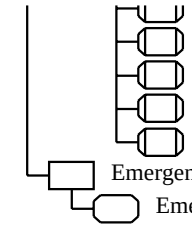
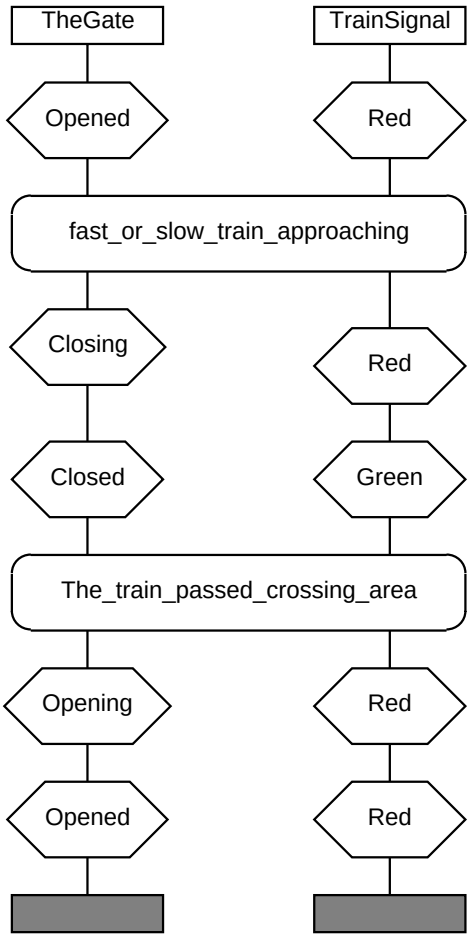
Table of Contents

SDT		rw	C:\Documents and Settings\Administrator\My Documents\SAM2002contest\Li_probert_williams\Railway.sdt	
		rw	C:\Documents and Settings\Administrator\My Documents\SAM2002contest\Li_probert_williams\	
	High Level Requirements Model MSCs	4		
	HL01	4	rw	HL01.msc
	HL02	5	rw	HL02.msc
	HL03	6	rw	HL03.msc
	HL04	7	rw	HL04.msc
	Critical Scenario Model MSCs	8		
	CS01	8	rw	CS01.msc
	CS02	9	rw	CS02.msc
	CS03	10	rw	CS03.msc
	Manual Override Model MSCs	11		
	MC01	11	rw	MC01.msc
	MC02	12	rw	MC02.msc
	Emergency Handling MSCs	13		
	EM01	13	rw	EM01.msc
	EM02	14	rw	EM02.msc
	EM03	15	rw	EM03.msc
	EM04	16	rw	EM04.msc
	EM05	17	rw	EM05.msc
	SDL System Structure	18		
	RailwayCrossing	18	rw	RailwayCrossing.ssy
x:y	tr (...) : Tracks			
	Tracks	20	rw	Tracks.sbt
	OutSensor	21	rw	OutSensor.spr
	InSensor	22	rw	InSensor.spr
	TrainSignal	23	rw	TrainSignal.spr
	TrackControl	24	rw	TrackControl.spr
	CarSensor	26	rw	CarSensor.sbk
	CarSensor	27	rw	CarSensor.spr
	TheGate	28	rw	TheGate.sbk
	TheGate	29	rw	TheGate.spr
	Controller	30	rw	Controller.sbk
	Controller	31	rw	Controller.spr
	ResetData	46	rw	ResetData.spd
	ResetTimer	47	rw	ResetTimer.spd
	FindSender	48	rw	FindSender.spd



CloseReq	49	rw	CloseReq.spd
SetGreen	50	rw	SetGreen.spd
EmergencyOpenF	51	rw	EmergencyHandle.spd
EmergencyCloseF	52	rw	EmergencyCloseF.spd
EmergencySetRedF	53	rw	EmergencySetRedF.spd
EmergencySignal	54	rw	EmergencySignal.sbk
EmergencySignal	55	rw	EmergencySignal.spr
ModifiedCarSensor	56	rw	ModifiedCarSensor.spr
TestCases		rw	TestCases.ssc
Simulator Trace MSCs			
TestCase1_1		rw	TestCase1_1.msc
TestCase1_2		rw	TestCase1_2.msc
TestCase1_3		rw	TestCase1_3.msc
TestCase1_4		rw	TestCase1_4.msc
TestCase1_5		rw	TestCase1_5.msc
TestCase1_6		rw	TestCase1_6.msc
TestCase1_7		rw	TestCase1_7.msc
TestCase2_1		rw	TestCase2_1.msc
TestCase2_2		rw	TestCase2_2.msc
TestCase2_3		rw	TestCase2_3.msc
TestCase2_4		rw	TestCase2_4.msc
TestCase3_1		rw	TestCase3_1.msc
TestCase3_2		rw	TestCase3_2.msc
TestCase3_3		rw	TestCase3_3.msc
TestCase3_4		rw	TestCase3_4.msc
TestCase3_5		rw	TestCase3_5.msc
TestCase4_1		rw	TestCase4_1.msc
TestCase4_2		rw	TestCase4_2.msc
TestCase4_3		rw	TestCase4_3.msc
TestCase4_4		rw	TestCase4_4.msc
TestCase4_5		rw	TestCase4_5.msc
TestCase4_6		rw	TestCase4_6.msc
TestCase4_7		rw	TestCase4_7.msc
TestCase4_8		rw	TestCase4_8.msc

MSC HL01



/* High level requirement model #1

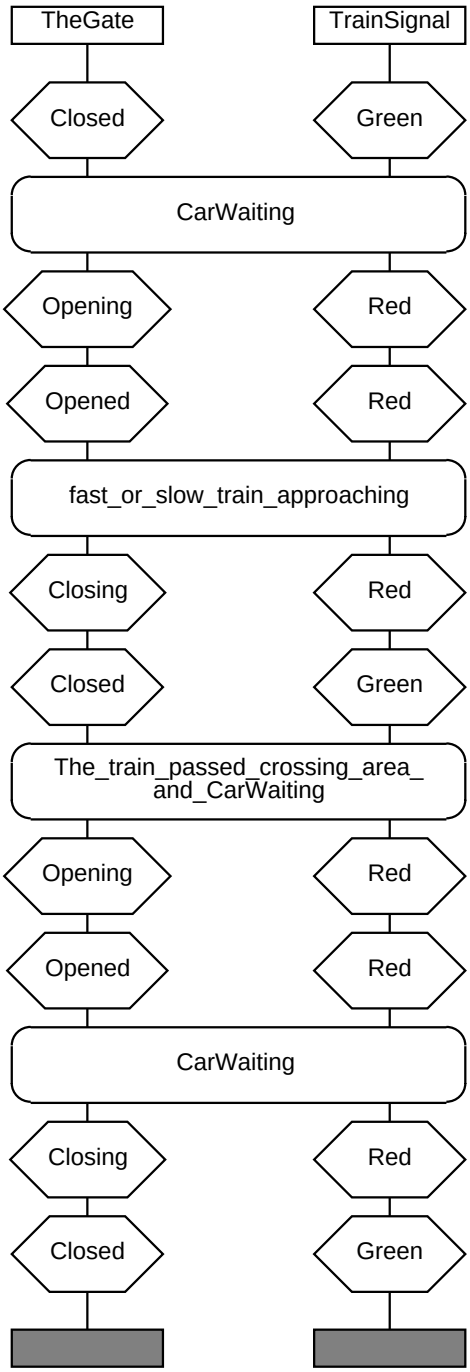
Simplest Generic Scenario (no car)
−− Applicable to all strategies

Start State: Gate is Open and TrainSignal is Red
End State: Gate is Open and TrainSignal is Red

Actions:
1. When a train (fast or slow) is approaching,
   close the gate, and set TrainSignal to Green.
2. After the train passed the crossing area,
   set TrainSignal to Red, and open the gate.

*/

MSC HL02



/* High level requirement model #2

All trains have precedence scenario
−− Applicable to Strategy 2

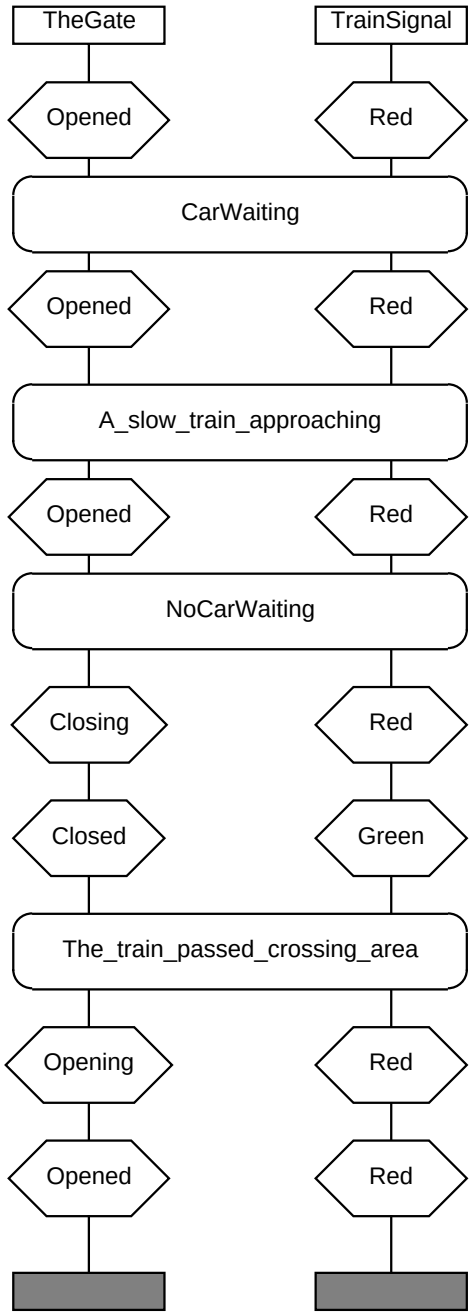
(In Strategy 2, normally the gate is closed)

Start State: Gate is Closed and TrainSignal is Green
End State: Gate is Closed and TrainSignal is Green

Actions:
1. Car waiting and no train, open the gate
2. A train (fast/slow) is approaching, close the gate
3. The train passed the crossing area, and car waiting, open the gate
4. No car waiting, close the gate.

*/

MSC HL03



/* High level requirement model #3

Normal slow train urban scenario
−− Applicable to Strategy 3

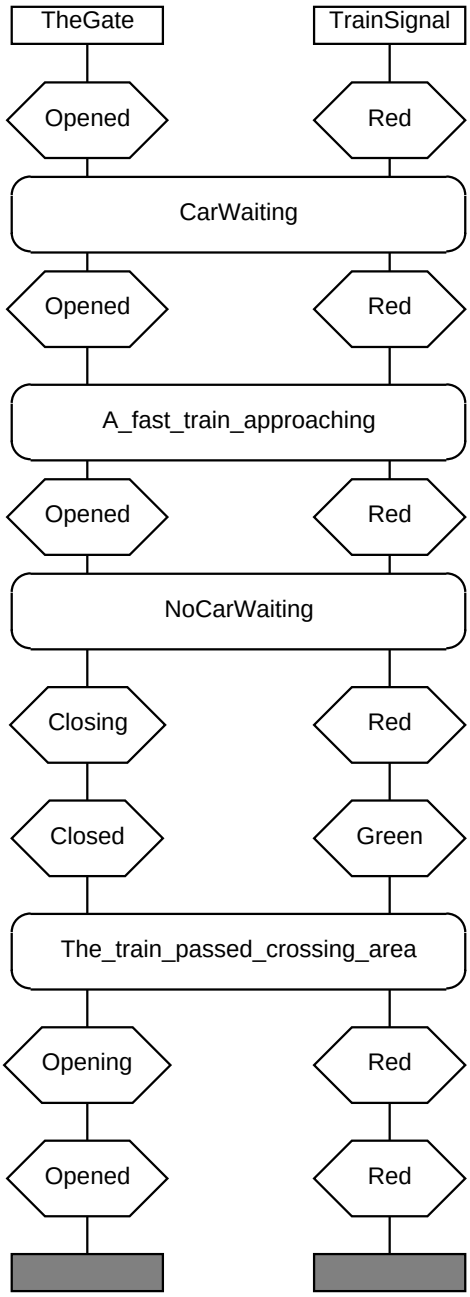
(In Strategy 3, normally the gate is open to expedite car traffic)

Start State: Gate is Open and TrainSignal is Red
End State: Gate is Open and TrainSignal is Red

Actions:
1. Car waiting, keep the gate open.
2. A slow train is approaching when car waiting,
   stop the train and keep the gate open.
3. No car waiting and a slow train is waiting,
   close the gate, and set TrainSignal Green.
4. The train has passed crossing area,
   set TrainSignal Red, and open the gate

*/

MSC HL04



/* High level requirement model #4

Car has precedence scenario
−− Applicable to Strategy 4

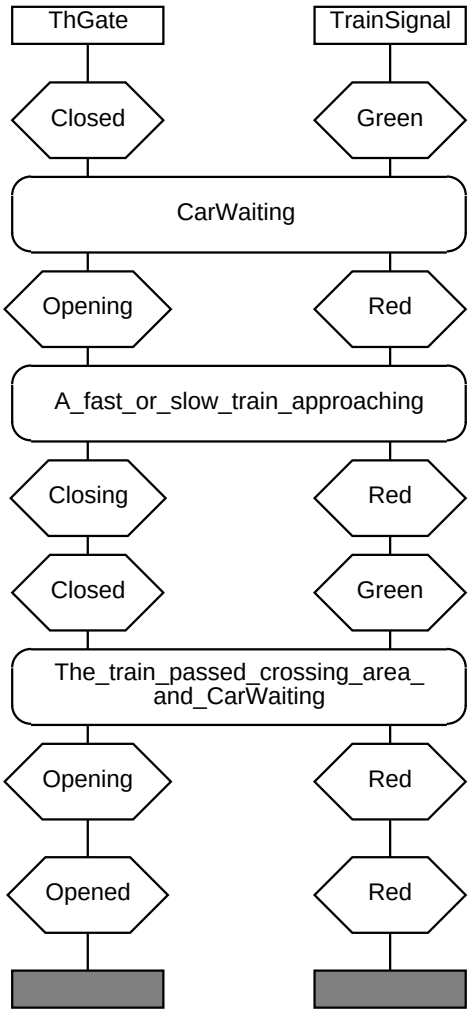
(In Strategy 4, normally the gate is open to expedite car traffic)

Start State: Gate is Opened and TrainSignal is Red
End State: Gate is Opened and TrainSignal is Red

Actions:
1. Car waiting, keep the gate open.
2. A fast train is approaching and car waiting,
   stop the train and keep the gate open
3. No car waiting, close the gate, and set TrainSignal Green
4. The train has passed crossing area,
   set TrainSignal Red and open the gate

*/

MSC CS01



/* Critical Scenario model #1

High Risk Critical Race Scenario − Trains have precedence
−− applicable to Strategy 2

(In Strategy 2, normally the gate is closed)

Start State: Gate is Closed and TrainSignal is Green
End State: Gate is Open and TrainSignal is Red

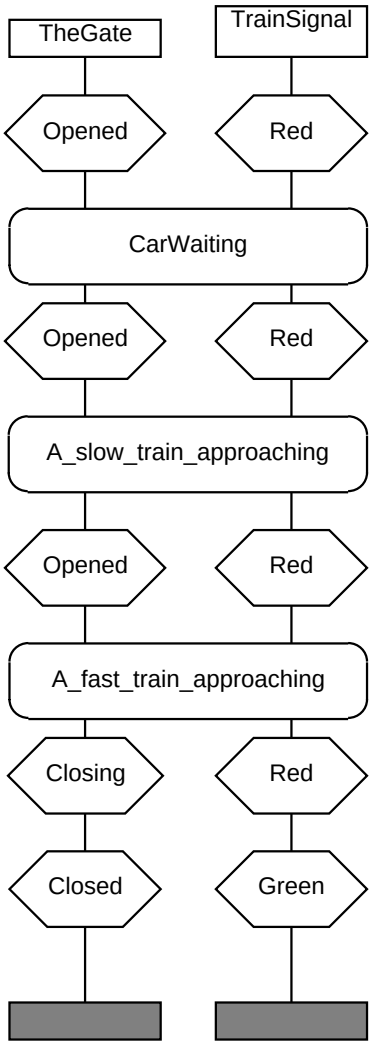
Actions:
1. Car waiting and no train,
set TrainSignal Red and open the gate

2. While the gate is raising,
a train (fast/slow) approaches.
Stop raising the gate, and close it instead.
Then set TrainSignal Green

3. When the train has passed crossing area,
set TrainSignal Red and open the gate

*/

MSC CS02



/* Critical Scenario model #2

Critical Race Scenario − Slow and Fast Train Arriving

−− Applicable to Strategy 3

(In Strategy 3, normally the gate is open to expedite car traffic)

Start State: Gate is Open and TrainSignal is Red
End State: Gate is Closed and TrainSignal is Green

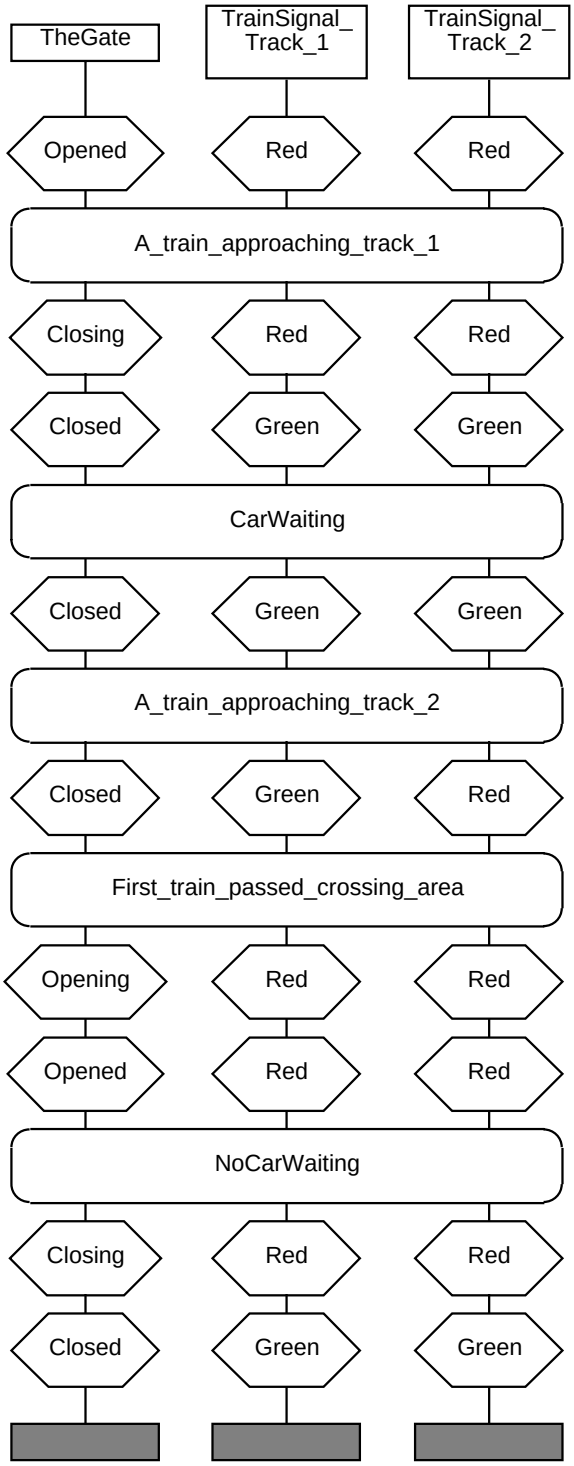
Actions:
1. Car waiting, keep the gate open.

2. A slow train approaches and car waiting,
stop the train and keep the gate open

3. A fast train approaches,
close the gate, set TrainSignal to Green and let all trains go

*/

MSC CS03



/* Critical Scenario model #3

Critical Race − Multiple Tracks and Trains −− Cars Have Precedence

−− Applicable to Strategy 4

(In Strategy 4, normally the gate is open to expedite car traffic)

Start State: Gate is Open and TrainSignal is Red
End State: Gate is Closed and TrainSignal is Green

Actions:
1. A train approaches Track 1,
   close the gate, and set TrainSignal to Green

2. Car waiting and the train is still in the crossing area,
   keep cars waiting, until the train leaves

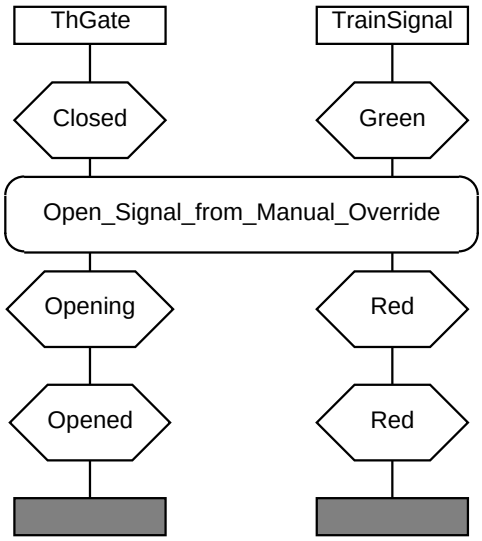
3. Another train approaches Track 2.
   Since car waiting and cars have precedence,
   stop the train by setting TrainSingal on Track 2 to Red,
   and keep the gate closed

4. When the train on Track 1 has passed the crossing area,
   set TrainSignal on Track 1 to Red, and open the gate for cars

5. When no car waiting,
   close the gate for the waiting train,
   and set TrainSignal on all tracks to Green.

*/

MSC MC01



/* Manual Override Scenario model #1

Manually Open Gate − Safe Scenario

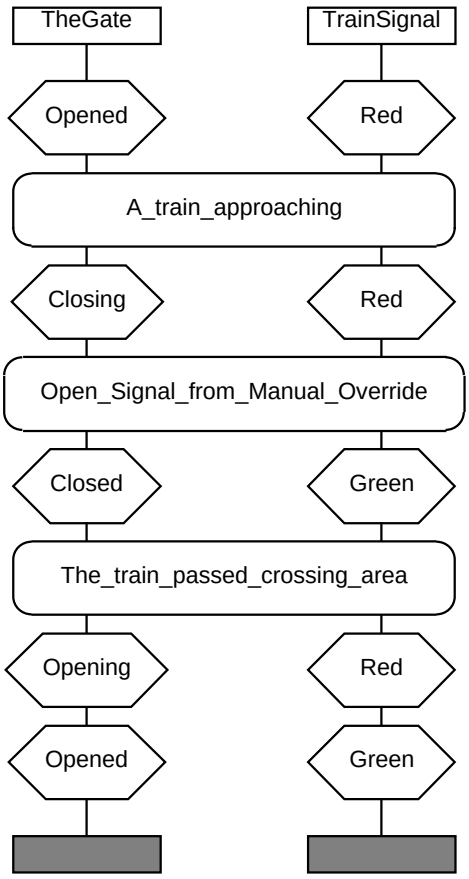
Start State: Gate is Closed and TrainSignal is Green
End State: Gate is Open and TrainSignal is Red

Actions:

Send Open signal manually
−− No train in crossing area, open the gate

*/

MSC MC02



/* Manual Override Scenario model #2

Unsafe Manually Stopping Gate Closing − Train Approaching

Start State: Gate is Open and TrainSignal is Red
End State: Gate is Open and TrainSignal is Red

Actions:

Send Open signal manually when the gate is lowing down

−− Check if there are trains approaching.

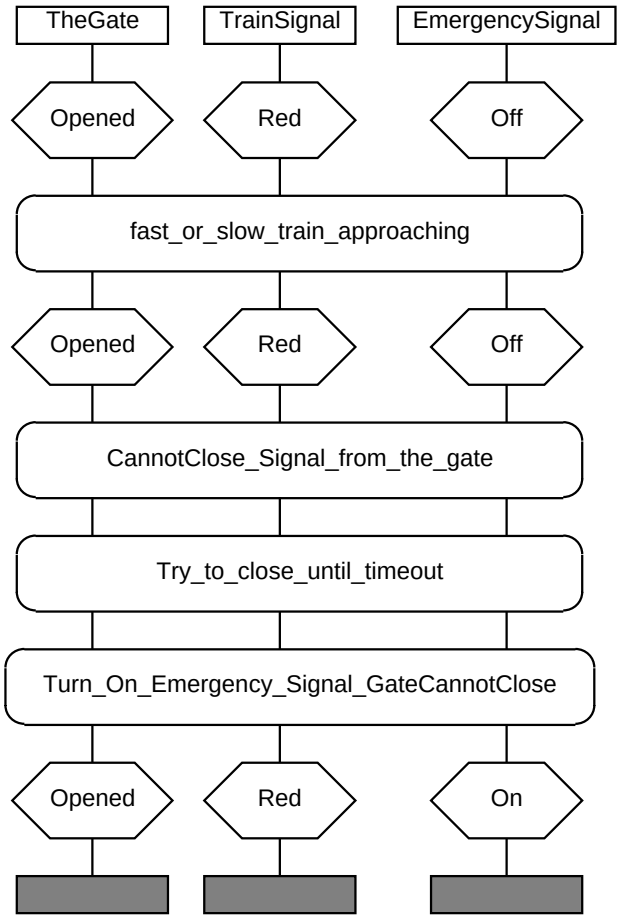
If there is one, continue closing the gate,
set TrainSignal to Green and let the train pass.

After the train passes, open the gate

Note: The unsafe action is not allowed by RCS

*/

MSC EM01



/* Emergency Handling Model #1

Gate Cannot Close Scenario
−− Applicable to all strategies

Start State: Gate is Open
TrainSignal is Red
EmergencySignal is Off

End State: Gate is Open
TrainSignal is Red
EmergencySignal is On

Actions:

1. A train approaches,
try to close the gate

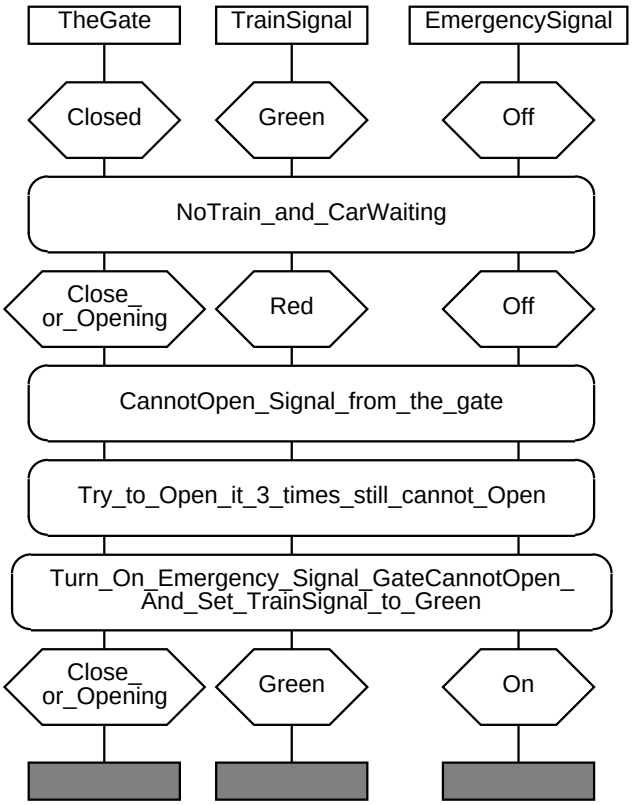
2. Gate cannot close
−− either half way down or cannot move at all

3. Try to close it again until timeout
(Train cannot stop safely after this moment)

4. Turn on Emergency Signal "GateCannotClose"
to inform Trains and Cars and maintenance personnel
RCS is locked

*/

MSC EM02



/* Emergency Handling Model #2

Gate Cannot Open Scenario
−− Applicable to all strategies

Start State: Gate is Closed
TrainSignal is Green
EmergencySignal is Off

End State: Gate is Closed/half way up
TrainSignal is Green
EmergencySignal is On

Actions:

1. No train in crossing area and Car Waiting
set TrainSignal to Red and try to open the gate

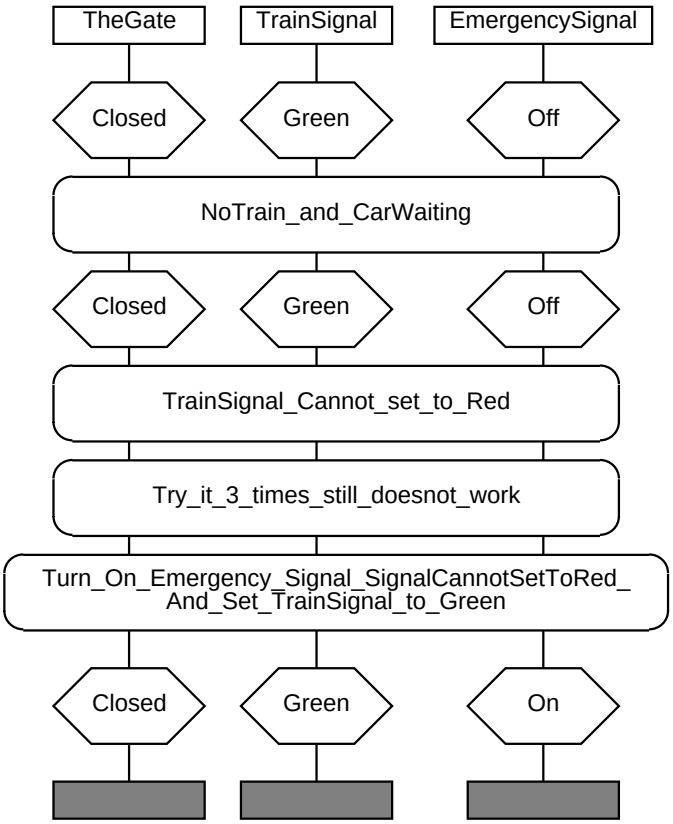
2. Gate cannot open
−− either half way up or cannot move at all

3. Try to open it again for 3 times

4. Still cannot open,
Turn on Emergency Signal "GateCannotOpen"
to inform maintenance personnel
and set TrainSignal to Green
to let all upcoming train pass

*/

MSC EM03



/* Emergency Handling Model #3

TrainSignal Cannot Set to Red Scenario
−− Applicable to all strategies

Start State: Gate is Closed
TrainSignal is Green
EmergencySignal is Off

End State: Gate is Closed
TrainSignal is Green
EmergencySignal is On

Actions:

1. No train in crossing area and Car Waiting
set TrainSignal to Red

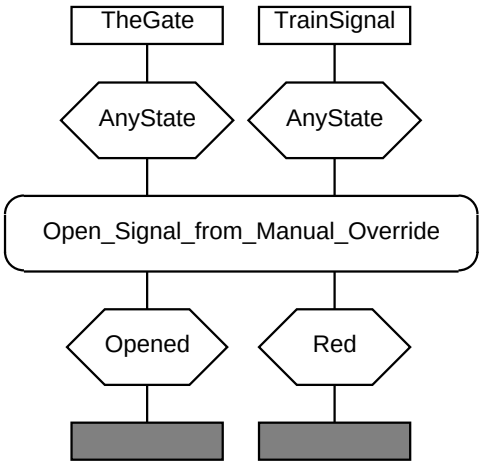
2. One of the TrainSignals cannot be set to Red

3. Try it again for 3 times

4. Still cannot set to Red,
Turn on Emergency Signal
"TrainSignal Cannot Set to Red"
to inform maintenance personnel
and set TrainSignal to Green
to let all upcoming train pass

*/

MSC EM04



/* Emergency Handling Model #4

Unlock RCS After Emergency Scenario −− Open gate
−− Applicable to all strategies

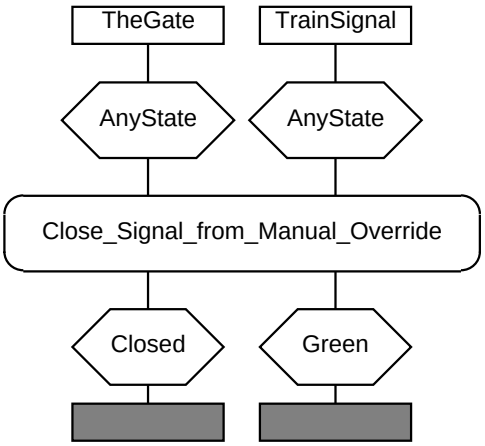
Start State: Any state
End State: Gate is Open
TrainSignal is Red

Actions:

A state after any emergency, send an Open signal
from Manual override, the gate will open
and TrainSignal is set Red. RCS is unlocked.

*/

MSC EM05



/* Emergency Handling Model #5

Unlock RCS After Emergency Scenario −− Close gate
−− Applicable to all strategies

Start State: Any state
End State: Gate is Closed
TrainSignal is Green

Actions:

A state after any emergency, send a Close signal
from Manual override, the gate will close
and TrainSignal is set Green. RCS is unlocked.

*/

system RailwayCrossing 1(2)

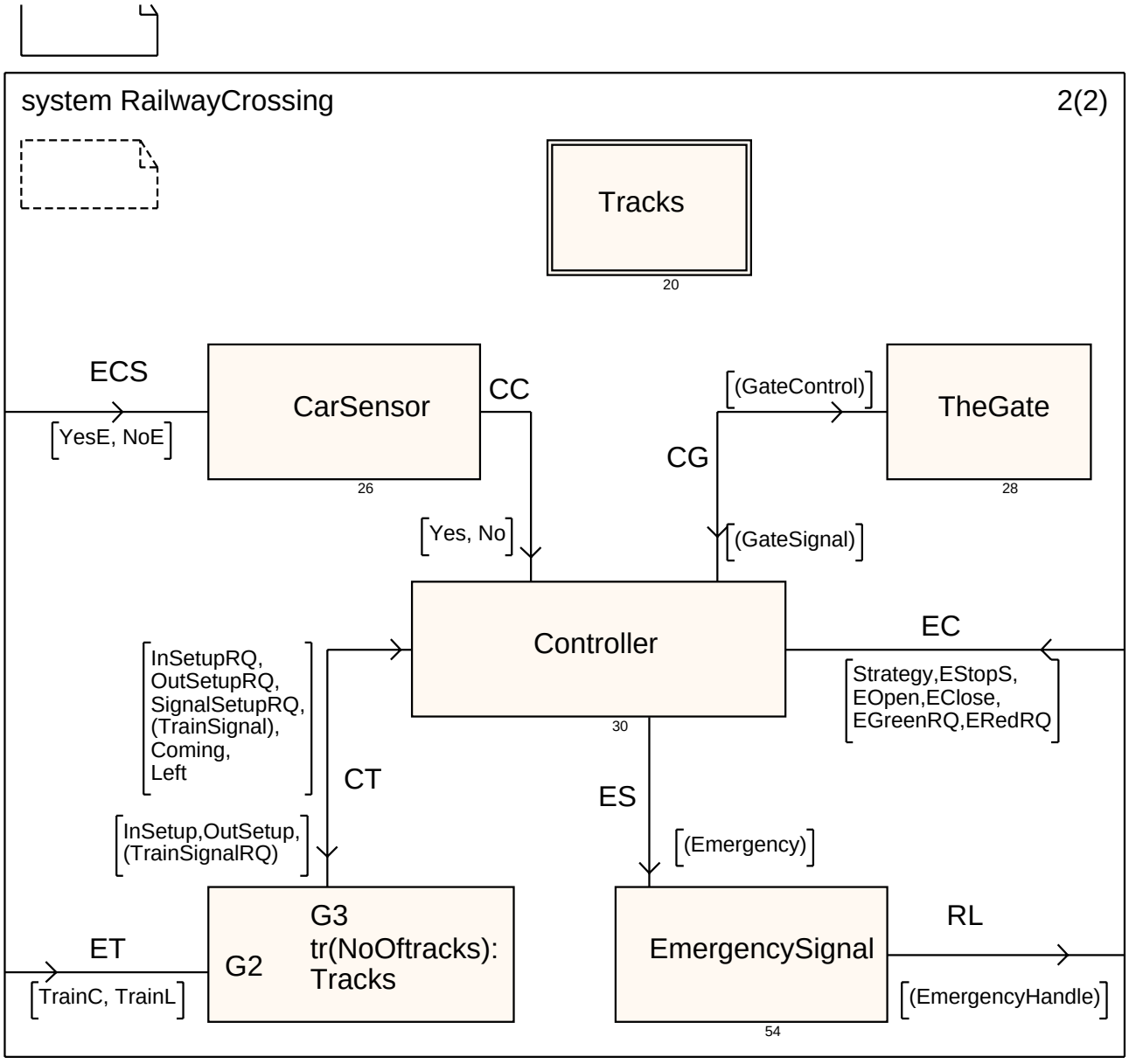
```
/* RailwayCross system consists of a controller, a car sensor, a gate, a set of
emergency  signal, and a number of tracks */
/* Each track consists of a track controller, an InSensor, an OutSensor and a set
of train signals */

/* Four Strategies can be used in this system, and they can be switched
dynamically when the system is running. They are:

 1−− Manual Override
 2 −−Trains take precedence
 3 −− Fast train − many cars − slow train
 4 −− Many cars − train */
```

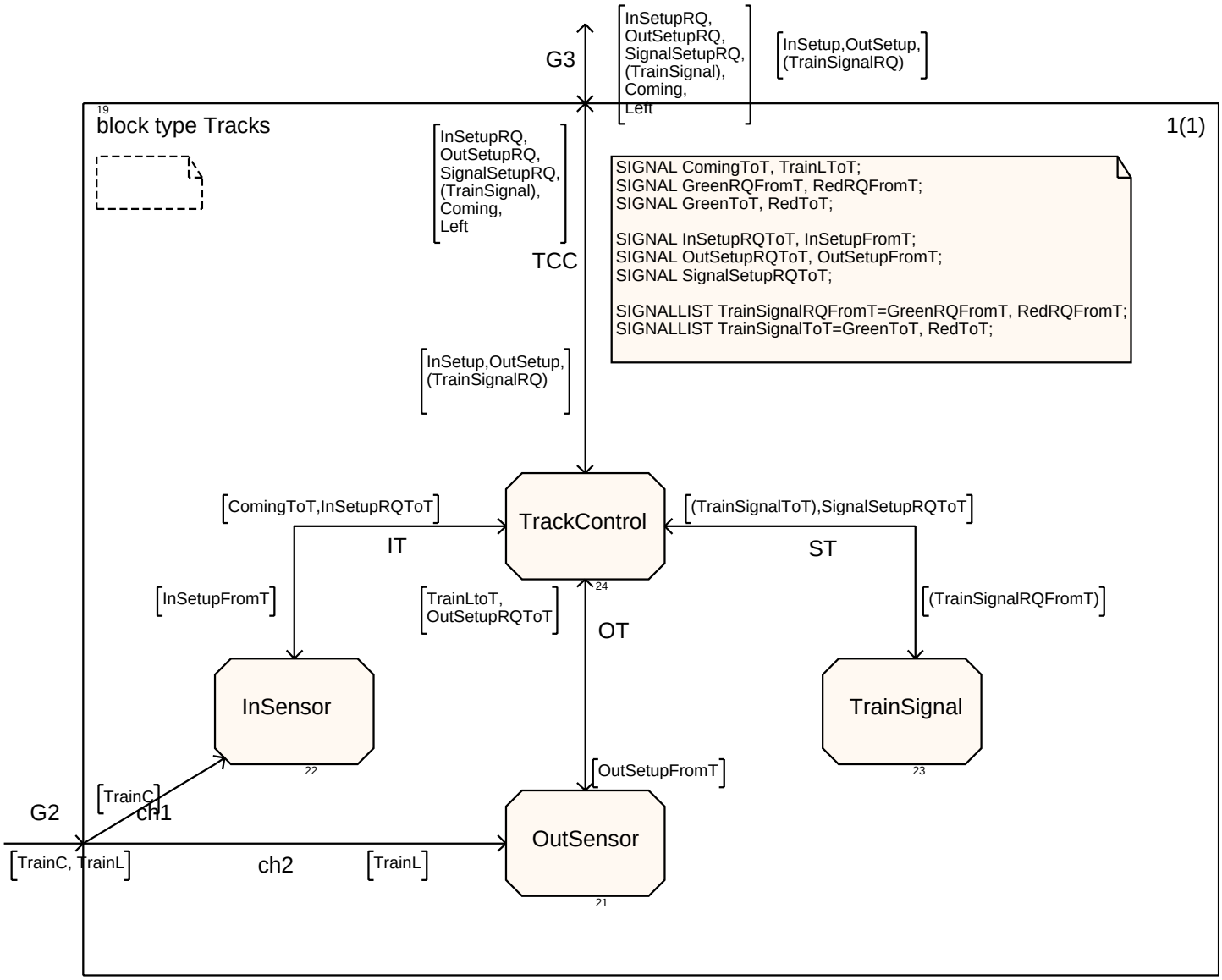
```
SIGNAL Yes, No,YesE,NoE; /* Yes−−more than one car waiting*/
SIGNAL TrainC, Coming;
SIGNAL TrainL, Left;
SIGNAL GreenRQ, RedRQ;
SIGNAL Green, Red;
SIGNAL Open, Close;
SIGNAL Opened, Closed, Raising, Lowering;

SIGNALLIST TrainSignalRQ=GreenRQ, RedRQ;
SIGNALLIST TrainSignal=Green, Red;
SIGNALLIST GateSignal=Opened, Closed, Raising, Lowering;
SIGNALLIST GateControl=Open, Close;

SYNONYM NoOfTracks Integer =2; /* Constant−−No of tracks*/
SYNONYM NoOfFastTracks Integer =1; /* Constant −− No of fast train tracks*/
```

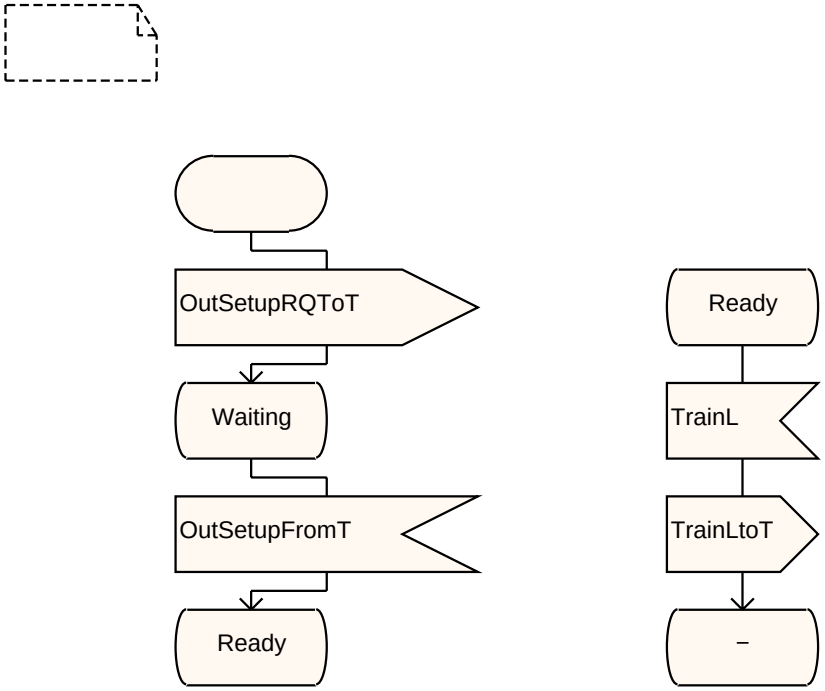
```
/* Emergency Signals */

SIGNAL OpenF, CloseF, SetRedF;
SIGNAL OpenFail, CloseFail, SetRedFail;

SIGNALLIST Emergency=OpenF, CloseF, SetRedF;
SIGNALLIST EmergencyHandle=OpenFail, CloseFail, SetRedFail;
```

```
/* Manual Override singals */

SIGNAL Strategy(Natural);
SIGNAL EStopS;
SIGNAL EOpen, EClose;
SIGNAL EGreenRQ, ERedRQ;
```

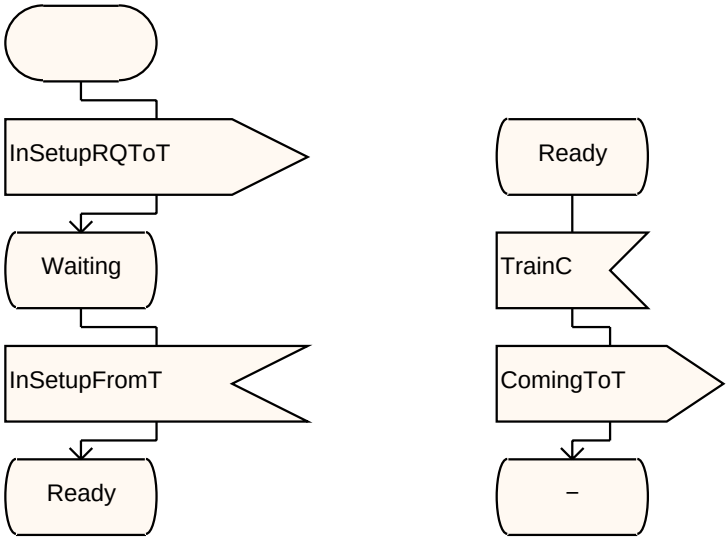
```
/* Setup signals */

SIGNAL InSetupRQ, InSetup;
SIGNAL OutSetupRQ, OutSetup;
SIGNAL SignalSetupRQ;
```


system RailwayCrossing
2(2)
Tracks
20
ECS
YesE, NoE
CarSensor
26
CC
Yes, No
(GateControl)
TheGate
28
CG
(GateSignal)
Controller
30
EC
Strategy,EStopS,
EOpen,EClose,
EGreenRQ,ERedRQ
InSetupRQ,
OutSetupRQ,
SignalSetupRQ,
(TrainSignal),
Coming,
Left
CT
InSetup,OutSetup,
(TrainSignalRQ)
ES
(Emergency)
ET
TrainC, TrainL
G2
G3
tr(NoOftracks):
Tracks
EmergencySignal
54
RL
(EmergencyHandle)

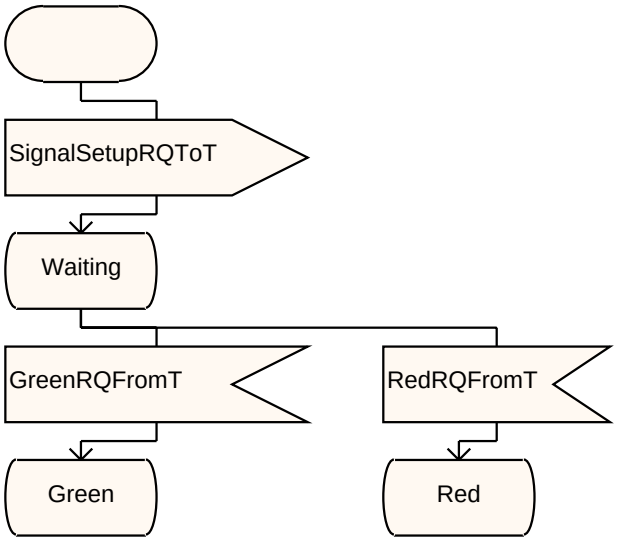

G3
InSetupRQ,
OutSetupRQ,
SignalSetupRQ,
(TrainSignal),
Coming,
Left
InSetup,OutSetup,
(TrainSignalRQ)
19
block type Tracks
1(1)
InSetupRQ,
OutSetupRQ,
SignalSetupRQ,
(TrainSignal),
Coming,
Left
TCC
SIGNAL ComingToT, TrainLToT;
SIGNAL GreenRQFromT, RedRQFromT;
SIGNAL GreenToT, RedToT;
SIGNAL InSetupRQToT, InSetupFromT;
SIGNAL OutSetupRQToT, OutSetupFromT;
SIGNAL SignalSetupRQToT;
SIGNALLIST TrainSignalRQFromT=GreenRQFromT, RedRQFromT;
SIGNALLIST TrainSignalToT=GreenToT, RedToT;
InSetup,OutSetup,
(TrainSignalRQ)
ComingToT,InSetupRQToT
IT
TrackControl
(TrainSignalToT),SignalSetupRQToT
ST
24
InSetupFromT
TrainLtoT,
OutSetupRQToT
OT
(TrainSignalRQFromT)
InSensor
22
TrainSignal
23
OutSetupFromT
TrainC
G2
ch1
OutSensor
TrainC, TrainL
ch2
TrainL
21

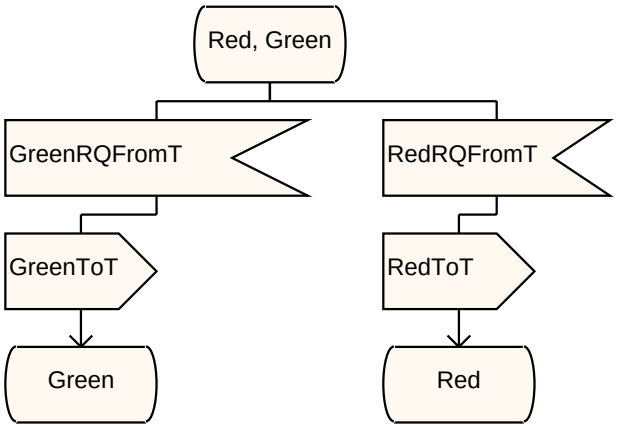

OutSetupRQToT
Waiting
OutSetupFromT
Ready
Ready
TrainL
TrainLtoT
–

process InSensor

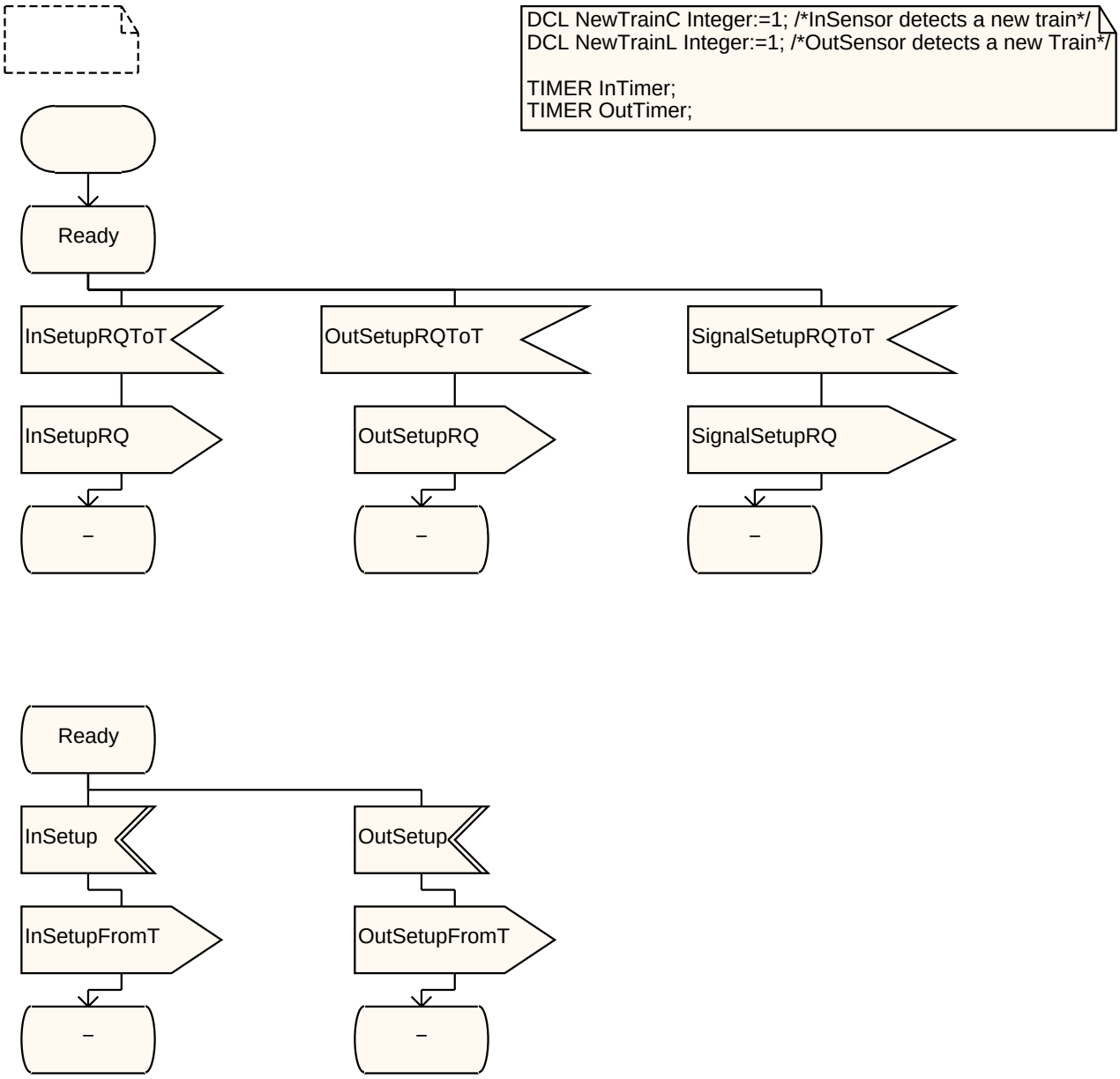


process TrainSignal



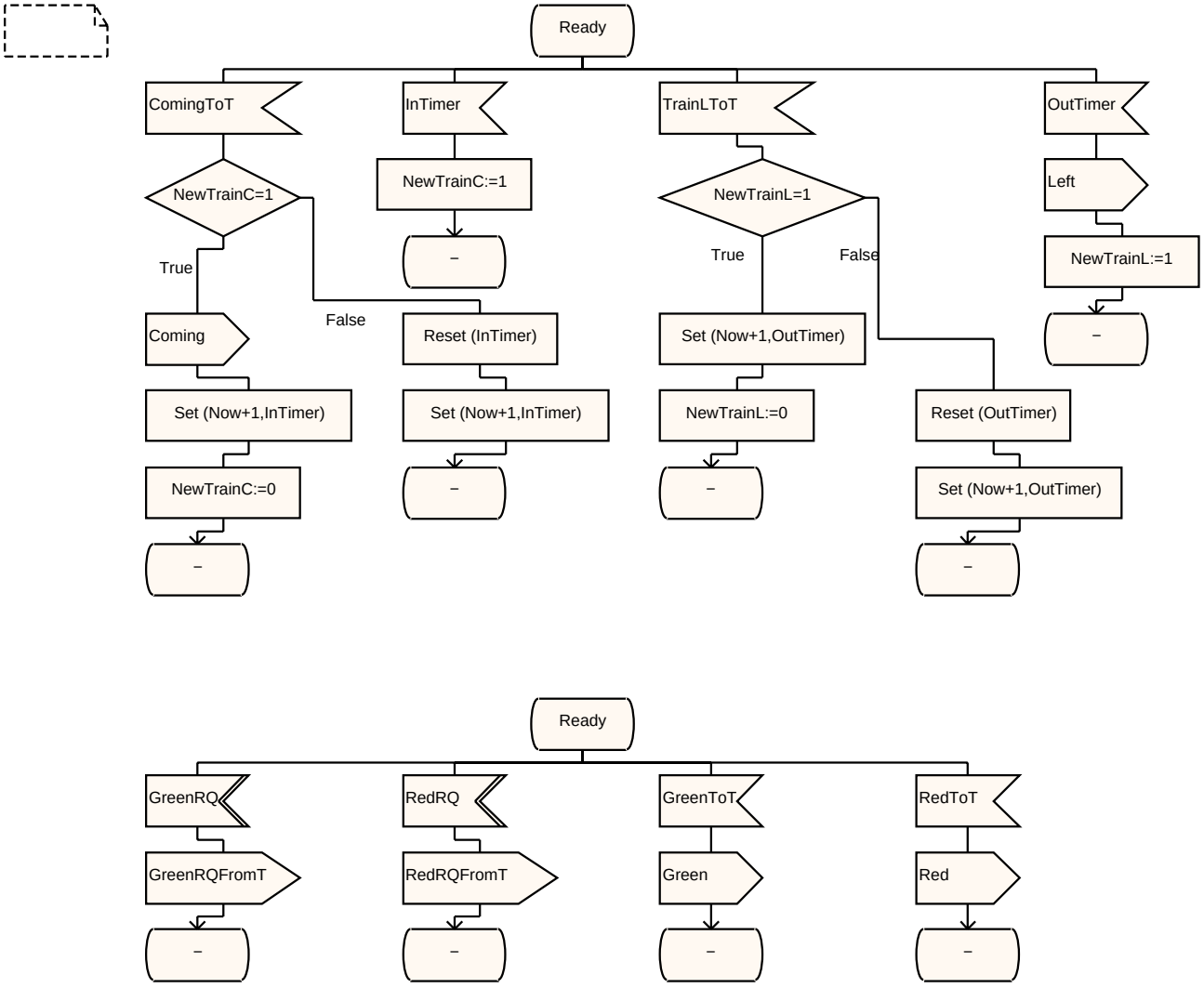


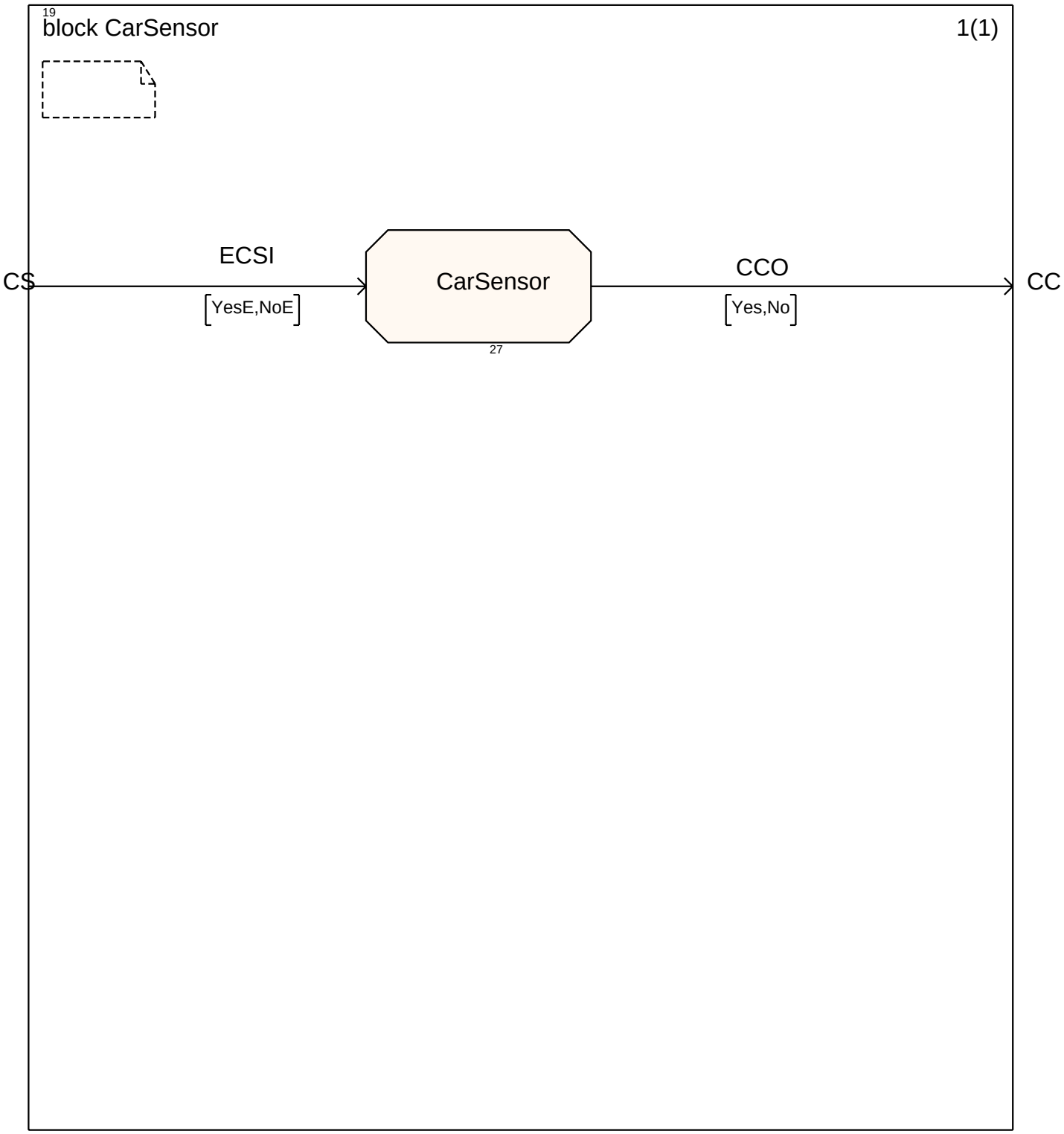
process TrackControl



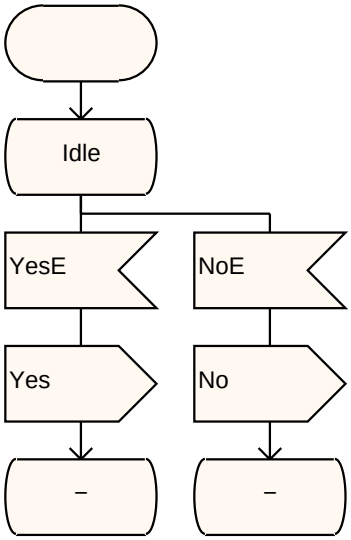
20

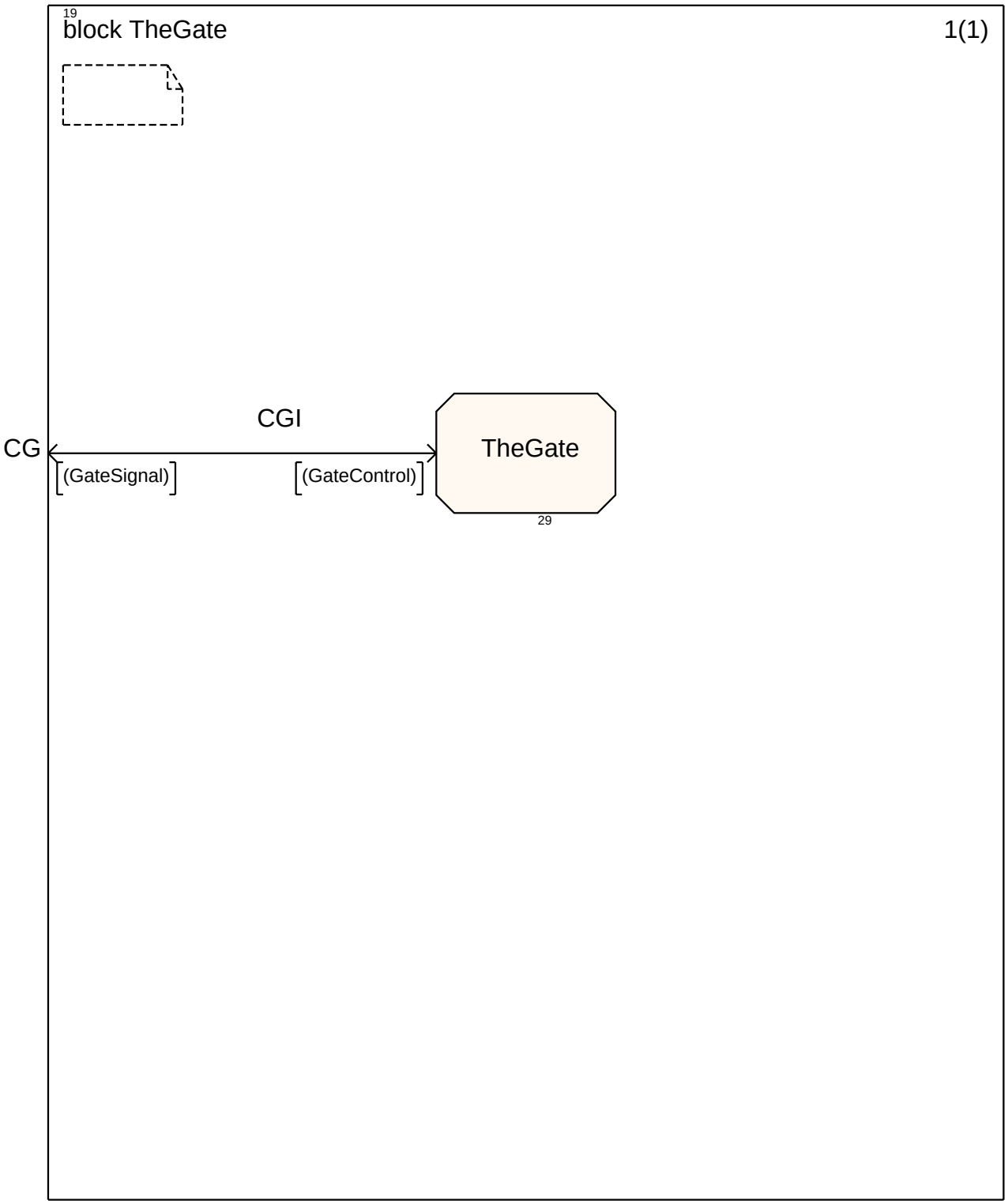
process TrackControl




19
block CarSensor
1(1)
ECSI
[YesE,NoE]
CarSensor
27
CCO
[Yes,No]
CS
CC

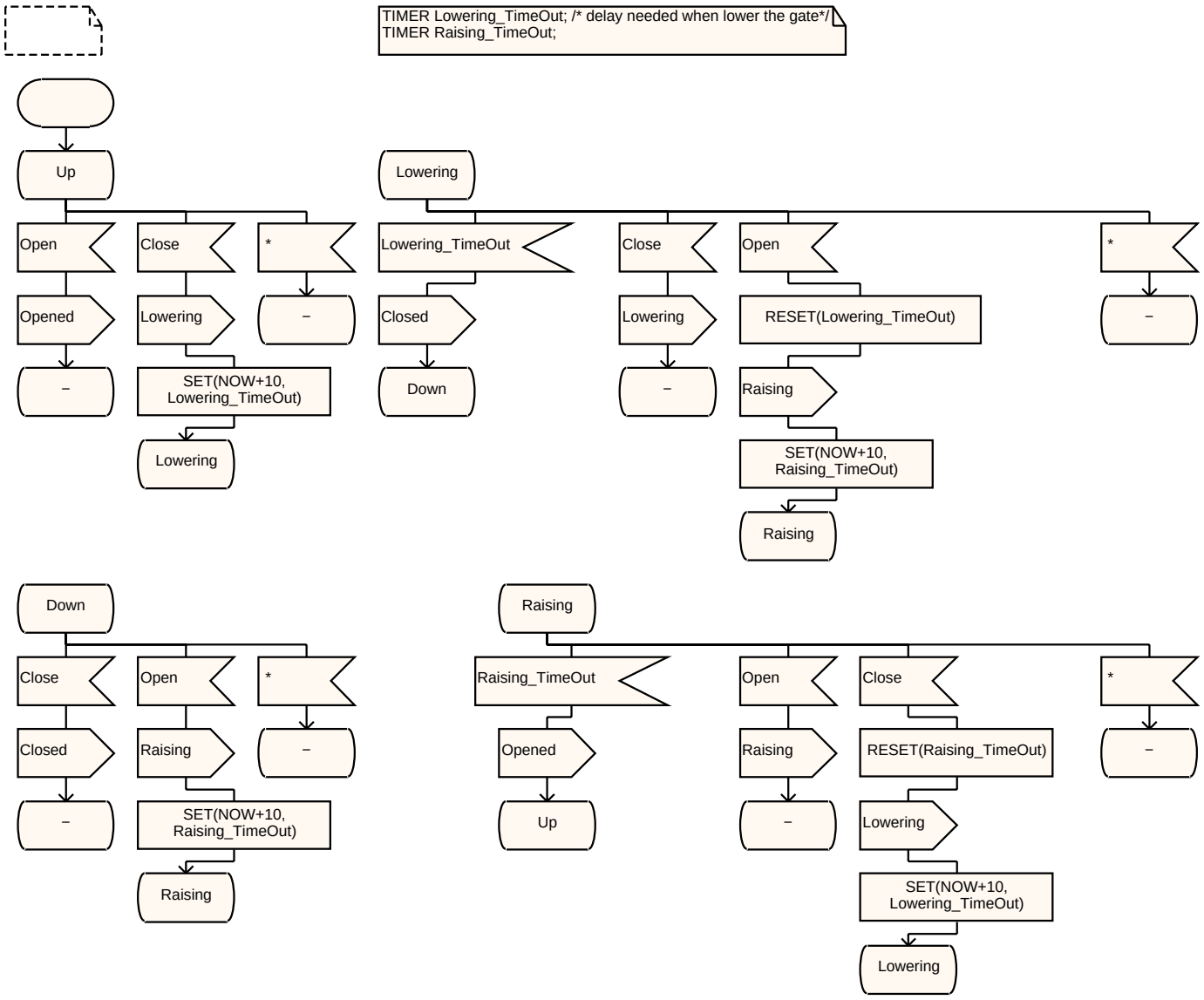
process CarSensor

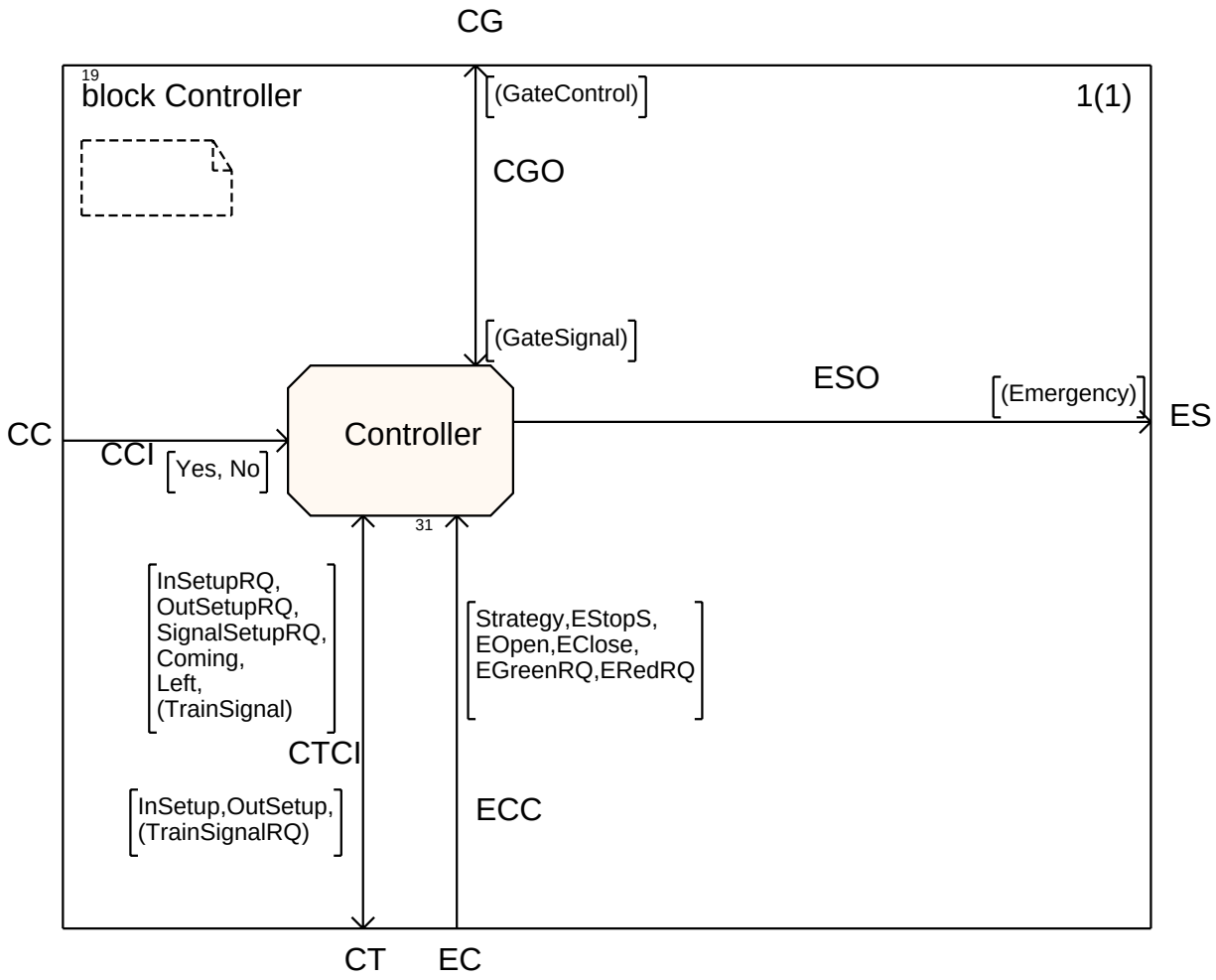



19
block TheGate
1(1)
CGI
CG
TheGate
(GateSignal)
(GateControl)
29

28

process TheGate




CG
19
block Controller
(GateControl)
1(1)
CGO
(GateSignal)
ESO
(Emergency)
ES
CC
CCI
Yes, No
Controller
31
InSetupRQ,
OutSetupRQ,
SignalSetupRQ,
Coming,
Left,
(TrainSignal)
Strategy,EStopS,
EOpen,EClose,
EGreenRQ,ERedRQ
CTCI
InSetup,OutSetup,
(TrainSignalRQ)
ECC
CT
EC

30

process <<block Controller>> Controller 1(15)

```
/* Strategies:

1 Manual Override −− any signal can be sent as long as it is safe
2Trains take precedence −− gate is normally closed
3 Fast train − many cars − slow train −− gate is normally open
   −− Fast train −− close the gate
   −− SlowTrain + NoCarWaiting −− close the gate
   −− SlowTrain + CarWaiting −− Leave the gate open
4 Many cars − train −− gate is normally open
   −− Any train comes, check if car Waiting.
      −− NoCarWaiting −− close the gate
      −− CarWaiting −− Leave the gate open

When gate is open, it is for cars, which means no train is moving in the
Crossing area, although some trains may be stopped and waiting.
Therefore, checking if  "NoOfTrains=0" is not necessary */
```

```
Newtype TrackPId Array(Integer, PId)
endnewtype;
DCL TrackNo TrackPId; /* PId of track control process */
```

```
Newtype TrackCount Array(Integer, Integer)
endnewtype;
DCL TrackCounter TrackCount; /* No of trains in a track*/
```

```
Newtype TrackLightType Array(Integer, Character)
endnewtype;
DCL TrackLight TrackLightType; /* Record each track's current light*/
```

```
DCL CarWaiting Integer:=0; /*CarWaiting(1), NoCarWaiting(0)*/
DCL NoOfTrains Integer:=0; /*In the crossing area*/
DCL NoOfSTrains Integer:=0; /* No. of trains stopped */

DCL n Integer :=0; /*counter*/
DCL m Integer :=0; /*counter*/
DCL ni Integer :=0; /*counter −−No. of InSensors already setup*/
DCL nO Integer :=0; /*counter −−No. of OutSensors already setup*/
DCL ns Integer :=0; /*counter −−No. of train signals already setup*/
DCL rs Integer :=0; /*counter −−Times of RedRQ sent, but no response*/

DCL SenderNo Integer := 0; /* Sender's track no */
```

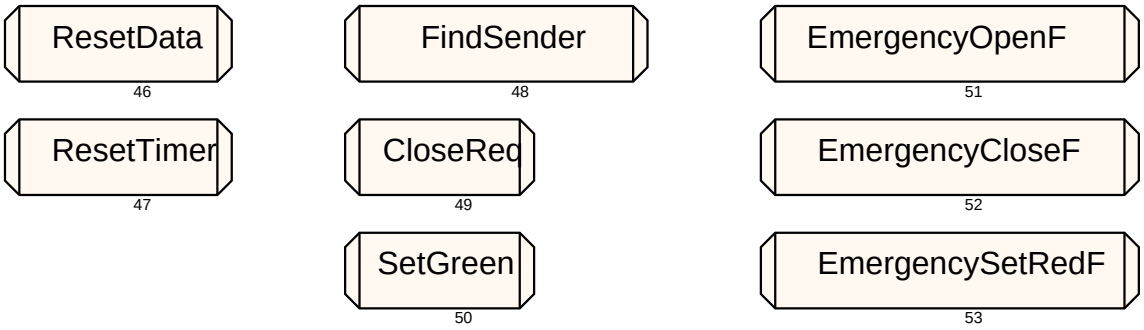
```
TIMER GateClose_TimeOut1;
TIMER GateClose_TimeOut2;
TIMER GateOpen_TimeOut1;
TIMER GateOpen_TimeOut2;
TIMER StopWaiting_TimeOut;
TIMER RedWaiting_TimeOut;
```

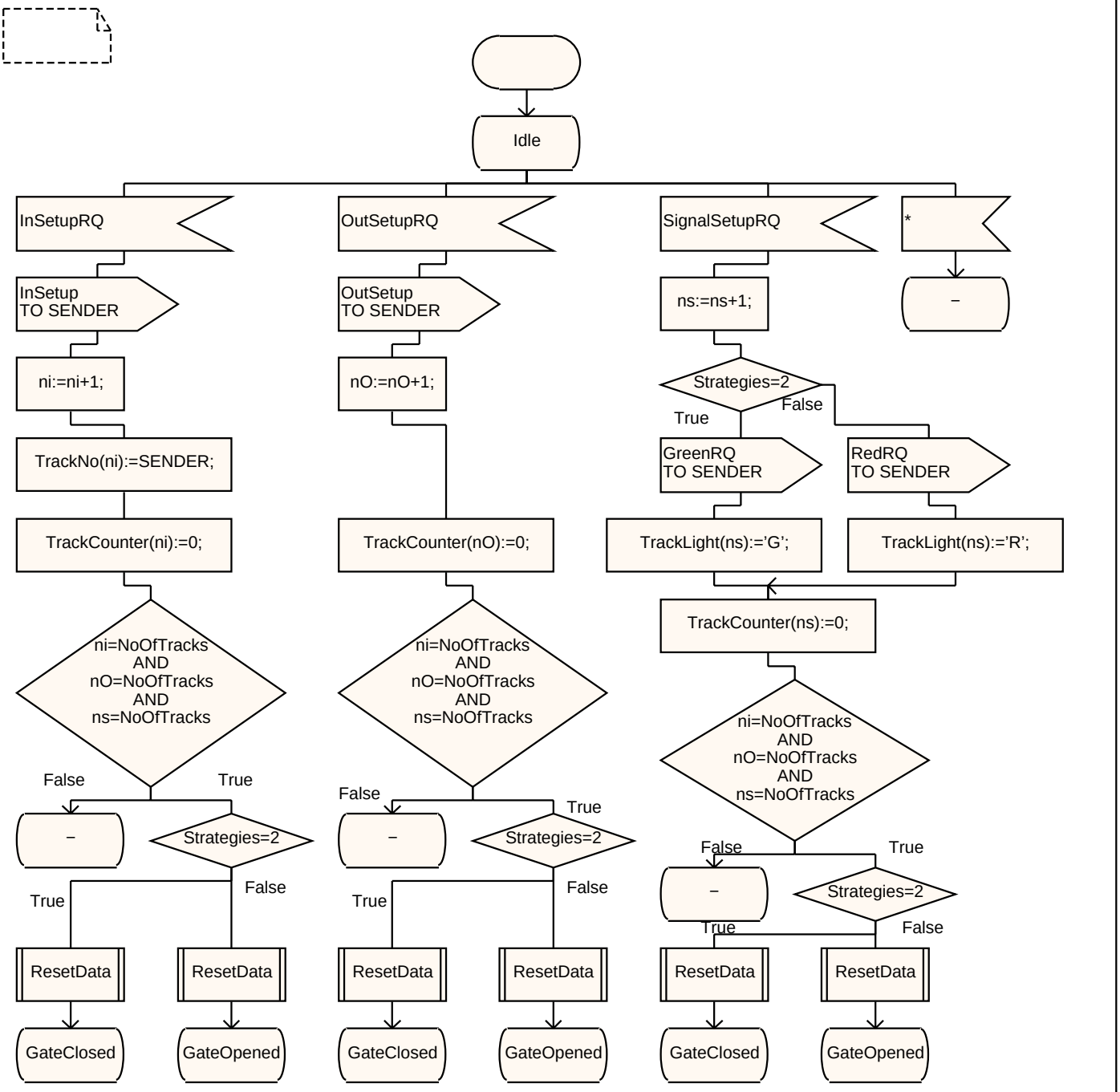
```
SYNTYPE sValue =Natural
  DEFAULT 3
  CONSTANTS 1:4
ENDSYNTYPE sValue;
```

```
DCL Strategies sValue;
DCL S Natural;
```

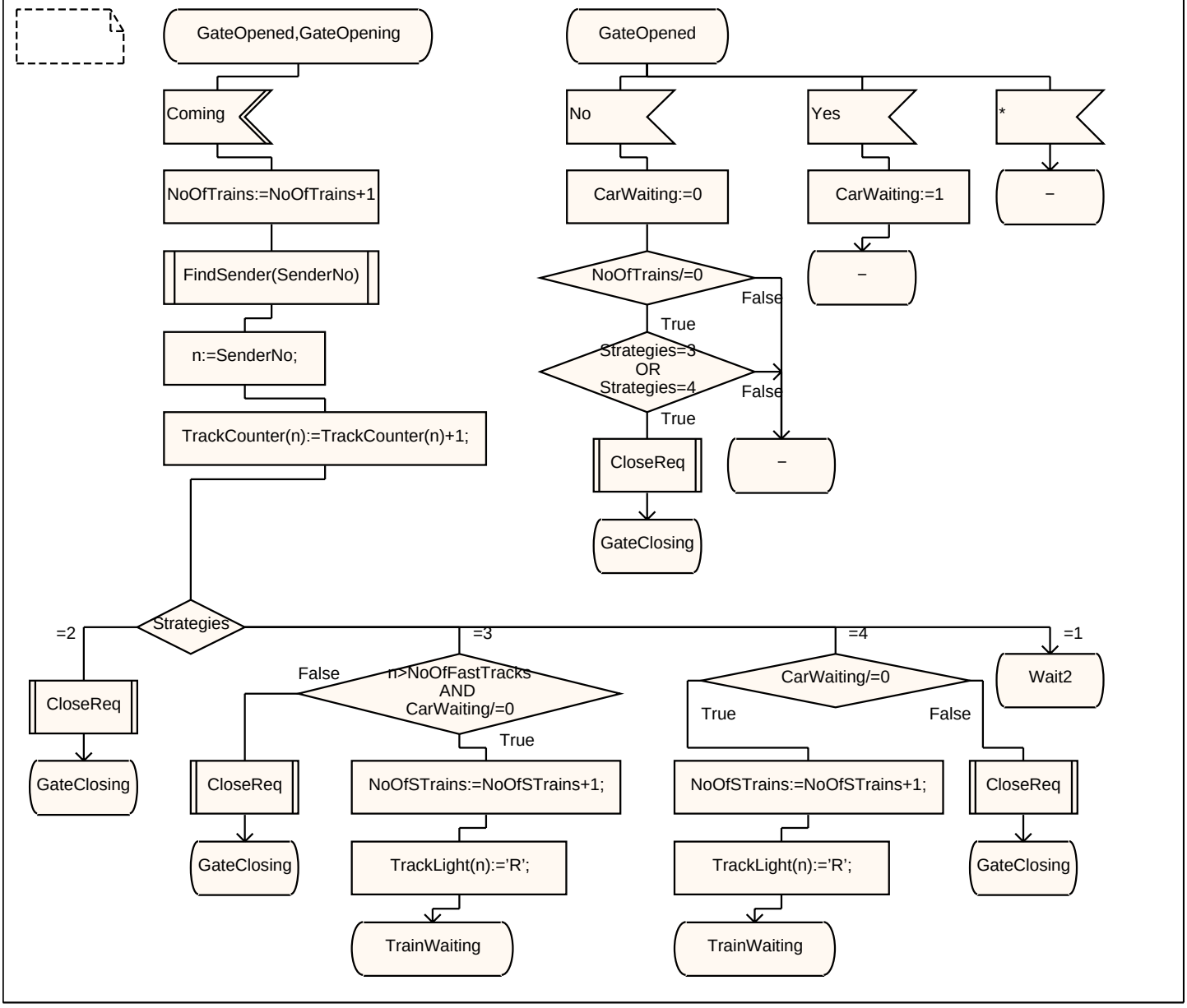
30

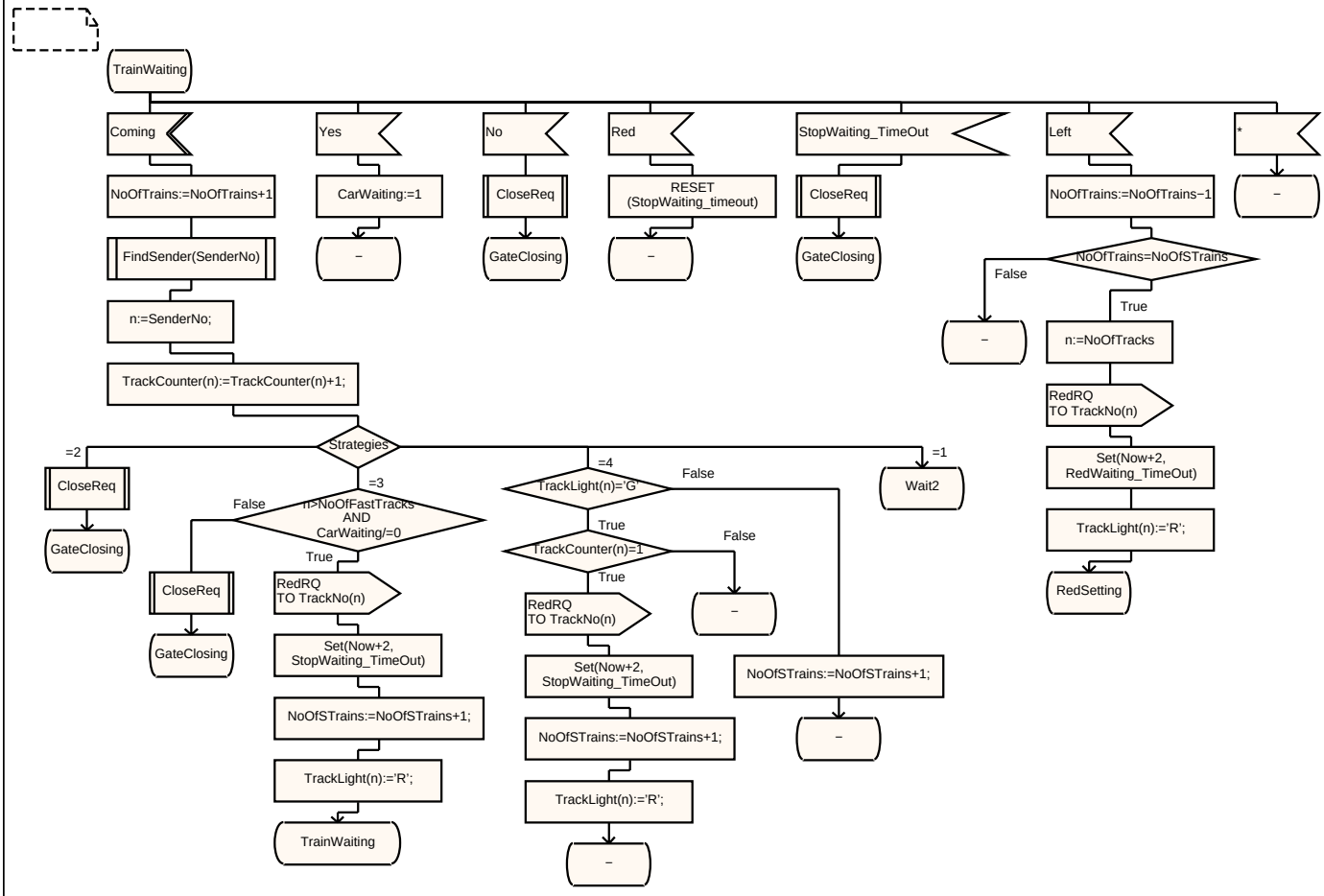
process <<block Controller>> Controller 2(15)

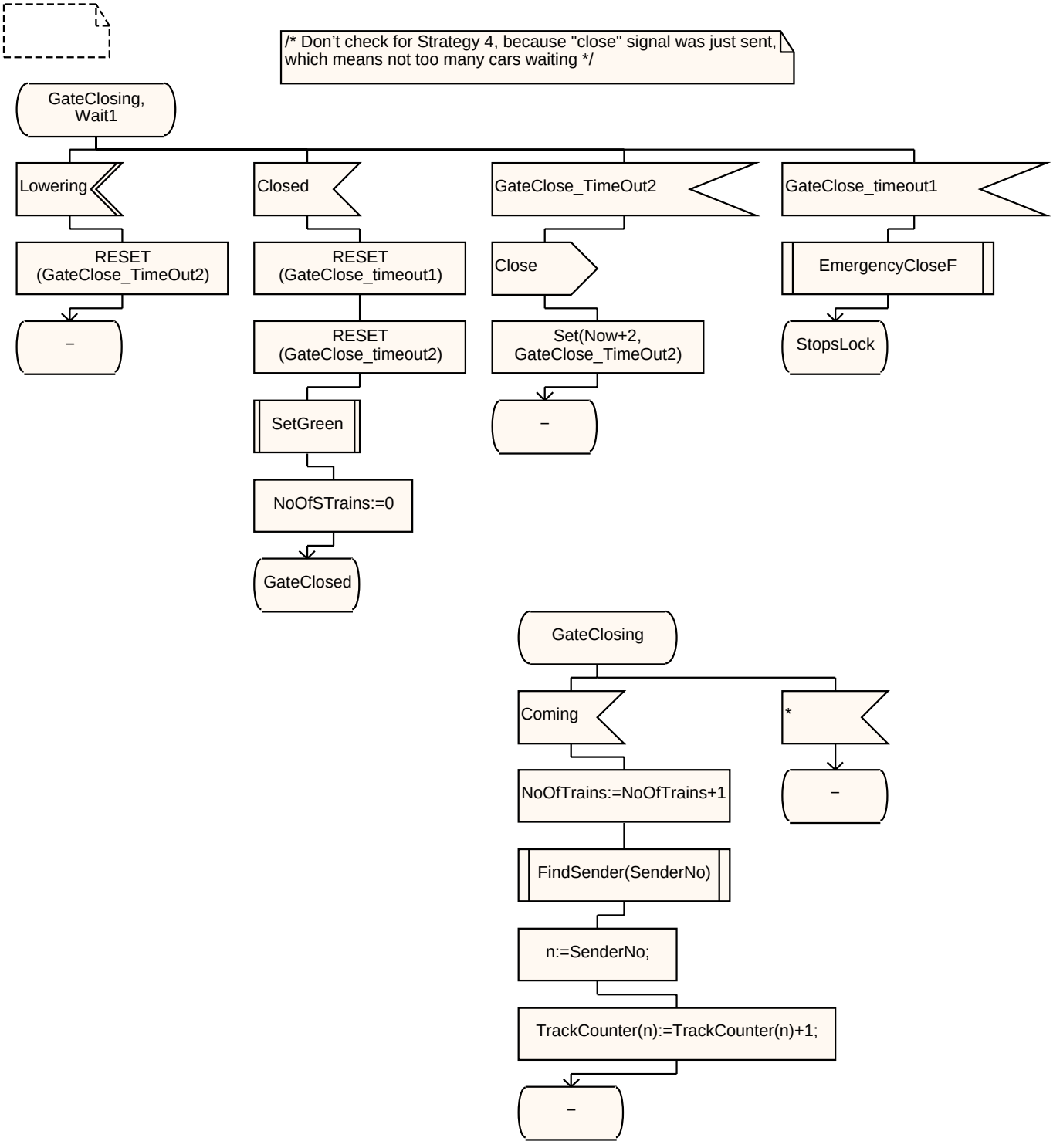



Idle
InSetupRQ
OutSetupRQ
SignalSetupRQ
*
InSetup
TO SENDER
OutSetup
TO SENDER
ns:=ns+1;
–
ni:=ni+1;
nO:=nO+1;
Strategies=2
True
False
TrackNo(ni):=SENDER;
GreenRQ
TO SENDER
RedRQ
TO SENDER
TrackCounter(ni):=0;
TrackCounter(nO):=0;
TrackLight(ns):='G';
TrackLight(ns):='R';
TrackCounter(ns):=0;
ni=NoOfTracks
AND
nO=NoOfTracks
AND
ns=NoOfTracks
ni=NoOfTracks
AND
nO=NoOfTracks
AND
ns=NoOfTracks
ni=NoOfTracks
AND
nO=NoOfTracks
AND
ns=NoOfTracks
False
True
–
Strategies=2
True
False
ResetData
ResetData
ResetData
ResetData
ResetData
ResetData
GateClosed
GateOpened
GateClosed
GateOpened
GateClosed
GateOpened

process <<block Controller>> Controller

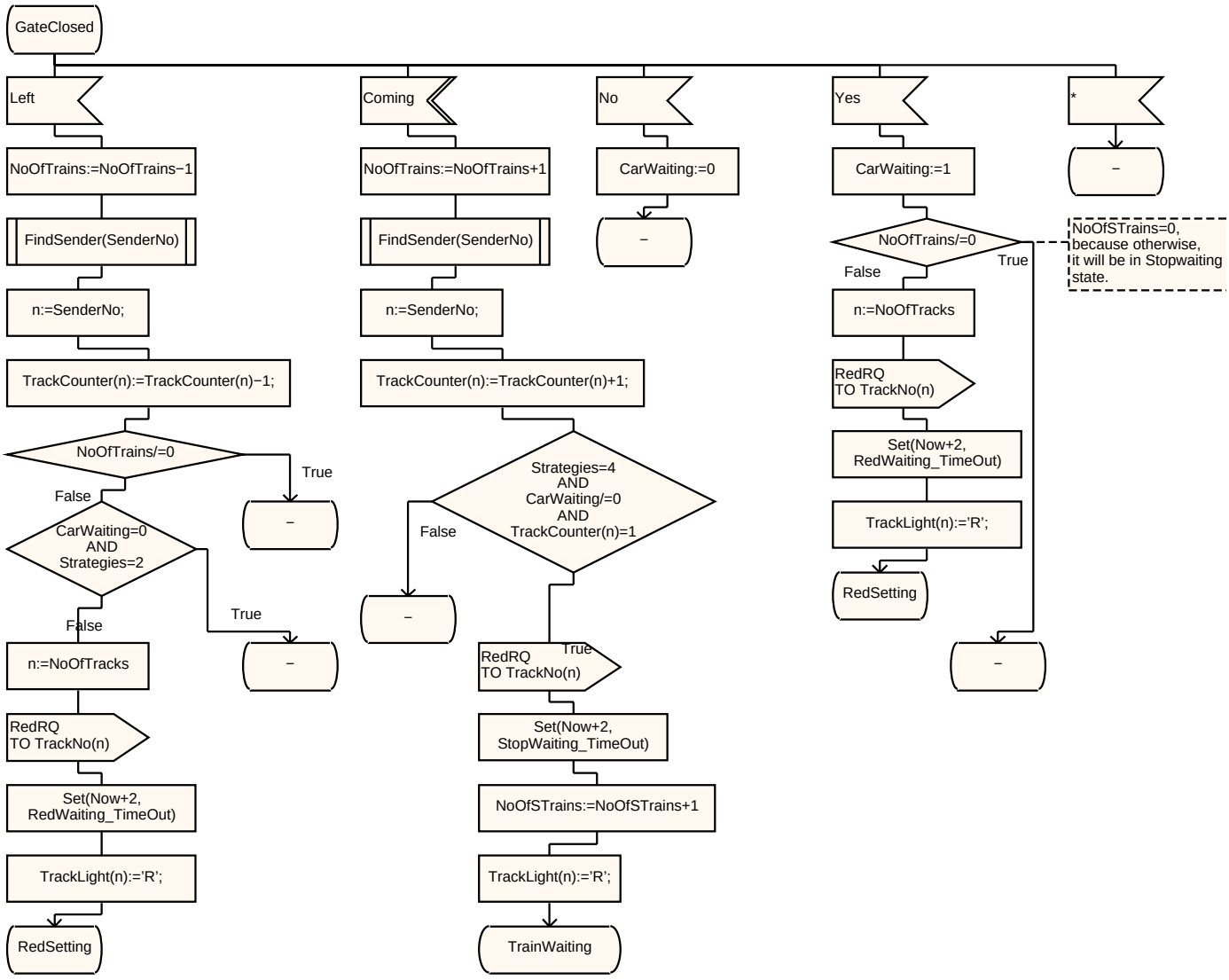


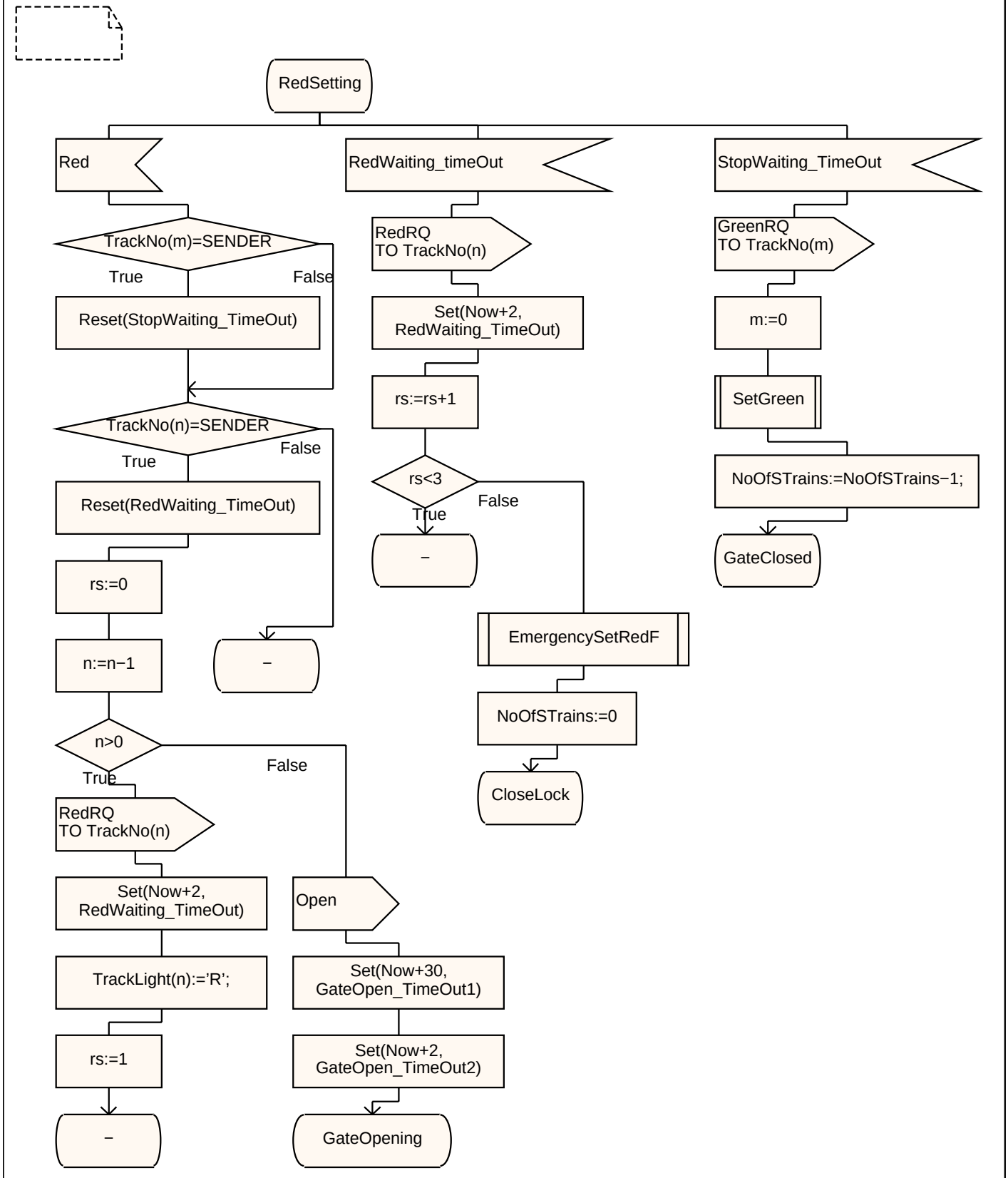

TrainWaiting
Coming
NoOfTrains:=NoOfTrains+1
FindSender(SenderNo)
n:=SenderNo;
TrackCounter(n):=TrackCounter(n)+1;
Strategies
=2
CloseReq
GateClosing
=3
n>NoOfFastTracks AND CarWaiting/=0
False
True
CloseReq
GateClosing
RedRQ TO TrackNo(n)
Set(Now+2, StopWaiting_TimeOut)
NoOfSTrains:=NoOfSTrains+1;
TrackLight(n):='R';
TrainWaiting
=4
TrackLight(n)='G'
False
True
TrackCounter(n)=1
False
True
RedRQ TO TrackNo(n)
Set(Now+2, StopWaiting_TimeOut)
NoOfSTrains:=NoOfSTrains+1;
TrackLight(n):='R';
–
NoOfSTrains:=NoOfSTrains+1;
–
=1
Wait2
Yes
CarWaiting:=1
–
No
CloseReq
GateClosing
Red
RESET (StopWaiting_timeout)
–
StopWaiting_TimeOut
CloseReq
GateClosing
Left
NoOfTrains:=NoOfTrains−1
NoOfTrains=NoOfSTrains
False
True
–
n:=NoOfTracks
RedRQ TO TrackNo(n)
Set(Now+2, RedWaiting_TimeOut)
TrackLight(n):='R';
RedSetting
*
–

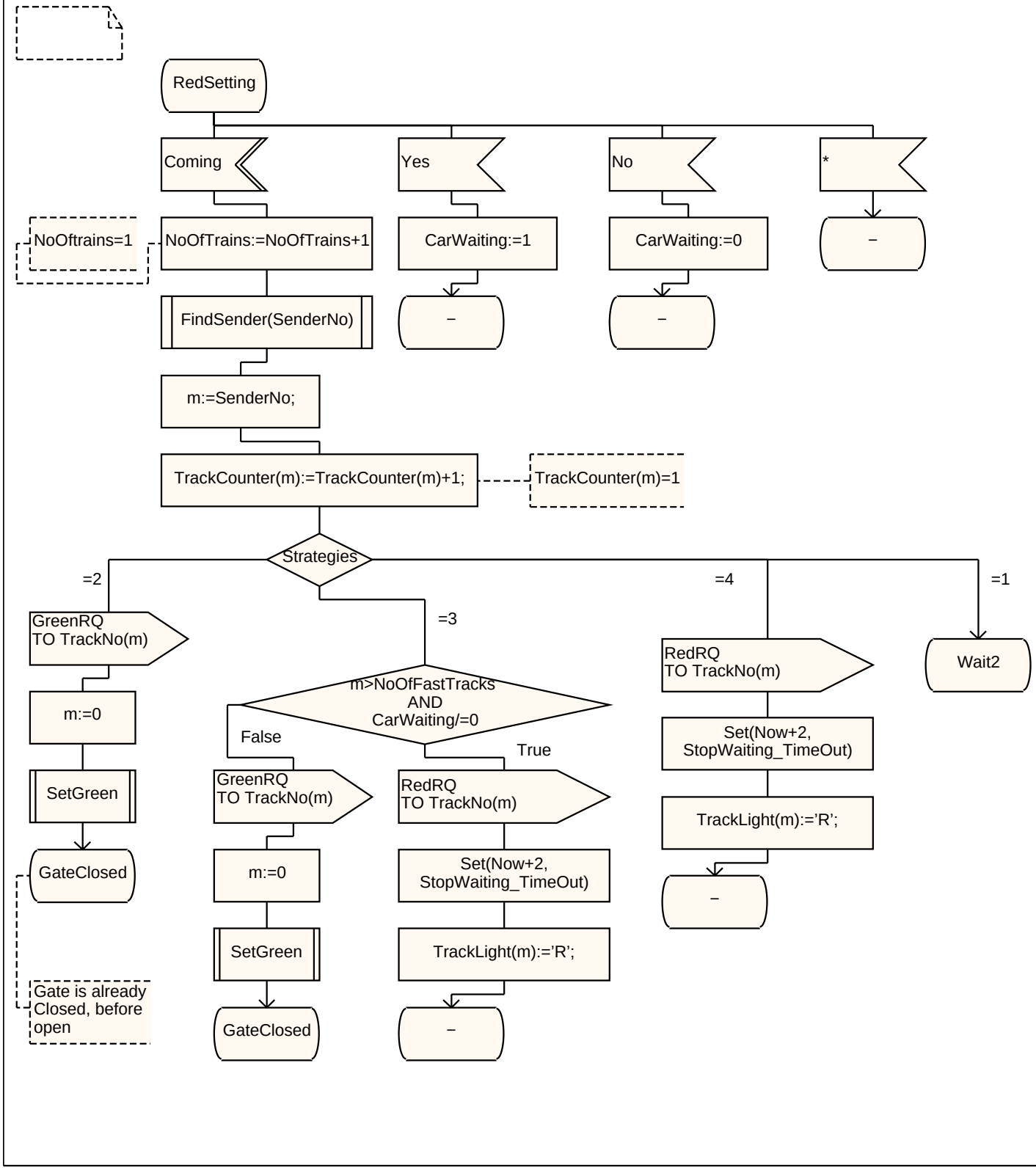

/* Don't check for Strategy 4, because "close" signal was just sent, which means not too many cars waiting */
GateClosing, Wait1
Lowering
Closed
GateClose_TimeOut2
GateClose_timeout1
RESET (GateClose_TimeOut2)
RESET (GateClose_timeout1)
Close
EmergencyCloseF
–
RESET (GateClose_timeout2)
Set(Now+2, GateClose_TimeOut2)
StopsLock
SetGreen
–
NoOfSTrains:=0
GateClosed
GateClosing
Coming
*
NoOfTrains:=NoOfTrains+1
–
FindSender(SenderNo)
n:=SenderNo;
TrackCounter(n):=TrackCounter(n)+1;
–

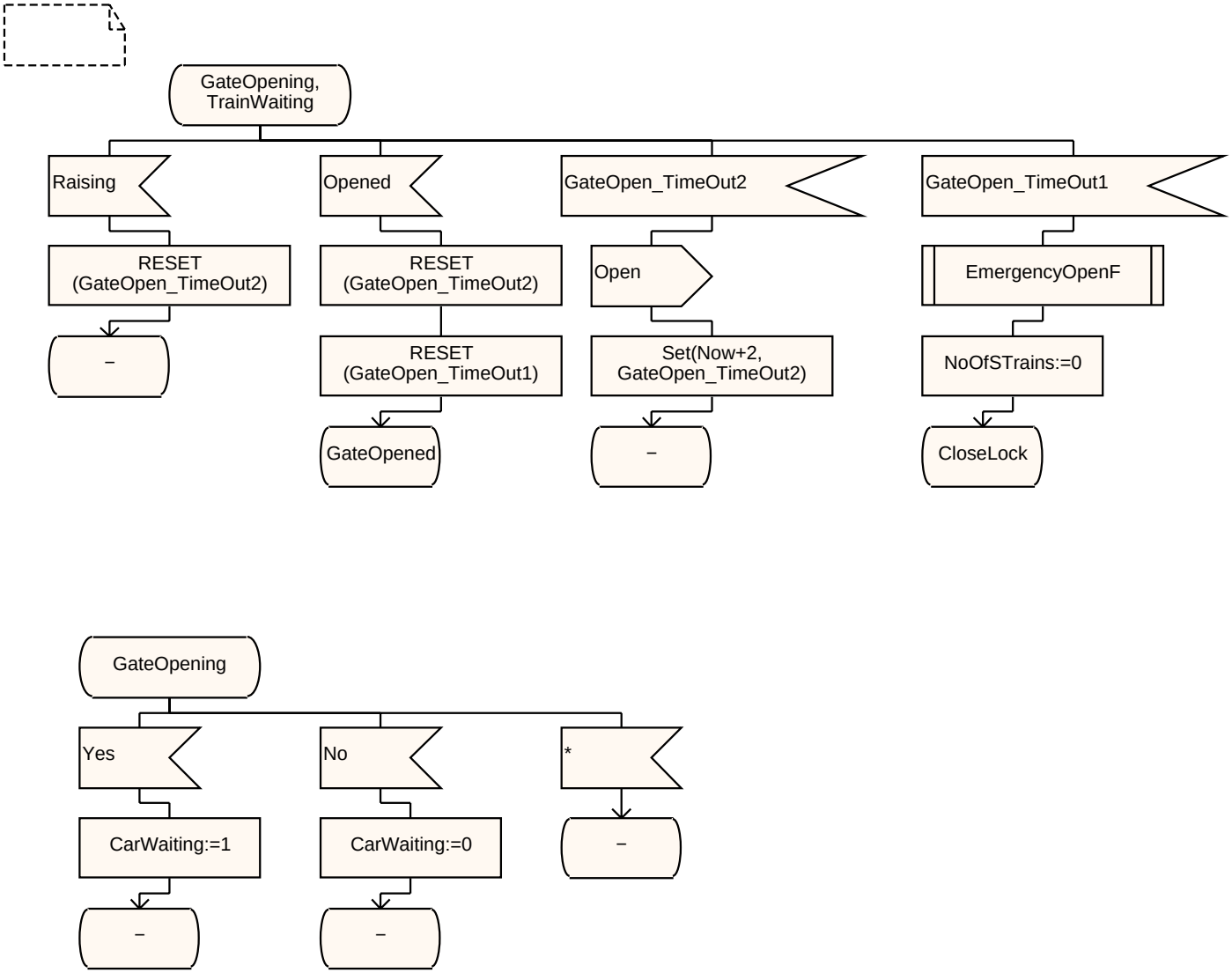
/* Strategy 4: if carWaiting, and no train running on the same track, stop the train, and try to open the gate
Other than this,
Always let train go, until no train in gate, then open the gate */

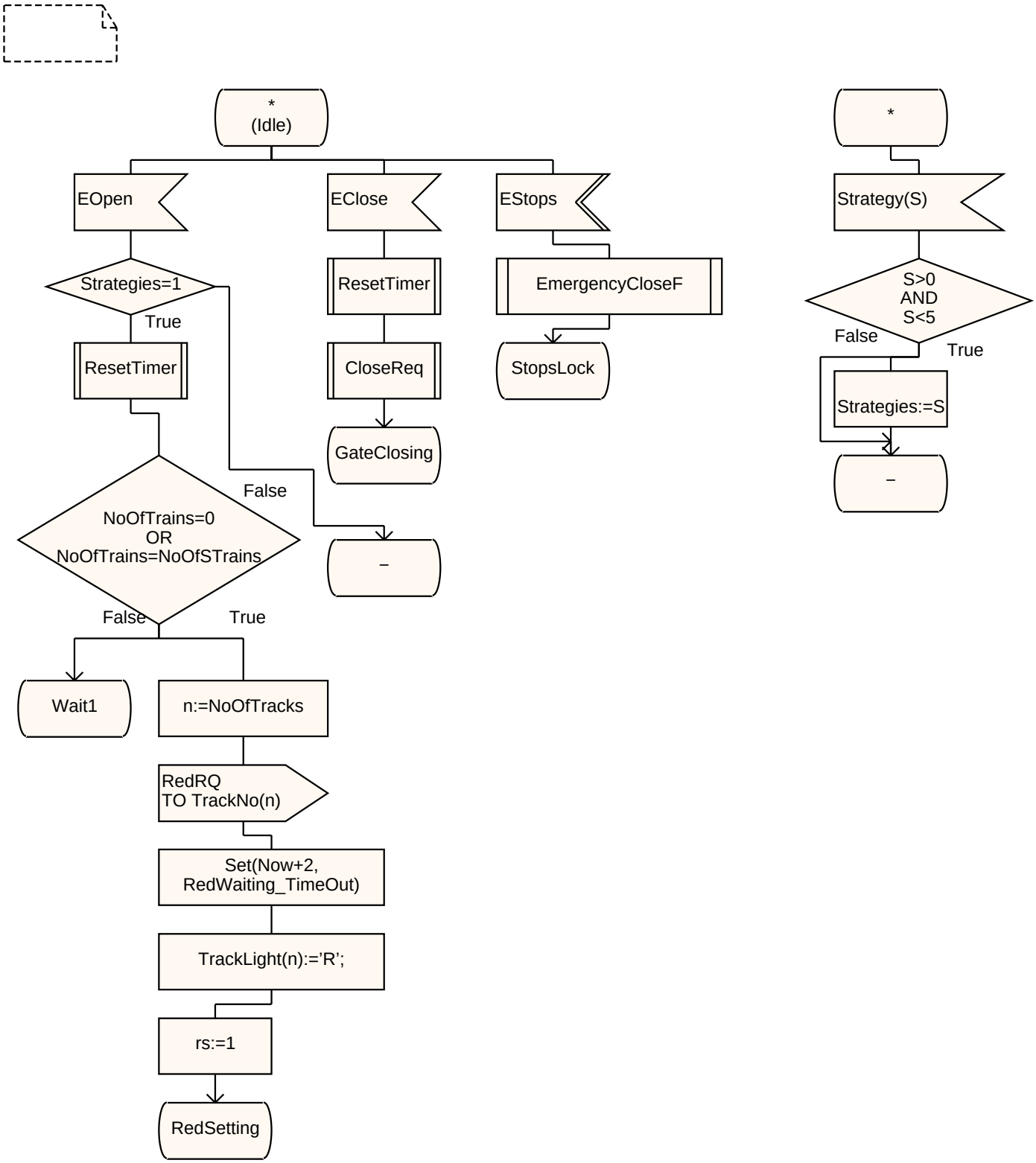
/* RedSetting−−Send "red" signal one by one, before send"Open" signal */

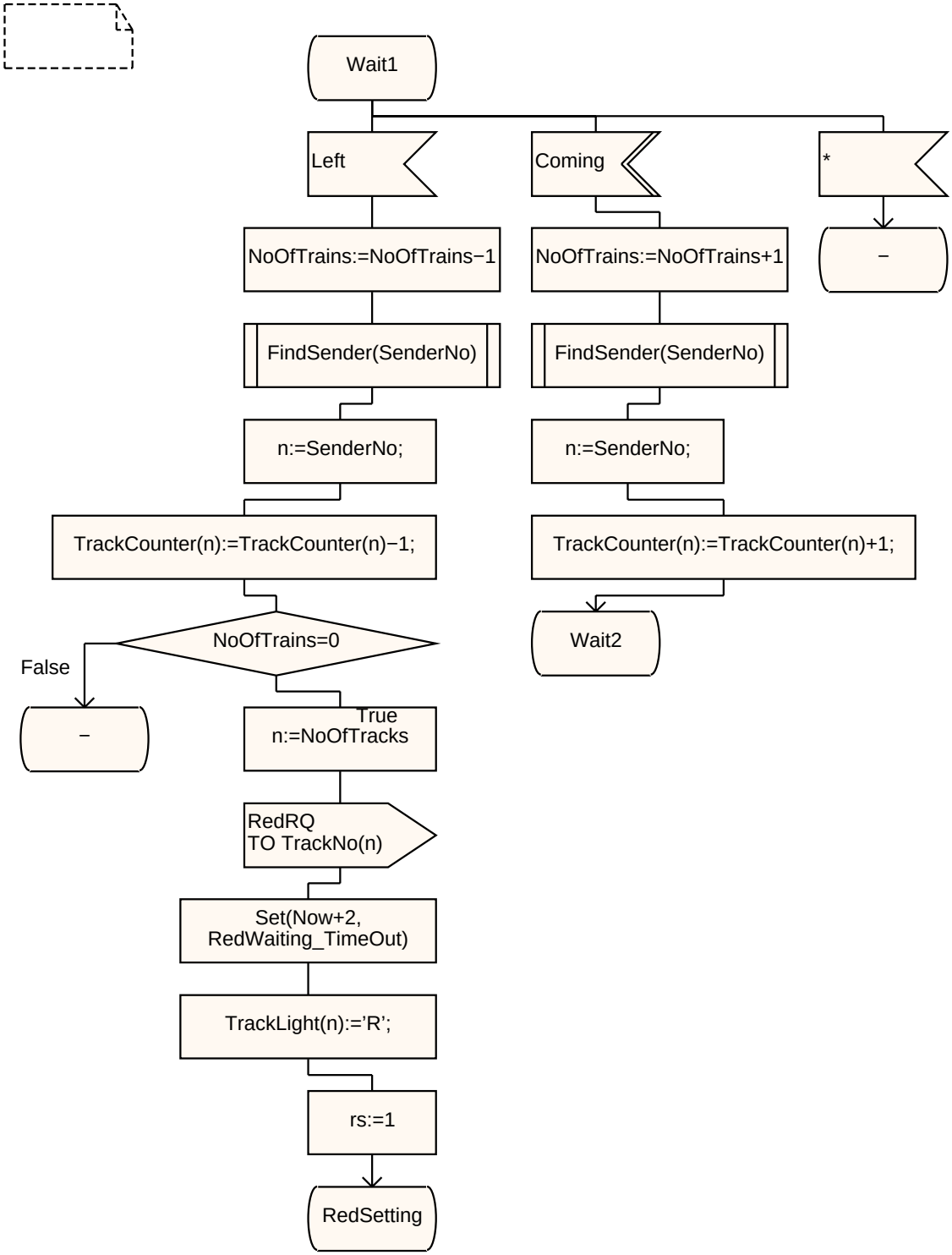


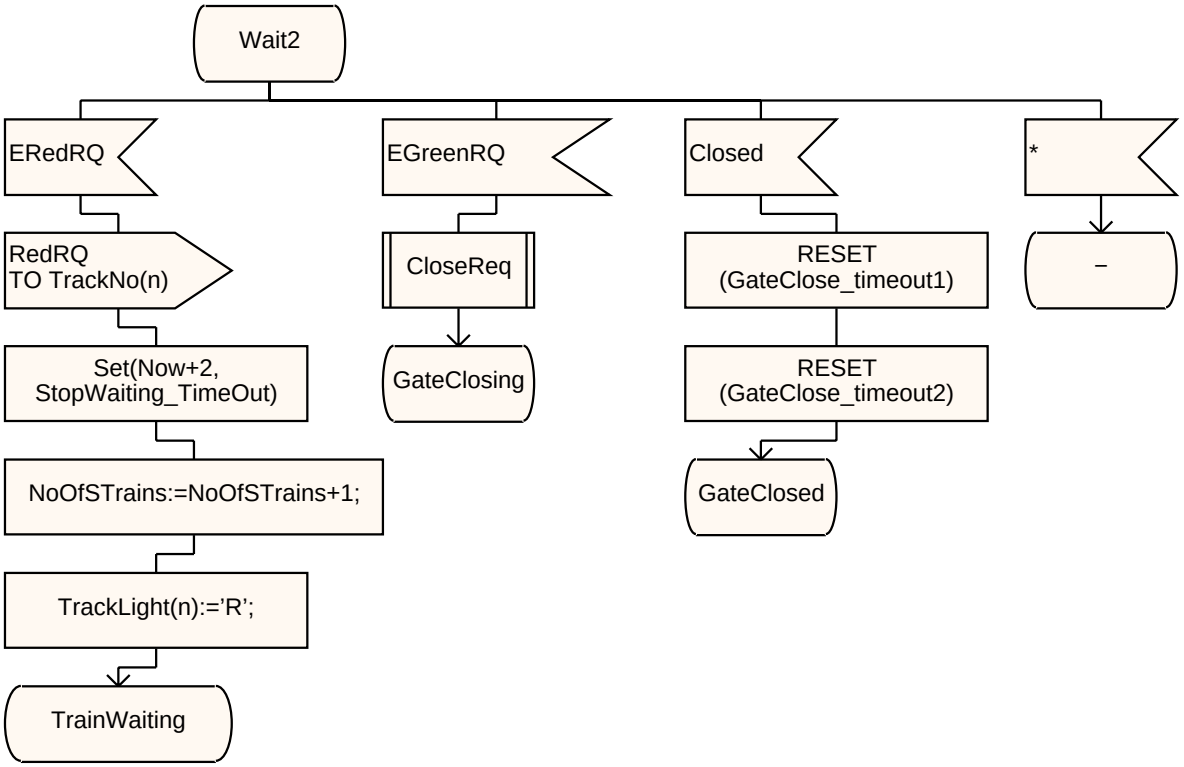

RedSetting
Red
RedWaiting_timeOut
StopWaiting_TimeOut
TrackNo(m)=SENDER
True
False
Reset(StopWaiting_TimeOut)
TrackNo(n)=SENDER
True
False
Reset(RedWaiting_TimeOut)
rs:=0
n:=n−1
n>0
True
False
RedRQ
TO TrackNo(n)
Set(Now+2,
RedWaiting_TimeOut)
TrackLight(n):='R';
rs:=1
−
−
RedRQ
TO TrackNo(n)
Set(Now+2,
RedWaiting_TimeOut)
rs:=rs+1
rs<3
True
False
−
EmergencySetRedF
NoOfSTrains:=0
CloseLock
Open
Set(Now+30,
GateOpen_TimeOut1)
Set(Now+2,
GateOpen_TimeOut2)
GateOpening
GreenRQ
TO TrackNo(m)
m:=0
SetGreen
NoOfSTrains:=NoOfSTrains−1;
GateClosed

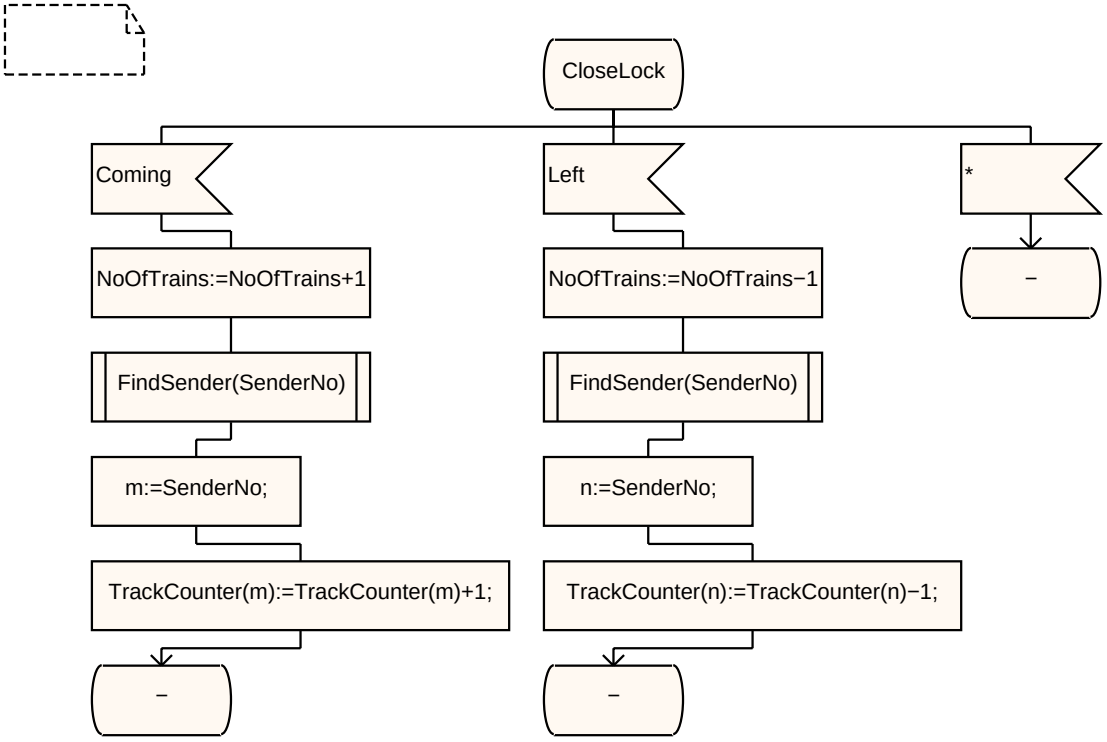

RedSetting
Coming
Yes
No
*
NoOftrains=1
NoOfTrains:=NoOfTrains+1
CarWaiting:=1
CarWaiting:=0
–
FindSender(SenderNo)
–
–
m:=SenderNo;
TrackCounter(m):=TrackCounter(m)+1;
TrackCounter(m)=1
Strategies
=2
=3
=4
=1
GreenRQ
TO TrackNo(m)
m:=0
SetGreen
GateClosed
Gate is already
Closed, before
open
m>NoOfFastTracks
AND
CarWaiting/=0
False
True
GreenRQ
TO TrackNo(m)
m:=0
SetGreen
GateClosed
RedRQ
TO TrackNo(m)
Set(Now+2,
StopWaiting_TimeOut)
TrackLight(m):='R';
–
RedRQ
TO TrackNo(m)
Set(Now+2,
StopWaiting_TimeOut)
TrackLight(m):='R';
–
Wait2

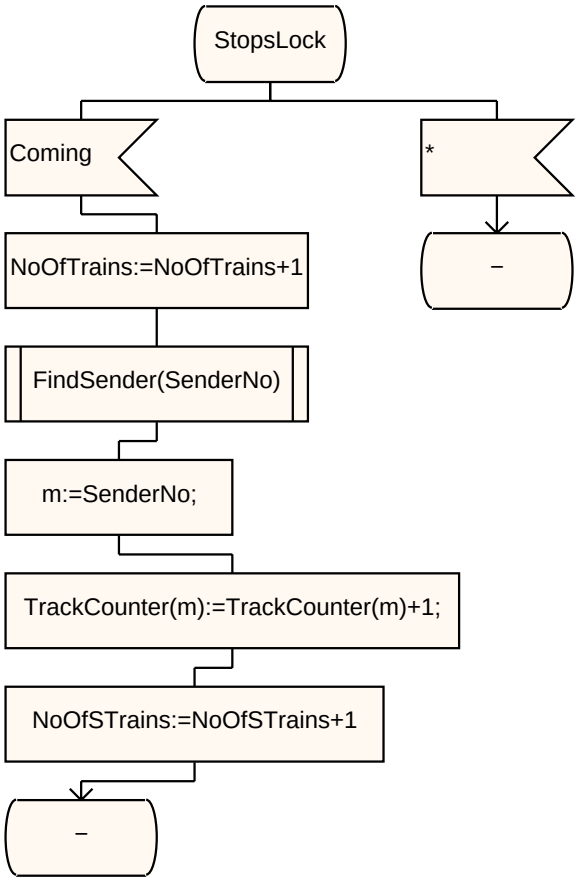

GateOpening, TrainWaiting
Raising
RESET (GateOpen_TimeOut2)
–
Opened
RESET (GateOpen_TimeOut2)
RESET (GateOpen_TimeOut1)
GateOpened
GateOpen_TimeOut2
Open
Set(Now+2, GateOpen_TimeOut2)
–
GateOpen_TimeOut1
EmergencyOpenF
NoOfSTrains:=0
CloseLock
GateOpening
Yes
CarWaiting:=1
–
No
CarWaiting:=0
–
*
–


*
(Idle)
EOpen
EClose
EStops
Strategies=1
True
ResetTimer
ResetTimer
EmergencyCloseF
CloseReq
StopsLock
GateClosing
False
NoOfTrains=0
OR
NoOfTrains=NoOfSTrains
–
False
True
Wait1
n:=NoOfTracks
RedRQ
TO TrackNo(n)
Set(Now+2,
RedWaiting_TimeOut)
TrackLight(n):='R';
rs:=1
RedSetting
*
Strategy(S)
S>0
AND
S<5
False
True
Strategies:=S
–

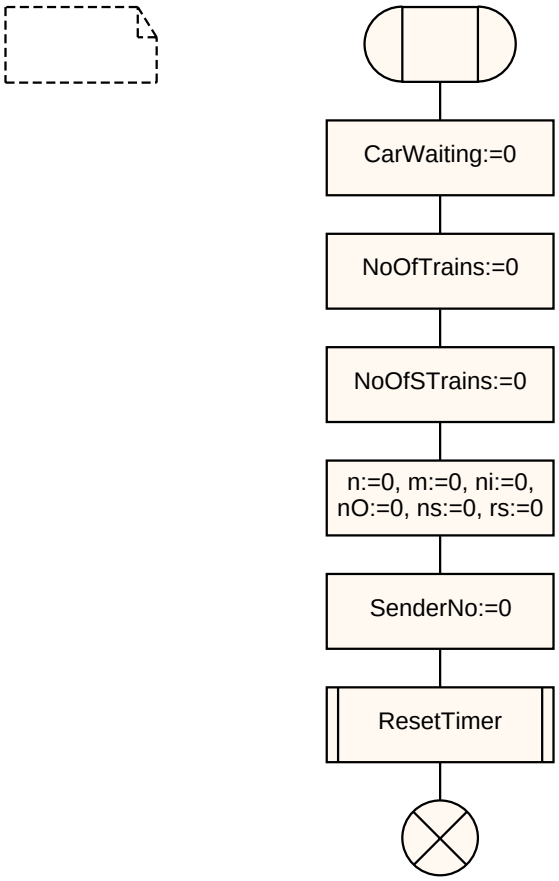

Wait1
Left
Coming
*
NoOfTrains:=NoOfTrains−1
NoOfTrains:=NoOfTrains+1
−
FindSender(SenderNo)
FindSender(SenderNo)
n:=SenderNo;
n:=SenderNo;
TrackCounter(n):=TrackCounter(n)−1;
TrackCounter(n):=TrackCounter(n)+1;
NoOfTrains=0
Wait2
False
True
−
n:=NoOfTracks
RedRQ
TO TrackNo(n)
Set(Now+2,
RedWaiting_TimeOut)
TrackLight(n):='R';
rs:=1
RedSetting


Wait2
ERedRQ
EGreenRQ
Closed
*
RedRQ
TO TrackNo(n)
CloseReq
RESET
(GateClose_timeout1)
–
Set(Now+2,
StopWaiting_TimeOut)
GateClosing
RESET
(GateClose_timeout2)
NoOfSTrains:=NoOfSTrains+1;
GateClosed
TrackLight(n):='R';
TrainWaiting

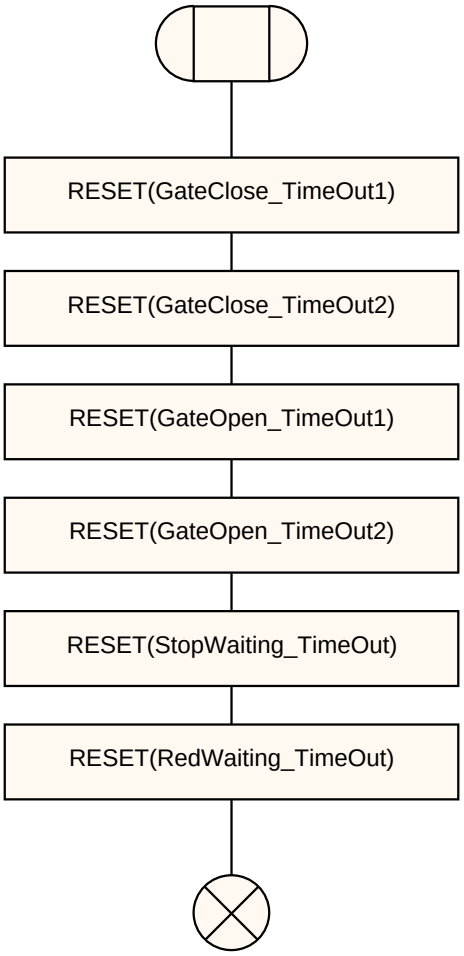

CloseLock
Coming
Left
*
NoOfTrains:=NoOfTrains+1
NoOfTrains:=NoOfTrains−1
−
FindSender(SenderNo)
FindSender(SenderNo)
m:=SenderNo;
n:=SenderNo;
TrackCounter(m):=TrackCounter(m)+1;
TrackCounter(n):=TrackCounter(n)−1;
−
−


StopsLock
Coming
*
NoOfTrains:=NoOfTrains+1
–
FindSender(SenderNo)
m:=SenderNo;
TrackCounter(m):=TrackCounter(m)+1;
NoOfSTrains:=NoOfSTrains+1
–

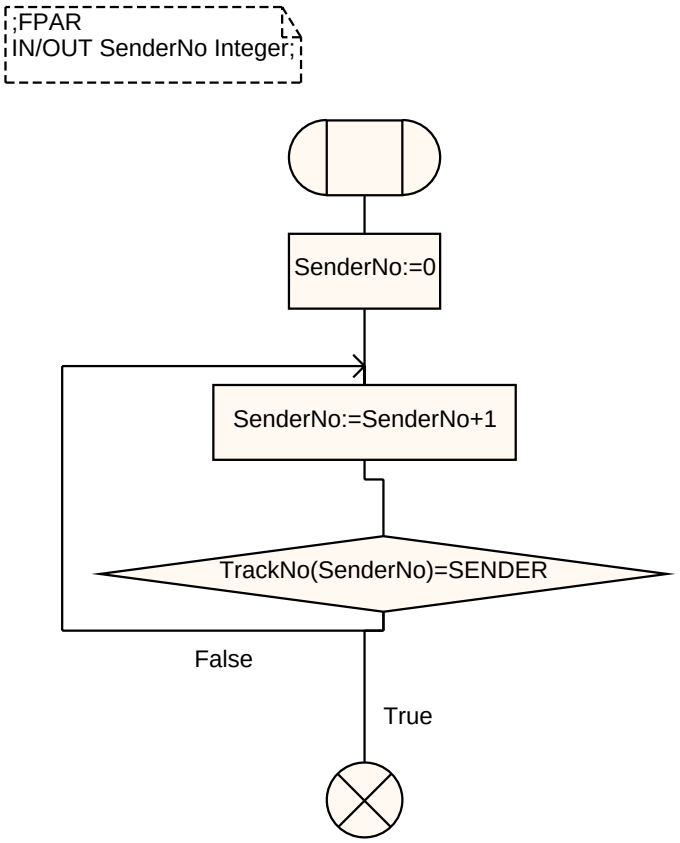
procedure ResetData



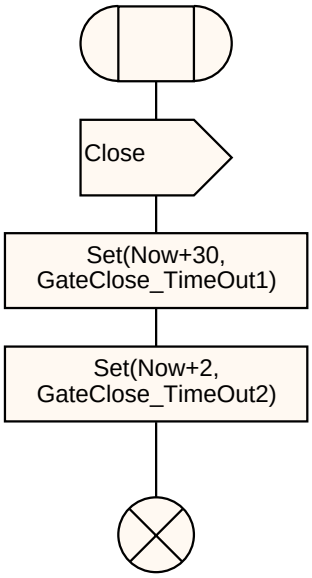
procedure ResetTimer



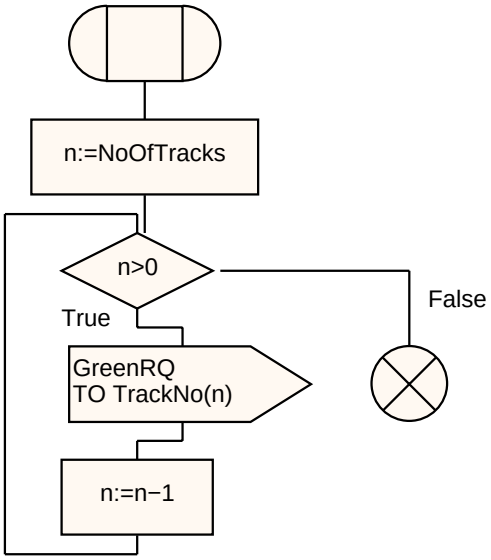
procedure FindSender



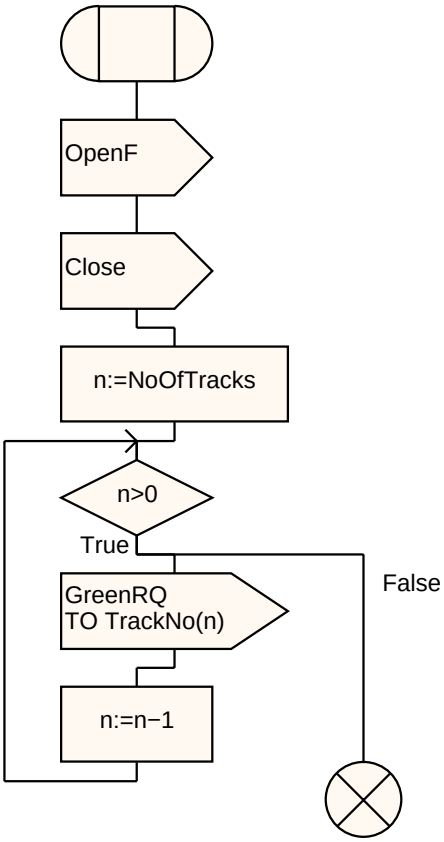
procedure CloseReq



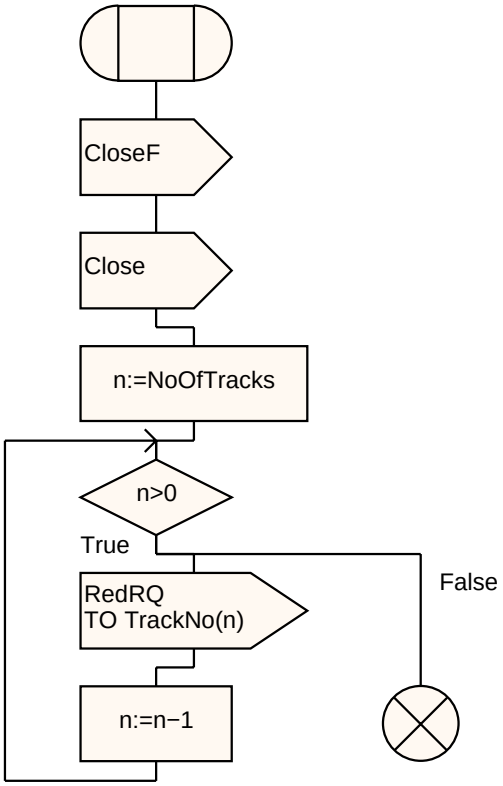
procedure SetGreen



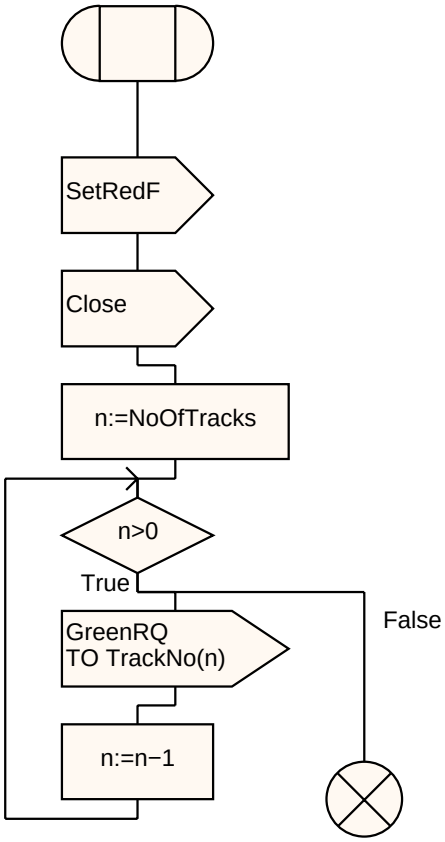
procedure EmergencyOpenF

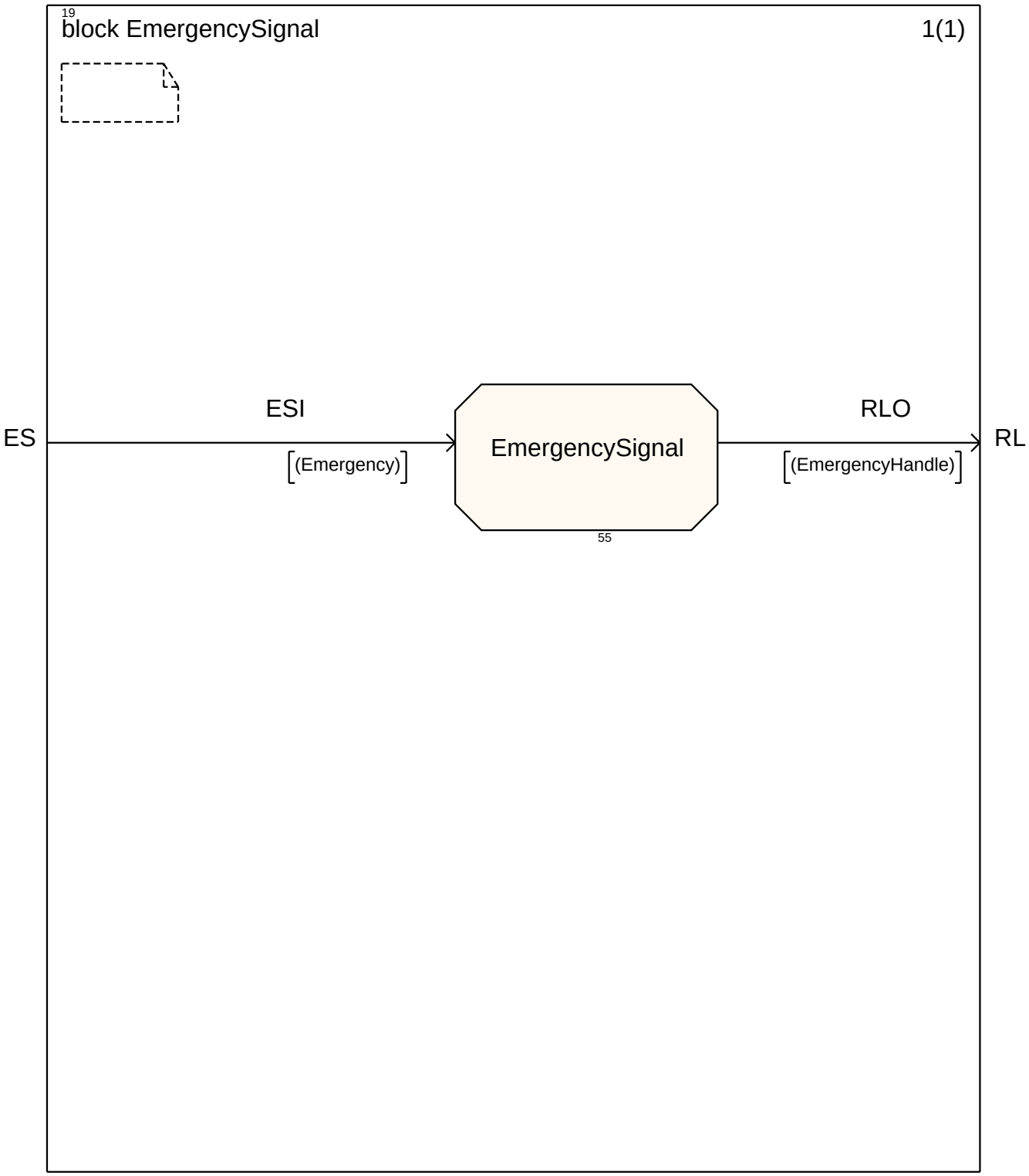


procedure EmergencyCloseF

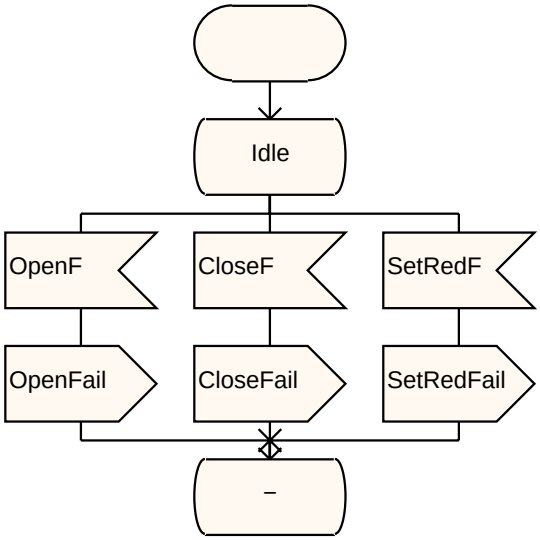


procedure EmergencySetRedF




19
block EmergencySignal
1(1)
ES
ESI
(Emergency)
EmergencySignal
55
RLO
(EmergencyHandle)
RL

process EmergencySignal



process ModifiedCarSensor 1(1)