

```

1  # =====
2  # FILE: DnB_Simulate.pm                                     7/08/2020
3  #
4  # SERVICES:  DnB OPS SIMULATION FUNCTIONS
5  #
6  # DESCRIPTION:
7  #   This perl module provides simulation related functions used by the DnB
8  #   model railroad control program when the -a option is specified on the
9  #   DnB.pl CLI. Simulated sensor values are used instead of real layout
10 #   sensor values. Once set, normal main loop processing is performed. The
11 #   step's sensor values are used for the specified time period. The 'Desc'
12 #   text is displayed on the console at the beginning of each step.
13 #
14 #   Hash key '00' is used to hold and persist the simulation control variables.
15 #
16 #   Refer to the %SensorBit hash in DnB.pl for bit position definitions. A
17 #   colon separated list of bit positions to be set to 1 are specified by the
18 #   'SensorBit' element. 'Time' is the step duration in seconds.
19 #
20 # PERL VERSION: 5.24.1
21 #
22 # =====
23 use strict;
24 # -----
25 # Package Declaration
26 # -----
27 package DnB_Simulate;
28 require Exporter;
29 our @ISA = qw(Exporter);
30
31 our @EXPORT = qw(
32     InitSimulation
33     SimulationStep
34     EndToEnd
35 );
36
37 use DnB_Turnout;
38 use DnB_Message;
39 use Storable 'dclone';
40 use Time::HiRes qw(sleep);
41
42 # =====
43 # FUNCTION:  InitSimulation
44 #
45 # DESCRIPTION:
46 #   This routine initializes the specified simulation data hash with the steps
47 #   for the requested train ops simulation.
48 #
49 # CALLING SYNTAX:
50 #   $result = &InitSimulation($Simulation, \%SimulationData);
51 #
52 # ARGUMENTS:
53 #   $Simulation          Simulation to run.
54 #   $SimulationData      Pointer to %SimulationData hash.
55 #
56 # RETURNED VALUES:
57 #   0 = Success,  1 = Error.
58 #
59 # ACCESSED GLOBAL VARIABLES:
60 #   None.

```

```

61 # =====
62 sub InitSimulation {
63     my($Simulation, $SimulationData) = @_;
64     my($result, @simSteps);
65
66     if ($Simulation eq 'EndToEnd') {
67         $result = &EndToEnd($SimulationData);
68
69         &DisplayDebug(0, "=====");
70         &DisplayDebug(0, "= EndToEnd Simulation Details =");
71         &DisplayDebug(0, "=====");
72
73         foreach my $step (sort keys(%$SimulationData)) {
74             next if ($step eq '00');
75             &DisplayDebug(0, "Step $step - " . $$SimulationData{$step}{'Desc'});
76             if (exists($$SimulationData{$step}{'SensorBit'})) {
77                 &DisplayDebug(0, "    SensorBit: " . $$SimulationData{$step}{'SensorBit'});
78             }
79             if (exists($$SimulationData{$step}{'Turnout'})) {
80                 &DisplayDebug(0, "    Turnout: " . $$SimulationData{$step}{'Turnout'});
81             }
82             if (exists($$SimulationData{$step}{'YardRoute'})) {
83                 &DisplayDebug(0, "    YardRoute: " . $$SimulationData{$step}{'YardRoute'});
84             }
85         }
86         &DisplayDebug(0, "=====");
87     }
88     else {
89         &DisplayError("InitSimulation, Invalid simulation: $Simulation");
90         return 1;
91     }
92
93     @simSteps = sort keys %$SimulationData;
94     $$SimulationData{'00'}{'MaxStep'} = $simSteps[-1];
95     &DisplayDebug(1, "InitSimulation - Simulation: $Simulation    MaxStep: " .
96         $$SimulationData{'00'}{'MaxStep'});
97     return 0;
98 }
99
100 # =====
101 # FUNCTION:  SimulationStep
102 #
103 # DESCRIPTION:
104 #     This routine is called to set turnout and SensorState values when running
105 #     a train ops simulation. The following hash sub-keys are recognized and
106 #     processed for each simulation step.
107 #
108 #     'SensorBit' is a colon (:) separated list of sensor and/or track block
109 #     names. The associated bit number is derived by %SensorBit 'Desc' lookup.
110 #     Ensure search term is unique to a single bit. e.g. 'B01:B03:S01' or
111 #     'B07:GC1 AprW:GC1 Road.'
112 #
113 #     'Turnout' is a comma separated list. Each list element is a colon separated
114 #     turnout number and position. e.g. 'T07:Open,T08:Close'.
115 #
116 #     'YardRoute' is a single entry that maps to a valid %YardRouteData index.
117 #     Input is similar to keypad from/to track number. e.g. '1->3' (track 1 to
118 #     track 3).
119 #
120 # CALLING SYNTAX:

```

```

121 # $result = &SimulationStep(\%SensorBit, \%SensorState1, \%SensorState2,
122 # \%SimulationData, \%TurnoutData, \%YardRouteData);
123 #
124 # ARGUMENTS:
125 # $SensorBit      Pointer to %SensorBit hash.
126 # $SensorState1   Pointer to $SensorState{'1'}.
127 # $SensorState2   Pointer to $SensorState{'2'}.
128 # $SimulationData Pointer to %SimulationData hash.
129 # $TurnoutData     Pointer to %TurnoutData hash.
130 # $YardRouteData  Pointer to %YardRouteData hash.
131 #
132 # RETURNED VALUES:
133 # 0 = Success, 1 = Error.
134 #
135 # ACCESSED GLOBAL VARIABLES:
136 # None.
137 # =====
138 sub SimulationStep {
139     my($SensorBit, $SensorState1, $SensorState2, $SimulationData, $TurnoutData,
140         $YardRouteData) = @_;
141     my($step, @bitDesc, $timeout, $sensorBits, $bits1, $bits2, $bits3, $bits4);
142     my(@match, @turnouts, $tNmbr, $tPos, $moveResult, $route);
143     my($cTime) = time;
144
145     if ($cTime >= $$SimulationData{'00'}{'Timeout'}) {
146         &PlaySound("B.wav", 70);
147         if ($$SimulationData{'00'}{'Step'} eq $$SimulationData{'00'}{'MaxStep'}) {
148             $$SimulationData{'00'}{'Step'} = '00';
149             sleep 0.1;
150             &PlaySound("B.wav", 70);
151         }
152         $step = $$SimulationData{'00'}{'Step'} + 1;
153         $step = "0${step}" if (length($step) == 1);
154         $$SimulationData{'00'}{'Step'} = $step;
155         &DisplayMessage("==> Simulation step $step - Delay: " .
156             $$SimulationData{$step}{'Time'} . " sec --> " .
157             $$SimulationData{$step}{'Desc'});
158
159         # ----- Process 'Turnout' key. -----
160         if (exists($$SimulationData{$step}{'Turnout'})) {
161             &DisplayDebug(0, "Turnout: " . $$SimulationData{$step}{'Turnout'});
162             @turnouts = split(',', $$SimulationData{$step}{'Turnout'});
163             foreach my $turnout (@turnouts) {
164                 if ($turnout =~ m/^\d{2}:(.+)/) {
165                     $tNmbr = $1;
166                     $tPos = $2;
167                     $moveResult = &MoveTurnout($tPos, $tNmbr, $TurnoutData);
168                     if ($moveResult == 1) {
169                         &DisplayError("SimulationStep, Failed to set turnout: " .
170                             $turnout);
171                     }
172                 }
173             }
174             else {
175                 &DisplayError("SimulationStep, Invalid turnout: $turnout");
176             }
177         }
178
179         # ----- Process 'YardRoute' key. -----
180         if (exists($$SimulationData{$step}{'YardRoute'})) {

```

```

181 &DisplayDebug(0, "YardRoute: " . $$SimulationData{$step}{'YardRoute'}));
182 if ($$SimulationData{$step}{'YardRoute'} =~ m/^(\\d+)->(\\d+)/) {
183     $route = join("", "R", sprintf("%1x", ($1 -1)),
184                 sprintf("%1x", ($2 -1)));
185     &DisplayDebug(0, "YardRoute: $route");
186     if (exists($$YardRouteData{$route})) {
187         if ($$YardRouteData{'Control'}{'Inprogress'} == 0) {
188             $$YardRouteData{'Control'}{'Route'} = $route;
189             $$YardRouteData{'Control'}{'Inprogress'} = 1;
190             $$YardRouteData{'Control'}{'Step'} = 0;
191         }
192         else {
193             &DisplayWarning("SimulationStep, skipped '$route'. A yard " .
194                             "route operation is inprogress.");
195         }
196     }
197     else {
198         &DisplayError("SimulationStep, Invalid yard route: $route");
199     }
200 }
201 }
202
203 # ----- Process 'SensorBit' key. -----
204 if (exists($$SimulationData{$step}{'SensorBit'})) {
205     &DisplayDebug(0, "SensorBit: " . $$SimulationData{$step}{'SensorBit'});
206     @bitDesc = split(':', $$SimulationData{$step}{'SensorBit'});
207     $sensorBits = 0; # Clear all bit positions.
208     foreach my $bit (@bitDesc) {
209         @match = grep { $$SensorBit{$_}{'Desc'} =~ /$bit/ } keys
210             %$SensorBit;
211         if ($#match == 0) {
212             $sensorBits = $sensorBits | (1 << $match[0]); # Position and add bit.
213         }
214         else {
215             &DisplayError("SimulationStep, Invalid sensor bit: '$bit' " .
216                             "match: '" . join(",", @match) . "'");
217         }
218     }
219     $$SensorState1 = $sensorBits & 0xFFFF;
220     $bits1 = $$SensorState1 & 0xFF;
221     $bits2 = ($$SensorState1 >> 8) & 0xFF;
222     $$SensorState2 = ($sensorBits >> 16) & 0xFFFF;
223     $bits3 = $$SensorState2 & 0xFF;
224     $bits4 = ($$SensorState2 >> 8) & 0xFF;
225     $timeout = $$SimulationData{$step}{'Time'};
226     $$SimulationData{'00'}{'Timeout'} = $cTime + $timeout;
227     &DisplayDebug(0, "
228         33222222 22221111 111111");
229     &DisplayDebug(0, "
230         10987654 32109876 54321098 76543210");
231     &DisplayDebug(0, "SimulationStep - Timeout: $timeout SensorState2: " .
232         sprintf("%0.8b", $bits4) . " " . sprintf("%0.8b", $bits3) .
233         " SensorState1: " . sprintf("%0.8b", $bits2) . " " .
234         sprintf("%0.8b", $bits1));
235 }
236 }
237 return 0;
238 }
239 # =====

```

```

240 # FUNCTION: EndToEnd
241 #
242 # DESCRIPTION:
243 # This routine returns the EndToEnd simulation steps. Sensor values are set
244 # corresponding to the following train movements. Refer to the description
245 # of the SimulationStep routine for information about the specification of
246 # sensor bits, yard routes, and turnout positioning in this hash.
247 #
248 # Train 1:
249 # From holdover B01 upgrade to yard B10 via B07.
250 # From yard B10 downgrade via B09 to holdover B02.
251 # Train 2:
252 # From holdover B02 upgrade to yard B09 via B08.
253 # From yard B09 downgrade via B07 to holdover B01.
254 #
255 # Yard routes are set to position turnouts for train transit of blocks B09
256 # and B10 since the associated turnouts are not automatically set by sensor
257 # input. For exercise purposes, turnouts on yard track 5 are opened and will
258 # be closed as part of the yard routes.
259 #
260 # CALLING SYNTAX:
261 # $result = &EndToEnd(\%SimulationData);
262 #
263 # ARGUMENTS:
264 # $SimulationData Pointer to %SimulationData hash.
265 #
266 # RETURNED VALUES:
267 # 0 = Success, 1 = Error.
268 #
269 # ACCESSED GLOBAL VARIABLES:
270 # None.
271 # =====
272 sub EndToEnd {
273     my($SimulationData) = @_ ;
274
275     my %EndToEnd = (
276         '00' => {'Step' => '00', 'MaxStep' => 0, 'Timeout' => 0},
277 # ===
278         '01' => {'Desc' => 'Train 1 in holdover B01.',
279                 'Time' => 3, 'SensorBit' => 'B01'},
280
281         '02' => {'Desc' => 'Train 1 holdover leaving B01. aT03o, aT01o',
282                 'Time' => 2, 'SensorBit' => 'B01:S03'},
283
284         '03' => {'Desc' => 'Train 1 upgrade enters B03.',
285                 'Time' => 2, 'SensorBit' => 'B03:B01:S01'},
286
287         '04' => {'Desc' => 'Train 1 upgrade in-transit B03.',
288                 'Time' => 5, 'SensorBit' => 'B03',
289                 'Turnout' => 'T08:Open,T10:Open'}, # exercise
290
291         '05' => {'Desc' => 'Train 1 upgrade enters B05.',
292                 'Time' => 2, 'SensorBit' => 'B05:B03'},
293
294         '06' => {'Desc' => 'Train 1 upgrade in-transit B05.',
295                 'Time' => 5, 'SensorBit' => 'B05'},
296
297         '07' => {'Desc' => 'Train 1 upgrade leaving B05. aT06o',
298                 'Time' => 2, 'SensorBit' => 'B06:B05:S06'},
299

```

```

300 '08' => {'Desc' => 'Train 1 upgrade in-transit B06. T07c',
301         'Time' => 5, 'SensorBit' => 'B06:S07',
302         'Turnout' => 'T07:Close'},          # set for B07.
303
304 '09' => {'Desc' => 'Train 1 upgrade enters B07.',
305         'Time' => 2, 'SensorBit' => 'B07:B06:S07:S08'},
306
307 '10' => {'Desc' => 'Train 1 upgrade in-transit B07. GC1 active',
308         'Time' => 4, 'SensorBit' => 'B07:GC1 AprE'},
309
310 '11' => {'Desc' => 'Train 1 upgrade in-transit B07. Route to B10',
311         'Time' => 2, 'SensorBit' => 'B07:GC1 Road:GC1 AprE',
312         'YardRoute' => '1->3'},
313
314 '12' => {'Desc' => 'Train 1 upgrade enters B10. Route B10',
315         'Time' => 2, 'SensorBit' => 'B10:B07:GC1 AprW:GC1 Road',
316         'YardRoute' => '3->3'},
317
318 '13' => {'Desc' => 'Train 1 yard in-transit B10. Route to B08',
319         'Time' => 5, 'SensorBit' => 'B10',
320         'Turnout' => 'T12:Open,T13:Open,T14:Open,T15:Open', # exercise
321         'YardRoute' => '3->2'},
322 # ===
323 '14' => {'Desc' => 'Train 1 yard leaving B10. GC2 active',
324         'Time' => 3, 'SensorBit' => 'B10:GC2 AprW'},
325
326 '15' => {'Desc' => 'Train 1 downgrade enters B08.',
327         'Time' => 2, 'SensorBit' => 'B08:B10:GC2 Road:GC2 AprW'},
328
329 '16' => {'Desc' => 'Train 1 downgrade in-transit B08.',
330         'Time' => 2, 'SensorBit' => 'B08:B10:GC2 AprE:GC2 Road'},
331
332 '17' => {'Desc' => 'Train 1 downgrade in-transit B08. aT07o',
333         'Time' => 2, 'SensorBit' => 'B08:S09:GC2 AprE'},
334
335 '18' => {'Desc' => 'Train 1 downgrade enters B06.',
336         'Time' => 2, 'SensorBit' => 'B06:B08:S09:S07'},
337
338 '19' => {'Desc' => 'Train 1 downgrade in-transit B06.',
339         'Time' => 5, 'SensorBit' => 'B06:S07'},
340
341 '20' => {'Desc' => 'Train 1 downgrade enters B04.',
342         'Time' => 5, 'SensorBit' => 'B04:B06'},
343
344 '21' => {'Desc' => 'Train 1 downgrade leaving B04. aT05c',
345         'Time' => 3, 'SensorBit' => 'B03:B04:S05'},
346
347 '22' => {'Desc' => 'Train 1 downgrade in-transit B03. B01 occupied',
348         'Time' => 5, 'SensorBit' => 'B03:B01'},
349
350 '23' => {'Desc' => 'Train 1 downgrade in-transit B03. aT01o aT03c',
351         'Time' => 3, 'SensorBit' => 'B03:B01:S01'},
352
353 '24' => {'Desc' => 'Train 1 downgrade enters B02.',
354         'Time' => 3, 'SensorBit' => 'B02:B03:B01'},
355 # ===
356 '25' => {'Desc' => 'Train 1 holdover in-transit B02.',
357         'Time' => 5, 'SensorBit' => 'B02'},
358
359 '26' => {'Desc' => 'Train 2 holdover leaving B02. aT02o aT01c',

```

```

360         'Time' => 2, 'SensorBit' => 'B02:S02'},
361
362 '27' => {'Desc' => 'Train 2 upgrade enters B03.',
363         'Time' => 2, 'SensorBit' => 'B03:B02:S01'},
364
365 '28' => {'Desc' => 'Train 2 upgrade in-transit B03.',
366         'Time' => 5, 'SensorBit' => 'B03',
367         'Turnout' => 'T08:Open,T10:Open'},      # exercise
368
369 '29' => {'Desc' => 'Train 2 upgrade enters B05.',
370         'Time' => 2, 'SensorBit' => 'B05:B03'},
371
372 '30' => {'Desc' => 'Train 2 upgrade in-transit B05.',
373         'Time' => 5, 'SensorBit' => 'B05'},
374
375 '31' => {'Desc' => 'Train 2 upgrade leaving B05. aT06o',
376         'Time' => 2, 'SensorBit' => 'B06:B05:S06'},
377
378 '32' => {'Desc' => 'Train 2 upgrade in-transit B06. T07o.',
379         'Time' => 4, 'SensorBit' => 'B06:S07',
380         'Turnout' => 'T07:Open'},              # set for B08.
381
382 '33' => {'Desc' => 'Train 2 upgrade enters B08.',
383         'Time' => 2, 'SensorBit' => 'B08:B06:S07:S09'},
384
385 '34' => {'Desc' => 'Train 2 upgrade in-transit B08. GC2 active',
386         'Time' => 4, 'SensorBit' => 'B08:GC2 AprE'},
387
388 '35' => {'Desc' => 'Train 2 upgrade in-transit B08. Route to B09',
389         'Time' => 2, 'SensorBit' => 'B08:GC2 Road:GC2 AprE',
390         'YardRoute' => '2->4'},
391
392 '36' => {'Desc' => 'Train 2 upgrade enters B09. Route B09',
393         'Time' => 2, 'SensorBit' => 'B09:B08:GC2 AprW:GC2 Road',
394         'YardRoute' => '4->4'},
395
396 '37' => {'Desc' => 'Train 2 yard in-transit B09. Route to B07',
397         'Time' => 5, 'SensorBit' => 'B09',
398         'Turnout' => 'T16:Open,T17:Open',      # exercise
399         'YardRoute' => '4->1'},
400 # ===
401 '38' => {'Desc' => 'Train 2 yard in-transit B09. GC1 active',
402         'Time' => 3, 'SensorBit' => 'B09:GC1 AprW'},
403
404 '39' => {'Desc' => 'Train 2 downgrade enters B07.',
405         'Time' => 2, 'SensorBit' => 'B07:B09:GC1 Road:GC1 AprW'},
406
407 '40' => {'Desc' => 'Train 2 downgrade in-transit B07.',
408         'Time' => 2, 'SensorBit' => 'B07:GC1 AprE:GC1 Road'},
409
410 '41' => {'Desc' => 'Train 2 downgrade in-transit B07. S08, GC1 AprE.',
411         'Time' => 2, 'SensorBit' => 'B07:S08:GC1 AprE'},
412
413 '42' => {'Desc' => 'Train 2 downgrade enters B06. aT07c',
414         'Time' => 2, 'SensorBit' => 'B06:B07:S07:S08'},
415
416 '43' => {'Desc' => 'Train 2 downgrade in-transit B06.',
417         'Time' => 5, 'SensorBit' => 'B06:S07'},
418
419 '44' => {'Desc' => 'Train 2 downgrade enters B04.',

```

```

420         'Time' => 5, 'SensorBit' => 'B04:B06'},
421
422     '45' => {'Desc' => 'Train 2 downgrade leaving B04. aT05c',
423             'Time' => 3, 'SensorBit' => 'B03:B04:S05'},
424
425     '46' => {'Desc' => 'Train 2 downgrade in-transit B03. B02 occupied',
426             'Time' => 5, 'SensorBit' => 'B03:B02'},
427
428     '47' => {'Desc' => 'Train 2 downgrade in-transit B03. aT01c aT02c',
429             'Time' => 3, 'SensorBit' => 'B03:B02:S01'},
430
431     '48' => {'Desc' => 'Train 2 downgrade enters B01.',
432             'Time' => 3, 'SensorBit' => 'B01:B03:B02'},
433
434     '49' => {'Desc' => 'Train 2 holdover in-transit B01.',
435             'Time' => 5, 'SensorBit' => 'B01',
436             'Turnout' => 'T16:Close,T17:Close'},      # exercise
437 # ===
438     '50' => {'Desc' => 'Cycle complete. Reset all sensor bits.',
439             'Time' => 3, 'SensorBit' => ''}
440 );
441
442 %SimulationData = %{ dclone(\%EndToEnd) };
443 return 0;
444 }
445
446 return 1;
447

```