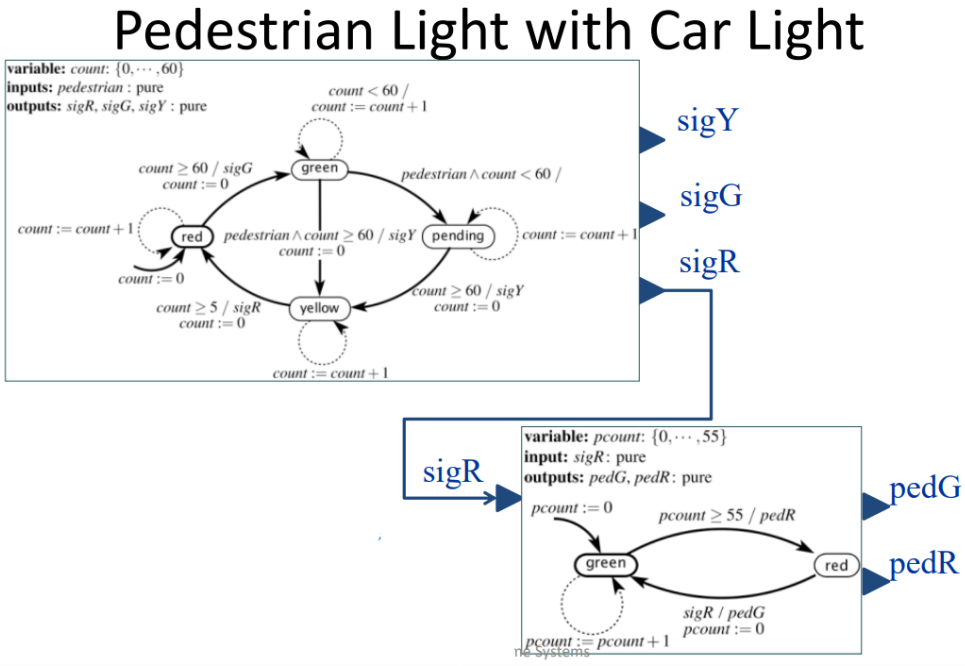
پاسخ تمرین سری دوم مبانی سیستم‌های نهفته و بیدرنگ

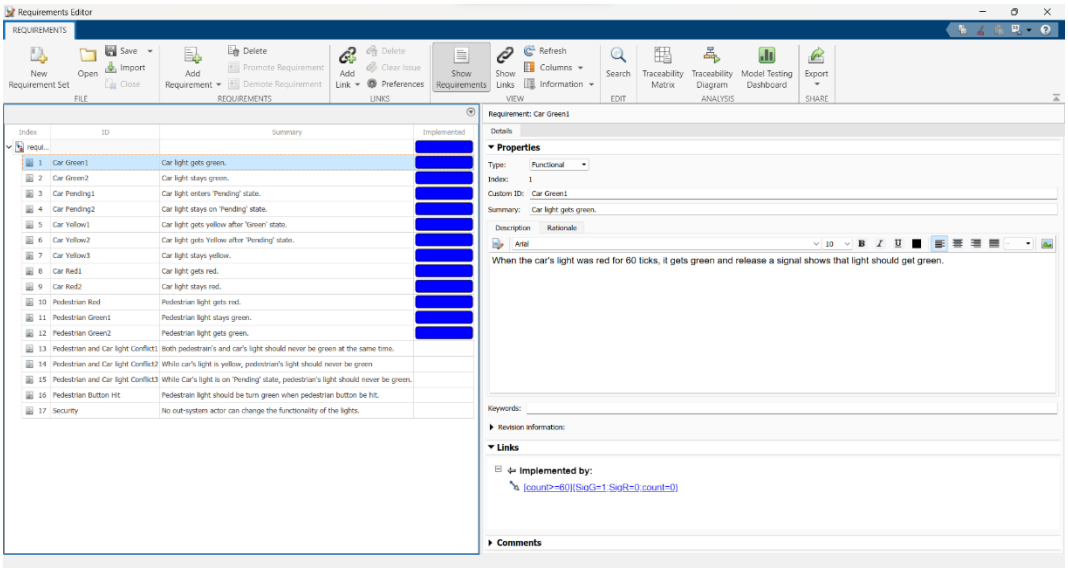
۱. در این سوال قرار است مدل چراغ راهنمایی ماشین‌ها و عابر پیاده (که در مرجع Lee به آن پرداخته شده است) توسط محیط نرم‌افزار Simulink پیاده‌سازی شود.

این مدل در مرجع Lee به صورت زیر نمایش داده شده است:



شکل ۱: مدل چراغ راهنمایی ماشین و عابرپیاده

أ. در این بخش به کمک Simulink Requirements، نیازمندی‌های functional و extra-functional مدل را بیان می‌کنیم. این نیازمندی‌های در Simulink Requirements به صورت زیر است:

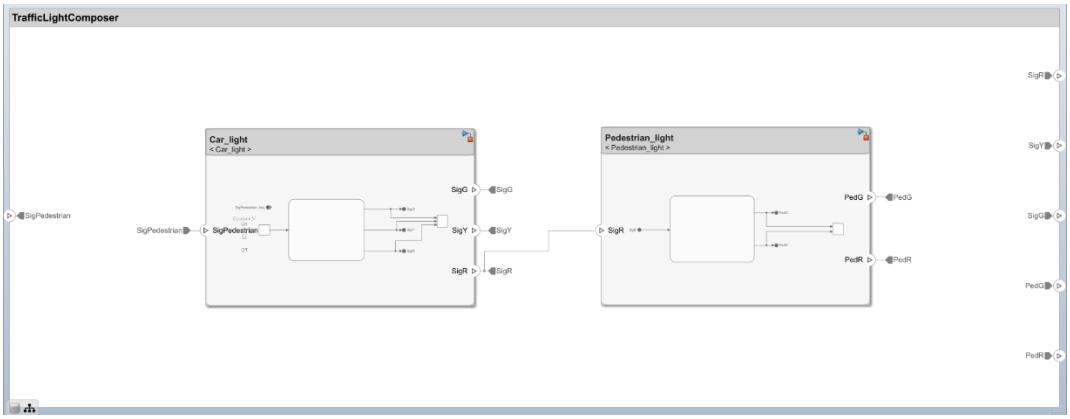


شکل ۲: نیازمندی‌های تعریف شده مدل چراغ راهنمایی در Simulink Requirements

از نیازمندی ۱ تا ۱۲ (که نیازمند پیاده‌سازی نیز هستند)، نیازمندی‌های کارکردی و از نیازمندی ۱۳ تا ۱۷، نیازمندی‌های فراکارکردی را داریم.

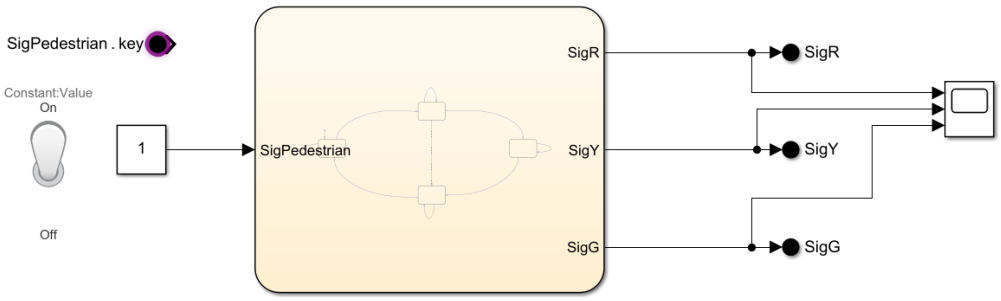
هر کدام از نیازمندی‌ها یک خلاصه و توضیح دارند. نیازمندی‌های کارکردی نیازمند پیاده‌سازی نیز هستند؛ در نتیجه هر کدام را به transaction متناسب با آن assign می‌کنیم. در نیازمندی‌های کارکردی تعریف شده در ابزار Simulink Requirements، transaction متناظر با آن نیازمندی در بخش Links (پایین سمت راست) قابل مشاهده است.

ب. طبق مدل نمایش داده شده در بالا (شکل ۱)، در این سیستم نیازمند دو component هستیم. کامپوننت سمت چپ با نام Car_light نشان‌دهنده بخش چراغ راهنمایی ماشین‌ها و کامپوننت سمت راست با نام Pedestrian_light نشان‌دهنده بخش چراغ راهنمایی عابران پیاده است. همچنین این سیستم یک سیگنال ورودی به نام SigPedestrian است که زمانی که عابر دکمه عابر پیاده را بفشارد، این سیگنال present خواهد شد. همچنین ۵ خروجی داریم که به ترتیب از بالا به پایین نمایانگر سیگنال قرمز بودن، سیگنال زرد بودن، سیگنال سبز بودن چراغ ماشین‌ها و سیگنال قرمز و سبز بودن چراغ عابران پیاده است. همچنین همانطور که در شکل ۱ دیدیم، می‌دانیم که برای اینکه چراغ عابران سبز شود، باید ابتدا چراغ ماشین‌ها قرمز شده باشند. در نتیجه سیگنالی که نشان می‌دهد چراغ راهنمایی قرمز است را به component عابران می‌دهیم.



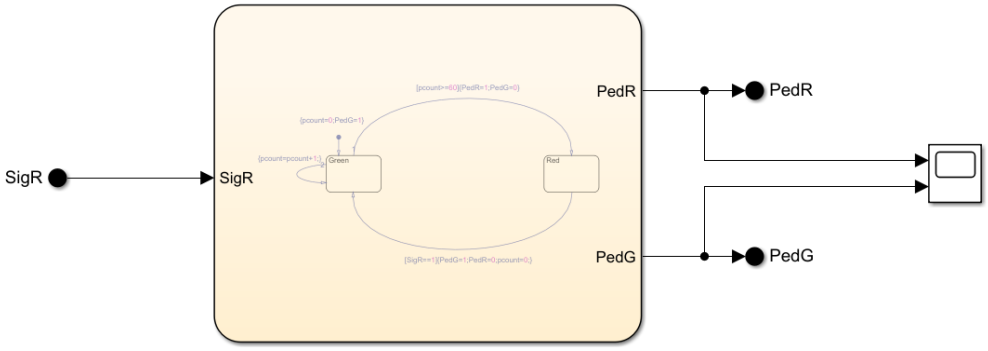
شکل ۳: شمای کلی معماری سیستم به همراه ورودی و خروجی و واسط میان دو component

درون کامپوننت Car_light، یک کلید برای ۰ و ۱ کردن ورودی (دکمه عابر) داریم و خروجی ما سیگنال‌های قرمز و زرد و سبز است (بسته به شرایط، یکی از آن‌ها برابر ۱ و دوتای دیگر ۰ می‌شوند).



شکل ٤:کامپوننت Car_light

درون کامپوننت Pedestrian_light، یک ورودی سیگنال قرمز کامپوننت Car_light را داریم و خروجی این کامپوننت، سیگنال‌های قرمز و سبز است که بسته به شرایط، یکی از آن‌ها ۱ و دیگری برابر ۰ است.

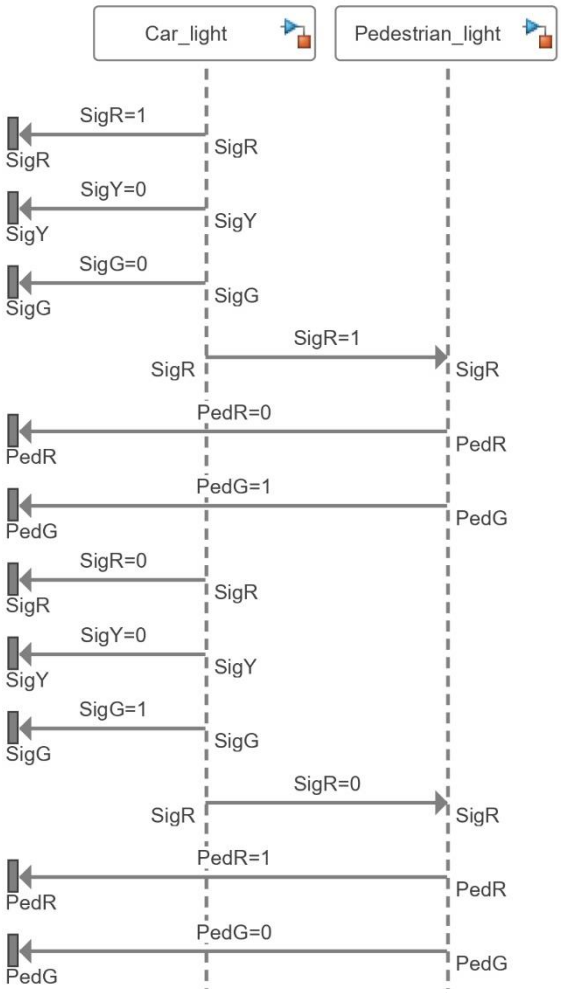


شکل ۵: کامپوننت Pedestrian_light

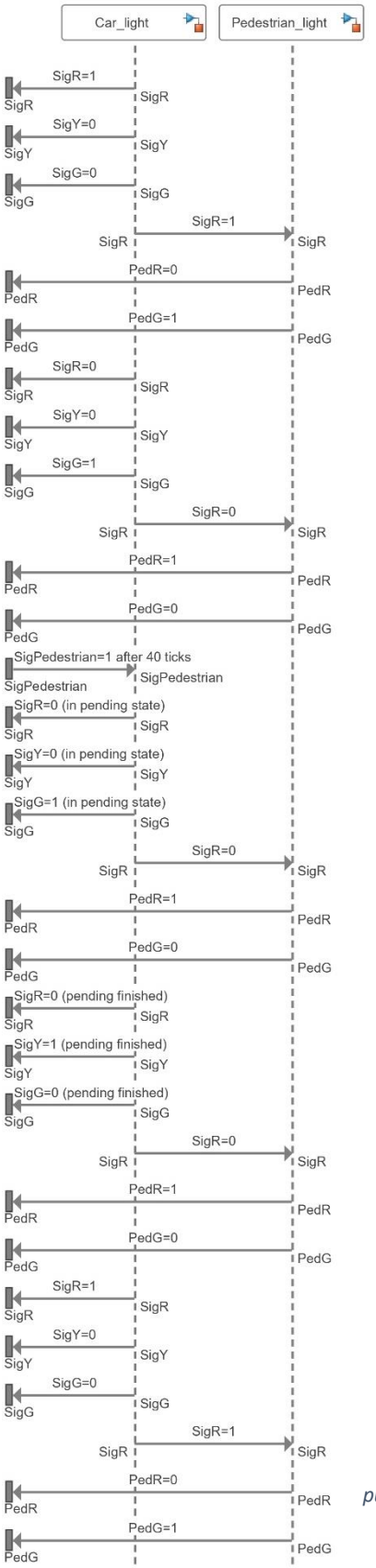
ج. مهم‌ترین سناریوهای این مدل شامل موارد زیر است:

- حالت اول:

زمانی که چراغ ماشین‌ها سبز است، هیچ عابری نباشد که دکمه عابر را فشار دهد. در این صورت، بعد از گذشت ۶۰ واحد زمانی ابتدایی که چراغ قرمز است، چراغ تا همیشه سبز می‌ماند.



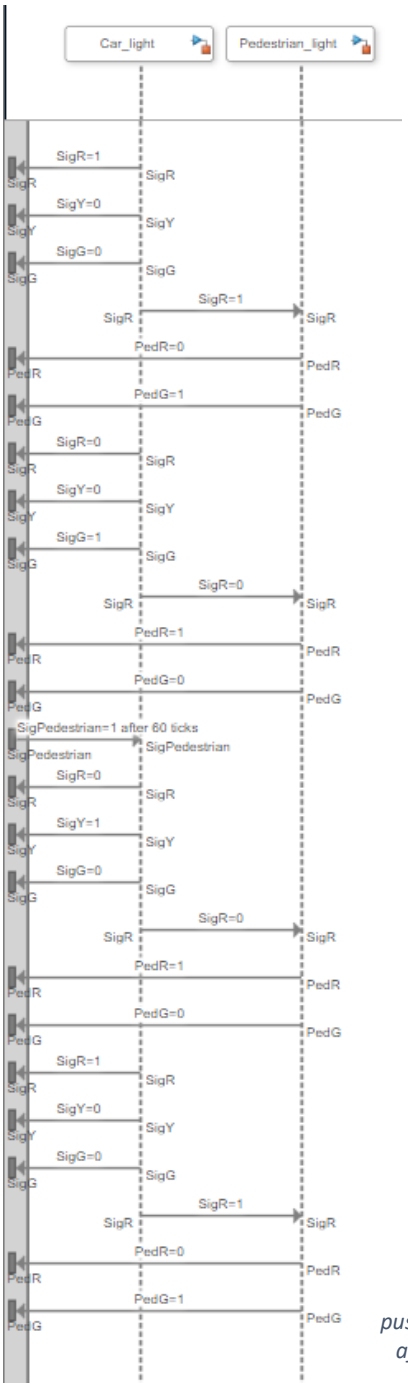
شکل ۶: no pedestrian button sequence diagram



شکل ۷: *pushed pedestrian button after 40 ticks sequence diagram*

- حالت دوم:

بعد از گذشتن حالت ابتدایی و سبز شدن چراغ ماشین‌ها بعد از ۶۰ واحد زمانی، هنگامی که چراغ ماشین‌ها کمتر از ۶۰ واحد زمانی سبز (مثلا در لحظه t=40 این دکمه فشرده شد) بود، عابری دکمه pedestrian را بفشارد. در این حالت، کامپوننت Pedestrian_light همچنان مقدار PedR=1 را روی خروجی می‌گذارد و کامپوننت Car_light نیز مقدار PedG=1 را روی خروجی قرار می‌دهد. حال در کامپوننت Car_light در استیت pending هستیم. زمانی که ۶۰ واحد زمانی گذشت، از این استیت بیرون می‌آییم و سیگنال SigY=1 را به خروجی می‌دهیم و پس از ۵ واحد زمانی، چراغ ماشین‌ها قرمز می‌شود و سپس چراغ عابرها سبز می‌گردد.



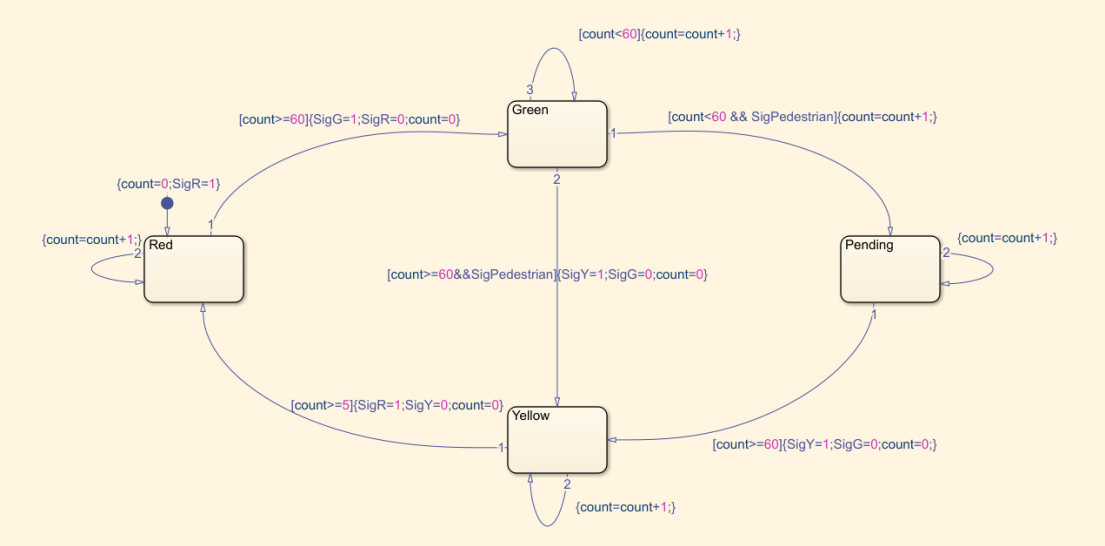
شکل ۸: *pushed pedestrian button after 60 ticks sequence diagram*

- حالت سوم:

بعد از گذشت حالت ابتدایی چراغ و سبز شدن آن، زمانی که به اندازه ۶۰ واحد زمانی چراغ سبز بود، عابری دکمه عابر را بفشارد. در این صورت، به محیط خارج سیگنال SigY=1 می‌دهد تا چراغ ماشین‌ها زرد شود. در این زمان همچنان چراغ عابر قرمز است. پس از گذشت ۵ واحد زمانی، چراغ ماشین‌ها قرمز می‌شود و سیگنال SigR=1 به component بعدی یعنی pedestrian_light می‌رود و چراغ عابر نیز سبز می‌شود.

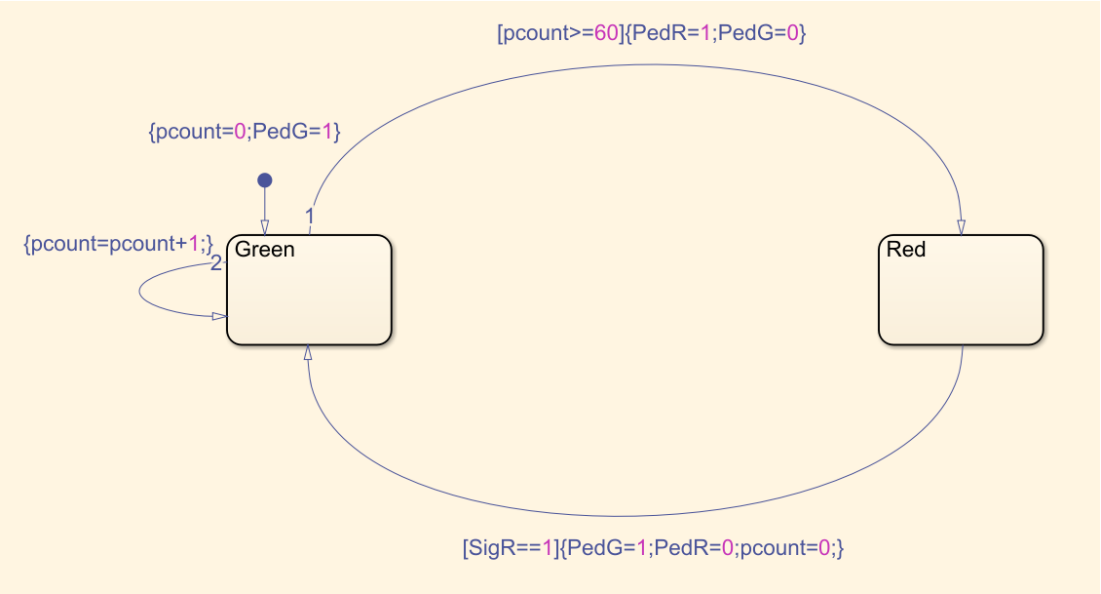
د. درون هر یک از componentها نیازمند یک StateChart به منظور طراحی تفضیلی آن component نیازمندیم. هر کدام از این StateChartها نیز مطابق با مدل ارائه شده در شکل ۱ هستند.

در تصویر زیر، ماشین میلی مرتبط با جزء Car_light دیده می‌شود:



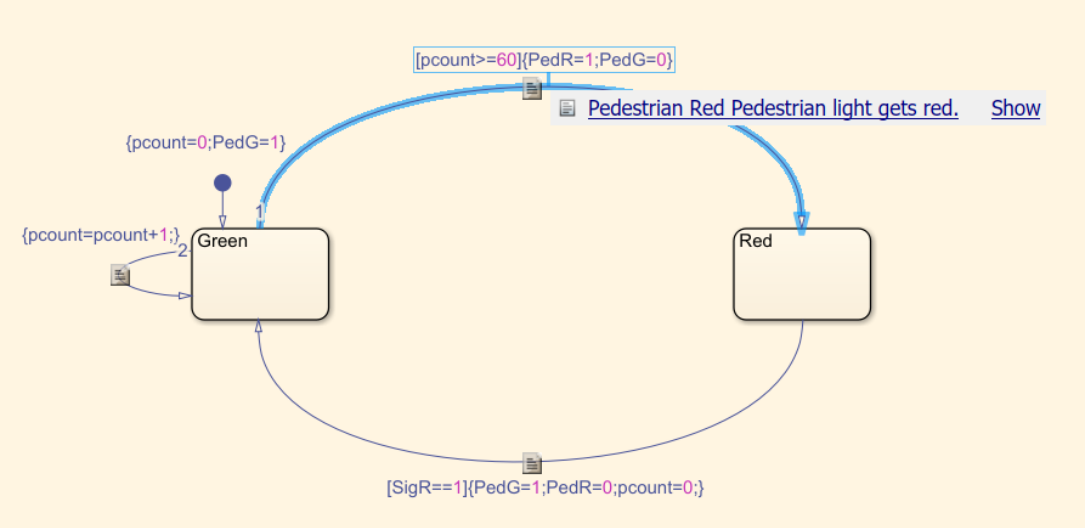
شکل ۹: ماشین میلی کامپوننت Car_light

در تصویر زیر، ماشین میلی مرتبط با جزء Pedestrian_light دیده می‌شود:



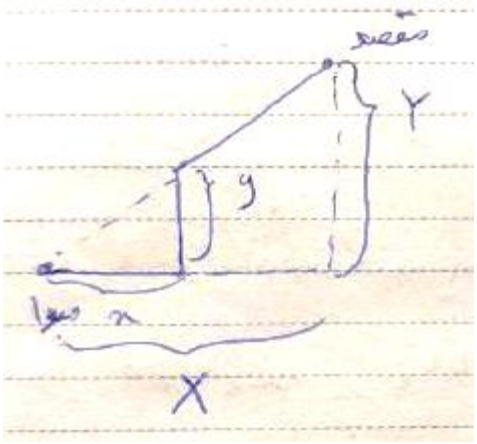
شکل ۱۰:ماشین میلی کامپوننت Pedestrian_light

ه. نیازمندی‌های تعریف شده در بخش أ در این مرحله باید به transaction متناظر خود assign شوند. برای این کار وارد بخش Simulink Requirements می‌شویم و نیازمندی مرتبط را با drag and drop به یال متناظر اختصاص می‌دهیم. در تصویر زیر، یک نمونه از نیازمندی اختصاص‌یافته را مشاهده می‌کنید:

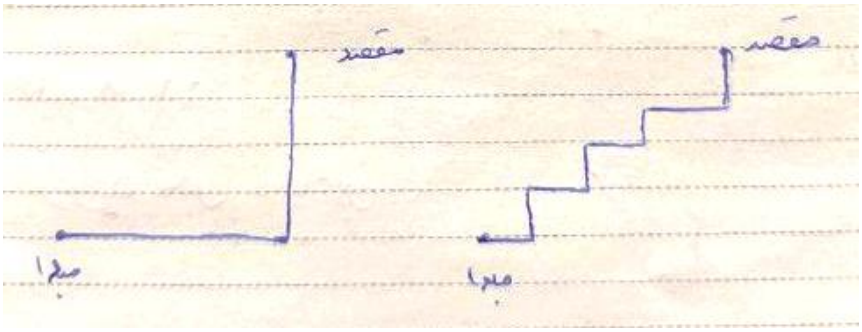


شکل ۱۱:تخصیص دادن نیازمندی شماره ۱۰ به transaction مربوطه

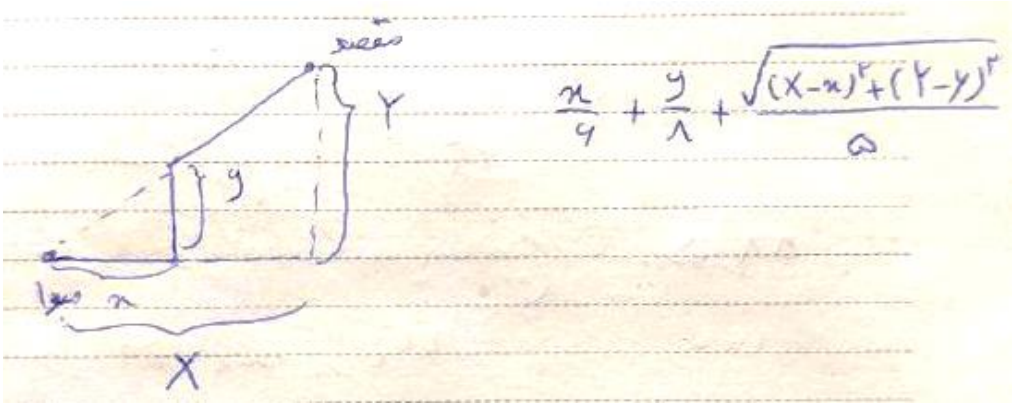
۲. برای رسیدن از مبدا به مقصد، ربات قسمتی از مسیر را در جهت محورx ، مقداری را در جهت محور y و مقدار باقی‌مانده را به صورت مستقیم طی می‌کند.



توجه شود که هر چقدر هم تغییر جهت داشته باشیم، باز هم مقادیر ذکر شده تغییر نمی‌کنند. برای مثال در دو شکل زیر این مقادیر برابرند:

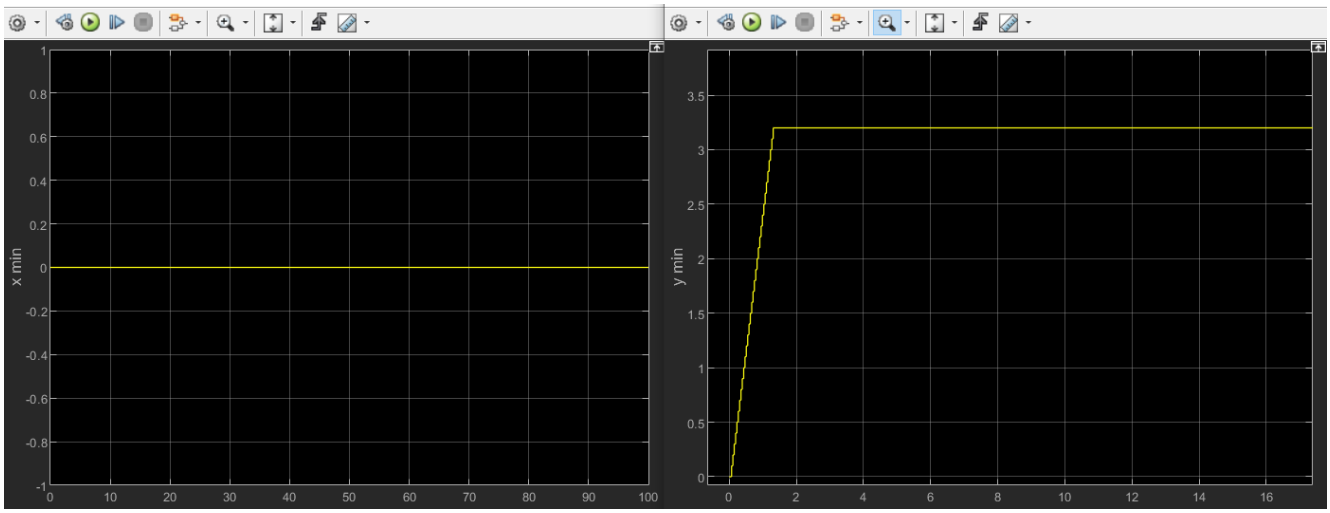


کل زمان لازم برای رسیدن از مبدا به مقصد از طریق فرمول زیر به دست می‌آید:



هدف این است که این مقادیر x و y طوری تعیین شوند که زمان مصرفی کمینه شود. این مقادیر را می‌توان با استفاده از یک ماشین حالت به دست آورد، به این صورت که مقادیر x و y را با واحدهای کوچک افزایش می‌دهیم و زمان مصرفی را محاسبه می‌کنیم و x و y زمان کمینه را در دو متغیر x_min و y_min ذخیره می‌کنیم.

در مثال ذکر شده، x_minو y_min به صورت زیر خواهند بود:

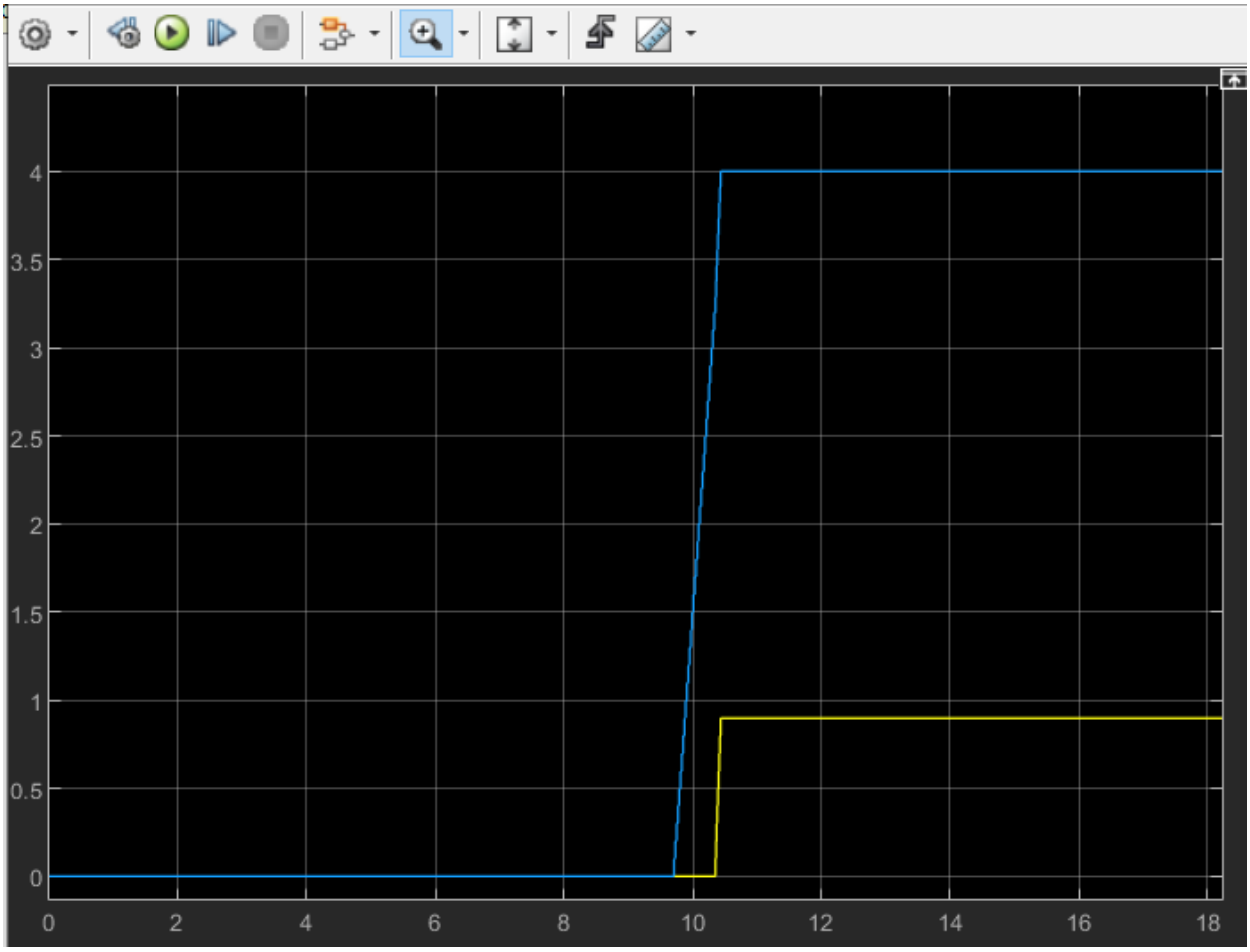


جواب را به وسیله‌ی حل معادله چک می‌کنیم:

$$\min\left\{\frac{x}{6}+\frac{y}{8}+\frac{1}{5}\sqrt{(1-x)^2+(4-y)^2}\;\middle|\;0<x<1\right\}\approx 0.656125 \text{ at } (x,y)\approx (0,\ 3.19936)$$

سپس وارد ماشین اصلی می‌شویم که در آن ۶ حالت داریم: توقف، حرکت به راست، چپ، بالا، پایین و مستقیم. برای انتخاب مسیر، دو حالت را در نظر می‌گیریم: ۱- مسیر مستقیم ۲- ابتدا مسیر افقی و سپس عمودی

ابتدا ربات را در جهت محور x حرکت می‌دهیم تا به اندازه‌ی x_min در این راستا حرکت کند، سپس همین کار را برای محور y انجام می‌دهیم و در نهایت در راستای مستقیم. نحوه‌ی تغییر مقادیر x و y ربات در این مثال به صورت زیر خواهد بود:


4
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
14
16
18