به نام خدا

گزارش پروژه 1 درس فرمال
راضیه دوره گرد - 819199419

Processes:

I assumed that the below processes can meet the requirements in this phase:

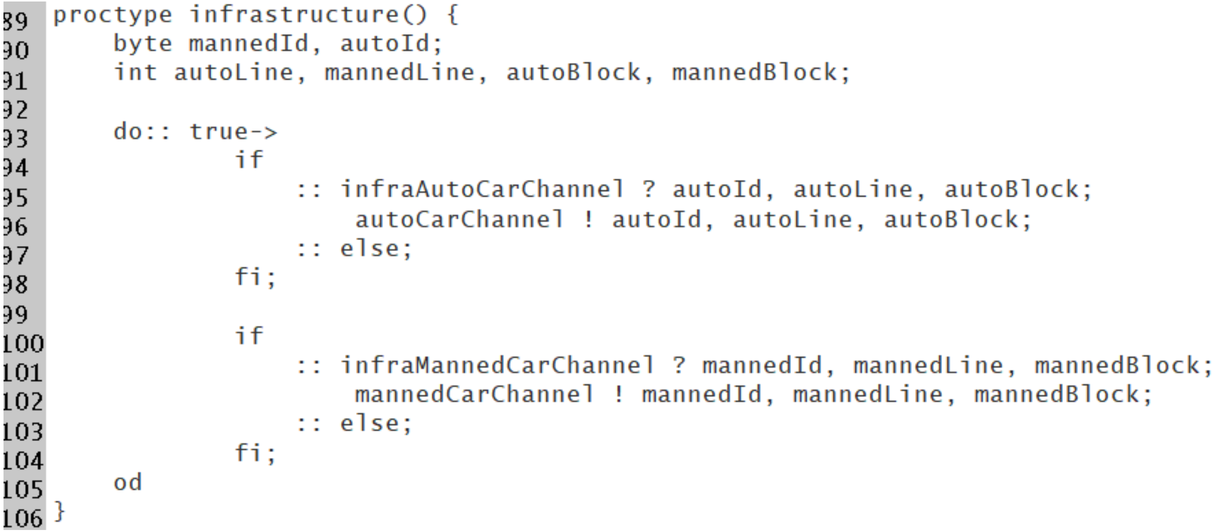
1- Autonomous car

2- Manned car

3-Infrastructure

4- Controller

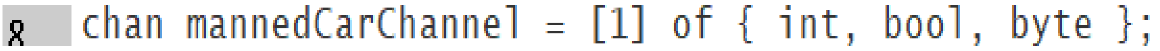
** The Infrastructure works like a proxy between the cars and the controller process and passes the id and location of each car to the controller:



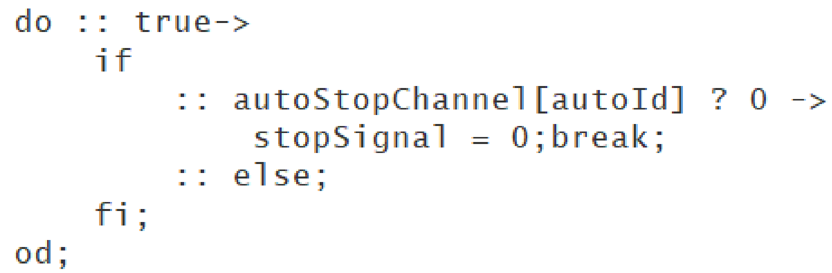
```
proctype infrastructure() {
    byte mannedId, autoId;
    int autoLine, mannedLine, autoBlock, mannedBlock;

    do:: true->
            if
                :: infraAutoCarChannel ? autoId, autoLine, autoBlock;
                    autoCarChannel ! autoId, autoLine, autoBlock;
                :: else;
            fi;

            if
                :: infraMannedCarChannel ? mannedId, mannedLine, mannedBlock;
                    mannedCarChannel ! mannedId, mannedLine, mannedBlock;
                :: else;
            fi;
    od
}
```

** Stop mechanism: each autonomous car waits for a stop channel to become true and stops in that case, otherwise, is allowed to keep moving:

** There is a channel for each channel and they are distinguished by their ids

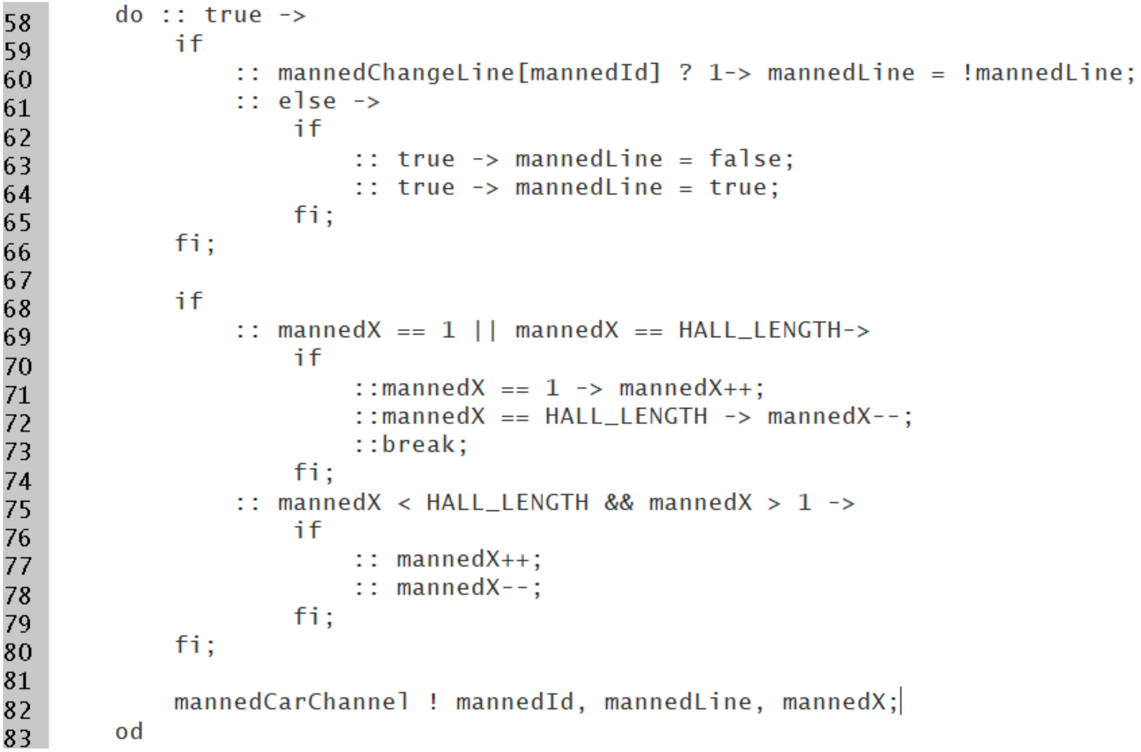


```
chan mannedCarChannel = [1] of { int, bool, byte };
```



```
do :: true->
     if
         :: autoStopChannel[autoId] ? 0 ->
             stopSignal = 0;break;
         :: else;
     fi;
od;
```

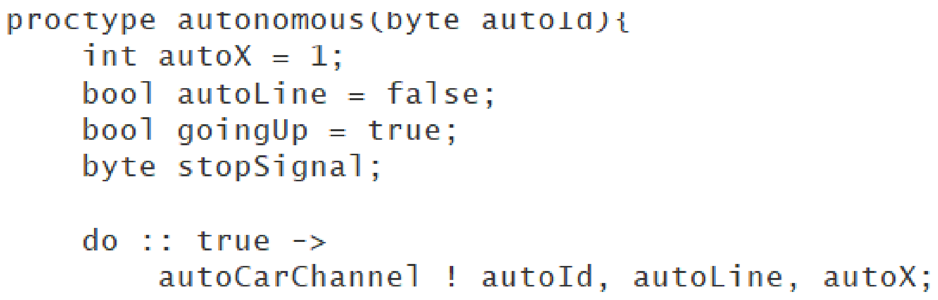
** The nondeterminism in a manned car’s movements has modeled as below:



```
do :: true ->
    if
        :: mannedChangeLine[mannedId] ? 1-> mannedLine = !mannedLine;
        :: else ->
            if
                :: true -> mannedLine = false;
                :: true -> mannedLine = true;
            fi;
    fi;

    if
        :: mannedX == 1 || mannedX == HALL_LENGTH->
            if
                ::mannedX == 1 -> mannedX++;
                ::mannedX == HALL_LENGTH -> mannedX--;
                ::break;
            fi;
        :: mannedX < HALL_LENGTH && mannedX > 1 ->
            if
                :: mannedX++;
                :: mannedX--;
            fi;
    fi;

    mannedCarChannel ! mannedId, mannedLine, mannedX;
od
```

** Broadcasting the location and IDs: on each movement of each type of car, the block number, the line (which is a boolean for convenience because there are only 2 lines) and the carID is sent to either channel of autoCarChannel or mannedCarChannel.

** Both autoCarChannel and mannedCarChannel have a capacity of 1, why?
Autonomous cars are highly critical due to the fact that humans’ lives are related to their criticality, thus this kind of system should be HARD-REAL-TIME to make sure they meet their deadlines and nobody gets hurt. One approach to design these systems is to make them SYNC to enforce some determinism to the behavior, however, based on Promela’s documentation, there are not mechanisms to model synchronized systems like clocks, etc.
Considering the points above, I decided to make the channels’ capacity 1 to make sure while assessing 2 cars’ locations, no accident occurs between the rest of the cars. This is because they get blocked as they want to send their location to the channel of 1 capacity.

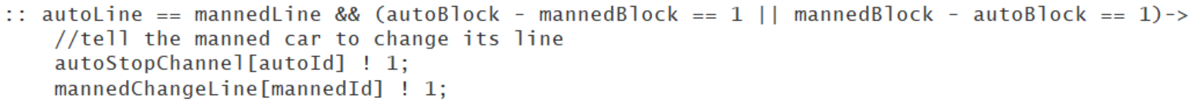


```
proctype autonomous(byte autoId){
    int autoX = 1;
    bool autoLine = false;
    bool goingUp = true;
    byte stopSignal;

    do :: true ->
        autoCarChannel ! autoId, autoLine, autoX;
```

** Change line mechanism:

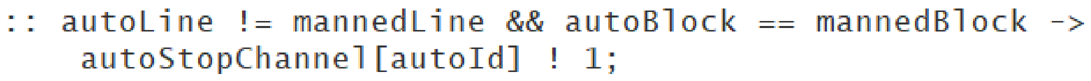
If an autonomous and a manned car are close, there are 2 cases:

1- they are on the same line ⇒ the manned car should change its line:



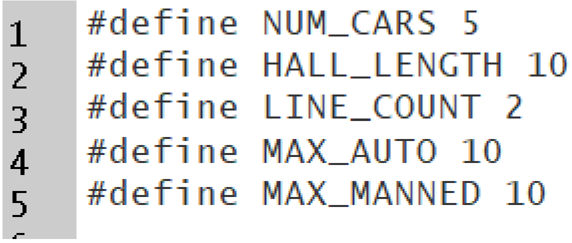
```
:: autoLine == mannedLine && (autoBlock - mannedBlock == 1 || mannedBlock - autoBlock == 1)->
    //tell the manned car to change its line
    autoStopChannel[autoId] ! 1;
    mannedChangeLine[mannedId] ! 1;
```

2- they are on different lines ⇒ only stop signal will be issued:



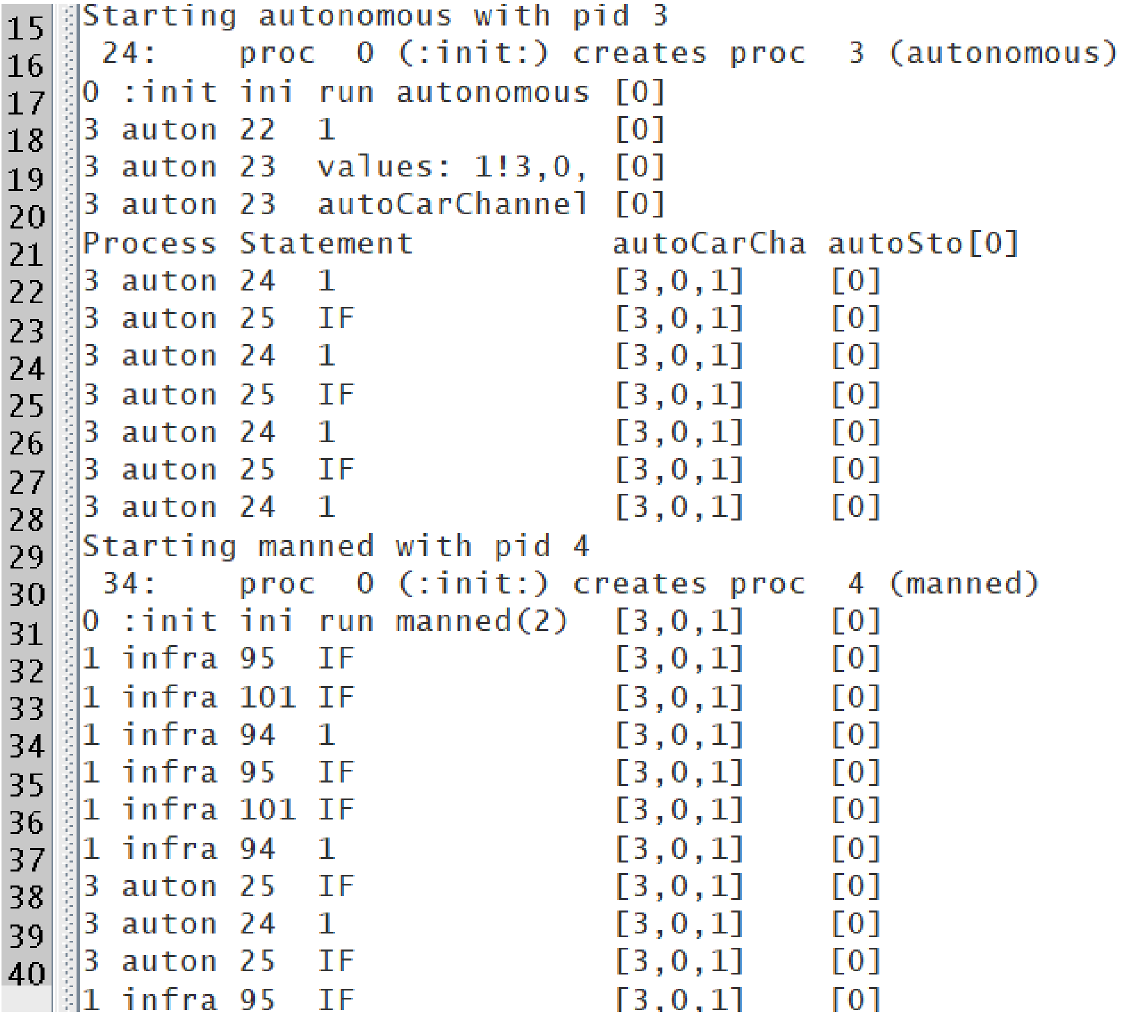
```
:: autoLine != mannedLine && autoBlock == mannedBlock ->
    autoStopChannel[autoId] ! 1;
```

** For the sake of convenience, some constants were defined:

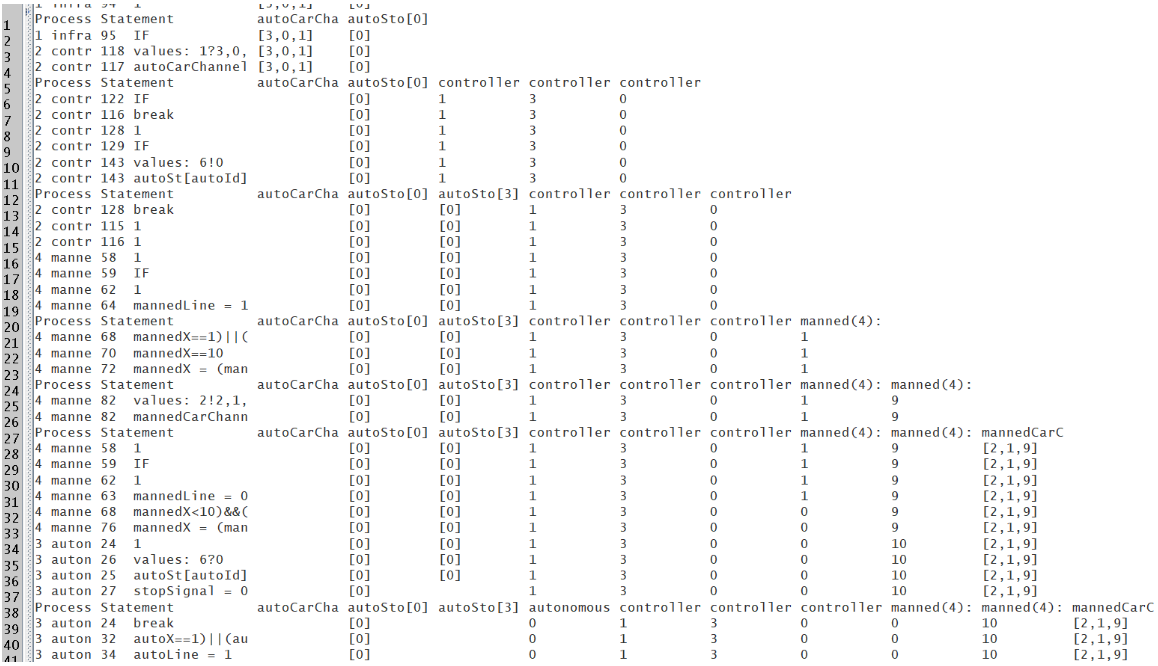


```
#define NUM_CARS 5
#define HALL_LENGTH 10
#define LINE_COUNT 2
#define MAX_AUTO 10
#define MAX_MANNED 10
```

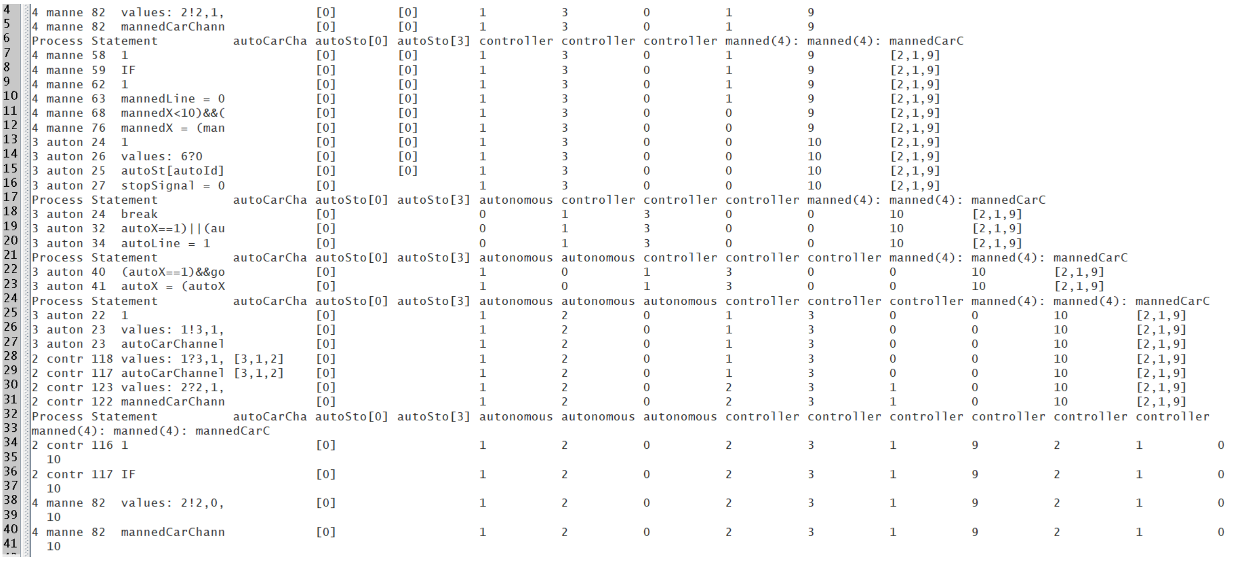
یک نمونه تریس از برنامه که ماشین ها از طریق اینفرا لایو لوکیشن میفرستن برای کنترلر و کنترلر چک میکندشون:



```
Starting autonomous with pid 3
 24:    proc  0 (:init:) creates proc  3 (autonomous)
0 :init ini run autonomous [0]
3 auton 22  1              [0]
3 auton 23  values: 1!3,0, [0]
3 auton 23  autoCarChannel [0]
Process Statement          autoCarCha autoSto[0]
3 auton 24  1              [3,0,1]    [0]
3 auton 25  IF             [3,0,1]    [0]
3 auton 24  1              [3,0,1]    [0]
3 auton 25  IF             [3,0,1]    [0]
3 auton 24  1              [3,0,1]    [0]
3 auton 25  IF             [3,0,1]    [0]
3 auton 24  1              [3,0,1]    [0]
Starting manned with pid 4
 34:    proc  0 (:init:) creates proc  4 (manned)
0 :init ini run manned(2)  [3,0,1]    [0]
1 infra 95  IF             [3,0,1]    [0]
1 infra 101 IF             [3,0,1]    [0]
1 infra 94  1              [3,0,1]    [0]
1 infra 95  IF             [3,0,1]    [0]
1 infra 101 IF             [3,0,1]    [0]
1 infra 94  1              [3,0,1]    [0]
3 auton 25  IF             [3,0,1]    [0]
3 auton 24  1              [3,0,1]    [0]
3 auton 25  IF             [3,0,1]    [0]
1 infra 95  IF             [3,0,1]    [0]
```



```
1 infra 94  1              [3,0,1]    [0]
Process Statement          autoCarCha autoSto[0]
1 infra 95  IF             [3,0,1]    [0]
2 contr 118 values: 1?3,0, [3,0,1]    [0]
2 contr 117 autoCarChannel [3,0,1]    [0]
Process Statement          autoCarCha autoSto[0] controller controller controller
2 contr 122 IF                        [0]        1          3          0
2 contr 116 break                     [0]        1          3          0
2 contr 128 1                         [0]        1          3          0
2 contr 129 IF                        [0]        1          3          0
2 contr 143 values: 6!0               [0]        1          3          0
2 contr 143 autoSt[autoId]            [0]        1          3          0
Process Statement          autoCarCha autoSto[0] autoSto[3] controller controller controller
2 contr 128 break                     [0]        [0]        1          3          0
2 contr 115 1                         [0]        [0]        1          3          0
2 contr 116 1                         [0]        [0]        1          3          0
4 manne 58  1                         [0]        [0]        1          3          0
4 manne 59  IF                        [0]        [0]        1          3          0
4 manne 62  1                         [0]        [0]        1          3          0
4 manne 64  mannedLine = 1            [0]        [0]        1          3          0
Process Statement          autoCarCha autoSto[0] autoSto[3] controller controller controller manned(4):
4 manne 68  mannedX==1)||(            [0]        [0]        1          3          0          1
4 manne 70  mannedX==10               [0]        [0]        1          3          0          1
4 manne 72  mannedX = (man            [0]        [0]        1          3          0          1
Process Statement          autoCarCha autoSto[0] autoSto[3] controller controller controller manned(4): manned(4):
4 manne 82  values: 2!2,1,            [0]        [0]        1          3          0          1          9
4 manne 82  mannedCarChann            [0]        [0]        1          3          0          1          9
Process Statement          autoCarCha autoSto[0] autoSto[3] controller controller controller manned(4): manned(4): mannedCarC
4 manne 58  1                         [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 59  IF                        [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 62  1                         [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 63  mannedLine = 0            [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 68  mannedX<10)&&(            [0]        [0]        1          3          0          0          9          [2,1,9]
4 manne 76  mannedX = (man            [0]        [0]        1          3          0          0          9          [2,1,9]
3 auton 24  1                         [0]        [0]        1          3          0          0          10         [2,1,9]
3 auton 26  values: 6?0               [0]        [0]        1          3          0          0          10         [2,1,9]
3 auton 25  autoSt[autoId]            [0]        [0]        1          3          0          0          10         [2,1,9]
3 auton 27  stopSignal = 0            [0]                   1          3          0          0          10         [2,1,9]
Process Statement          autoCarCha autoSto[0] autoSto[3] autonomous controller controller controller manned(4): manned(4): mannedCarC
3 auton 24  break                     [0]                   0          1          3          0          0          10         [2,1,9]
3 auton 32  autoX==1)||(au            [0]                   0          1          3          0          0          10         [2,1,9]
3 auton 34  autoLine = 1              [0]                   0          1          3          0          0          10         [2,1,9]
```



```
4 manne 82  values: 2!2,1,            [0]        [0]        1          3          0          1          9
4 manne 82  mannedCarChann            [0]        [0]        1          3          0          1          9
Process Statement          autoCarCha autoSto[0] autoSto[3] controller controller controller manned(4): manned(4): mannedCarC
4 manne 58  1                         [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 59  IF                        [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 62  1                         [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 63  mannedLine = 0            [0]        [0]        1          3          0          1          9          [2,1,9]
4 manne 68  mannedX<10)&&(            [0]        [0]        1          3          0          0          9          [2,1,9]
4 manne 76  mannedX = (man            [0]        [0]        1          3          0          0          9          [2,1,9]
3 auton 24  1                         [0]        [0]        1          3          0          0          10         [2,1,9]
3 auton 26  values: 6?0               [0]        [0]        1          3          0          0          10         [2,1,9]
3 auton 25  autoSt[autoId]            [0]        [0]        1          3          0          0          10         [2,1,9]
3 auton 27  stopSignal = 0            [0]                   1          3          0          0          10         [2,1,9]
Process Statement          autoCarCha autoSto[0] autoSto[3] autonomous controller controller controller manned(4): manned(4): mannedCarC
3 auton 24  break                     [0]                   0          1          3          0          0          10         [2,1,9]
3 auton 32  autoX==1)||(au            [0]                   0          1          3          0          0          10         [2,1,9]
3 auton 34  autoLine = 1              [0]                   0          1          3          0          0          10         [2,1,9]
Process Statement          autoCarCha autoSto[0] autoSto[3] autonomous autonomous controller controller controller manned(4): manned(4): mannedCarC
3 auton 40  (autoX==1)&&go            [0]                   1          0          1          3          0          0          10         [2,1,9]
3 auton 41  autoX = (autoX            [0]                   1          0          1          3          0          0          10         [2,1,9]
Process Statement          autoCarCha autoSto[0] autoSto[3] autonomous autonomous autonomous controller controller controller manned(4): manned(4): mannedCarC
3 auton 22  1                         [0]                   1          2          0          1          3          0          0          10         [2,1,9]
3 auton 23  values: 1!3,1,            [0]                   1          2          0          1          3          0          0          10         [2,1,9]
3 auton 23  autoCarChannel            [0]                   1          2          0          1          3          0          0          10         [2,1,9]
2 contr 118 values: 1?3,1, [3,1,2]    [0]                   1          2          0          1          3          0          0          10         [2,1,9]
2 contr 117 autoCarChannel [3,1,2]    [0]                   1          2          0          1          3          0          0          10         [2,1,9]
2 contr 123 values: 2?2,1,            [0]                   1          2          0          2          3          1          0          10         [2,1,9]
2 contr 122 mannedCarChann            [0]                   1          2          0          2          3          1          0          10         [2,1,9]
Process Statement          autoCarCha autoSto[0] autoSto[3] autonomous autonomous autonomous controller controller controller controller controller controller
manned(4): manned(4): mannedCarC
2 contr 116 1                         [0]                   1          2          0          2          3          1          9          2          1          0
  10
2 contr 117 IF                        [0]                   1          2          0          2          3          1          9          2          1          0
  10
4 manne 82  values: 2!2,0,            [0]                   1          2          0          2          3          1          9          2          1          0
  10
4 manne 82  mannedCarChann            [0]                   1          2          0          2          3          1          9          2          1          0
  10
```