به نام خدا

نام اعضای گروه :

عرفان ابراهیمی

سپهر رضایی

زمستان ۹۹

پروژه ۱

سیستم کروز کنترل :

با توجه به صورت سوال و مشخص شدن استیت های این سییستم کروز کنترل دارای ۴ استیت است که به ترتیب عبارتند از :

- پدال گاز
- پدال ترمز
- داشبورد
- کنترلر

که برای هندل کردن این سیستم و هندل کردن بازه کنترل سرعت بر روی بازه ۴۰ تا ۱۶۰ کیلومتر بر ساعت باید به نحوی عمل کند که یک استیت پدال گاز تا زمانی که سرعت به محدوده کنترلی ما نرسیده است شروع به افزایش سرعت می کند و این افزایش سرعت تا زمانی رخ میدهد که این سرعت در این بازه کنترلی قرار داشته باشد زمانی که سرعت از یک حدی بالا می رود سیستم پدال ترمز شروع به ترمز کردن و کاهش سرعت انجام می دهد سیستم داشبورد نیز شامل دو دکمه است دکمه on که برای روشن کردن سیستم کروز کنترل و دکمه off برای خاموش کردن سیستم کروز کنترل است . همچنین دکمه به اسم Set-Accl وظیفه دوگانه دارد اگر اتوموبیل در این استیت قرار داشته باشد این بدین معناست که دکمه on فشرده شده است و سیستم شروع می کند برای تنظیم کردن سرعت و ... در آن بازه انتخابی اگر اتوموبیل در حالت کروز کنترلر باشد با رفتن به این استیت سیستم شروع به افزایش سرعت می کند هم چنین دکمه Resume-Decel اگر راننده اتومبیل در حالت کروز به پدال ترمز برخورد کند ، CCS در حالت غیرفعال خواهد بود. زدن دکمه Resume-Decel در حالت غیرفعال ، به

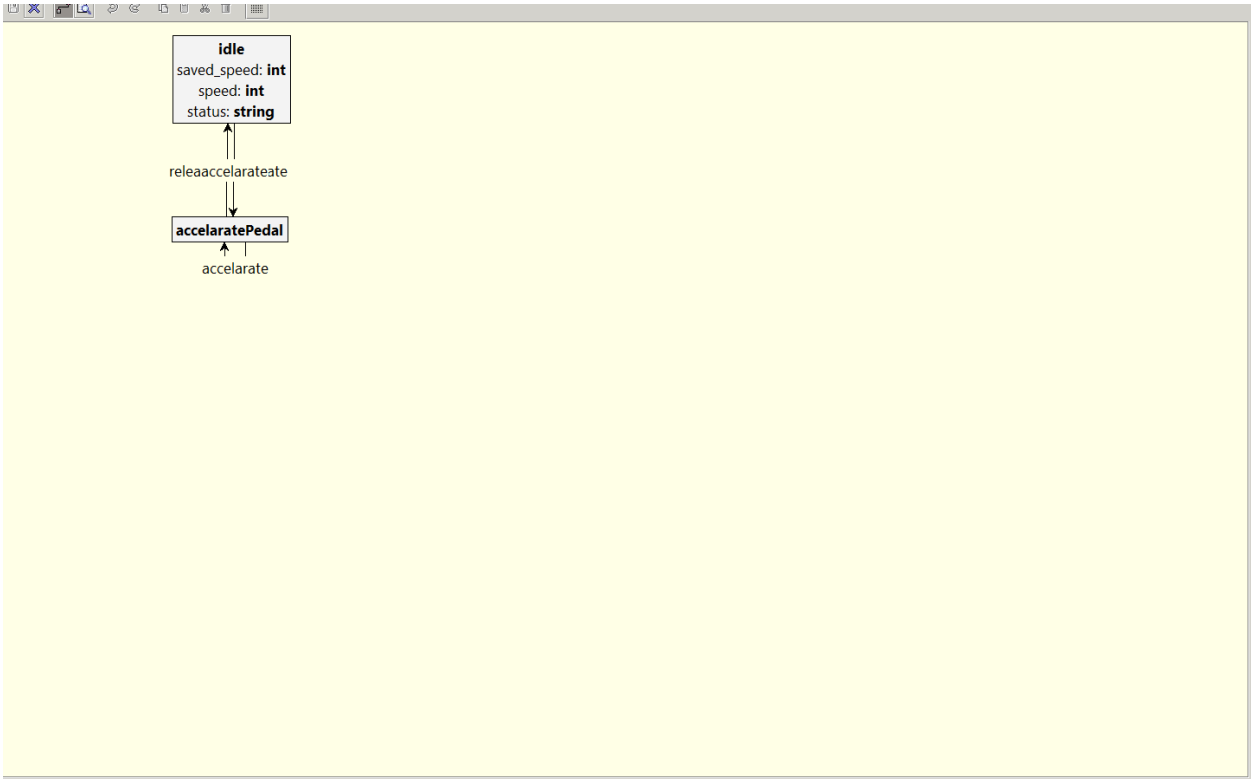
ماشین فرمان می دهد تا به آخرین تنظیمات سرعت بازگردد. اگر اتومبیل در حالت کروز کنترل باشد و دکمه Resume-Decel زده شود ، نگه داشتن دکمه Resume-Decel باعث کند شدن سرعت خودرو می شود.

همچنین وقتی CCS در ماشین از طریق کنترل کننده داشبورد فعال می شود ، کنترلر رفتار صحیح CCS را مدیریت می کند. با فشار دادن پدال گاز ، CCS غیرفعال شده و سرعت افزایش می یابد. با آزاد شدن شتاب دهنده ، CCS با آخرین سرعت تنظیم شده خود از سر می گیرد. اگر در هر زمان از شتاب سرعت CCS تنظیم شود ، CCS سرعت تنظیم شده قدیمی را با سرعت جدید جایگزین می کند. کنترلر کنترل کننده دارای ویژگی های ایمنی زیر است:

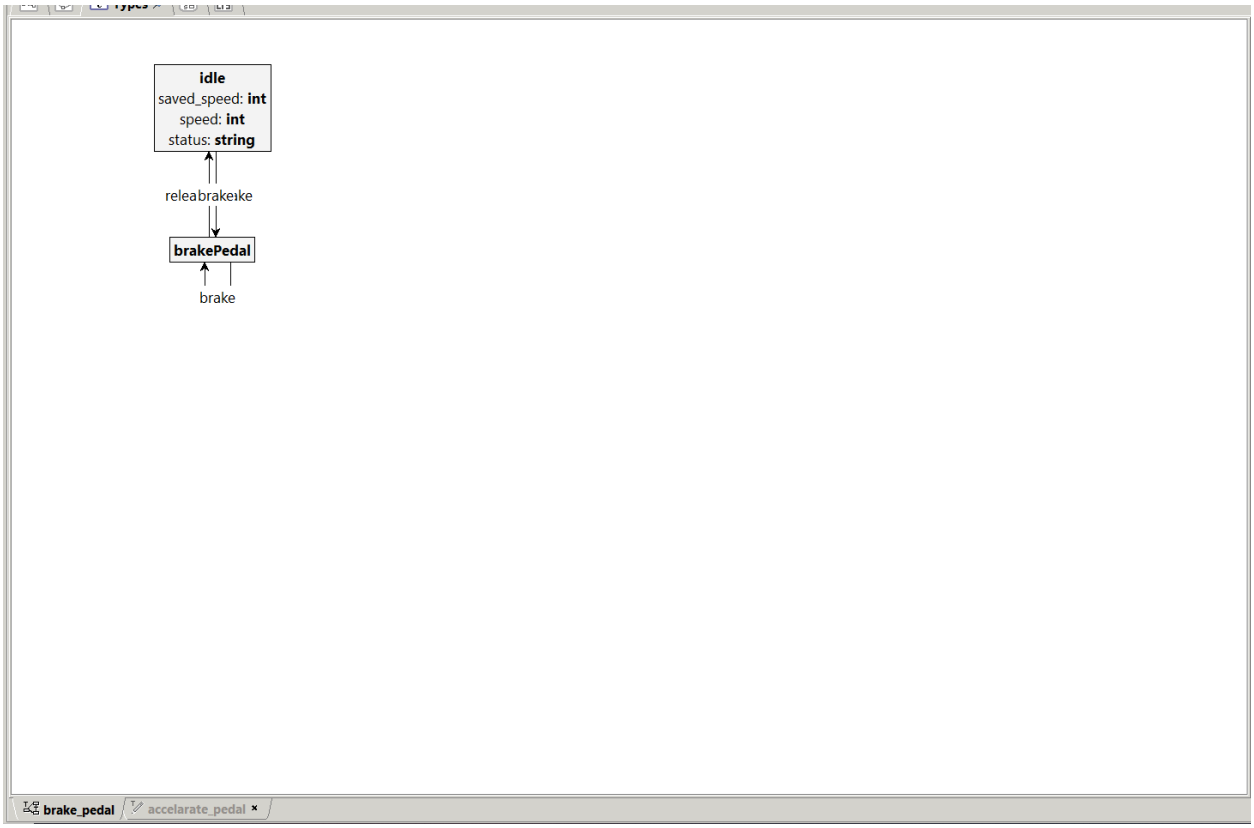
۱- هنگامی که سرعت خودرو یا کمتر از 45 کیلومتر در ساعت یا بالاتر از 160 کیلومتر در ساعت باشد ، CCS به طور خودکار غیرفعال می شود.

۲- با فعال شدن سیستم ضد قفل ترمز ، CCS به طور خودکار غیرفعال می شود.

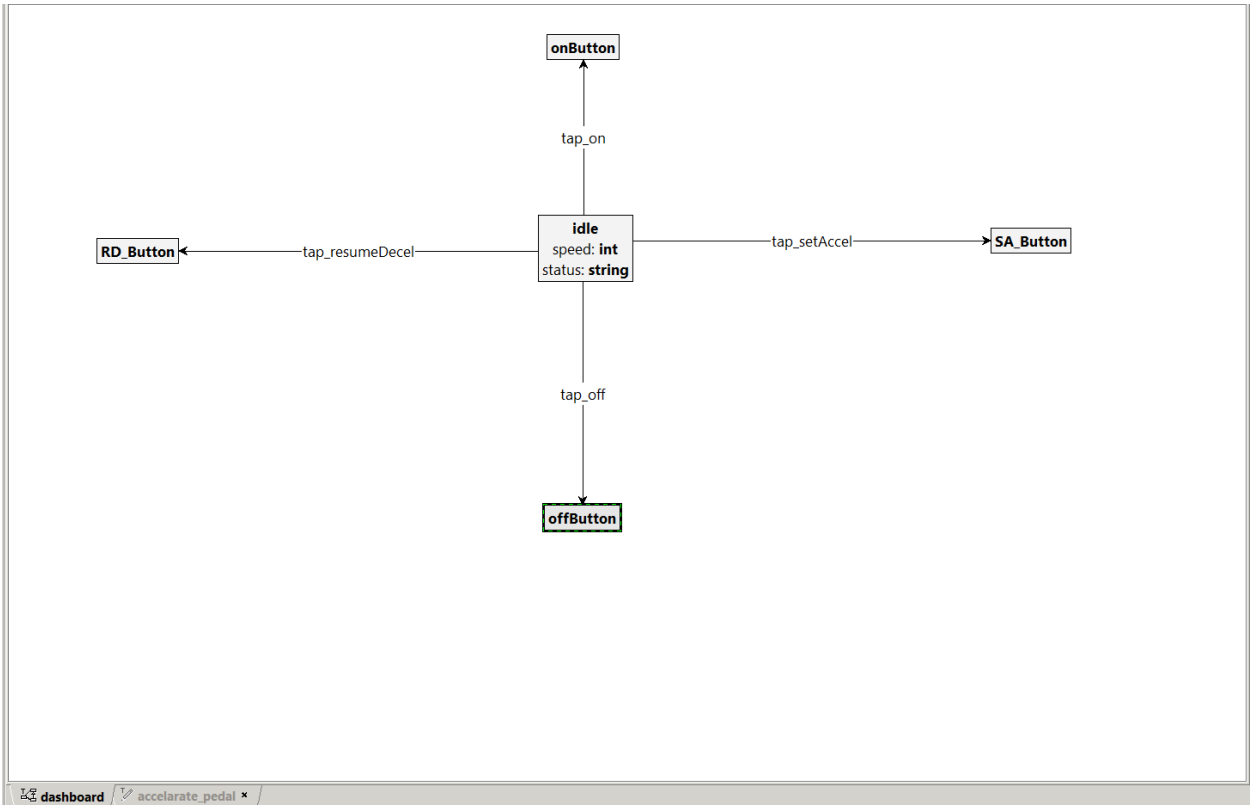
استیت افزایش سرعت را به صورت زیر در گروه رسم می کنیم :



همچنین سیستم پدال ترمز نیز به صورت زیر در گروو رسم می کنیم :



همچنین سیستم داشبورد که در بالا توضیح دادیم نیز به شکل زیر پیاده سازی می کنیم :

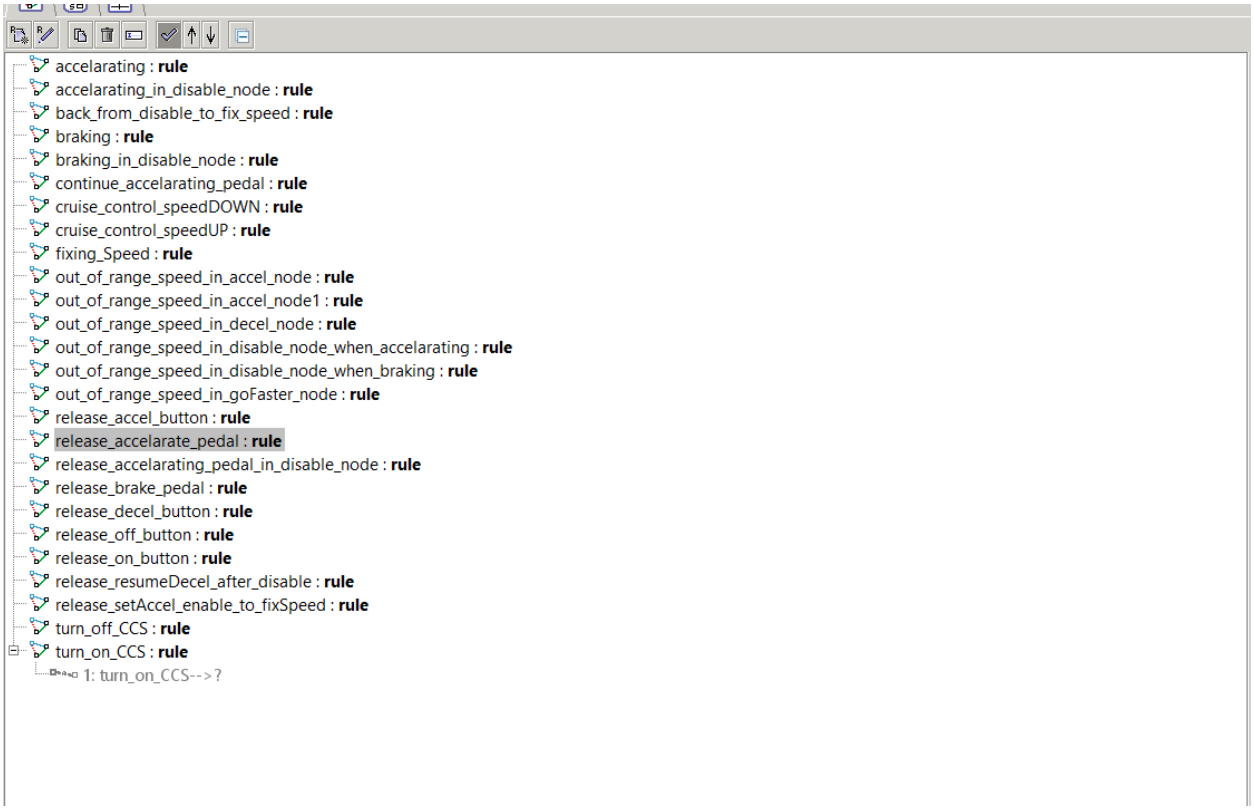


که این نحوه پیاده سازی مطابق با FA های موجود در سوال رسم کردیم همچنین بعد از تعریف این استیت به تعریف rule های موجود می پردازیم که به ازای هر transition یک رول در بالا رسم میکنیم که رول های تعداد زیاد هستند که به مختصر برخی از آن ها را شرح می دهیم :

- Fix_speed وظیفه دارد وقتی سیستم در حالت fixSpeed باشد و وضعیت کروز کنترل به صورت فعال باشد ترنزیشن tap_setAccl را که در بالا توضیح دادیم فراخوانی می کند

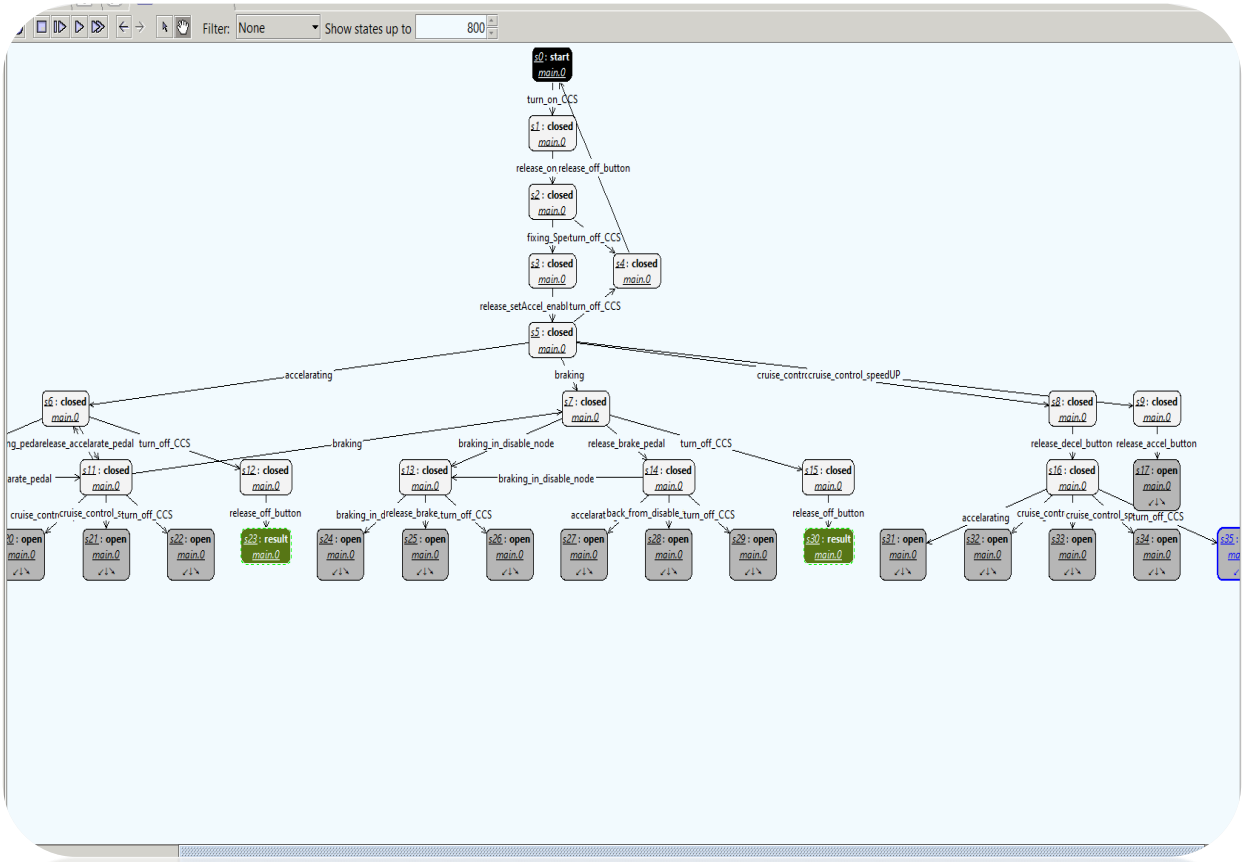
- **rule** : release_accelarate_pedal نیز زمانی که سرعت به رنج تنظیم شده نرسیده باشد با افزایش سرعت و قرار دادن سیستم در حالت goFaseter و فشردن پدال گاز سرعت سیستم را افزایش می دهد .

و به طور کلی رول ها و قوانینی که با توجه به صورت سوال تعریف می کنیم برای مثال کاهش سرعت و فشردن پدال ترمز زمانی که سرعت از رنج تعریف شده خارج بشود و یا بعد از فشرده شدن دکمه set_Accl و سیستم کروز در حالت غیر غعال باشد که سیستم شروع به افزایش سرعت می کند که تا به بازه تعریف شده برسد و همچنین رول های دیگر که لیست تمامی رول ها را در این بخش از پروژه به وضوح مشاهده می کنید :



برای اجرا پروژه کافی است فاید موجود در پوشه project1 را به صورت loadGrammer بزنید و سپس در منو تب پایین قسمت استارت کلیک کنید در

پنجره باز شده به حالت state می رویم و دکمه پلی را فشار می دهیم نمونه اجرا شده را در تصویر پایین مشاهده می کنید :



پروژه ۲

با توجه به تعریف پروژه که داریم دستگاهی داریم که پیامی را برای یک کنترلر ارسال می کند که ارسال کننده با توجه به رنگ چراغ و وضعیت بالا بودن گیت دستور به انجام تغییری در وضعیت گیت و تغییر رنگ چراغ می پردازد با توجه به صورت سوال استیت های زیر مورد بررسی قرار گرفته است :

- اگر چراغ سبز باشد دستور گیت raised را به کنرتلر ارسال می کند .
- اگر چراغ زرد باشد و گیت پایین باشد دستور گیت closed را به کنترلر ارسال می کند .
- اگر چراغ زرد باشد و گیت بالا باشد دستور گیت open را به کنترلر ارسال می کند
- اگر جراغ قرمز باشد دستور گیت lowered را ارسال می کند

مواردی که در کنترلر مورد بررسی قرار گرفته است :

- اگر دستور گیت open به کنترلر برسد چراغ را به رنگ قرمز در می آورد و نیز speed به حالت stop می رود .
- اگر دستور گیت lowered به کنترلر ارسال شود چراغ را به رنگ زرد در می آورد و نیز speed به حالت low می رود.
- اگر دستور گیت closed را به کنترلر برسد چراغ به رنگ سبز در می آید و نیز speed به حالت normal می رود.

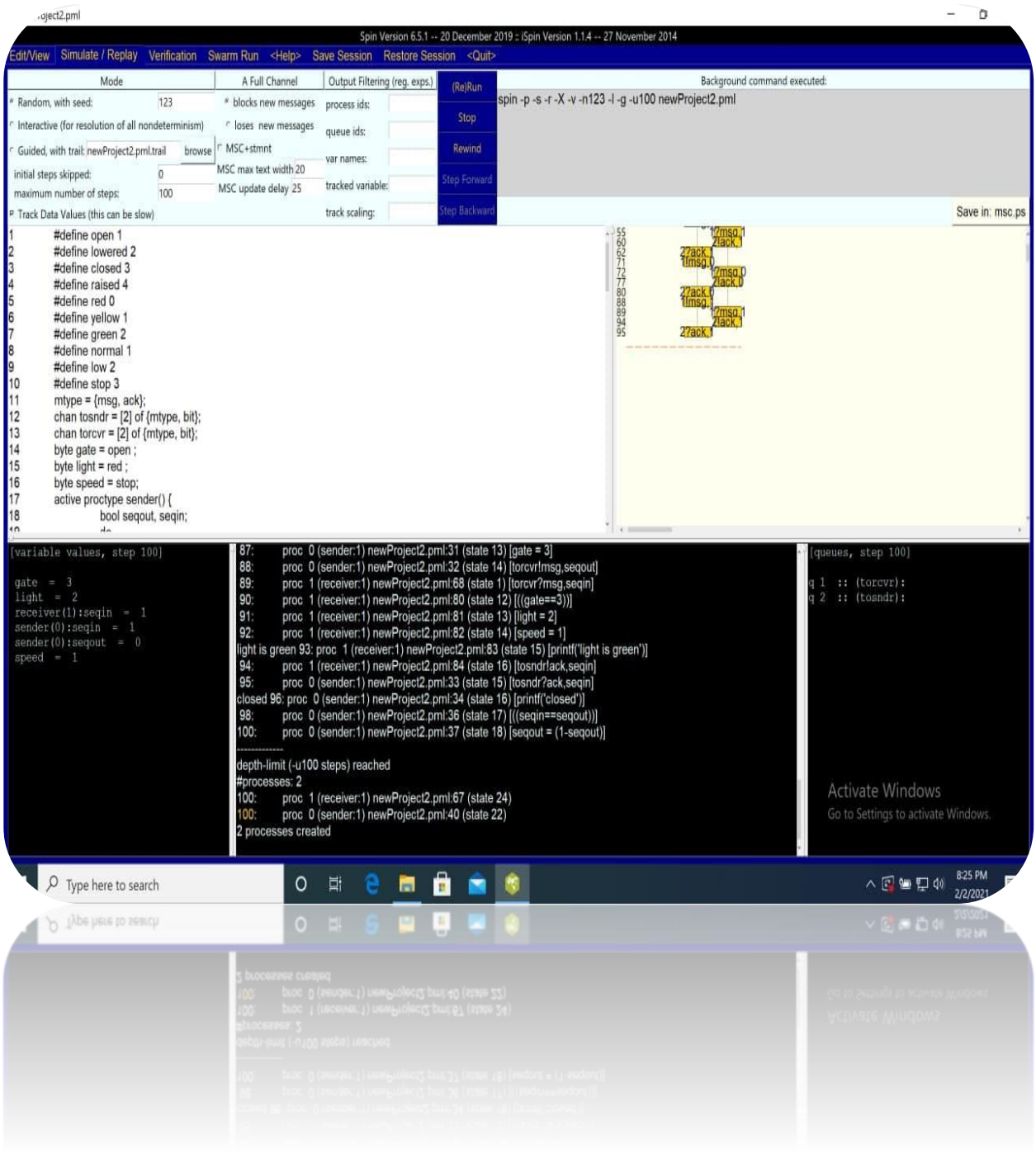
- اگر دستور گیت raised به کنترلر برسد جراغ به رنگ زرد در می آید و نیز speed به حالت low می رود .

کد ها شامل دو ماژول هستند که به صورت activeproctype اجرا می شوند سه ماژول به شرح زیر می باشند :

- ماژول sender برای کنترل کردن گیت برای دستورات حرکتی گیت
- ماژولreceiver کنترل کردن رنگ چراغ و همچنین سرعت قطار

کلیه بخش های این کد به صورت چنل پیاده سازی شده است که برای اجرا فایل موجود در پروژه در قسمت project2 را در اسپین کپی کرده و یا open کرده و در قسمت وریفیکیشن تعداد اسیت ها را مقدار دلخواه برای مثال ۱۰۰۰ قرار می دهیم سپس پروژه رو اجرا میکنیم در قسمت چکینگ .

برای مثال نمونه کد اجرا شده را در تصویر پایین مشاهده می کنید :


oject2.pml
Spin Version 6.5.1 -- 20 December 2019 :: iSpin Version 1.1.4 -- 27 November 2014
Edit/View Simulate / Replay Verification Swarm Run <Help> Save Session Restore Session <Quit>
Mode
Random, with seed: 123
Interactive (for resolution of all nondeterminism)
Guided, with trail: newProject2.pml.trail browse
initial steps skipped: 0
maximum number of steps: 100
Track Data Values (this can be slow)
A Full Channel
blocks new messages
loses new messages
MSC+stmnt
MSC max text width 20
MSC update delay 25
Output Filtering (reg. exps.)
process ids:
queue ids:
var names:
tracked variable:
track scaling:
(Re)Run
Stop
Rewind
Step Forward
Step Backward
Background command executed:
spin -p -s -r -X -v -n123 -l -g -u100 newProject2.pml
Save in: msc.ps
1 #define open 1
2 #define lowered 2
3 #define closed 3
4 #define raised 4
5 #define red 0
6 #define yellow 1
7 #define green 2
8 #define normal 1
9 #define low 2
10 #define stop 3
11 mtype = {msg, ack};
12 chan tosndr = [2] of {mtype, bit};
13 chan torcvr = [2] of {mtype, bit};
14 byte gate = open ;
15 byte light = red ;
16 byte speed = stop;
17 active proctype sender() {
18 bool seqout, seqin;
[variable values, step 100]
gate = 3
light = 2
receiver(1):seqin = 1
sender(0):seqin = 1
sender(0):seqout = 0
speed = 1
87: proc 0 (sender:1) newProject2.pml:31 (state 13) [gate = 3]
88: proc 0 (sender:1) newProject2.pml:32 (state 14) [torcvr!msg,seqout]
89: proc 1 (receiver:1) newProject2.pml:68 (state 1) [torcvr?msg,seqin]
90: proc 1 (receiver:1) newProject2.pml:80 (state 12) [((gate==3))]
91: proc 1 (receiver:1) newProject2.pml:81 (state 13) [light = 2]
92: proc 1 (receiver:1) newProject2.pml:82 (state 14) [speed = 1]
light is green 93: proc 1 (receiver:1) newProject2.pml:83 (state 15) [printf('light is green')]
94: proc 1 (receiver:1) newProject2.pml:84 (state 16) [tosndr!ack,seqin]
95: proc 0 (sender:1) newProject2.pml:33 (state 15) [tosndr?ack,seqin]
closed 96: proc 0 (sender:1) newProject2.pml:34 (state 16) [printf('closed')]
98: proc 0 (sender:1) newProject2.pml:36 (state 17) [((seqin==seqout))]
100: proc 0 (sender:1) newProject2.pml:37 (state 18) [seqout = (1-seqout)]
depth-limit (-u100 steps) reached
#processes: 2
100: proc 1 (receiver:1) newProject2.pml:67 (state 24)
100: proc 0 (sender:1) newProject2.pml:40 (state 22)
2 processes created
[queues, step 100]
q 1 :: (torcvr):
q 2 :: (tosndr):
Activate Windows
Go to Settings to activate Windows.
Type here to search
8:25 PM
2/2/2021

پروژه ۳

در این پروژه با توجه به صورت مسئله داریم که دو نقش در سیستم قرار دارد که نقش اول متشکل از publisher (ناشر) و subscriber (مشترکین) می باشد که publisher یک چیز را منتشر می کند و subscriber این چیز را دنبال می‌کند . که با توجه به خواسته سوال باید دو شرط زیر برقرار باشد :

- فقط مشترکین باید اطلاعات منتشر شده توسط ناشر را دریافت کنند
- همه مشترکین اطلاعات یکسانی را دریافت می کنند.

در حل این سوال چهار استیت در نظر گرفته شده است:

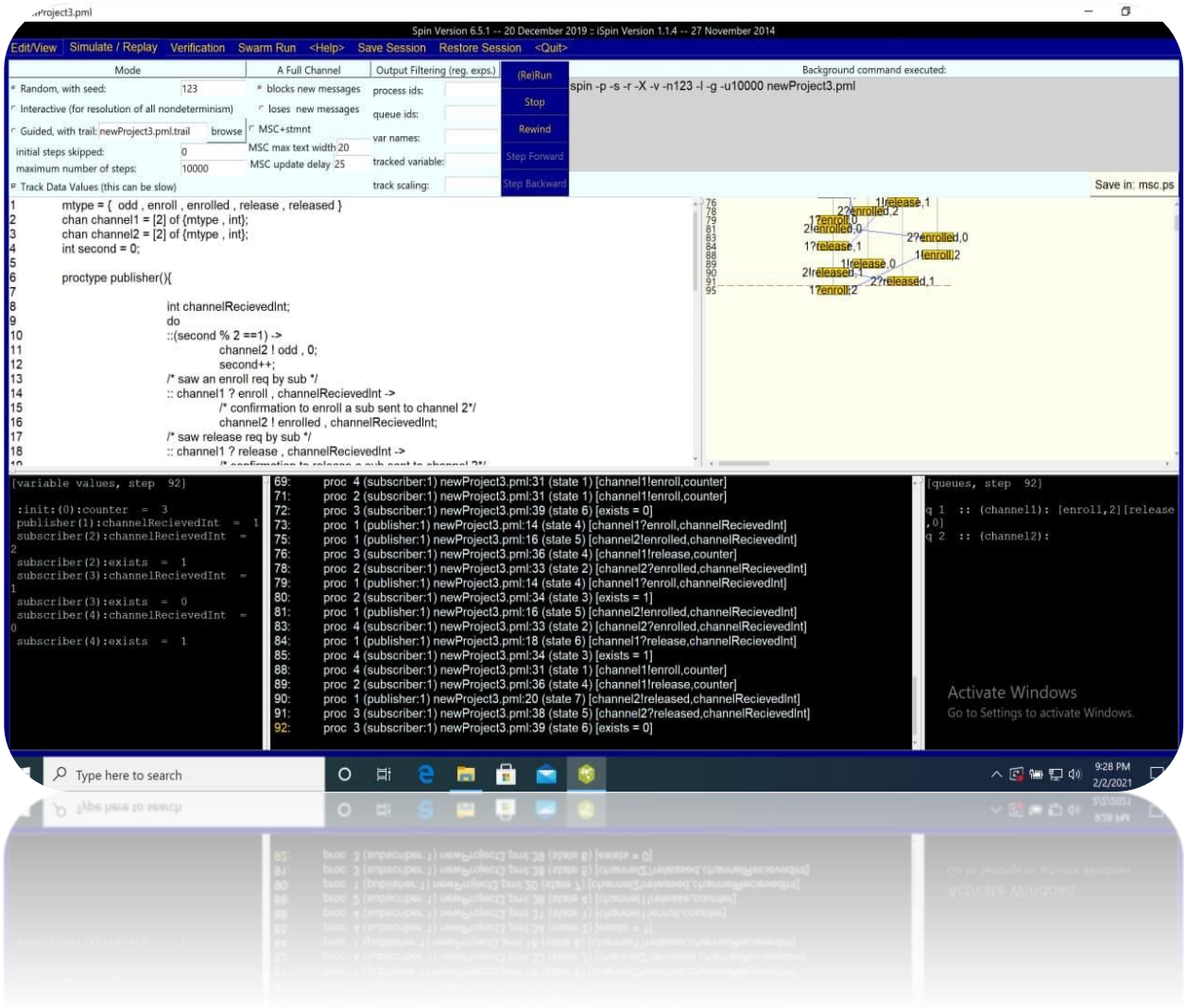
- Enroll : زمانی است که یک از مشترکین درخواست انرول کردن در یک چیزی که ناشر منتشر کرده است می کند در این استیت قرار میگیریم .
- Enrolled : زمانی که مشترکین در خواست انرول دادند و ناشر درخواست را قبول کرد به این استیت می رویم و یک ایدی را در این مرحله به عنوان ack می فرستیم و عضویت یک مشترک را در داخل آن فرآیند (process) یک می کند
- Release : زمانی که یک مشترک توسط یک ناشر انرول شده باشد مشترک درخواست ریلیز می دهد .
- Released: زمانی در این استیت قرار می گیریم که ناشر درخواست یک مشترک را قبول کرده باشد و ناشر یک ack را برای او ارسال کرده باشد عضویت یک مشترک را در داخل آن فرآیند (process) صفر می کند .

در ماژول ناشر یک ثانیه شمار وجود دارد که در چنل در هر ثانیه فرد یک مسیج odd برادکست می کند.

ماژول های موجود در این پروژه دو تا هستند که عبارتند از :

- Publisher
- Subscriber

که کد موجود را در پوشه project3 مشاهده می کنید .



لینک دسترسی به پروژه در گیت هاب :

https://github.com/erfangoodboy/formalMethods9901