_click_sig_2w_2}
                    click_sig_2w_3={this.sparrow_click_sig_2w_3}
                    click_sig_2e={this.sparrow_click_sig_2e}
                    throw_sw_1={this.sparrow_throw_sw_1}
                    throw_sw_3={this.sparrow_throw_sw_3}
                />
                <PA
                    status={this.state.status_pa}
                    click_sig_2w_1={this.pa_click_sig_2w_1}
                    click_sig_2w_2={this.pa_click_sig_2w_2}
                    click_sig_4w={this.pa_click_sig_4w}
                    click_sig_2e={this.pa_click_sig_2e}
                    click_sig_4e={this.pa_click_sig_4e}
                    throw_sw_1={this.pa_throw_sw_1}
                    throw_sw_3={this.pa_throw_sw_3}
                />
                <Port
                    status={this.state.status_port}
                    click_sig_2w={this.port_click_sig_2w}
                    click_sig_2e_1={this.port_click_sig_2e_1}
                    click_sig_2e_2={this.port_click_sig_2e_2}
```

```
    throw_sw_1={this.port_throw_sw_1}
/>
<BC
    status={this.state.status_bc}
    click_sig_2w={this.bc_click_sig_2w}
    click_sig_2e={this.bc_click_sig_2e}
    click_sig_4e={this.bc_click_sig_4e}
    throw_sw_1={this.bc_throw_sw_1}
/>
<OV
    status={this.state.status_ov}
    click_sig_2w={this.ov_click_sig_2w}
    click_sig_2ws={this.ov_click_sig_2ws}
    click_sig_2e={this.ov_click_sig_2e}
    throw_sw_1={this.ov_throw_sw_1}
/>
<Howells
    status={this.state.status_howells}
    click_sig_2w={this.howells_click_sig_2w}
    click_sig_2e={this.howells_click_sig_2e}
    click_sig_2es={this.howells_click_sig_2es}
    throw_sw_3={this.howells_throw_sw_3}
/>
<Hall
    status={this.state.status_hall}
    click_sig_2w={this.hall_click_sig_2w}
    click_sig_4w={this.hall_click_sig_4w}
    click_sig_2e={this.hall_click_sig_2e}
    click_sig_4e={this.hall_click_sig_4e}
    throw_sw_1={this.hall_throw_sw_1}
/>
<HudsonJunction
    status={this.state.status_hudson}
    click_sig_2w={this.hudson_click_sig_2w}
    click_sig_2ws={this.hudson_click_sig_2ws}
    click_sig_2e={this.hudson_click_sig_2e}
    click_sig_2es={this.hudson_click_sig_2es}
    throw_sw_1={this.hudson_throw_sw_1}
    throw_sw_3={this.hudson_throw_sw_3}
/>
<CentralValley
    status={this.state.status_valley}
    click_sig_1w={this.valley_click_sig_1w}
    click_sig_2w={this.valley_click_sig_2w}
    click_sig_1e={this.valley_click_sig_1e}
    throw_sw_21={this.valley_throw_sw_21}
/>
<Harriman
    status={this.state.status_harriman}
    click_sig_1w={this.harriman_click_sig_1w}
```

```
    click_sig_1e={this.harriman_click_sig_1e}
    click_sig_2e={this.harriman_click_sig_2e}
    click_sig_3e={this.harriman_click_sig_3e}
    throw_sw_21={this.harriman_throw_sw_21}
    throw_sw_32={this.harriman_throw_sw_32}
/>
<Sterling
    status={this.state.status_sterling}
    click_sig_2w={this.sterling_click_sig_2w}
    click_sig_2ws={this.sterling_click_sig_2ws}
    click_sig_1e={this.sterling_click_sig_1e}
    throw_sw_21={this.sterling_throw_sw_21}
/>


{/* BERGEN COUNTY LINE SECTION */}
{/* Tracks */}
<BergenTracks
    blocks={this.state.status_bergenLine}
    symbols={this.state.symbols_bergenLine}
/>
{/* Interlockings */}
<BT
    status={this.state.status_bt}
    click_sig_2w1={this.bt_click_sig_2w1}
    click_sig_2w2={this.bt_click_sig_2w2}
    click_sig_4w={this.bt_click_sig_4w}
    click_sig_2e={this.bt_click_sig_2e}
    click_sig_4e={this.bt_click_sig_4e}
    throw_sw_1={this.bt_throw_sw_1}
    throw_sw_3={this.bt_throw_sw_3}
    throw_sw_5={this.bt_throw_sw_5}
/>
<PascackJunction
    status={this.state.status_pascack}
    click_sig_2w={this.pascack_click_sig_2w}
    click_sig_4w={this.pascack_click_sig_4w}
    click_sig_2e={this.pascack_click_sig_2e}
    click_sig_4e={this.pascack_click_sig_4e}
    throw_sw_1={this.pascack_throw_sw_1}
    throw_sw_3={this.pascack_throw_sw_3}
/>
<HX
    status={this.state.status_hx}
    click_sig_2w1={this.hx_click_sig_2w1}
    click_sig_2w2={this.hx_click_sig_2w2}
    click_sig_2w3={this.hx_click_sig_2w3}
    click_sig_4w={this.hx_click_sig_4w}
    click_sig_2e={this.hx_click_sig_2e}
    click_sig_4e={this.hx_click_sig_4e}
```

```
    throw_sw_1={this.hx_throw_sw_1}
    throw_sw_3={this.hx_throw_sw_3}
    throw_sw_5={this.hx_throw_sw_5}
/>


{/* MAIN LINE SECTION */}
{/* Tracks */}
<MainLineTracks
    blocks={this.state.status_mainLine}
    symbols={this.state.symbols_mailLine}
/>
{/* Interlockings */}
<Hilburn
    status={this.state.status_hilburn}
    click_sig_2w_1={this.hilburn_click_sig_2w_1}
    click_sig_2w_2={this.hilburn_click_sig_2w_2}
    click_sig_2e={this.hilburn_click_sig_2e}
    click_sig_4e={this.hilburn_click_sig_4e}
    throw_sw_1={this.hilburn_throw_sw_1}
/>
<SF
    status={this.state.status_sf}
    click_sig_2w={this.sf_click_sig_2w}
    click_sig_4w={this.sf_click_sig_4w}
    click_sig_2e={this.sf_click_sig_2e}
    click_sig_4e_1={this.sf_click_sig_4e_1}
    click_sig_4e_2={this.sf_click_sig_4e_2}
    throw_sw_1={this.sf_throw_sw_1}
    throw_sw_3={this.sf_throw_sw_3}
/>
<WC
    status={this.state.status_wc}
    click_sig_2w_1={this.wc_click_sig_2w_1}
    click_sig_2w_2={this.wc_click_sig_2w_2}
    click_sig_4w={this.wc_click_sig_4w}
    click_sig_2e_1={this.wc_click_sig_2e_1}
    click_sig_2e_2={this.wc_click_sig_2e_2}
    click_sig_4e={this.wc_click_sig_4e}
    throw_sw_1={this.wc_throw_sw_1}
    throw_sw_3={this.wc_throw_sw_3}
    throw_sw_5={this.wc_throw_sw_5}
    throw_sw_7={this.wc_throw_sw_7}
/>
<RidgewoodJunction
    status={this.state.status_ridgewood}
    click_sig_2w_1={this.ridgewood_click_sig_2w_1}
    click_sig_2w_2={this.ridgewood_click_sig_2w_2}
    click_sig_4w={this.ridgewood_click_sig_4w}
    click_sig_6w={this.ridgewood_click_sig_6w}
```

```
    click_sig_2e={this.ridgewood_click_sig_2e}
    click_sig_4e={this.ridgewood_click_sig_4e}
    click_sig_6e={this.ridgewood_click_sig_6e}
    throw_sw_1={this.ridgewood_throw_sw_1}
    throw_sw_3={this.ridgewood_throw_sw_3}
    throw_sw_5={this.ridgewood_throw_sw_5}
    throw_sw_7={this.ridgewood_throw_sw_7}
    throw_sw_9={this.ridgewood_throw_sw_9}
/>
<Suscon
    status={this.state.status_suscon}
    click_sig_2w={this.suscon_click_sig_2w}
    click_sig_2e={this.suscon_click_sig_2e}
    click_sig_4w={this.suscon_click_sig_4w}
    click_sig_4e={this.suscon_click_sig_4e}
    throw_sw_1={this.suscon_throw_sw_1}
    throw_sw_3={this.suscon_throw_sw_3}
/>
<Mill
    status={this.state.status_mill}
    click_sig_2w={this.mill_click_sig_2w}
    click_sig_2e={this.mill_click_sig_2e}
    click_sig_4w={this.mill_click_sig_4w}
    click_sig_4e={this.mill_click_sig_4e}
    throw_sw_1={this.mill_throw_sw_1}
    throw_sw_3={this.mill_throw_sw_3}
/>
<WestSecaucus
    status={this.state.status_westSecaucus}
    click_sig_2w={this.westSecaucus_click_sig_2w}
    click_sig_2e={this.westSecaucus_click_sig_2e}
    click_sig_4w={this.westSecaucus_click_sig_4w}
    click_sig_4e={this.westSecaucus_click_sig_4e}
    throw_sw_1={this.westSecaucus_throw_sw_1}
    throw_sw_3={this.westSecaucus_throw_sw_3}
/>
<Laurel
    status={this.state.status_laurel}
    click_sig_2w={this.laurel_click_sig_2w}
    click_sig_4w={this.laurel_click_sig_4w}
    click_sig_8w={this.laurel_click_sig_8w}
    click_sig_10w={this.laurel_click_sig_10w}
    click_sig_6e={this.laurel_click_sig_6e}
    click_sig_12e={this.laurel_click_sig_12e}
    click_sig_4e={this.laurel_click_sig_4e}
    click_sig_8e={this.laurel_click_sig_8e}
    throw_sw_1={this.laurel_throw_sw_1}
    throw_sw_3={this.laurel_throw_sw_3}
    throw_sw_7={this.laurel_throw_sw_7}
    throw_sw_9={this.laurel_throw_sw_9}
```

```
                    throw_sw_11={this.laurel_throw_sw_11}
                    throw_sw_13={this.laurel_throw_sw_13}
                />
            </div>
        );
    }
    // ---- END render() ----

    //
-------------------------------------------------------------------------
----------------------
    // All of the following function are the only way to get the event
handers (below) and passed
    // into each component to access the fuctions in the CTC
controler, it's a very cumbersum way
    // to accomplish this, but its the only way I was able to find. I
would like to change this
    // one day in the future if I find a more streamlined way
    //
-------------------------------------------------------------------------
----------------------

    /* Bergen County Line Event Handlers */
    /* Functions for the HX Interlocking */
    /**
     * hx_click_sig_2w1()
     * @summary The event handler for Signal #2w-1
     */
    hx_click_sig_2w1 = () => {
        // Get the backend function for corresponding signal
        // Passing reference the next blocks
        ctc.get_hx().click_sig_2w1(
            this.state.status_bergenLine.block_pascack_hx_1,
            this.state.status_bergenLine.block_pascack_hx_2
        );
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_click_sig_2w1() ----

    /**
     * hx_click_sig_2w2()
     * @summary The event handler for the Signal #2w2
     */
    hx_click_sig_2w2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hx().click_sig_2w2(
            this.state.status_bergenLine.block_pascack_hx_1,
```

```
            this.state.status_bergenLine.block_pascack_hx_2
        );
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_click_sig_2w2() ----

    /**
     * hx_click_sig_2w3()
     * @summary The event handler for the Signal #2w3
     */
    hx_click_sig_2w3 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hx().click_sig_2w3(
            this.state.status_bergenLine.block_pascack_hx_1,
            this.state.status_bergenLine.block_pascack_hx_2
        );
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_click_sig_2w3() ----

    /**
     * hx_click_sig_4w()
     * @summary The event handler for the Signal #4w
     */
    hx_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hx().click_sig_4w(
            this.state.status_bergenLine.block_pascack_hx_2
        );
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_click_sig_4w() ----

    /**
     * hx_click_sig_2e()
     * @summary The event handler for the Signal 2e
     */
    hx_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hx().click_sig_2e(
            this.state.status_bergenLine.block_hx_laurel_1,
```

```
            this.state.status_bergenLine.block_hx_croxton_2,
            this.state.status_bergenLine.block_hx_croxton_1
        );
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_click_sig_2e() ----

    /**
     * hx_click_sig_4e()
     * @summary The event handler for the Signal 4e
     */
    hx_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hx().click_sig_4e(
            this.state.status_bergenLine.block_hx_laurel_1,
            this.state.status_bergenLine.block_hx_laurel_2,
            this.state.status_bergenLine.block_hx_croxton_2,
            this.state.status_bergenLine.block_hx_croxton_1
        );
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_click_sig_4e() ----

    /**
     * hx_throw_sw_1()
     * @summary The event handler for switch #1
     */
    hx_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hx().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_throw_sw_1() ----

    /**
     * hx_throw_sw_3()
     * @summary The event handler for switch #3
     */
    hx_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hx().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_hx:
```

```
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_throw_sw_3() ----

    /**
     * hx_throw_sw_5()
     * @summary The event handler for switch #5
     */
    hx_throw_sw_5 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hx().throw_sw_5();
        // Set the state of the Interlocking
        this.setState({status_hx:
ctc.get_hx().get_interlocking_status()});
    }
    // ---- END hx_throw_sw_5() ----
    /* END Functions for the HX Interlocking */


    /* Functions for the Pascack Junction Interlocking */
    /**
     * pascack_click_sig_2w()
     * @summary Event handler for the signal #2w
     */
    pascack_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pascack().click_sig_2w(
            this.state.status_bergenLine.block_bt_pascack_1,
            this.state.status_bergenLine.block_bt_pascack_2
        );
        // Set the state of the Interlocking
        this.setState({status_pascack:
ctc.get_pascack().get_interlocking_status()});
    }
    // ---- END pascack_click_sig_2w() ----

    /**
     * pascack_click_sig_4w()
     * @summary Event handler for the signal #4w
     */
    pascack_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pascack().click_sig_4w(
            this.state.status_bergenLine.block_bt_pascack_1,
            this.state.status_bergenLine.block_bt_pascack_2
        );
        // Set the state of the Interlocking
        this.setState({status_pascack:
```

```
ctc.get_pascack().get_interlocking_status()});
    }
    // ---- END pascack_click_sig_4w() ----

    /**
     * pascack_click_sig_2e()
     * @summary Event handler for the signal #2e
     */
    pascack_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pascack().click_sig_2e(
            this.state.status_bergenLine.block_pascack_hx_1,
            this.state.status_bergenLine.block_pascack_hx_2
        );
        // Set the state of the Interlocking
        this.setState({status_pascack:
ctc.get_pascack().get_interlocking_status()});
    }
    // ---- END pascack_click_sig_2e() ----

    /**
     * pascack_click_sig_4e()
     * @summary Event handler for the signal #4e
     */
    pascack_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pascack().click_sig_4e(
            this.state.status_bergenLine.block_pascack_hx_1,
            this.state.status_bergenLine.block_pascack_hx_2
        );
        // Set the state of the Interlocking
        this.setState({status_pascack:
ctc.get_pascack().get_interlocking_status()});
    }
    // ---- END pascack_click_sig_4e() ----

    /**
     * pascack_throw_sw_1()
     * @summary The event handler for switch #1
     */
    pascack_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_pascack().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_pascack:
ctc.get_pascack().get_interlocking_status()});
    }
    // ---- END pascack_throw_sw_1() ----
```

```
    /**
     * pascack_throw_sw_3()
     * @summary The event handler for switch #3
     */
    pascack_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_pascack().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_pascack:
ctc.get_pascack().get_interlocking_status()});
    }
    // ---- END pascack_throw_sw_1() ----
    /* END Functions for the Pascack Junction Interlocking */


    /* Functions for the BT Interlocking */
    /**
     * bt_click_sig_2w1()
     * @summary Event handler for the signal #2w1
     */
    bt_click_sig_2w1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bt().click_sig_2w1(
            this.state.status_bergenLine.block_ridgewood_bt_1,
            this.state.status_bergenLine.block_ridgewood_bt_2
        );
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_click_sig_2w1() ----

    /**
     * bt_click_sig_2w2()
     * @summary Event handler for the signal #2w2
     */
    bt_click_sig_2w2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bt().click_sig_2w2(
            this.state.status_bergenLine.block_ridgewood_bt_1,
            this.state.status_bergenLine.block_ridgewood_bt_2
        );
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_click_sig_2w1() ----
```

```
    /**
     * bt_click_sig_4w()
     * @summary Event handler for the signal #4
     */
    bt_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bt().click_sig_4w(
            this.state.status_bergenLine.block_ridgewood_bt_1,
            this.state.status_bergenLine.block_ridgewood_bt_2
        );
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_click_sig_2w1() ----

    /**
     * bt_click_sig_2e()
     * @summary Event handler for the signal #2e
     */
    bt_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bt().click_sig_2e(
            this.state.status_bergenLine.block_bt_pascack_1,
            this.state.status_bergenLine.block_bt_pascack_2,
            this.state.status_bergenLine.block_bt_nysw
        );
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_click_sig_2w1() ----

    /**
     * bt_click_sig_4e()
     * @summary Event handler for the signal #4e
     */
    bt_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bt().click_sig_4e(
            this.state.status_bergenLine.block_bt_pascack_1,
            this.state.status_bergenLine.block_bt_pascack_2,
            this.state.status_bergenLine.block_bt_nysw
        );
        // Set the state of the Interlocking
        this.setState({status_bt:
```

```
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_click_sig_2w1() ----

    /**
     * bt_throw_sw_1()
     * @summary The event handler for switch #1
     */
    bt_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_bt().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_throw_sw_1() ----

    /**
     * bt_throw_sw_3()
     * @summary The event handler for switch #3
     */
    bt_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_bt().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_throw_sw_3() ----

    /**
     * bt_throw_sw_5()
     * @summary The event handler for switch #5
     */
    bt_throw_sw_5 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_bt().throw_sw_5();
        // Set the state of the Interlocking
        this.setState({status_bt:
ctc.get_bt().get_interlocking_status()});
    }
    // ---- END bt_throw_sw_5() ----
    /* END Functions for the BT Interlocking */
    /* END Bergen County Line Event Handlers */




    /* Southern Tier Event Handlers */
    /* Functions for CP Sparrow */
```

```
    /**
     * sparrow_click_sig_2w_1()
     * @summary The event handler for Signal #2w_1
     */
    sparrow_click_sig_2w_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sparrow().click_sig_2w_1(
            this.state.status_tier.block_bingo_sparrow
        );
        // Set the state of the Interlocking
        this.setState({status_sparrow:
ctc.get_sparrow().get_interlocking_status()});
    }
    // ---- END sparrow_click_sig_2w_1() ----

    /**
     * sparrow_click_sig_2w_2()
     * @summary The event handler for Signal #2w_2
     */
    sparrow_click_sig_2w_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sparrow().click_sig_2w_2(
            this.state.status_tier.block_bingo_sparrow
        );
        // Set the state of the Interlocking
        this.setState({status_sparrow:
ctc.get_sparrow().get_interlocking_status()});
    }
    // ---- END sparrow_click_sig_2w_2() ----

    /**
     * sparrow_click_sig_2w_3()
     * @summary The event handler for Signal #2w_3
     */
    sparrow_click_sig_2w_3 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sparrow().click_sig_2w_3(
            this.state.status_tier.block_bingo_sparrow
        );
        // Set the state of the Interlocking
        this.setState({status_sparrow:
ctc.get_sparrow().get_interlocking_status()});
    }
    // ---- END sparrow_click_sig_2w_3() ----

    /**
     * sparrow_click_sig_2e()
```

```
     * @summary The event handler for Signal #2e
     */
    sparrow_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sparrow().click_sig_2e(
            this.state.status_tier.block_sparrow_pa_1,
            this.state.status_tier.block_sparrow_pa_2,
            this.state.status_tier.block_sparrow_cripple
        );
        // Set the state of the Interlocking
        this.setState({status_sparrow:
ctc.get_sparrow().get_interlocking_status()});
    }
    // ---- END sparrow_click_sig_2e() ----

    /**
     * sparrow_throw_sw_1()
     * @summary The event handler for switch #1
     */
    sparrow_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_sparrow().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_sparrow:
ctc.get_sparrow().get_interlocking_status()});
    }
    // ---- END sparrow_throw_sw_1() ----

    /**
     * sparrow_throw_sw_3()
     * @summary The event handler for switch #3
     */
    sparrow_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_sparrow().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_sparrow:
ctc.get_sparrow().get_interlocking_status()});
    }
    // ---- END sparrow_throw_sw_3() ----
    /* END Functions for CP Sparrow */


    /* Functions for CP PA */
    /**
     * pa_click_sig_2w_1()
     * @summary The event handler for Signal #2w_1
     */
    pa_click_sig_2w_1 = () => {
```

```
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pa().click_sig_2w_1(
            this.state.status_tier.block_sparrow_pa_1,
            this.state.status_tier.block_sparrow_pa_2,
            this.state.status_tier.block_buckleys_west
        );
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_click_sig_2w_1() ----

    /**
     * pa_click_sig_2w_2()
     * @summary The event handler for Signal #2w_2
     */
    pa_click_sig_2w_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pa().click_sig_2w_2(
            this.state.status_tier.block_sparrow_pa_1,
            this.state.status_tier.block_sparrow_pa_2,
            this.state.status_tier.block_buckleys_west
        );
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_click_sig_2w_2() ----

    /**
     * pa_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    pa_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pa().click_sig_4w(
            this.state.status_tier.block_sparrow_pa_2,
            this.state.status_tier.block_buckleys_west
        );
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_click_sig_4w() ----

    /**
     * pa_click_sig_2e()
```

```
     * @summary The event handler for Signal #2e
     */
    pa_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pa().click_sig_2e(
            this.state.status_tier.block_pa_port_1,
            this.state.status_tier.block_port_yard_west
        );
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_click_sig_2e() ----

    /**
     * pa_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    pa_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_pa().click_sig_4e(
            this.state.status_tier.block_pa_port_1,
            this.state.status_tier.block_pa_bc_2,
            this.state.status_tier.block_port_yard_west
        );
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_click_sig_4e() ----

    /**
     * pa_throw_sw_1()
     * @summary The event handler for switch #1
     */
    pa_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_pa().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_throw_sw_1() ----

    /**
     * pa_throw_sw_3()
     * @summary The event handler for switch #3
     */
```

```
    pa_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_pa().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_pa:
ctc.get_pa().get_interlocking_status()});
    }
    // ---- END pa_throw_sw_3() ----
    /* END Functions for CP PA */


    /* Functions for CP Port */
    /**
     * pa_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    port_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_port().click_sig_2w(
            this.state.status_tier.block_pa_port_1,
            this.state.status_tier.block_port_yard_east
        );
        // Set the state of the Interlocking
        this.setState({status_port:
ctc.get_port().get_interlocking_status()});
    }
    // ---- END port_click_sig_2w() ----

    /**
     * pa_click_sig_2e_1()
     * @summary The event handler for Signal #2e_1
     */
    port_click_sig_2e_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_port().click_sig_2e_1(
            this.state.status_tier.block_port_bc_1
        );
        // Set the state of the Interlocking
        this.setState({status_port:
ctc.get_port().get_interlocking_status()});
    }
    // ---- END port_click_sig_2e_1() ----

    /**
     * pa_click_sig_2e_2()
     * @summary The event handler for Signal #2e_2
     */
    port_click_sig_2e_2 = () => {
```

```
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_port().click_sig_2e_2(
            this.state.status_tier.block_port_bc_1
        );
        // Set the state of the Interlocking
        this.setState({status_port:
ctc.get_port().get_interlocking_status()});
    }
    // ---- END port_click_sig_2e_2() ----

    /**
     * port_throw_sw_1()
     * @summary The event handler for switch #1
     */
    port_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_port().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_port:
ctc.get_port().get_interlocking_status()});
    }
    // ---- END port_throw_sw_1() ----
    /* END Functions for CP Port */


    /* Functions for CP BC */
    /**
     * bc_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    bc_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bc().click_sig_2w(
            this.state.status_tier.block_port_bc_1,
            this.state.status_tier.block_pa_bc_2
        );
        // Set the state of the Interlocking
        this.setState({status_bc:
ctc.get_bc().get_interlocking_status()});
    }
    // ---- END port_click_sig_2w() ----

    /**
     * bc_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    bc_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
```

```
        // Passing reference the next blocks
        ctc.get_bc().click_sig_2e(
            this.state.status_tier.block_bc_ov_1
        );
        // Set the state of the Interlocking
        this.setState({status_bc:
ctc.get_bc().get_interlocking_status()});
    }
    // ---- END port_click_sig_2e() ----

    /**
     * bc_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    bc_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_bc().click_sig_4e(
            this.state.status_tier.block_bc_ov_1
        );
        // Set the state of the Interlocking
        this.setState({status_bc:
ctc.get_bc().get_interlocking_status()});
    }
    // ---- END port_click_sig_4e() ----

    /**
     * bc_throw_sw_1()
     * @summary The event handler for switch #1
     */
    bc_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_bc().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_bc:
ctc.get_bc().get_interlocking_status()});
    }
    // ---- END bc_throw_sw_1() ----
    /* END Functions for CP BC */


    /* Functions for CP OV */
    /**
     * ov_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    ov_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ov().click_sig_2w(
```

```
            this.state.status_tier.block_bc_ov_1
        );
        // Set the state of the Interlocking
        this.setState({status_ov:
ctc.get_ov().get_interlocking_status()});
    }
    // ---- END ov_click_sig_2w() ----

    /**
     * ov_click_sig_2ws()
     * @summary The event handler for Signal #2ws
     */
    ov_click_sig_2ws = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ov().click_sig_2ws(
            this.state.status_tier.block_bc_ov_1
        );
        // Set the state of the Interlocking
        this.setState({status_ov:
ctc.get_ov().get_interlocking_status()});
    }
    // ---- END ov_click_sig_2ws() ----

    /**
     * ov_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    ov_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ov().click_sig_2e(
            this.state.status_tier.block_ov_howells_1,
            this.state.status_tier.block_ov_howells_2
        );
        // Set the state of the Interlocking
        this.setState({status_ov:
ctc.get_ov().get_interlocking_status()});
    }
    // ---- END ov_click_sig_2e() ----

    /**
     * ov_throw_sw_1()
     * @summary The event handler for switch #1
     */
    ov_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_ov().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_ov:
```

```
ctc.get_ov().get_interlocking_status()});
    }
    // ---- END ov_throw_sw_1() ----
    /* END Functions for CP OV */


    /* Functions for CP Howells */
    /**
     * howells_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    howells_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_howells().click_sig_2w(
            this.state.status_tier.block_ov_howells_1,
            this.state.status_tier.block_ov_howells_2
        );
        // Set the state of the Interlocking
        this.setState({status_howells:
ctc.get_howells().get_interlocking_status()});
    }
    // ---- END howells_click_sig_2w() ----

    /**
     * howells_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    howells_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_howells().click_sig_2e(
            this.state.status_tier.block_howells_hall_1
        );
        // Set the state of the Interlocking
        this.setState({status_howells:
ctc.get_howells().get_interlocking_status()});
    }
    // ---- END howells_click_sig_2e() ----

    /**
     * howells_click_sig_2es()
     * @summary The event handler for Signal #2es
     */
    howells_click_sig_2es = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_howells().click_sig_2es(
            this.state.status_tier.block_howells_hall_1
        );
```

```
        // Set the state of the Interlocking
        this.setState({status_howells:
ctc.get_howells().get_interlocking_status()});
    }
    // ---- END howells_click_sig_2es() ----

    /**
     * howells_throw_sw_3()
     * @summary The event handler for switch #3
     */
    howells_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_howells().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_howells:
ctc.get_howells().get_interlocking_status()});
    }
    // ---- END howells_throw_sw_3() ----
    /* END Functions for CP Howells */


    /* Functions for CP Hall */
    /**
     * hall_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    hall_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hall().click_sig_2w(
            this.state.status_tier.block_howells_hall_1
        );
        // Set the state of the Interlocking
        this.setState({status_hall:
ctc.get_hall().get_interlocking_status()});
    }
    // ---- END hall_click_sig_2w() ----

    /**
     * hall_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    hall_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hall().click_sig_4w(
            this.state.status_tier.block_howells_hall_1,
            this.state.status_tier.block_hall_yard
        );
        // Set the state of the Interlocking
```

```
        this.setState({status_hall:
ctc.get_hall().get_interlocking_status()});
    }
    // ---- END hall_click_sig_4w() ----

    /**
     * hall_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    hall_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hall().click_sig_2e(
            this.state.status_tier.block_hall_hudson_1,
            this.state.status_tier.block_hall_hudson_2
        );
        // Set the state of the Interlocking
        this.setState({status_hall:
ctc.get_hall().get_interlocking_status()});
    }
    // ---- END hall_click_sig_2e() ----

    /**
     * hall_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    hall_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hall().click_sig_4e(
            this.state.status_tier.block_hall_hudson_2
        );
        // Set the state of the Interlocking
        this.setState({status_hall:
ctc.get_hall().get_interlocking_status()});
    }
    // ---- END hall_click_sig_4e() ----

    /**
     * hall_throw_sw_1()
     * @summary The event handler for switch #1
     */
    hall_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hall().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_hall:
ctc.get_hall().get_interlocking_status()});
    }
    // ---- END hall_throw_sw_1() ----
```

```
    /* END Functions for CP Hall */


    /* Functions for CP Hudson Junction */
    /**
     * hudson_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    hudson_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hudson().click_sig_2w(
            this.state.status_tier.block_hall_hudson_1,
            this.state.status_tier.block_hall_hudson_2
        );
        // Set the state of the Interlocking
        this.setState({status_hudson:
ctc.get_hudson().get_interlocking_status()});
    }
    // ---- END hudson_click_sig_2w() ----

    /**
     * hudson_click_sig_2ws()
     * @summary The event handler for Signal #2ws
     */
    hudson_click_sig_2ws = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hudson().click_sig_2ws(
            this.state.status_tier.block_hall_hudson_1,
            this.state.status_tier.block_hall_hudson_2
        );
        // Set the state of the Interlocking
        this.setState({status_hudson:
ctc.get_hudson().get_interlocking_status()});
    }
    // ---- END hudson_click_sig_2ws() ----

    /**
     * hudson_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    hudson_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hudson().click_sig_2e(
            this.state.status_tier.block_hudson_valley_1,
            this.state.status_tier.block_hudson_nysw
        );
        // Set the state of the Interlocking
```

```
        this.setState({status_hudson:
ctc.get_hudson().get_interlocking_status()});
    }
    // ---- END hudson_click_sig_2e() ----

    /**
     * hudson_click_sig_2es()
     * @summary The event handler for Signal #2es
     */
    hudson_click_sig_2es = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hudson().click_sig_2es(
            this.state.status_tier.block_hudson_valley_1,
            this.state.status_tier.block_hudson_nysw
        );
        // Set the state of the Interlocking
        this.setState({status_hudson:
ctc.get_hudson().get_interlocking_status()});
    }
    // ---- END hudson_click_sig_2es() ----

    /**
     * hudson_throw_sw_1()
     * @summary The event handler for switch #1
     */
    hudson_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hudson().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_hudson:
ctc.get_hudson().get_interlocking_status()});
    }
    // ---- END hudson_throw_sw_1() ----

    /**
     * hudson_throw_sw_3()
     * @summary The event handler for switch #3
     */
    hudson_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hudson().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_hudson:
ctc.get_hudson().get_interlocking_status()});
    }
    // ---- END hudson_throw_sw_3() ----
    /* END Functions for CP Hudson Junction */
```

```
    /* Functions for CP Central Valley */
    /**
     * valley_click_sig_1w()
     * @summary The event handler for Signal #1w
     */
    valley_click_sig_1w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_valley().click_sig_1w(
            this.state.status_tier.block_hudson_valley_1
        );
        // Set the state of the Interlocking
        this.setState({status_valley:
ctc.get_valley().get_interlocking_status()});
    }
    // ---- END valley_click_sig_1w() ----

    /**
     * valley_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    valley_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_valley().click_sig_2w(
            this.state.status_tier.block_hudson_valley_1
        );
        // Set the state of the Interlocking
        this.setState({status_valley:
ctc.get_valley().get_interlocking_status()});
    }
    // ---- END valley_click_sig_2w() ----

    /**
     * valley_click_sig_1e()
     * @summary The event handler for Signal #1e
     */
    valley_click_sig_1e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_valley().click_sig_1e(
            this.state.status_tier.block_valley_harriman_1,
            this.state.status_tier.block_valley_harriman_2
        );
        // Set the state of the Interlocking
        this.setState({status_valley:
ctc.get_valley().get_interlocking_status()});
    }
    // ---- END valley_click_sig_1e() ----
```

```
    /**
     * valley_throw_sw_21()
     * @summary The event handler for switch #21
     */
    valley_throw_sw_21 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_valley().throw_sw_21();
        // Set the state of the Interlocking
        this.setState({status_valley:
ctc.get_valley().get_interlocking_status()});
    }
    // ---- END valley_throw_sw_21() ----
    /* END Functions for CP Central Valley */


    /* Functions for CP Harriman */
    /**
     * harriman_click_sig_1w()
     * @summary The event handler for Signal #1w
     */
    harriman_click_sig_1w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_harriman().click_sig_1w(
            this.state.status_tier.block_valley_harriman_1,
            this.state.status_tier.block_valley_harriman_2,
            this.state.status_tier.block_harriman_industrial
        );
        // Set the state of the Interlocking
        this.setState({status_harriman:
ctc.get_harriman().get_interlocking_status()});
    }
    // ---- END harriman_click_sig_1w() ----

    /**
     * harriman_click_sig_1e()
     * @summary The event handler for Signal #1e
     */
    harriman_click_sig_1e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_harriman().click_sig_1e(
            this.state.status_tier.block_harriman_sterling_1
        );
        // Set the state of the Interlocking
        this.setState({status_harriman:
ctc.get_harriman().get_interlocking_status()});
    }
    // ---- END harriman_click_sig_1e() ----
```

```
    /**
     * harriman_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    harriman_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_harriman().click_sig_2e(
            this.state.status_tier.block_harriman_sterling_1
        );
        // Set the state of the Interlocking
        this.setState({status_harriman:
ctc.get_harriman().get_interlocking_status()});
    }
    // ---- END harriman_click_sig_2e() ----

    /**
     * harriman_click_sig_3e()
     * @summary The event handler for Signal #3e
     */
    harriman_click_sig_3e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_harriman().click_sig_3e(
            this.state.status_tier.block_harriman_sterling_1
        );
        // Set the state of the Interlocking
        this.setState({status_harriman:
ctc.get_harriman().get_interlocking_status()});
    }
    // ---- END harriman_click_sig_3e() ----

    /**
     * harriman_throw_sw_21()
     * @summary The event handler for switch #21
     */
    harriman_throw_sw_21 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_harriman().throw_sw_21();
        // Set the state of the Interlocking
        this.setState({status_harriman:
ctc.get_harriman().get_interlocking_status()});
    }
    // ---- END harriman_throw_sw_21() ----

    /**
     * harriman_throw_sw_32()
     * @summary The event handler for switch #32
     */
    harriman_throw_sw_32 = () => {
```

```
        // Get the backend function for the corresponding switch
        ctc.get_harriman().throw_sw_32();
        // Set the state of the Interlocking
        this.setState({status_harriman:
ctc.get_harriman().get_interlocking_status()});
    }
    // ---- END harriman_throw_sw_32() ----
    /* END Functions for CP Harriman */


    /* Functions for CP Sterling */
    /**
     * sterling_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    sterling_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sterling().click_sig_2w(
            this.state.status_tier.block_harriman_sterling_1
        );
        // Set the state of the Interlocking
        this.setState({status_sterling:
ctc.get_sterling().get_interlocking_status()});
    }
    // ---- END sterling_click_sig_2w() ----

    /**
     * sterling_click_sig_2ws()
     * @summary The event handler for Signal #2ws
     */
    sterling_click_sig_2ws = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sterling().click_sig_2ws(
            this.state.status_tier.block_harriman_sterling_1
        );
        // Set the state of the Interlocking
        this.setState({status_sterling:
ctc.get_sterling().get_interlocking_status()});
    }
    // ---- END sterling_click_sig_2ws() ----

    /**
     * sterling_click_sig_1e()
     * @summary The event handler for Signal #1e
     */
    sterling_click_sig_1e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
```

```
        ctc.get_sterling().click_sig_1e(
            this.state.status_tier.block_sterling_sf,
            this.state.status_tier.block_sterling_hilburn
        );
        // Set the state of the Interlocking
        this.setState({status_sterling:
ctc.get_sterling().get_interlocking_status()});
    }
    // ---- END sterling_click_sig_1e() ----

    /**
     * sterling_throw_sw_21()
     * @summary The event handler for switch #21
     */
    sterling_throw_sw_21 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_sterling().throw_sw_21();
        // Set the state of the Interlocking
        this.setState({status_sterling:
ctc.get_sterling().get_interlocking_status()});
    }
    // ---- END sterling_throw_sw_21() ----
    /* END Functions for CP Sterling */
    /* END Southern Tier Event Handlers */




    /* Main Line Event Handlers */
    /* Functions for Hilburn Interlocking */
    /**
     * hilburn_click_sig_2w_1()
     * @summary The event handler for Signal #2w_1
     */
    hilburn_click_sig_2w_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hilburn().click_sig_2w_1(
            this.state.status_mainLine.block_sterling_hilburn
        );
        // Set the state of the Interlocking
        this.setState({status_hilburn:
ctc.get_hilburn().get_interlocking_status()});
    }
    // ---- END hilburn_click_sig_2w_1() ----

    /**
     * hilburn_click_sig_2w_2()
     * @summary The event handler for Signal #2w_2
     */
```

```
    hilburn_click_sig_2w_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hilburn().click_sig_2w_2(
            this.state.status_mainLine.block_sterling_hilburn
        );
        // Set the state of the Interlocking
        this.setState({status_hilburn:
ctc.get_hilburn().get_interlocking_status()});
    }
    // ---- END hilburn_click_sig_2w_2() ----

    /**
     * hilburn_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    hilburn_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_hilburn().click_sig_2e(
            this.state.status_mainLine.block_hilburn_sf,
            this.state.status_mainLine.block_hilburn_yard_west
        );
        // Set the state of the Interlocking
        this.setState({status_hilburn:
ctc.get_hilburn().get_interlocking_status()});
    }
    // ---- END hilburn_click_sig_2e() ----

    /**
     * hilburn_throw_sw_1()
     * @summary The event handler for switch #1
     */
    hilburn_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_hilburn().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_hilburn:
ctc.get_hilburn().get_interlocking_status()});
    }
    // ---- END hilburn_throw_sw_1() ----
    /* END Functions for Hilburn Interlocking */


    /* Functions for SF Interlocking */
    /**
     * sf_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    sf_click_sig_2w = () => {
```

```
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sf().click_sig_2w(
            this.state.status_mainLine.block_sterling_sf,
            this.state.status_mainLine.block_hilburn_sf,
            this.state.status_mainLine.block_hilburn_yard_east
        );
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_click_sig_2w() ----

    /**
     * sf_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    sf_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sf().click_sig_4w(
            this.state.status_mainLine.block_hilburn_sf,
            this.state.status_mainLine.block_hilburn_yard_east
        );
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_click_sig_4w() ----

    /**
     * sf_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    sf_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sf().click_sig_2e(
            this.state.status_mainLine.block_sf_wc_1
        );
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_click_sig_2e() ----

    /**
     * sf_click_sig_4e_1()
     * @summary The event handler for Signal #4e_1
     */
```

```
    sf_click_sig_4e_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sf().click_sig_4e_1(
            this.state.status_mainLine.block_sf_wc_1,
            this.state.status_mainLine.block_sf_wc_2
        );
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_click_sig_4e_1() ----

    /**
     * sf_click_sig_4e_2()
     * @summary The event handler for Signal #4e_2
     */
    sf_click_sig_4e_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_sf().click_sig_4e_2(
            this.state.status_mainLine.block_sf_wc_1,
            this.state.status_mainLine.block_sf_wc_2
        );
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_click_sig_4e_2() ----

    /**
     * sf_throw_sw_1()
     * @summary The event handler for switch #1
     */
    sf_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_sf().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_throw_sw_1() ----

    /**
     * sf_throw_sw_3()
     * @summary The event handler for switch #3
     */
    sf_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_sf().throw_sw_3();
```

```
        // Set the state of the Interlocking
        this.setState({status_sf:
ctc.get_sf().get_interlocking_status()});
    }
    // ---- END sf_throw_sw_3() ----
    /* END Functions for SF Interlocking */


    /* Functions for WC Interlocking */
    /**
     * wc_click_sig_2w_1()
     * @summary The event handler for Signal #2w_1
     */
    wc_click_sig_2w_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_wc().click_sig_2w_1(
            this.state.status_mainLine.block_sf_wc_1,
            this.state.status_mainLine.block_sf_wc_2,
            this.state.status_mainLine.block_wc_yard
        );
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_click_sig_2w_1() ----

    /**
     * wc_click_sig_2w_2()
     * @summary The event handler for Signal #2w_2
     */
    wc_click_sig_2w_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_wc().click_sig_2w_2(
            this.state.status_mainLine.block_sf_wc_1,
            this.state.status_mainLine.block_sf_wc_2,
            this.state.status_mainLine.block_wc_yard
        );
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_click_sig_2w_2() ----

    /**
     * wc_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    wc_click_sig_4w = () => {
```

```
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_wc().click_sig_4w(
            this.state.status_mainLine.block_sf_wc_1,
            this.state.status_mainLine.block_sf_wc_2,
            this.state.status_mainLine.block_wc_yard
        )
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_click_sig_4w() ----

    /**
     * wc_click_sig_2e_1()
     * @summary The event handler for Signal #2e_1
     */
    wc_click_sig_2e_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_wc().click_sig_2e_1(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
            this.state.status_mainLine.block_wc_ridgewood_3
        );
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_click_sig_2e_1() ----

    /**
     * wc_click_sig_2e_2()
     * @summary The event handler for Signal #2e_2
     */
    wc_click_sig_2e_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_wc().click_sig_2e_2(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
            this.state.status_mainLine.block_wc_ridgewood_3
        );
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_click_sig_2e_2() ----

    /**
```

```
     * wc_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    wc_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_wc().click_sig_4e(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
            this.state.status_mainLine.block_wc_ridgewood_3
        );
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_click_sig_4e() ----

    /**
     * wc_throw_sw_1()
     * @summary The event handler for switch #1
     */
    wc_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_wc().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_throw_sw_1() ----

    /**
     * wc_throw_sw_3()
     * @summary The event handler for switch #3
     */
    wc_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_wc().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_throw_sw_3() ----

    /**
     * wc_throw_sw_5()
     * @summary The event handler for switch #5
     */
    wc_throw_sw_5 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_wc().throw_sw_5();
```

```
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_throw_sw_5() ----

    /**
     * wc_throw_sw_7()
     * @summary The event handler for switch #7
     */
    wc_throw_sw_7 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_wc().throw_sw_7();
        // Set the state of the Interlocking
        this.setState({status_wc:
ctc.get_wc().get_interlocking_status()});
    }
    // ---- END wc_throw_sw_7() ----
    /* END Functions for WC Interlocking */


    /* Functions for Ridgewood Junction Interlocking */
    /**
     * ridgewood_click_sig_2w_1()
     * @summary The event handler for Signal #2w_1
     */
    ridgewood_click_sig_2w_1 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_2w1(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
            this.state.status_mainLine.block_wc_ridgewood_3,
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_2w_1() ----

    /**
     * ridgewood_click_sig_2w_2()
     * @summary The event handler for Signal #2w_2
     */
    ridgewood_click_sig_2w_2 = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_2w2(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
```

```
            this.state.status_mainLine.block_wc_ridgewood_3,
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_2w_2() ----

    /**
     * ridgewood_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    ridgewood_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_4w(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
            this.state.status_mainLine.block_wc_ridgewood_3,
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_4w() ----

    /**
     * ridgewood_click_sig_6w()
     * @summary The event handler for Signal #6w
     */
    ridgewood_click_sig_6w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_6w(
            this.state.status_mainLine.block_wc_ridgewood_1,
            this.state.status_mainLine.block_wc_ridgewood_2,
            this.state.status_mainLine.block_wc_ridgewood_3,
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_6w() ----

    /**
     * ridgewood_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    ridgewood_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
```

```
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_2e(
            this.state.status_mainLine.block_ridgewood_suscon_1,
            this.state.status_mainLine.block_ridgewood_suscon_2,
            this.state.status_mainLine.block_ridgewood_suscon_3,
            this.state.status_mainLine.block_ridgewood_suscon_4
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_2e() ----

    /**
     * ridgewood_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    ridgewood_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_4e(
            this.state.status_mainLine.block_ridgewood_suscon_1,
            this.state.status_mainLine.block_ridgewood_suscon_2,
            this.state.status_mainLine.block_ridgewood_suscon_3,
            this.state.status_mainLine.block_ridgewood_suscon_4
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_4e() ----

    /**
     * ridgewood_click_sig_6e()
     * @summary The event handler for Signal #6e
     */
    ridgewood_click_sig_6e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_ridgewood().click_sig_6e(
            this.state.status_mainLine.block_ridgewood_suscon_1,
            this.state.status_mainLine.block_ridgewood_suscon_2,
            this.state.status_mainLine.block_ridgewood_suscon_3,
            this.state.status_mainLine.block_ridgewood_suscon_4
        );
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_click_sig_6e() ----
```

```
    /**
     * ridgewood_throw_sw_1()
     * @summary The event handler for switch #1
     */
    ridgewood_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_ridgewood().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_throw_sw_1() ----

    /**
     * ridgewood_throw_sw_3()
     * @summary The event handler for switch #3
     */
    ridgewood_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_ridgewood().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_throw_sw_3() ----

    /**
     * ridgewood_throw_sw_5()
     * @summary The event handler for switch #5
     */
    ridgewood_throw_sw_5 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_ridgewood().throw_sw_5();
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_throw_sw_5() ----

    /**
     * ridgewood_throw_sw_7()
     * @summary The event handler for switch #7
     */
    ridgewood_throw_sw_7 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_ridgewood().throw_sw_7();
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
```

```
    }
    // ---- END ridgewood_throw_sw_7() ----

    /**
     * ridgewood_throw_sw_9()
     * @summary The event handler for switch #9
     */
    ridgewood_throw_sw_9 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_ridgewood().throw_sw_9();
        // Set the state of the Interlocking
        this.setState({status_ridgewood:
ctc.get_ridgewood().get_interlocking_status()});
    }
    // ---- END ridgewood_throw_sw_9() ----
    /* END Functions for Ridgewood Junction Interlocking */


    /* Functions for Suscon Interlocking */
    /**
     * suscon_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    suscon_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_suscon().click_sig(
            "2W",
            this.state.status_mainLine.block_ridgewood_suscon_1,
            this.state.status_mainLine.block_ridgewood_suscon_2
        );
        // Set the state of the Interlocking
        this.setState({status_suscon:
ctc.get_suscon().get_interlocking_status()});
    }
    // ---- END suscon_click_sig_2w() ----

    /**
     * suscon_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    suscon_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_suscon().click_sig(
            "2E",
            this.state.status_mainLine.block_suscon_mill_1,
            this.state.status_mainLine.block_suscon_mill_2
        );
        // Set the state of the Interlocking
```

```
        this.setState({status_suscon:
ctc.get_suscon().get_interlocking_status()});
    }
    // ---- END suscon_click_sig_2e() ----

    /**
     * suscon_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    suscon_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_suscon().click_sig(
            "4W",
            this.state.status_mainLine.block_ridgewood_suscon_1,
            this.state.status_mainLine.block_ridgewood_suscon_2
        );
        // Set the state of the Interlocking
        this.setState({status_suscon:
ctc.get_suscon().get_interlocking_status()});
    }
    // ---- END suscon_click_sig_4w() ----

    /**
     * suscon_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    suscon_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_suscon().click_sig(
            "4E",
            this.state.status_mainLine.block_suscon_mill_1,
            this.state.status_mainLine.block_suscon_mill_2
        );
        // Set the state of the Interlocking
        this.setState({status_suscon:
ctc.get_suscon().get_interlocking_status()});
    }
    // ---- END suscon_click_sig_4e() ----

    /**
     * suscon_throw_sw_1()
     * @summary The event handler for switch #1
     */
    suscon_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_suscon().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_suscon:
```

```
ctc.get_suscon().get_interlocking_status()});
    }
    // ---- END suscon_throw_sw_1() ----

    /**
     * suscon_throw_sw_3()
     * @summary The event handler for switch #3
     */
    suscon_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_suscon().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_suscon:
ctc.get_suscon().get_interlocking_status()});
    }
    // ---- END suscon_throw_sw_3() ----
    /* END Functions for Suscon Interlocking */


    /* Functions for Mill Interlocking */
    /**
     * mill_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    mill_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_mill().click_sig(
            "2W",
            this.state.status_mainLine.block_suscon_mill_1,
            this.state.status_mainLine.block_suscon_mill_2
        );
        // Set the state of the Interlocking
        this.setState({status_mill:
ctc.get_mill().get_interlocking_status()});
    }
    // ---- END mill_click_sig_2w() ----

    /**
     * mill_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    mill_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_mill().click_sig(
            "2E",
            this.state.status_mainLine.block_mill_westSecaucus_1,
            this.state.status_mainLine.block_mill_westSecaucus_2
        );
```

```
        // Set the state of the Interlocking
        this.setState({status_mill:
ctc.get_mill().get_interlocking_status()});
    }
    // ---- END mill_click_sig_2e() ----

    /**
     * mill_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    mill_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_mill().click_sig(
            "4W",
            this.state.status_mainLine.block_suscon_mill_1,
            this.state.status_mainLine.block_suscon_mill_2
        );
        // Set the state of the Interlocking
        this.setState({status_mill:
ctc.get_mill().get_interlocking_status()});
    }
    // ---- END mill_click_sig_4w() ----

    /**
     * mill_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    mill_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_mill().click_sig(
            "4E",
            this.state.status_mainLine.block_mill_westSecaucus_1,
            this.state.status_mainLine.block_mill_westSecaucus_2
        );
        // Set the state of the Interlocking
        this.setState({status_mill:
ctc.get_mill().get_interlocking_status()});
    }
    // ---- END mill_click_sig_4e() ----

    /**
     * mill_throw_sw_1()
     * @summary The event handler for switch #1
     */
    mill_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_mill().throw_sw_1();
        // Set the state of the Interlocking
```

```
        this.setState({status_mill:
ctc.get_mill().get_interlocking_status()});
    }
    // ---- END mill_throw_sw_1() ----

    /**
     * mill_throw_sw_3()
     * @summary The event handler for switch #3
     */
    mill_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_mill().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_mill:
ctc.get_mill().get_interlocking_status()});
    }
    // ---- END mill_throw_sw_3() ----
    /* END Functions for Mill Interlocking */


    /* Functions for West Secaucus Interlocking */
    /**
     * westSecaucus_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    westSecaucus_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_westSecaucus().click_sig(
            "2W",
            this.state.status_mainLine.block_mill_westSecaucus_1,
            this.state.status_mainLine.block_mill_westSecaucus_2
        );
        // Set the state of the Interlocking
        this.setState({status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status()});
    }
    // ---- END westSecaucus_click_sig_2w() ----

    /**
     * westSecaucus_click_sig_2e()
     * @summary The event handler for Signal #2e
     */
    westSecaucus_click_sig_2e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_westSecaucus().click_sig(
            "2E",
            this.state.status_mainLine.block_westSecaucus_laurel_1,
            this.state.status_mainLine.block_westSecaucus_laurel_2
```

```
        );
        // Set the state of the Interlocking
        this.setState({status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status()});
    }
    // ---- END westSecaucus_click_sig_2e() ----

    /**
     * westSecaucus_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    westSecaucus_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_westSecaucus().click_sig(
            "4W",
            this.state.status_mainLine.block_mill_westSecaucus_1,
            this.state.status_mainLine.block_mill_westSecaucus_2
        );
        // Set the state of the Interlocking
        this.setState({status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status()});
    }
    // ---- END westSecaucus_click_sig_4w() ----

    /**
     * westSecaucus_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    westSecaucus_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_westSecaucus().click_sig(
            "4E",
            this.state.status_mainLine.block_westSecaucus_laurel_1,
            this.state.status_mainLine.block_westSecaucus_laurel_2
        );
        // Set the state of the Interlocking
        this.setState({status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status()});
    }
    // ---- END westSecaucus_click_sig_4e() ----

    /**
     * westSecaucus_throw_sw_1()
     * @summary The event handler for switch #1
     */
    westSecaucus_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_westSecaucus().throw_sw_1();
```

```
        // Set the state of the Interlocking
        this.setState({status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status()});
    }
    // ---- END westSecaucus_throw_sw_1() ----

    /**
     * westSecaucus_throw_sw_3()
     * @summary The event handler for switch #3
     */
    westSecaucus_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_westSecaucus().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_westSecaucus:
ctc.get_westSecaucus().get_interlocking_status()});
    }
    // ---- END westSecaucus_throw_sw_3() ----
    /* END Functions for West Secaucus Interlocking */


    /* Functions for Laurel Interlocking */
    /**
     * laurel_click_sig_2w()
     * @summary The event handler for Signal #2w
     */
    laurel_click_sig_2w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_2w(
            this.state.status_mainLine.block_hx_laurel_2,
            this.state.status_mainLine.block_westSecaucus_laurel_1,
            this.state.status_mainLine.block_hx_laurel_1
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_2w() ----

    /**
     * laurel_click_sig_4w()
     * @summary The event handler for Signal #4w
     */
    laurel_click_sig_4w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_4w(
            this.state.status_mainLine.block_hx_laurel_2,
            this.state.status_mainLine.block_westSecaucus_laurel_1,
```

```
            this.state.status_mainLine.block_hx_laurel_1
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_4w() ----

    /**
     * laurel_click_sig_8w()
     * @summary The event handler for Signal #8w
     */
    laurel_click_sig_8w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_8w(
            this.state.status_mainLine.block_hx_laurel_2,
            this.state.status_mainLine.block_westSecaucus_laurel_1,
            this.state.status_mainLine.block_hx_laurel_1,
            this.state.status_mainLine.block_westSecaucus_laurel_2
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_8w() ----

    /**
     * laurel_click_sig_10w()
     * @summary The event handler for Signal #10w
     */
    laurel_click_sig_10w = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_10w(
            this.state.status_mainLine.block_hx_laurel_2,
            this.state.status_mainLine.block_westSecaucus_laurel_1,
            this.state.status_mainLine.block_hx_laurel_1,
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_10w() ----

    /**
     * laurel_click_sig_6e()
     * @summary The event handler for Signal #6e
     */
    laurel_click_sig_6e = () => {
```

```
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_6e(
            this.state.status_mainLine.block_westEnd_laurel_1,
            this.state.status_mainLine.block_westEnd_laurel_2,
            this.state.status_mainLine.block_westEnd_laurel_3,
            this.state.status_mainLine.block_westEnd_laurel_4
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_6e() ----

    /**
     * laurel_click_sig_12e()
     * @summary The event handler for Signal #12e
     */
    laurel_click_sig_12e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_12e(
            this.state.status_mainLine.block_westEnd_laurel_1,
            this.state.status_mainLine.block_westEnd_laurel_2,
            this.state.status_mainLine.block_westEnd_laurel_3,
            this.state.status_mainLine.block_westEnd_laurel_4
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_12e() ----

    /**
     * laurel_click_sig_4e()
     * @summary The event handler for Signal #4e
     */
    laurel_click_sig_4e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_4e(
            this.state.status_mainLine.block_westEnd_laurel_1,
            this.state.status_mainLine.block_westEnd_laurel_2,
            this.state.status_mainLine.block_westEnd_laurel_3,
            this.state.status_mainLine.block_westEnd_laurel_4
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
```

```
    // ---- END laurel_click_sig_4e() ----

    /**
     * laurel_click_sig_8e()
     * @summary The event handler for Signal #8e
     */
    laurel_click_sig_8e = () => {
        // Get the backend function for the corresponding signal
        // Passing reference the next blocks
        ctc.get_laurel().click_sig_8e(
            this.state.status_mainLine.block_westEnd_laurel_4
        );
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_click_sig_8e() ----

    /**
     * laurel_throw_sw_1()
     * @summary The event handler for switch #1
     */
    laurel_throw_sw_1 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_laurel().throw_sw_1();
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_throw_sw_1() ----

    /**
     * laurel_throw_sw_3()
     * @summary The event handler for switch #3
     */
    laurel_throw_sw_3 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_laurel().throw_sw_3();
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_throw_sw_3() ----

    /**
     * laurel_throw_sw_7()
     * @summary The event handler for switch #7
     */
    laurel_throw_sw_7 = () => {
        // Get the backend function for the corresponding switch
```

```
        ctc.get_laurel().throw_sw_7();
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_throw_sw_7() ----

    /**
     * laurel_throw_sw_11()
     * @summary The event handler for switch #11
     */
    laurel_throw_sw_11 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_laurel().throw_sw_11();
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_throw_sw_11() ----

    /**
     * laurel_throw_sw_13()
     * @summary The event handler for switch #13
     */
    laurel_throw_sw_13 = () => {
        // Get the backend function for the corresponding switch
        ctc.get_laurel().throw_sw_13();
        // Set the state of the Interlocking
        this.setState({status_laurel:
ctc.get_laurel().get_interlocking_status()});
    }
    // ---- END laurel_throw_sw_13() ----
    /* END Functions for Laurel Interlocking */
}

// Export the panel to be drawn on the screen
export default MainLine;
```