Mechanical Simulation
755 Phoenix Drive, Ann Arbor MI, 48108, USA
Phone: 734 668-2930 • Fax: 734 668-2877 • Email: info@carsim.com

Sine with Dwell Test in CarSim

Many countries are requiring Electronic Stability Control (ESC) safety systems on new vehicles. In the USA, nearly all vehicles sold after September 2011 must be equipped with ESC and comply with the regulation FMVSS 126, published in 2007. The FMVSS 126 requirements were more or less duplicated into the United Nations ECE R13H, which applies to European Union countries and many countries in Asia. Both FMVSS 126 and ECE R13H define a test procedure that uses a driving maneuver known as the *Sine with Dwell*.

OEMs simulate the sine with dwell maneuver in order to comply with both regulations. In the case of FMVSS 126, the OEM must be confident that any vehicle sold in the USA would pass the specified procedure if tested. A combination of testing and simulation is done to gain this confidence. In the case of ECE R13H, the regulation specifically allows simulation to be used for some vehicles, if the simulation has been validated through comparison with physical test results.

Shortly after FMVSS 126 was published in 2007, Mechanical Simulation posted a tech memo and example run on the carsim.com web site for use with CarSim (the release at that time was version 7.01b). Since then, OEMs have gained experience using simulation to comply with both FMVSS

126 and ECE R13H, and CarSim has been further developed to provide more built-in programming to simplify the simulation of this test series.

This memo describes how the sine with dwell performance test can be simulated in CarSim, and is demonstrated with example procedures included in CarSim.

This memo assumes that you are familiar with the basic use of CarSim. At a minimum, you should have gone through the *CarSim Quick Start Guide* and you should have some familiarity with the way VS solvers work, as documented in the VS Solvers Manual. We also recommend that you first read another tech memo, *Making Advanced Procedures with VS Commands*, which introduces methods for using VS commands to automate sequences of tests. The examples in that memo involve a simpler type of test: steady-state circular turning used to obtain the understeer gradient for a vehicle. The circular testing method is related to ESC testing and provides a good introduction for the types of VS commands covered here.

The procedure described in this memo manipulates a normalized sine with dwell waveform using the capabilities of configurable functions that are used throughout CarSim. Details for this type of manipulation are provided in an appendix.

The detailed reference material for VS Commands is provided in the VS Commands Reference Manual.

Summary of the ESC Test

FMVSS 126 is described in an 88-page document that is part of the Federal Register, Vol. 72, No. 66, April 6, 2007. The standard itself (571.126) is presented in six pages (pp. 17310 – 17315); the description needed to simulate the test is presented in just a few paragraphs.

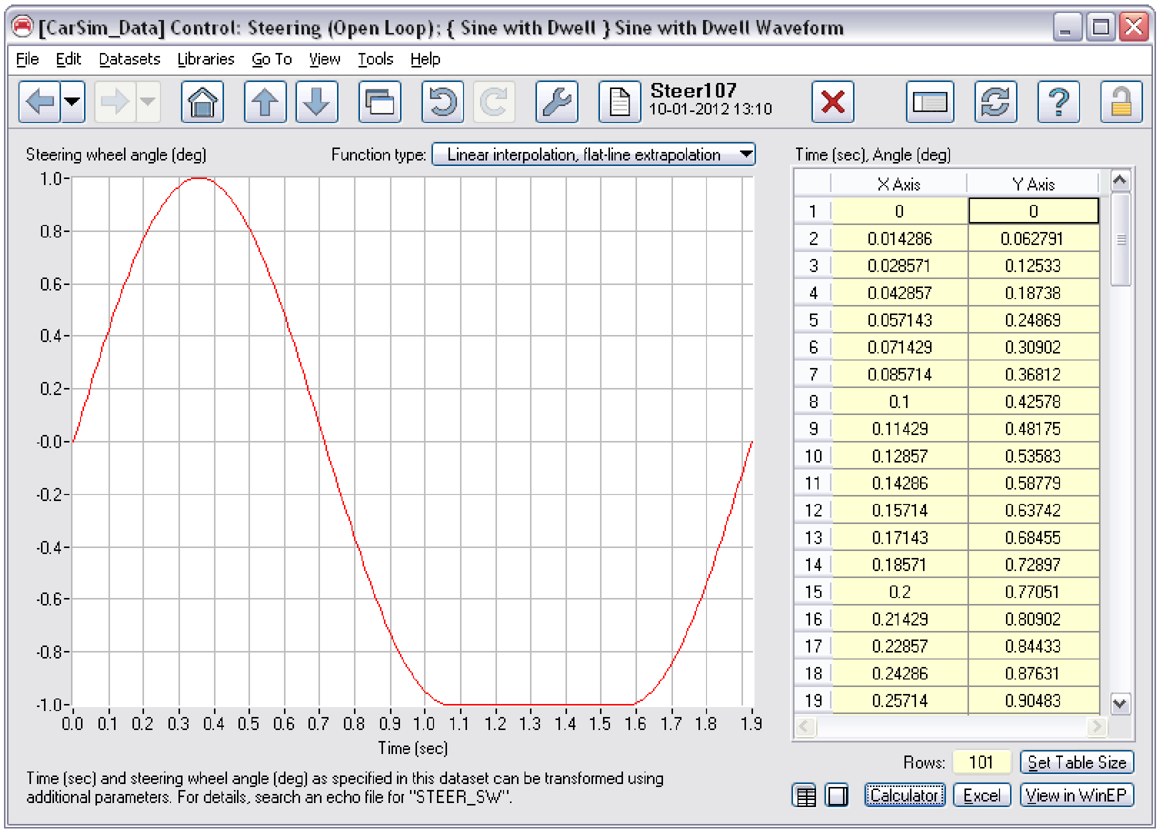
Note The FMVSS 126 test procedure includes many specifications that are not directly relevant to simulation and which are not covered in this memo. These are discussed in more detail in the last section of this memo.

Slowly Increasing Steer

The simulation test procedure begins with a “Slowly Increasing Steer Test” intended to determine the steering wheel angle *A* associated with a lateral acceleration of 0.3g for a speed of 80 km/h with the steering angle increasing at a rate of 13.5 deg/s. When performed via physical testing, three tests are repeated for counter-clockwise steering up to a lateral acceleration of 0.5 g; the steering angle at exactly 0.3 g’s of lateral acceleration is calculated using linear regression fro each test, and the three results are averaged. The sequence is repeated for clockwise steering, and the absolute angles obtained in the two directions are averaged to obtain *A*.

Sine with Dwell

A “sine with dwell” test involves bringing the vehicle to a speed slightly above 80 km/h with no steering or braking, letting it coast in the highest gear to 80 km/h, and then using a robot to apply a steering control with the shape shown in Figure 1. The shape is defined as a sinusoid with a frequency of 0.7 Hz that pauses for 0.5 seconds after reaching the second peak. Without the dwell, the period would be 1/0.700 = 1.429 seconds; with the 0.5 second dwell, the control ends at 1.929 seconds.


[CarSim_Data] Control: Steering (Open Loop): { Sine with Dwell } Sine with Dwell Waveform
File Edit Datasets Libraries Go To View Tools Help
Steer107
10-01-2012 13:10
Steering wheel angle (deg)
1.0 0.8 0.6 0.4 0.2 -0.0 -0.2 -0.4 -0.6 -0.8 -1.0
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9
Time (sec)
Set Table Size
Calculator Excel View in WinEP

Function type: Linear interpolation, flat-line extrapolation

Time (sec), Angle (deg)

	X Axis	Y Axis
1	0	0
2	0.014286	0.062791
3	0.028571	0.12533
4	0.042857	0.18738
5	0.057143	0.24869
6	0.071429	0.30902
7	0.085714	0.36812
8	0.1	0.42578
9	0.11429	0.48175
10	0.12857	0.53583
11	0.14286	0.58779
12	0.15714	0.63742
13	0.17143	0.68455
14	0.18571	0.72897
15	0.2	0.77051
16	0.21429	0.80902
17	0.22857	0.84433
18	0.24286	0.87631
19	0.25714	0.90483

Rows: 101

Time (sec) and steering wheel angle (deg) as specified in this dataset can be transformed using additional parameters. For details, search an echo file for "STEER_SW".

Figure 1. Sine with dwell waveform, with amplitude of 1°.

Two series of sine with dwell tests are done, one with initial counter-clockwise steering (positive steer, as shown in the figure), and the second with initial clockwise steering.

In each case, the amplitude is first set to 1.5*A*, where *A* is the reference amplitude obtained with the slowly increasing steer test.

Notes The amplitude shown in Figure 1 is 1°; this is done to simplify scaling the control to multiples of *A* as will be shown later.

Details for working with this dataset are provided in the appendix.

For example, Figure 2 shows plots of two variables for the two slowly increasing steer tests and three sine with dwell tests for a vehicle with the ESC disabled. When the lateral acceleration reached 0.3 g's, the steering wheel angle *A* was 39.9°. Therefore, the first sine with dwell test amplitude (1.5*A*) was 59.9°.

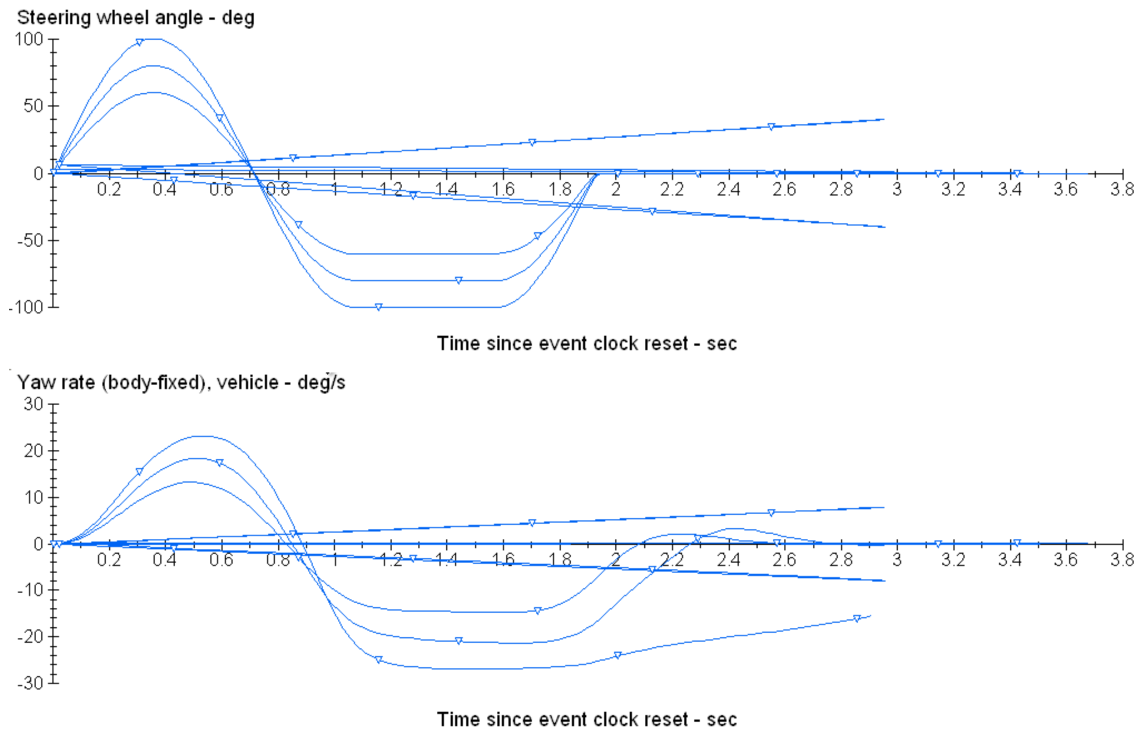


Figure 2. Vehicle response for sine with dwell without ESC.

The standard identifies several critical times for each test in the series. When the steering starts, this time is called the Beginning of Steer (BOS). When the waveform completes, the time is called the Completion of Steer (COS). The plots shown in Figure 2 use a local time in which BOS is always zero; therefore, COS is always 1.929. Four times are used to evaluate the vehicle response:

1. Starting after the first zero crossing of the steering wheel angle (0.714 seconds after BOS), the vehicle yaw rate is monitored for a peak value. For example, in the figure, the peak value for the last test is -27.0°/s at T = 1.457 after BOS.
2. For tests where the steering amplitude is 5.*0A* or more, at 1.07 seconds after BOS the lateral displacement of the vehicle mass center must be at least 1.83 m (6 ft) relative to the start of the test for vehicles with GVW of 3,500 kg or less. For vehicles with GVW greater than 3,500 kg, the required lateral displacement is 1.52 m (5 ft).
3. At 1.0 s after COS (2.929 seconds after BOS) the instant yaw rate must be less than 35% of the peak yaw rate; otherwise, the vehicle fails the test. In this example, 35% of the peak yaw rate for the third test is –9.45°; the yaw rate at 2.93 s is -15.2°/s, which means the vehicle fails the test.
4. At 1.75 s after COS (3.679 s after BOS) the instant yaw rate must be less than 20% of the peak yaw rate; otherwise, the vehicle fails the test.

If the vehicle passes a test, the steering amplitude is increased by 0.5*A* and another test is run.

The sequence is completed when the steering amplitude reaches 6.5*A* (11 tests) or 300°; the steering wheel angle for the last test is not allowed to be greater than 300°. If the amplitude reaches 6.5*A* and this amplitude is less than 270°, then the final test uses an angle of 270°. If the vehicle with ESC passes the last test with initial counter-clockwise (positive) steering, another series of sine with dwell tests is made with the initial clockwise (negative) steering. For example, Figure 3 shows the complete sequence for a vehicle with ESC that passes the test series.

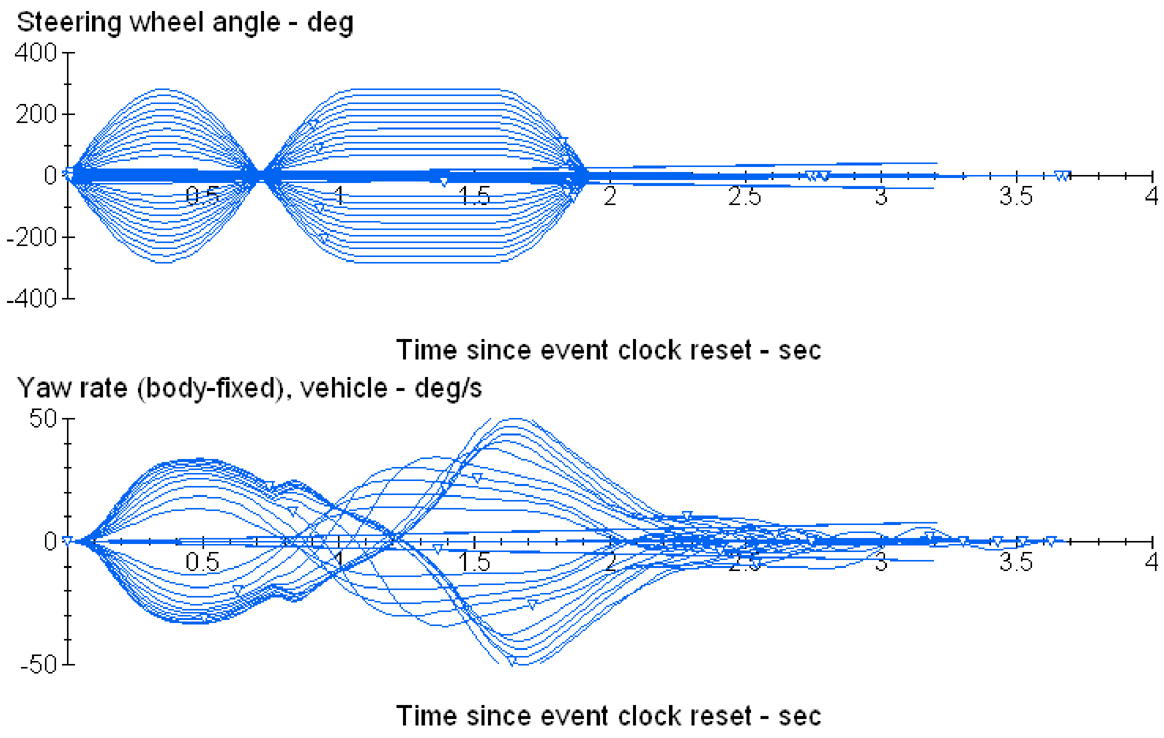


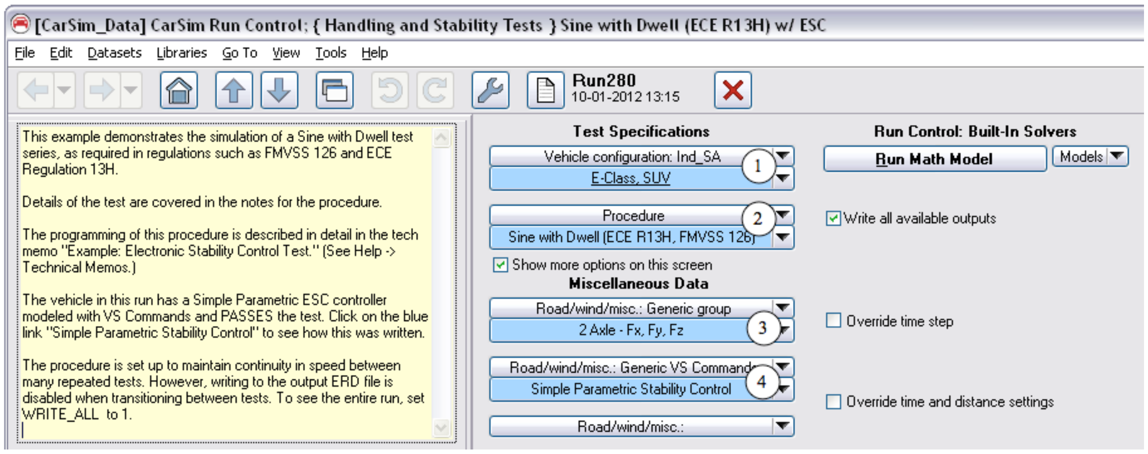
Figure 3. Response of vehicle with ESC for the full sequence.

Simulating the Sine with Dwell Procedure

With the capabilities of VehicleSim (VS) commands in CarSim, the test procedure described above can be performed in a single run. This section describes how this is done, using an example procedure provided in CarSim.

Several example runs in the category **Handling and Stability Tests** have names beginning with **Sine with Dwell (ECE R13H)**. The main example is the dataset **Sine with Dwell (ECE R13H) w/ ESC** (Figure 4).

The run is set up with a vehicle ①, a procedure ②, and an ESC controller model defined using VS commands ④. (This simple controller is described in a separate technical memo: *An Example Electronic Stability Controller*, available from the CarSim 8.2 **Help** menu.) A link is also included to show arrows for the tire forces ③ when the run is animated (Figure 5). In the animation figure, he tire force arrows show a significant longitudinal force at the right-front tire, due to the brake being actuated by the controller.



[CarSim_Data] CarSim Run Control: { Handling and Stability Tests } Sine with Dwell (ECE R13H) w/ ESC

File Edit Datasets Libraries Go To View Tools Help

Run280
10-01-2012 13:15

This example demonstrates the simulation of a Sine with Dwell test series, as required in regulations such as FMVSS 126 and ECE Regulation 13H.

Details of the test are covered in the notes for the procedure.

The programming of this procedure is described in detail in the tech memo "Example: Electronic Stability Control Test." (See Help -> Technical Memos.)

The vehicle in this run has a Simple Parametric ESC controller modeled with VS Commands and PASSES the test. Click on the blue link "Simple Parametric Stability Control" to see how this was written.

The procedure is set up to maintain continuity in speed between many repeated tests. However, writing to the output ERD file is disabled when transitioning between tests. To see the entire run, set WRITE_ALL to 1.

Test Specifications

Vehicle configuration: Ind_SA
E-Class, SUV ①

Procedure
Sine with Dwell (ECE R13H, FMVSS 126) ②

[✓] Show more options on this screen

Miscellaneous Data

Road/wind/misc.: Generic group
2 Axle - Fx, Fy, Fz ③

Road/wind/misc.: Generic VS Commands
Simple Parametric Stability Control ④

Road/wind/misc.:

Run Control: Built-In Solvers

Run Math Model | Models

[✓] Write all available outputs

[ ] Override time step

[ ] Override time and distance settings

Figure 4. Run Control screen for a sine with dwell test using VS commands to add ESC.

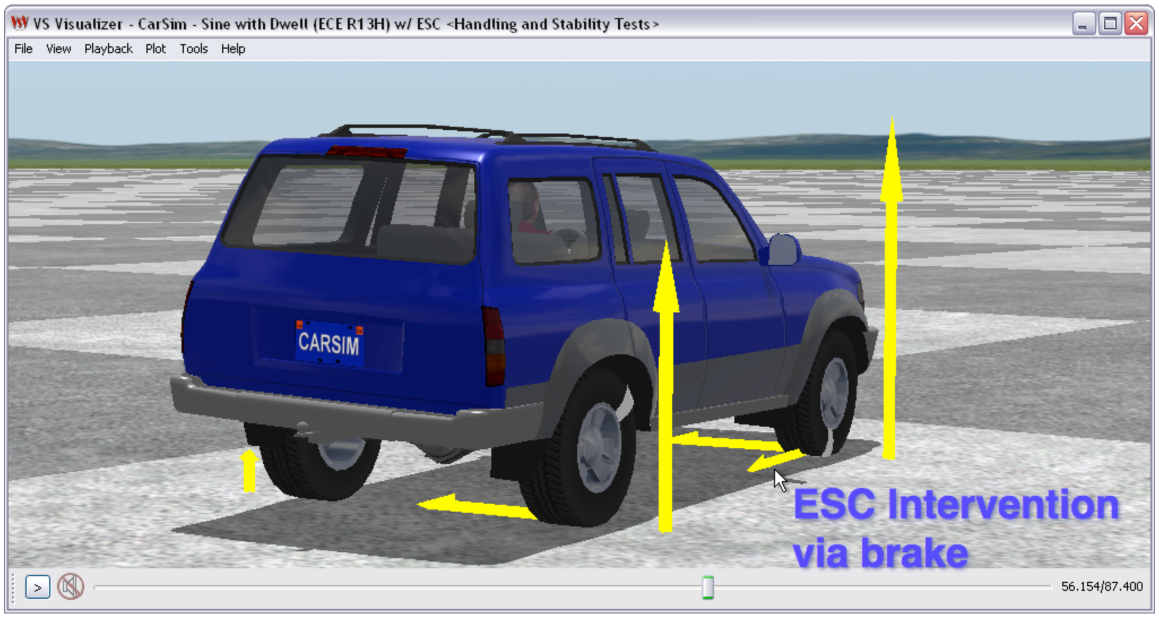
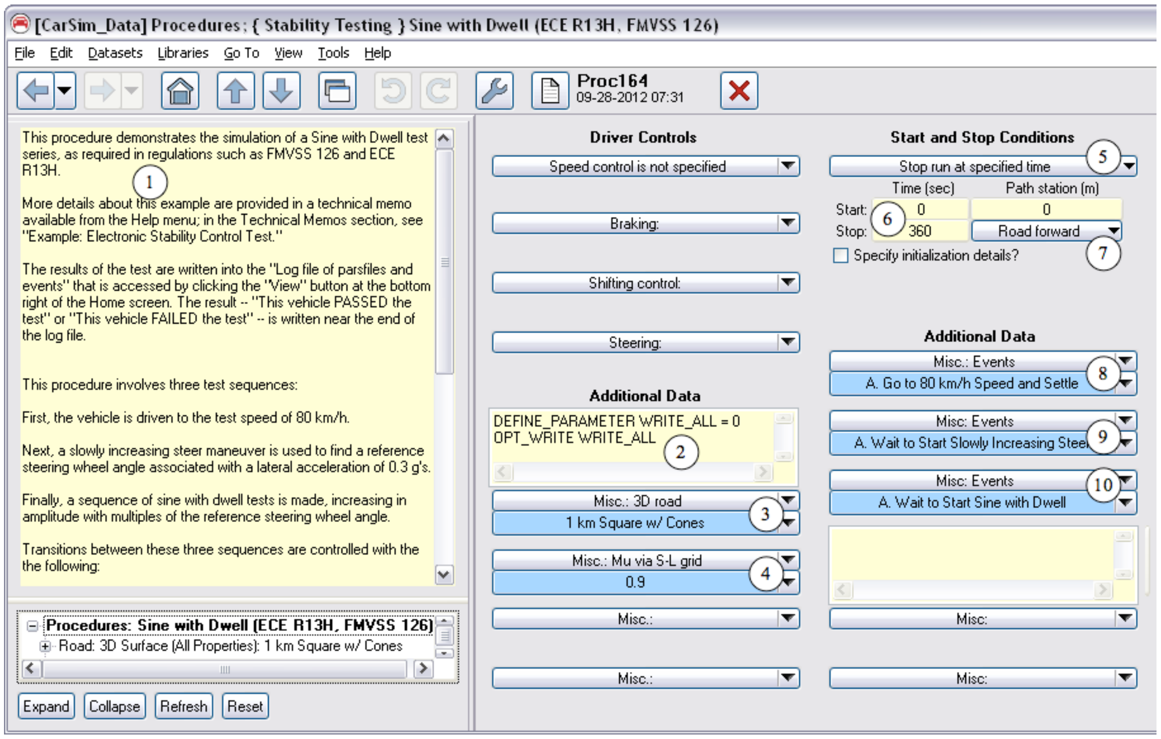

VS Visualizer - CarSim - Sine with Dwell (ECE R13H) w/ ESC <Handling and Stability Tests>
File View Playback Plot Tools Help
CARSIM
ESC Intervention via brake
56.154/87.400

Figure 5. Animation of the sine with dwell test when intervention occurs.

The ESC Testing Procedure

CarSim 8.2 includes two sine-with-dwell procedures. Figure 6 shows the one used for the example runs shown earlier, with some explanatory notes ①.

This procedure covers many changes in controls that are made as the run proceeds; therefore, none of the driver controls are set here. The tests take place on a flat, level surface ③ with a friction coefficient of 0.9 ④. CarSim supports two kinds of ground 3D descriptions: a grid in which Z is defined as a function of global X and Y, or a road, in which Z is defined as a function of station (distance along the road) and lateral distance from a road reference line. Both descriptions work OK for a flat level surface. A road dataset is used because it automatically provides a straight reference line that will be used during some parts of the sequence by the built-in CarSim driver model.


[CarSim_Data] Procedures: { Stability Testing } Sine with Dwell (ECE R13H, FMVSS 126)
File Edit Datasets Libraries Go To View Tools Help
Proc164
09-28-2012 07:31
Expand
Collapse
Refresh
Reset
1
2
3
4
5
6
7
8
9
10

This procedure demonstrates the simulation of a Sine with Dwell test series, as required in regulations such as FMVSS 126 and ECE R13H.

More details about this example are provided in a technical memo available from the Help menu; in the Technical Memos section, see "Example: Electronic Stability Control Test."

The results of the test are written into the "Log file of parsfiles and events" that is accessed by clicking the "View" button at the bottom right of the Home screen. The result -- "This vehicle PASSED the test" or "This vehicle FAILED the test" -- is written near the end of the log file.

This procedure involves three test sequences:

First, the vehicle is driven to the test speed of 80 km/h.

Next, a slowly increasing steer maneuver is used to find a reference steering wheel angle associated with a lateral acceleration of 0.3 g's.

Finally, a sequence of sine with dwell tests is made, increasing in amplitude with multiples of the reference steering wheel angle.

Transitions between these three sequences are controlled with the the following:

Procedures: Sine with Dwell (ECE R13H, FMVSS 126)
Road: 3D Surface (All Properties): 1 km Square w/ Cones

Driver Controls
Speed control is not specified
Braking:
Shifting control:
Steering:

Additional Data
```
DEFINE_PARAMETER WRITE_ALL = 0
OPT_WRITE WRITE_ALL
```
Misc.: 3D road
1 km Square w/ Cones
Misc.: Mu via S-L grid
0.9
Misc.:
Misc.:

Start and Stop Conditions
Stop run at specified time

	Time (sec)	Path station (m)
Start:	0	0
Stop:	360	Road forward

☐ Specify initialization details?

Additional Data
Misc.: Events
A. Go to 80 km/h Speed and Settle
Misc.: Events
A. Wait to Start Slowly Increasing Steer
Misc.: Events
A. Wait to Start Sine with Dwell
Misc.:
Misc.:

Figure 6. Procedure for the full series of sine with dwell tests, suitable for HIL simulation.

Because this procedure really involves a series of tests, the overall start and stop conditions are set to be as basic as possible, with a stop time set only as backup, in case the procedure doesn't terminate as intended (5, 6, 7).

Links are made to three sequences of events for this procedure. The first is used to bring the vehicle from rest up to the 80 km/h test speed 8, the second applies the slowly increasing steer tests 9 to establish the reference steering wheel angle *A*, and the third controls two series of sine with dwell tests 10. Three sets of events rather than one handle the procedure in order to provide modularity. For example, the slowly increasing steer sequence is used not only to set up sine with dwell tests, but also to set up fishhook tests for roll stability.

This procedure is set up to maintain continuity in speed throughout the series of tests. This is done to support hardware in the loop (HIL) systems in which a hardware ESC unit might get confused if any simulation shortcuts are taken (e.g., starting the run at the test speed of 80 km/h rather than at zero). The plots shown earlier show steering and yaw rate based on a clock that was reset with each test repeat. In contrast, Figure 7 shows portions of the overall sequence using absolute simulation time on the horizontal axis. Notice that much of the time is spent between tests. The extreme cornering that can occur during the sine with dwell test slows the vehicle considerably, and the speed must be brought up above 80 km/h for the next test.

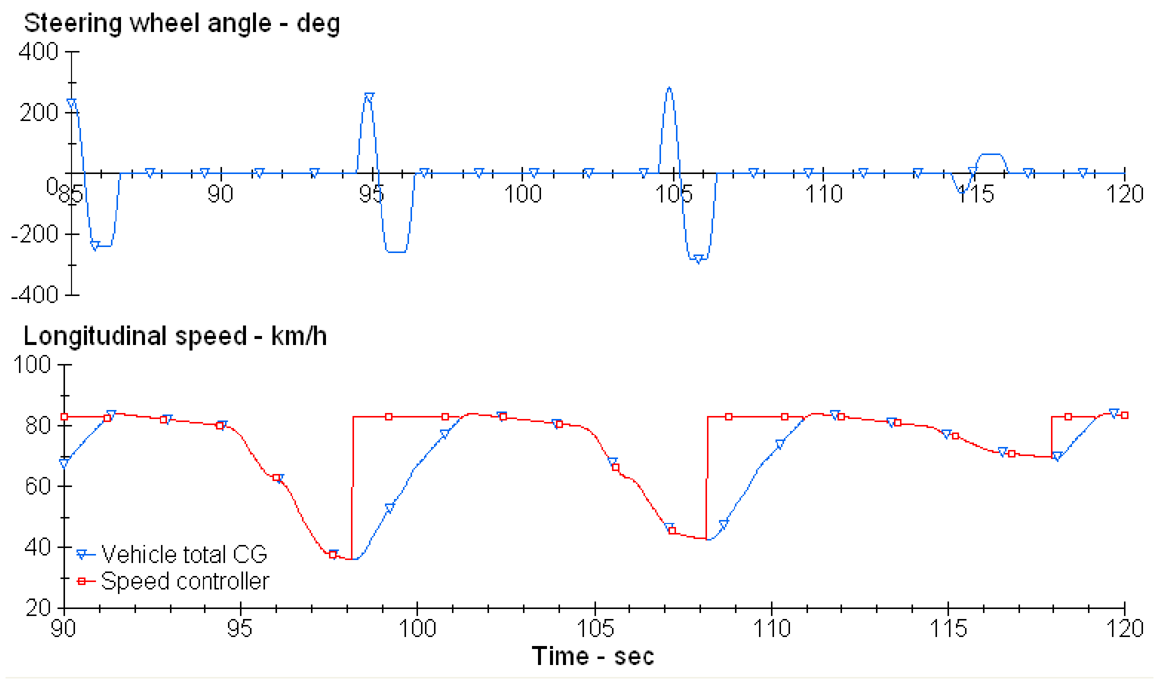


Figure 7. Time histories based on absolute simulation time.

Writing to file is normally controlled in these simulations such that the output files contain only the interesting parts of the vehicle motions. Writing to file is controlled by a system parameter `OPT_WRITE`, which is set initially to a new parameter `WRITE_ALL` created with the VS command `DEFINE_PARAMETER` ② (Figure 6). The new parameter `WRITE_ALL` is given a value of zero; hence, when it is used to set the value of `OPT_WRITE`, the effect is to turn off writing to file when this dataset is loaded. This is done to avoid writing data to file during the transitions between tests (Figure 8). Comparing Figure 7 and Figure 8, we see that the first sequence of sine with dwell tests that finishes about 110 s into the run (Figure 7) finishes at about 46 s into the recording where transitions are skipped.

If you want to see the entire simulation time history (211.4 s), you can set the value of `WRITE_ALL` to a non-zero value. (This was done to generate the plots shown in Figure 7.)

Note As noted earlier, the tech memo *Making Advanced Procedures with VS Commands* introduces and explains most of the techniques for using VS commands that are used in the sine with dwell procedures. This introductory tech memo is available from the **Help** menu.

The First Series of Events: Going to the Test Speed

Click on the blue link for the **Events** dataset ⑧ (Figure 6) that brings the vehicle up to the test speed. Figure 9 shows this dataset. It is one of three **Events** datasets in the category: **Go to Constant Speed**.

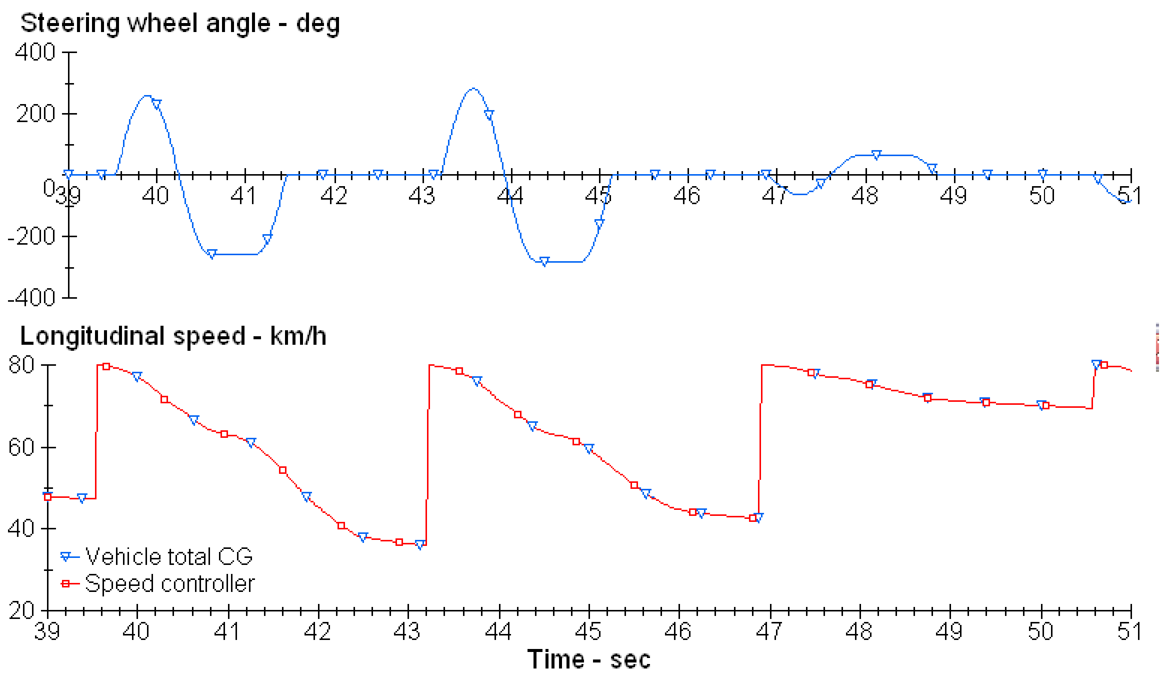
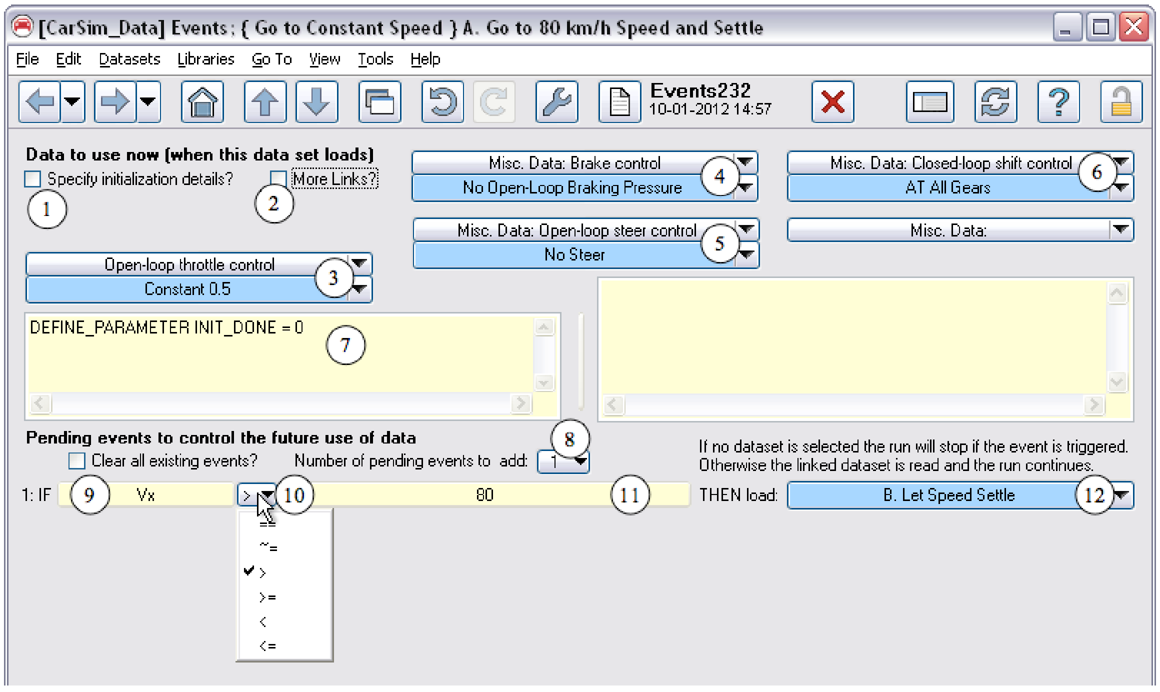


Figure 8. Time histories when writing to file is controlled to skip transitions.


[CarSim_Data] Events; { Go to Constant Speed } A. Go to 80 km/h Speed and Settle

Events232
10-01-2012 14:57

Data to use now (when this data set loads)
Specify initialization details?
More Links?

Misc. Data: Brake control: No Open-Loop Braking Pressure
Misc. Data: Closed-loop shift control: AT All Gears
Misc. Data: Open-loop steer control: No Steer
Misc. Data:
Open-loop throttle control: Constant 0.5

```
DEFINE_PARAMETER INIT_DONE = 0
```

Pending events to control the future use of data
Clear all existing events?
Number of pending events to add: 1

If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

1: IF Vx > 80 THEN load: B. Let Speed Settle

Figure 9. Events dataset to accelerate vehicle up to test speed.

This dataset is read along with the **Procedures** dataset, before the simulation starts and before the math model initializes. It sets up the main control settings that will be used at the start of the run:

- The speed control mode is set to open loop and the throttle is set to half (3), under the assumption that this will be sufficient to accelerate any CarSim vehicle from rest to a target speed of 80 km/h on a high friction surface in a reasonable manner.

- When the speed control was set to open loop, any closed-loop braking was disabled. Open loop braking is also set to zero ④.
- Steer control is set to open-loop, with the steering wheel angle set to zero ⑤.
- Shifting is performed with a closed-loop controller that will use the shift schedules for the vehicle, using all gears available in the vehicle ⑥.

The **Events** screen has options for controlling initialization; in most cases, including this one, the checkbox to enable these options is not checked ①. Another option is to show more links ②. When not checked (as in the figure) the miscellaneous fields available for providing VS commands (e.g., ⑦) are given more space.

The main distinction of this screen is that it can automatically generate VS commands to add pending events. In this case, one pending event is added ⑧. A pending event is defined with a *Boolean* expression (a logical expression that must have a value of true or false) of the form:

variable operator threshold

where *variable* is any parameter or variable in the math model that can be identified with a keyword (e.g., vehicle forward speed, `VX` ⑨), *operator* is a comparison operator available from a pull-down list ⑩, and *threshold* is any expression that can be evaluated by the VS solver (e.g., `80` ⑪). The *threshold* can be a number, a symbol, or an algebraic expression.

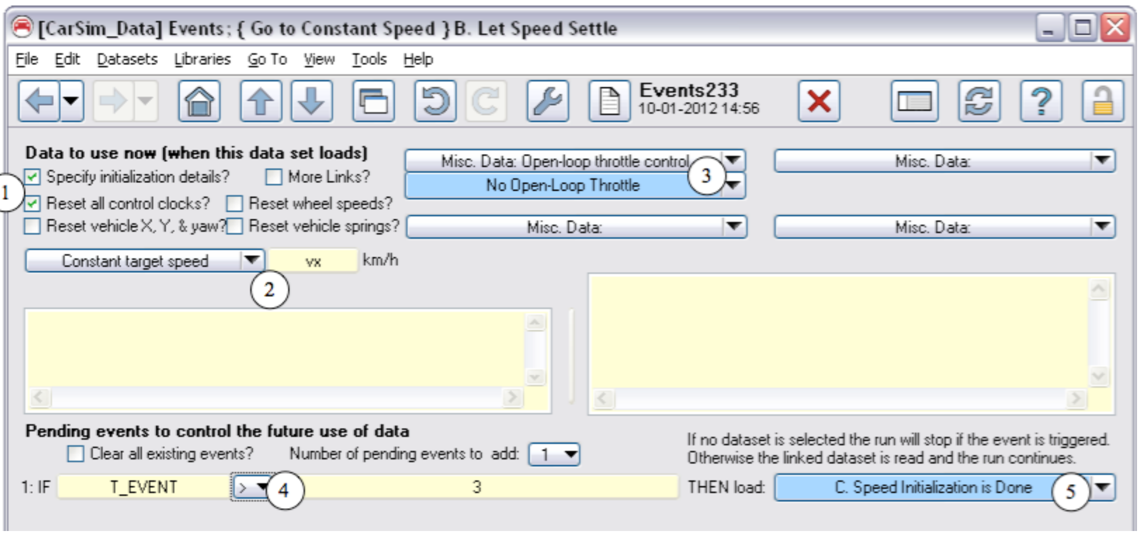
As the simulation runs, all pending events are evaluated each time step. The *threshold* expression is updated if necessary and compared to the current value of *variable*. (In this example, the threshold is simply a number; however, in other examples the threshold can be a formula that is applied before the comparison is made.) If the Boolean expression is false, no action is taken. However, if the Boolean expression is true, then the event is said to be *triggered* and one of two possible actions are taken:

1. If another dataset is specified (e.g., **B. Let Speed Settle** ⑫), then that dataset is read by the VS solver and processed, and the run continues using any changes in the vehicle properties or test conditions that are specified in the new dataset.
2. On the other hand, if there is no linked dataset associated with the pending event, then the simulation run is immediately terminated.

If there is a linked dataset, the VS solver removes the event that was just triggered from the list of pending events. Any other pending events remain pending by default. It is possible to clear all pending events, as will be seen later.

In addition to the built-in options selected on this screen, the VS command `DEFINE_PARAMETER` is used to add a new parameter `INIT_DONE` and assign it a value of zero. When the vehicle has finished this series of events, the parameter `INIT_DONE` will be given a non-zero value to indicate that this part of the simulation is done. It will be seen shortly that another test sequence (the slowly increasing steer) has a pending event waiting for this condition; when that event is triggered the next part of the procedure will become active.

When the vehicle speed exceeds 80 km/h, the VS solver will read the dataset file with the title **B. Let Speed Settle** ⑫, shown in Figure 10.


[CarSim_Data] Events ; { Go to Constant Speed } B. Let Speed Settle

Events233
10-01-2012 14:56

Data to use now (when this data set loads)

[✓] Specify initialization details? [ ] More Links?
[✓] Reset all control clocks? [ ] Reset wheel speeds?
[ ] Reset vehicle X, Y, & yaw? [ ] Reset vehicle springs?

Constant target speed: vx km/h

Misc. Data: Open-loop throttle control — No Open-Loop Throttle

Misc. Data:

Misc. Data:

Misc. Data:

Pending events to control the future use of data

[ ] Clear all existing events? Number of pending events to add: 1

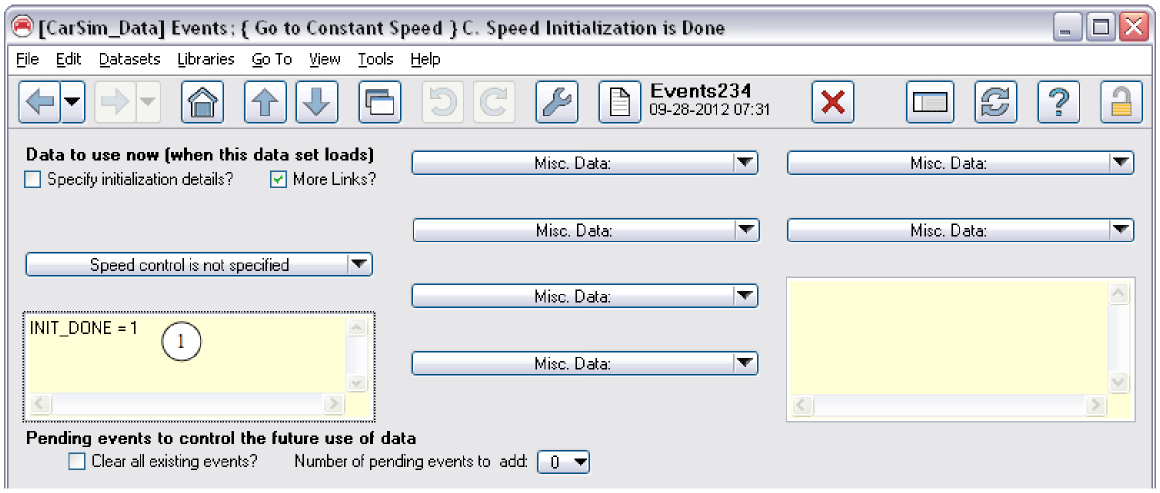
If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

1: IF T_EVENT > 3 THEN load: C. Speed Initialization is Done

Figure 10. Events dataset to let speed settle.

When this event dataset is loaded, the speed control option is changed from open loop to closed loop, with the target speed being set to the current vehicle speed (2). Open-loop throttle (previous set to 0.5) is turned off (3).

The checkbox to specify initialization details is checked (1), revealing more checkboxes related to details of initialization. In this case, the box is checked to reset all control clocks and the others are unchecked, indicating that the math model should stay "as is" regarding position, wheel speeds, and suspension springs. When the control clocks are reset, a built-in time variable `T_EVENT` is reset to zero. This variable is used to define a new pending event that will be triggered when it exceeds 3 s (4). This is intended to let the vehicle settle at the current speed. When three seconds have passed, the next **Events** dataset will be loaded: **C: Speed Initialization is Done**. As shown in Figure 11, the only change made by reading this dataset is that the parameter `INIT_DONE` is given a new value of 1. (Recall that it was initially zero.)


[CarSim_Data] Events ; { Go to Constant Speed } C. Speed Initialization is Done

Events234
09-28-2012 07:31

Data to use now (when this data set loads)

[ ] Specify initialization details? [✓] More Links?

Speed control is not specified

INIT_DONE = 1

Misc. Data:

Misc. Data:

Misc. Data:

Misc. Data:

Misc. Data:

Misc. Data:

Pending events to control the future use of data

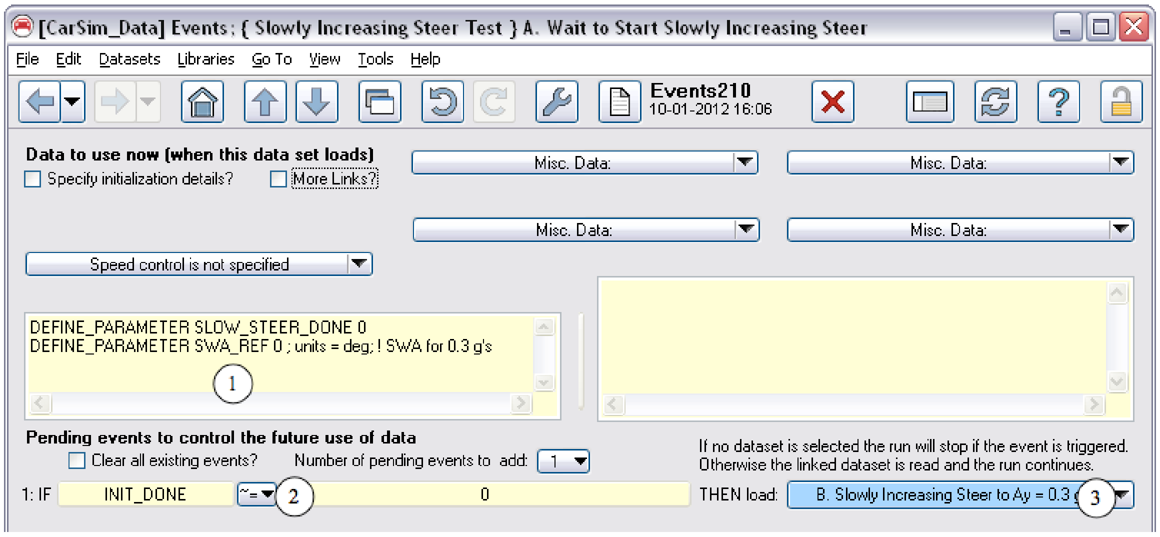
[ ] Clear all existing events? Number of pending events to add: 0

Figure 11, Events dataset that indicates the speed initialization is complete.

Because no pending events remain for this series, and no new events are defined on this screen, this dataset concludes the series involving speed initialization.

The Second Series of Events: Slowly Increasing Steer

Recall that the **Procedure** dataset (Figure 6, page 7) had links to three **Events** datasets. The second, with the title **A. Wait to Start Slowly Increasing Steer**, is shown in Figure 12.


[CarSim_Data] Events: { Slowly Increasing Steer Test } A. Wait to Start Slowly Increasing Steer

Events210
10-01-2012 16:06

Data to use now (when this data set loads)
Specify initialization details? ☐
More Links? ☐
Speed control is not specified

```
DEFINE_PARAMETER SLOW_STEER_DONE 0
DEFINE_PARAMETER SWA_REF 0 ; units = deg; ! SWA for 0.3 g's
```

Pending events to control the future use of data
Clear all existing events? ☐
Number of pending events to add: 1
If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.
1: IF INIT_DONE ~= 0 THEN load: B. Slowly Increasing Steer to Ay = 0.3 g

Figure 12. Events dataset to wait to start the slowly increasing steer tests.

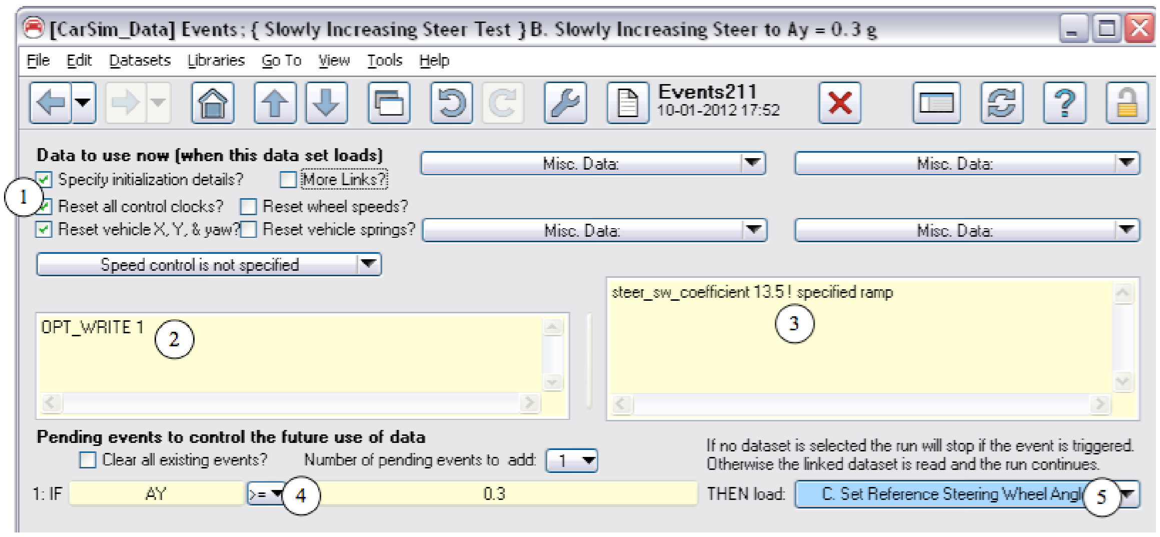
This dataset defines an event based on the parameter `INIT_DONE`, which was created and updated in the previous series. The pending event checks for the condition that `INIT_DONE` is not equal to zero (2); when this happens, the dataset **B: Slowly Increasing Steer to Ay = 0.3 g's** is loaded (3). In order for this event to be valid, the new parameter `INIT_DONE` must already exist. This is achieved by making sure the link to the **Events** dataset that defines `INIT_DONE` appears before the link to this dataset on the **Procedures** screen (Figure 6).

This dataset also uses the VS command `DEFINE_PARAMETER` to define two new parameters: `SLOW_STEER_DONE` and `SWA_REF` (1). The `SLOW_STEER_DONE` parameter will be used to indicate when this series of events is complete. (Once defined, it can be referenced in any datasets that are loaded after this one.)

When the previous sequence is complete and `INIT_DONE` is no longer equal to zero, the event is triggered and the dataset shown in Figure 13 is loaded.

Recall that the VS solver was instructed via the parameter `OPT_WRITE` to not write any data to file at the start of the run. At the time that this dataset (Figure 13) is loaded, the vehicle has settled into the proper speed and is ready to start the slowly increasing steer. This is the first instant in the simulation where the behavior might be of interest for animation and plotting. Therefore, the `OPT_WRITE` parameter is set to 1 (2), instructing the VS solver to start writing to file.

The checkbox to specify initialization details is checked (1), as are two related checkboxes: the option to reset all clocks, and the option to reset the vehicle position (X, Y, and Yaw). One of the clocks is used to define open-loop steering wheel angle. Because the clock now starts at zero, the slowly increasing steer can be defined in CarSim simply with a linear coefficient multiplied by the steering clock time. The coefficient (`STEER_SW_COEFFICIENT`) is given the value 13.5 °/s ③ as specified in the standards.



[CarSim_Data] Events : { Slowly Increasing Steer Test } B. Slowly Increasing Steer to Ay = 0.3 g

Events211
10-01-2012 17:52

Data to use now (when this data set loads)
☑ Specify initialization details? ☐ More Links?
☑ Reset all control clocks? ☐ Reset wheel speeds?
☑ Reset vehicle X, Y, & yaw? ☐ Reset vehicle springs?

Misc. Data:
Misc. Data:
Misc. Data:
Misc. Data:

Speed control is not specified

```
OPT_WRITE 1
```

```
steer_sw_coefficient 13.5 ! specified ramp
```

Pending events to control the future use of data
☐ Clear all existing events? Number of pending events to add: 1

If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

1: IF AY >= 0.3 THEN load: C. Set Reference Steering Wheel Angl

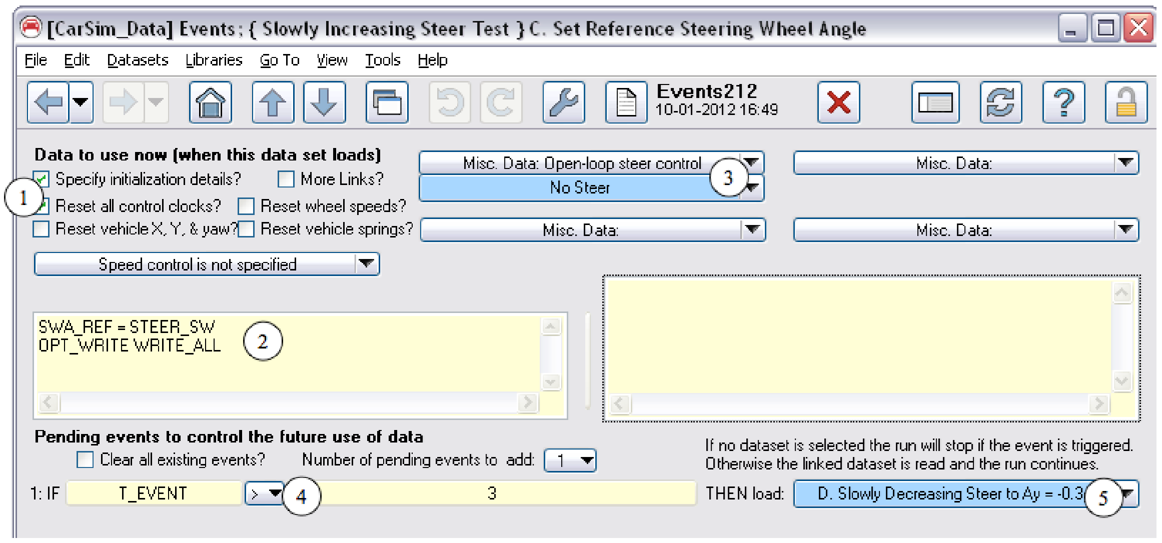
Figure 13. Events dataset to start increasing steer slowly until Ay reaches 0.3 g's.

Steering and all other open-loop controls are represented in CarSim as VS configurable functions that can be constants, linear coefficients, tables, or equations. The calculation method is identified by a keyword that identifies the configurable function and indicates the type of calculation. The root name for the keyword is found by looking at the screen for setting steering as a function of time (visit this screen using the **Libraries** menu, e.g., Figure 1 on page 3). In this case, the root name is `STEER_SW`. The calculation method is specified with a suffix: `STEER_SW_CONSTANT` indicates steering is a constant, `STEER_SW_COEFFICIENT` indicates steering is calculated by multiplying a coefficient with time, `STEER_SW_TABLE` indicates steering is calculated using a table of numbers and a specified interpolation method, and `STEER_SW_EQUATION` indicates steering is calculated using a formula provided at runtime. Details on VS configurable functions are provided from the **Help** menu in the documents **Driver Controls** and **VS Solver Programs**.

The ESC standards require this test to continue to a lateral acceleration level of 0.5 g's and that the steering corresponding to 0.3 g's be calculated using linear regression. However, in a perfectly repeatable simulation environment, only a single test is needed up to a limit of 0.3 g's. Because the simulation results do not include measurement noise, the steering wheel angle that exists at the millisecond where absolute lateral acceleration reaches 0.3 g's is simply saved.

A pending event is defined to check the lateral acceleration ④. When it reaches 0.3 g's, the next **Events** dataset will be loaded ⑤. Figure 14 shows the dataset that will be loaded when the acceleration limit is reached.

At the instant that this dataset is loaded (the lateral acceleration has reached 0.3 g's) the steering wheel angle (`STEER_SW`) defines the reference *A* needed later for the sine with dwell tests. Hence, the current value is saved using the parameter `SWA_REF` that was defined earlier (②, Figure 12).


[CarSim_Data] Events: { Slowly Increasing Steer Test } C. Set Reference Steering Wheel Angle
File Edit Datasets Libraries Go To View Tools Help
Events212
10-01-2012 16:49
1
2
3
4
5

Data to use now (when this data set loads)
[✓] Specify initialization details? ☐ More Links?
[✓] Reset all control clocks? ☐ Reset wheel speeds?
☐ Reset vehicle X, Y, & yaw? ☐ Reset vehicle springs?
Speed control is not specified
Misc. Data: Open-loop steer control: No Steer

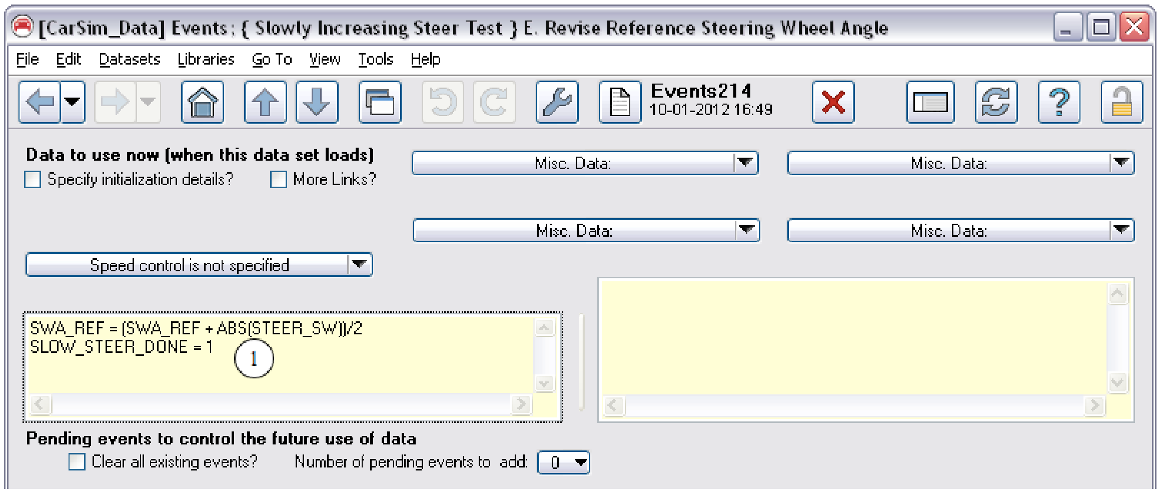
```
SWA_REF = STEER_SW
OPT_WRITE WRITE_ALL
```

Pending events to control the future use of data
☐ Clear all existing events? Number of pending events to add: 1
If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.
1: IF T_EVENT > 3 THEN load: D. Slowly Decreasing Steer to Ay = -0.3

Figure 14. Events dataset to set the reference steer angle and settle before the next test.

The next test in the standard is a repeat with the opposite direction (slowly decreasing steer). To prepare for the next test, the steering is set to zero (3), the local clocks are reset (1), and a pending event is added to wait 3 s for the vehicle to settle (4). When the time has elapsed, a new dataset is loaded to slowly decrease steer until the lateral acceleration reaches a value of -0.3 g's (5). During this settling period, writing to file is once again disabled by setting the system parameter `OPT_WRITE` to the parameter `WRITE_ALL` (2).

The next dataset, **D. Slowly Decreasing Steer to Ay = -0.3 g's**, is nearly identical to the dataset for slowly increasing steer shown earlier (Figure 13). The only differences are that the sign for the steering coefficient and lateral acceleration are negative, and that the next dataset to load in the sequence is **E. Revise Reference Steering Wheel Angle** (Figure 15).


[CarSim_Data] Events: { Slowly Increasing Steer Test } E. Revise Reference Steering Wheel Angle
File Edit Datasets Libraries Go To View Tools Help
Events214
10-01-2012 16:49
1

Data to use now (when this data set loads)
☐ Specify initialization details? ☐ More Links?
Speed control is not specified

```
SWA_REF = (SWA_REF + ABS(STEER_SW))/2
SLOW_STEER_DONE = 1
```

Pending events to control the future use of data
☐ Clear all existing events? Number of pending events to add: 0

Figure 15. Events dataset to conclude the slowly increasing steer tests.

When loaded, the vehicle has just reached the lateral acceleration level of -0.3 g's. The reference steering wheel angle needed for the sine with dwell tests is set to the average of the saved value and the absolute value of the current steering (1).

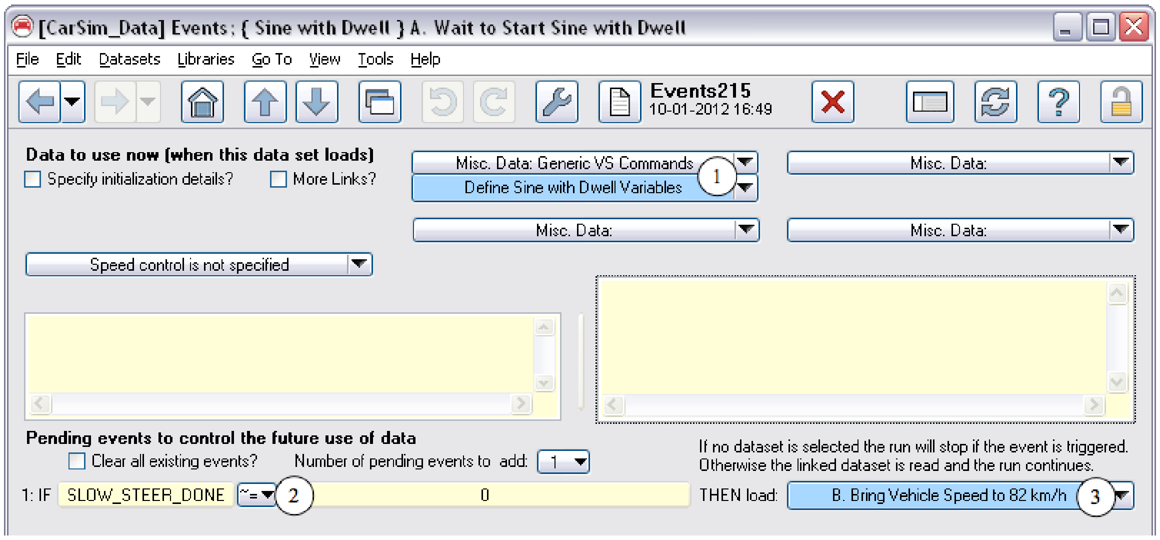
Because no pending events remain for this series, and no new events are defined on this screen, this dataset concludes the series involving slowly increasing speed.

The Third Series of Events: Sine with Dwell Steering

The third link to an **Events** dataset from the **Procedure** dataset (Figure 6, page 7) loads the dataset with the title **A. Wait to Start Sine with Dwell**.

Setting up the First Sine with Dwell Test Series

Figure 16 shows the first dataset in the series. As with other datasets linked to the **Procedures** dataset, it is loaded before the simulation run starts. Any output variables that will be written to file (e.g., for plotting) must be defined before the output files are created when the run starts. Several new outputs are defined for this test, and the definitions are all provided in a linked dataset ① using the **Generic: VS Commands** screen.


[CarSim_Data] Events: { Sine with Dwell } A. Wait to Start Sine with Dwell

Events215
10-01-2012 16:49

Data to use now (when this data set loads)
☐ Specify initialization details? ☐ More Links?

Misc. Data: Generic VS Commands: Define Sine with Dwell Variables ①
Misc. Data:
Misc. Data:
Misc. Data:

Speed control is not specified

Pending events to control the future use of data
☐ Clear all existing events? Number of pending events to add: 1

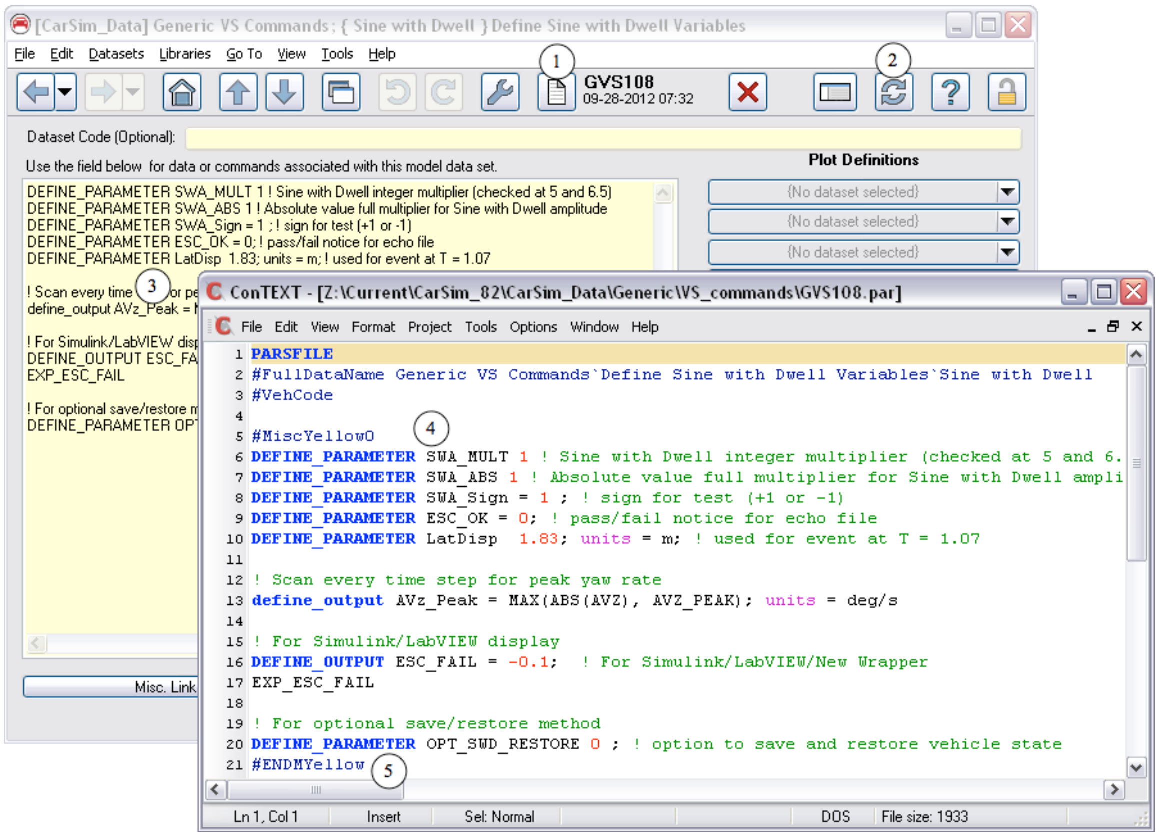
1: IF SLOW_STEER_DONE ~= ② 0 THEN load: B. Bring Vehicle Speed to 82 km/h ③

If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

Figure 16. Events dataset to set up sine with dwell series.

The **Events** dataset also sets up an event based on the parameter `SLOW_STEER_DONE`. The pending event checks for the condition that `SLOW_STEER_DONE` is not equal to zero ②; when this happens, the previous series has been completed and the sine with dwell series is started by loading the next dataset ③.

Figure 17 show the **Generic VS Commands** dataset used to define new parameters and output variables that are unique for the sine with dwell series of tests. This is the preferred screen for locating multiple VS Commands because the yellow field is large ③; however, it can be easier to read and edit using a text editor by clicking the **Parsfile** button ①. The associated text file (in this case, with the name `GVS108.par`) has the information shown on the screen plus additional information used to manage the database. Notice that the text is color-coded, with VS commands such as `DEFINE_PARAMETER` and `DEFINE_OUTPUT` shown in bold color.


[CarSim_Data] Generic VS Commands ; { Sine with Dwell } Define Sine with Dwell Variables
File Edit Datasets Libraries Go To View Tools Help
1
2
GVS108
09-28-2012 07:32
Dataset Code (Optional):
Use the field below for data or commands associated with this model data set.
Plot Definitions
{No dataset selected}
{No dataset selected}
{No dataset selected}
3
Misc. Link
ConTEXT - [Z:\Current\CarSim_82\CarSim_Data\Generic\VS_commands\GVS108.par]
File Edit View Format Project Tools Options Window Help
4
5
Ln 1, Col 1
Insert
Sel: Normal
DOS
File size: 1933

```
DEFINE_PARAMETER SWA_MULT 1 ! Sine with Dwell integer multiplier (checked at 5 and 6.5)
DEFINE_PARAMETER SWA_ABS 1 ! Absolute value full multiplier for Sine with Dwell amplitude
DEFINE_PARAMETER SWA_Sign = 1 ; ! sign for test (+1 or -1)
DEFINE_PARAMETER ESC_OK = 0; ! pass/fail notice for echo file
DEFINE_PARAMETER LatDisp  1.83; units = m; ! used for event at T = 1.07
```

```
PARSFILE
#FullDataName Generic VS Commands`Define Sine with Dwell Variables`Sine with Dwell
#VehCode

#MiscYellow0
DEFINE_PARAMETER SWA_MULT 1 ! Sine with Dwell integer multiplier (checked at 5 and 6.
DEFINE_PARAMETER SWA_ABS 1 ! Absolute value full multiplier for Sine with Dwell ampli
DEFINE_PARAMETER SWA_Sign = 1 ; ! sign for test (+1 or -1)
DEFINE_PARAMETER ESC_OK = 0; ! pass/fail notice for echo file
DEFINE_PARAMETER LatDisp  1.83; units = m; ! used for event at T = 1.07

! Scan every time step for peak yaw rate
define_output AVz_Peak = MAX(ABS(AVZ), AVZ_PEAK); units = deg/s

! For Simulink/LabVIEW display
DEFINE_OUTPUT ESC_FAIL = -0.1;  ! For Simulink/LabVIEW/New Wrapper
EXP_ESC_FAIL

! For optional save/restore method
DEFINE_PARAMETER OPT_SWD_RESTORE 0 ; ! option to save and restore vehicle state
#ENDMYellow
```

Figure 17. Defining variables with the Generic VS Commands screen.

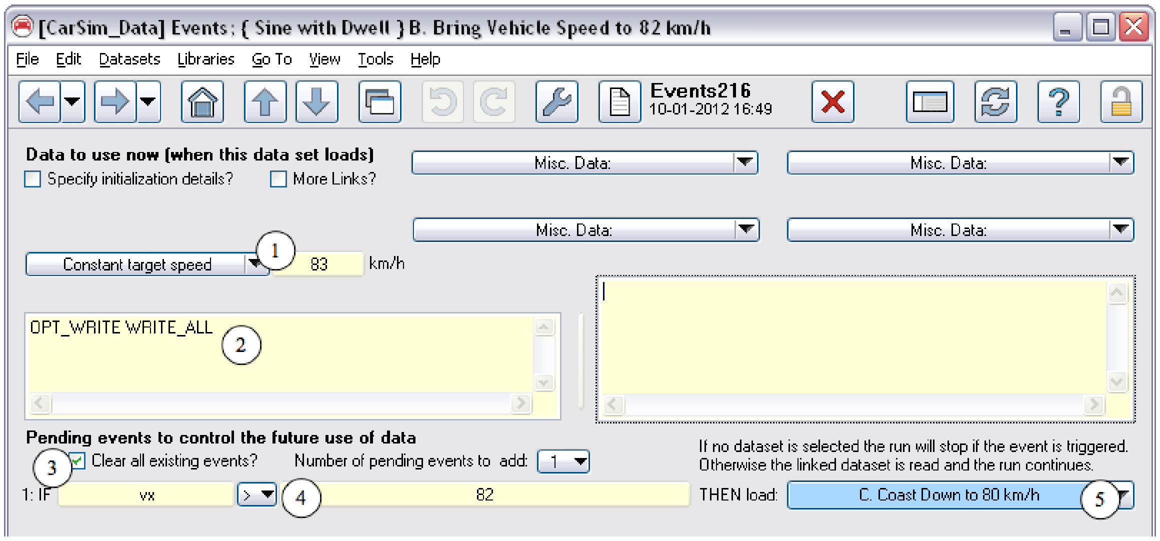
If you use a text editor, it is important to only edit the text that appears in the big yellow field ③. In the text file, the contents of the big yellow field start after a line that reads `#MiscYellow0` ④, and continue until a line `#ENDMYellow` ⑤. Be sure not to edit anything outside the lines that define the beginning and end of the field. Once done editing, save the file; otherwise, CarSim will not use the changes made. To see the changes made by editing the text file directly simply go back to the CarSim window. If by any chance you don't see the changes, click the **Refresh** button ② to force CarSim to read the parsfile again.

Two new outputs are defined using the VS command `DEFINE_OUTPUT`. One is the peak yaw rate `AVz_Peak`, used to evaluate the success or failure of the vehicle and ESC at two specific times in each sine with dwell test, as described earlier. The other is a variable `ESC_FAIL` that can be exported to Simulink or other external software to indicate whether the vehicle has passed or failed the test. (It is not needed for the Windows-only examples, but is handy when this example is extended in driving simulators and HIL setups.)

The other new variables are defined with the `DEFINE_PARAMETER` command, and are used to scale the sine with dwell control at the series progresses. One of these, `SWA_MULT`, is a multiplier that is given values of 1.5 to 6.5 in increments of 0.5. It is given an initial value here of 1, so that when it is incremented by 0.5 it will have the correct value for the first sine with dwell maneuver.

Setting Up the Vehicle Initial Conditions for a Sine with Dwell Test

Once the slowly increasing steer tests are done, the parameter `SLOW_STEER_DONE` is set to one, the pending event (Figure 16) triggers, and the sine with dwell testing begins with the loading of the data shown in Figure 18.


[CarSim_Data] Events ; { Sine with Dwell } B. Bring Vehicle Speed to 82 km/h

Events216
10-01-2012 16:49

Data to use now (when this data set loads)
Specify initialization details? ☐
More Links? ☐

Constant target speed: 83 km/h

OPT_WRITE WRITE_ALL

Pending events to control the future use of data
Clear all existing events? ✓
Number of pending events to add: 1

If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

1: IF vx > 82 THEN load: C. Coast Down to 80 km/h

Figure 18. Events dataset to start the sine with dwell testing.

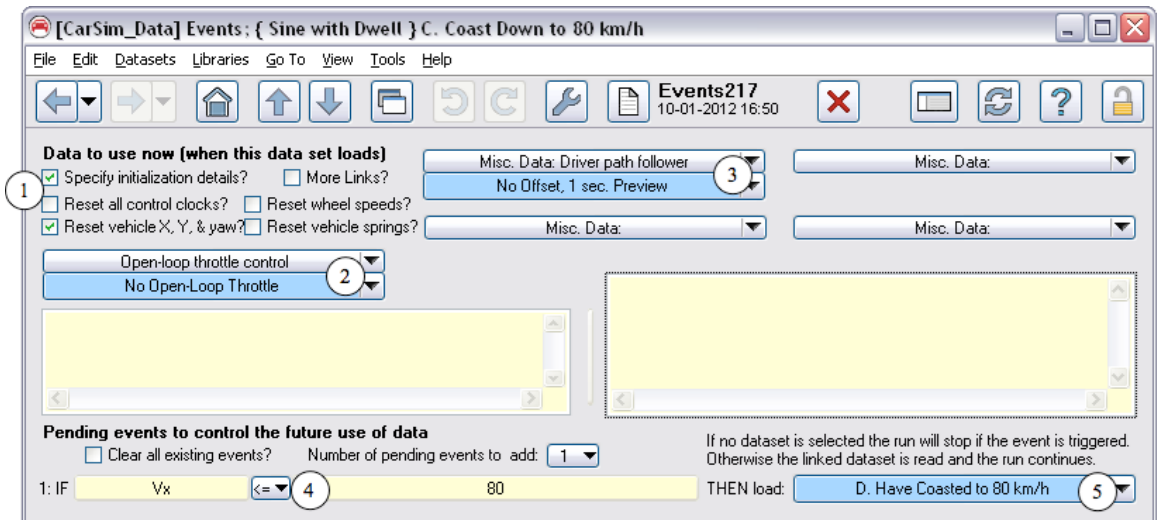
This dataset has four effects. First, the closed-loop speed controller is enabled with a target speed of 83 km/h to bring the speed up ①. Second, writing to file is disabled for this speed transition by setting `OPT_WRITE` to the parameter `WRITE_ALL` ②. Third, any pending events that existed when this dataset is loaded are cleared ③. Fourth, a pending event is defined to trigger when the speed reaches 82 km/h ④. When this speed is reached, a dataset is loaded to begin the coast down specified in the sine with dwell test ⑤.

Note At this time, it might not be clear why the box is checked to clear all pending events ③. The reason is that this dataset will be loaded many times during a run, to set up for each sine with dwell test. Some of these times when it is loaded later, there will be multiple pending events that must be cleared.

Once the speed has increased to 82 km/h the dataset shown in Figure 19 is loaded.

Now that the speed has reached 82 km/h, this dataset changes to open-loop speed control with no throttle to simulate a coast down ②. It also activates closed-loop steer control ③ to have the vehicle follow the road reference line (the road reference line follows the global X axis). To avoid unwanted transient steering due to the driver model adjusting the vehicle yaw, some checkboxes are selected to perform initialization when this dataset is loaded ①. The vehicle is repositioned to have the global X and Y coordinates reset to zero, along with the yaw angle. Thus, the vehicle is perfectly positioned when the closed-loop steering controller is engaged.

Although the vehicle position is reset, other variables are not: it retains roll, pitch, height, and all velocity properties that might be sensed by the ESC.



[CarSim_Data] Events ; { Sine with Dwell } C. Coast Down to 80 km/h

Events217
10-01-2012 16:50

Data to use now (when this data set loads)
1
[✓] Specify initialization details? [ ] More Links?
[ ] Reset all control clocks? [ ] Reset wheel speeds?
[✓] Reset vehicle X, Y, & yaw? [ ] Reset vehicle springs?

Misc. Data: Driver path follower — No Offset, 1 sec. Preview 3
Misc. Data:
Misc. Data:
Misc. Data:

Open-loop throttle control — No Open-Loop Throttle 2

Pending events to control the future use of data
[ ] Clear all existing events? Number of pending events to add: 1

If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

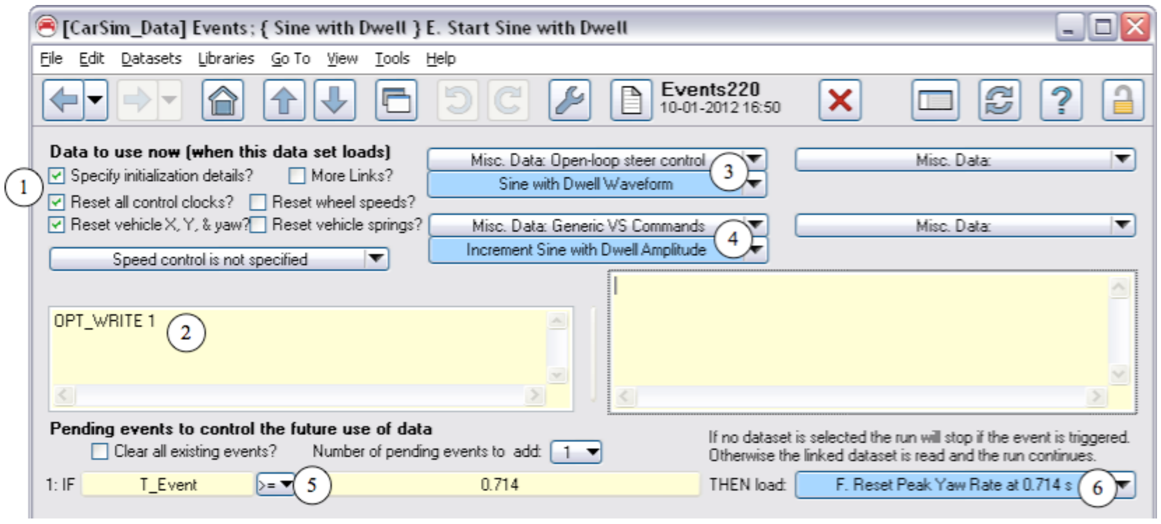
1: IF Vx <= 4 80 THEN load: D. Have Coasted to 80 km/h 5

Figure 19. Events dataset to coast down at start of sine with dwell test.

Note The example ESC included in CarSim has very simple logic. However, the procedure is intended to work with more sophisticated ESC algorithms, including hardware that might check for potential errors in the signals such as a discontinuous change in speed.

Running a Sine with Dwell Test

When the vehicle has coasted down to 80 km/h, the dataset **D. Have Coasted to 80 m/h** is loaded ⑤. That dataset in turn defines two pending events, one of which will supports an advanced option that will be described later and the other (the default) that immediately loads the dataset **E. Start Sine with Dwell**, shown in Figure 20. This is the main dataset that sets up a sine with dwell test.



[CarSim_Data] Events ; { Sine with Dwell } E. Start Sine with Dwell

Events220
10-01-2012 16:50

Data to use now (when this data set loads)
1
[✓] Specify initialization details? [ ] More Links?
[✓] Reset all control clocks? [ ] Reset wheel speeds?
[✓] Reset vehicle X, Y, & yaw? [ ] Reset vehicle springs?

Misc. Data: Open-loop steer control — Sine with Dwell Waveform 3
Misc. Data: Generic VS Commands — Increment Sine with Dwell Amplitude 4
Misc. Data:
Misc. Data:

Speed control is not specified

OPT_WRITE 1 2

Pending events to control the future use of data
[ ] Clear all existing events? Number of pending events to add: 1

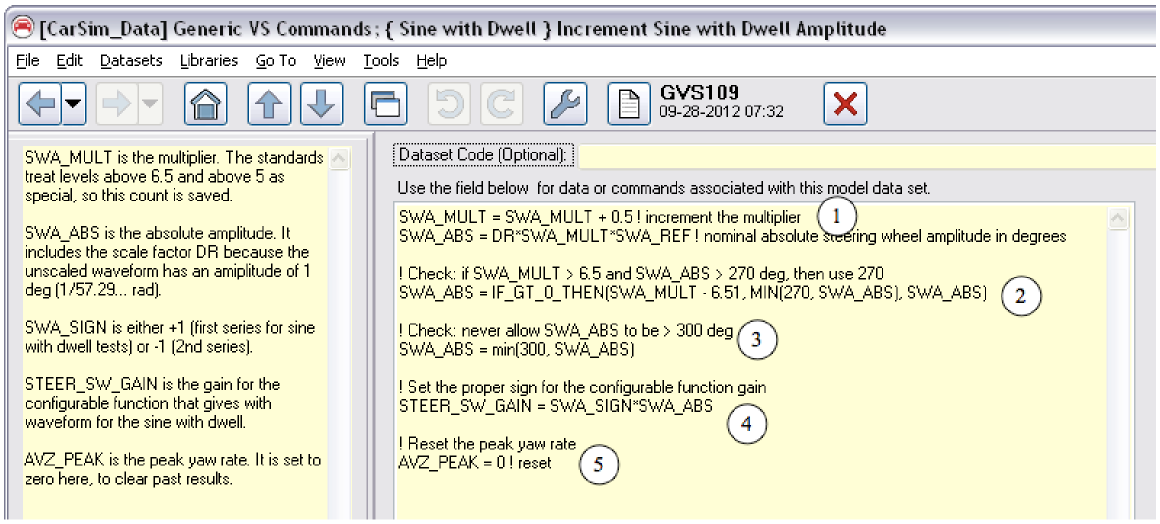
If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

1: IF T_Event >= 5 0.714 THEN load: F. Reset Peak Yaw Rate at 0.714 s 6

Figure 20. Events dataset to start a sine with dwell maneuver.

When this dataset is loaded, some initializations are done with the vehicle model (1). The control clocks are reset. This defines the Beginning of Control (BOS) to be zero in plots such as those shown earlier in Figure 2 and Figure 3. Also, the vehicle is positioned at the initial starting location (X and Y are moved to the start of the road) and heading (Yaw is parallel to the road). Writing to file is enabled (2).

The sine-with-dwell open-loop control is loaded (3). As shown earlier, the waveform that is loaded is normalized with amplitude of 1.0 degree. The `STEER_SW_GAIN` parameter is used to assign the amplitude according to the ESC standards. (See the appendix for details on how configurable functions are transformed.) To do this, the dataset includes a link to some VS commands with the title **Increment Sine with Dwell Amplitude** (4). Figure 21 shows this dataset.


[CarSim_Data] Generic VS Commands: { Sine with Dwell } Increment Sine with Dwell Amplitude
File Edit Datasets Libraries Go To View Tools Help
GVS109
09-28-2012 07:32
Dataset Code (Optional):
Use the field below for data or commands associated with this model data set.
1
2
3
4
5

SWA_MULT is the multiplier. The standards treat levels above 6.5 and above 5 as special, so this count is saved.

SWA_ABS is the absolute amplitude. It includes the scale factor DR because the unscaled waveform has an amiplitude of 1 deg (1/57.29... rad).

SWA_SIGN is either +1 (first series for sine with dwell tests) or -1 (2nd series).

STEER_SW_GAIN is the gain for the configurable function that gives with waveform for the sine with dwell.

AVZ_PEAK is the peak yaw rate. It is set to zero here, to clear past results.

```
SWA_MULT = SWA_MULT + 0.5 ! increment the multiplier
SWA_ABS = DR*SWA_MULT*SWA_REF ! nominal absolute steering wheel amplitude in degrees

! Check: if SWA_MULT > 6.5 and SWA_ABS > 270 deg, then use 270
SWA_ABS = IF_GT_0_THEN(SWA_MULT - 6.51, MIN(270, SWA_ABS), SWA_ABS)

! Check: never allow SWA_ABS to be > 300 deg
SWA_ABS = min(300, SWA_ABS)

! Set the proper sign for the configurable function gain
STEER_SW_GAIN = SWA_SIGN*SWA_ABS

! Reset the peak yaw rate
AVZ_PEAK = 0 ! reset
```

Figure 21. Scaling the sine with dwell steering control.

These equations set the gain for the steering wheel angle (`STEER_SW_GAIN`). To do so, they make use of variables introduced at the start of the run (Figure 17, page 16):

- `SWA_MULT` is an integer multiplier specified in the standard. For the first test in a series it is 1.5, and is increased by 0.5 for each subsequent test in the series. (Recall that the initial value was 1 (Figure 17) to allow for the increment made here.)
- `SWA_ABS` is the absolute value of the gain `STEER_SW_GAIN`.
- `SWA_Sign` is ±1, depending on the direction of steering (clockwise or counter-clockwise).

Recall that the reference steering wheel angle `SWA_REF` was defined in the slowly increasing steer tests as the absolute steering wheel angle associated with a lateral acceleration of 0.3 g's.

Looking at the equations in Figure 21, we see that `SWA_MULT` is incremented by 0.5 and used to define the absolute value of the gain `SWA_ABS` (1). Internally, all calculations involving angles assume units of radians. Because the steering wheel angle shown in Figure 1 and Figure 36 is "normalized" for an amplitude of 1°, it has an amplitude in radians of 1/`DR`, where `DR` is a built-in VS constant for "Degrees per Radian" with a value of about 57.3 (180/π). To provide proper scaling, the equation for `SWA_ABS` includes a factor `DR` to normalize the steering control to 1.0 radian.

FMVSS 126 limits the number of tests in each direction to 11 (when the multiplier would be 6.5). If the eleventh test (with amplitude 6.5*A*) would give an amplitude less than 270°, then a value of 270° must be used. This provision is included using the VS function `IF_GT_0_THEN` to possibly adjust `SWA_ABS` ②.

If the calculated amplitude `DR*SWA_MUL*SWA_REF` ever exceeds 300°, then the amplitude is limited to 300° and this becomes the final test. This provision is included using the `MIN` function to adjust `SWA_ABS` ③.

The amplitude is applied simply using the VS scale factor `STEER_SW_GAIN`, including the sign convention (`SWA_SIGN` is +1 for counter-clockwise steering, –1 for clockwise steering) ④.

Evaluating the Vehicle Response in the Sine with Dwell Test

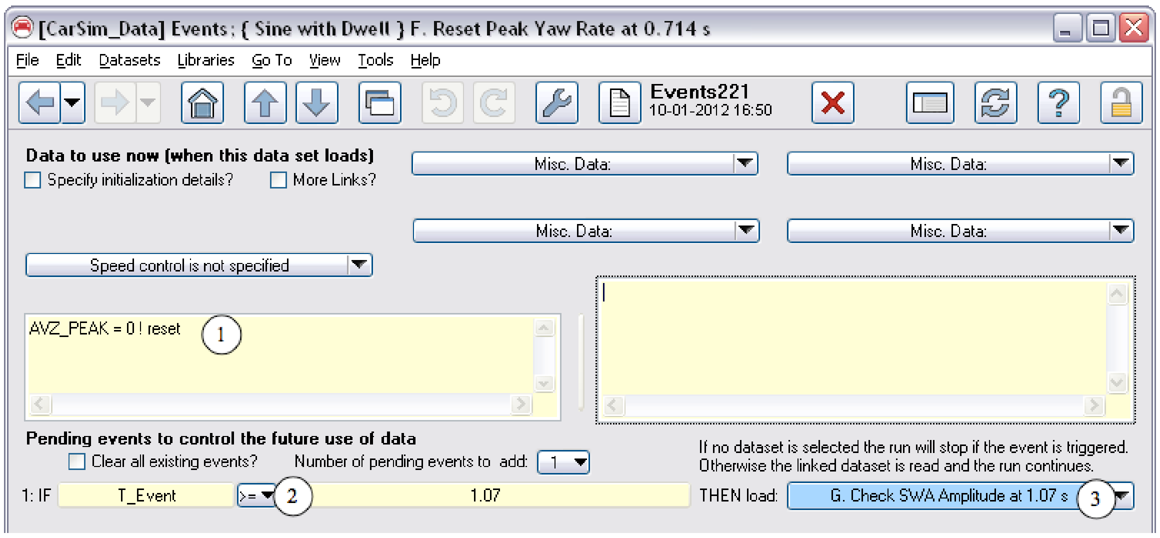
Recall that there are four instants in time during a sine with dwell test where the vehicle response is evaluated. Using the Beginning of Control (BOS) time as the reference, these are:

1. At 0.714 s after BOS the steering wheel angle crosses zero and the peak vehicle yaw statistic is reset to zero.
2. At 1.070 s after BOS when the steering amplitude is 5.*0A* or more, the lateral displacement of the vehicle mass center must be at least 1.83 m for vehicles with GVW of 3,500 kg or less, or 1.52 m for larger vehicles.
3. At 2.929 s after BOS (1.000 s COS) the vehicle yaw rate must not exceed 35% of the peak yaw rate for this iteration.
4. At 3.679 s after BOS (1.750 COS) the vehicle yaw rate must not exceed 20% of the peak yaw rate for this iteration.

When the local time `T_Event` reaches 0.714 s, the pending event set at BOS (⑤, Figure 20) is triggered and the VS solver loads the dataset shown in Figure 22. This dataset sets the peak yaw rate statistic (`AVZ_PEAK`) to zero and adds an event for the next checkpoint, when `T_Event` reaches 1.07 s after BOS.

When the next checkpoint is reached at `T_Event` ≥ 1.07 s, dataset G adds two pending events (Figure 23).

These two events define a logical branch, because the second event is based on a condition that is always true: `T > 0`. (`T` is the simulation time, which is always greater than zero after the simulation starts.) Pending events are evaluated in the same sequence in which they were added. This means that the second event might not be evaluated, in case both the first event ① is true and the linked dataset ② clears other pending events when it is loaded. (We will see shortly that this is the case for dataset H ②.) Usually, when more than one pending events are defined, and the last event is based on a condition that is always true, the intent is to choose between two mutually exclusive options.


[CarSim_Data] Events: { Sine with Dwell } F. Reset Peak Yaw Rate at 0.714 s
File Edit Datasets Libraries Go To View Tools Help
Events221
10-01-2012 16:50
Misc. Data:
Misc. Data:
Misc. Data:
Misc. Data:
1
2
3

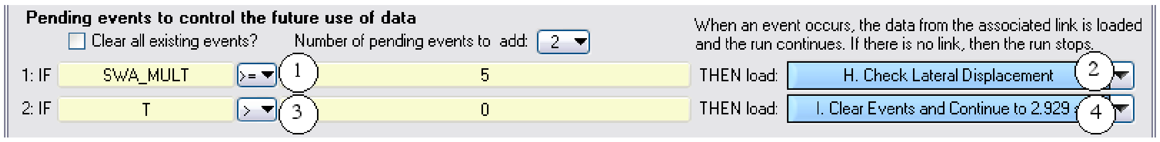
Data to use now (when this data set loads)
Specify initialization details? ☐ More Links? ☐
Speed control is not specified

AVZ_PEAK = 0 ! reset

Pending events to control the future use of data
☐ Clear all existing events? Number of pending events to add: 1
If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

1: IF	T_Event	>=	1.07	THEN load:	G. Check SWA Amplitude at 1.07 s

Figure 22. Events dataset to reset peak yaw rate and set the next checkpoint.



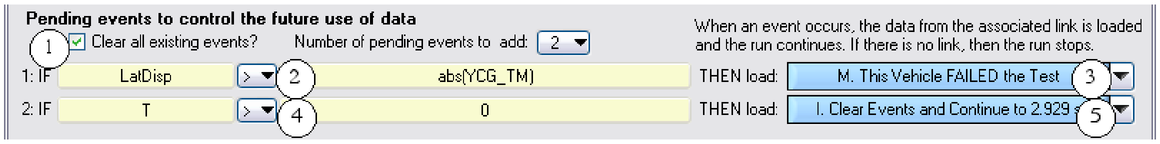
Pending events to control the future use of data
☐ Clear all existing events? Number of pending events to add: 2
When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

1: IF	SWA_MULT	>=	5	THEN load:	H. Check Lateral Displacement
2: IF	T	>	0	THEN load:	I. Clear Events and Continue to 2.929

Figure 23. New pending events added for Events dataset G to create a logical branch.

In this case, the first option is for the case where the steering control multiplier `SWA_MULT` has reached 5. In this case, the lateral position of the vehicle mass center must be checked to ensure it has moved a specified distance `LatDisp` (set to either 1.83 m or 1.52 m, depending on the size of the vehicle). The second option is to continue without checking the lateral displacement. If the second event is evaluated, it will always trigger, loading dataset H.

Figure 24 shows the part of the **Events** screen that has data for dataset H: all pending events are cleared ① (including the check on `T` from the previous Event G (Figure 23) that was not yet triggered), and two pending events are added.



Pending events to control the future use of data
☑ Clear all existing events? Number of pending events to add: 2
When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

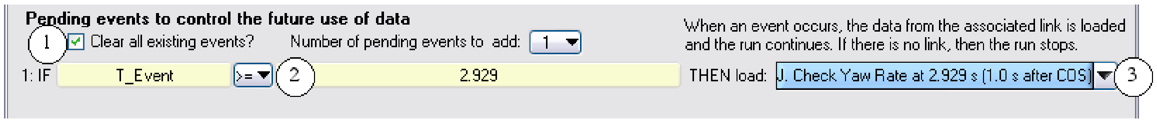
1: IF	LatDisp	>	abs(YCG_TM)	THEN load:	M. This Vehicle FAILED the Test
2: IF	T	>	0	THEN load:	I. Clear Events and Continue to 2.929

Figure 24. New pending events added by Events dataset H to check lateral displacement.

This dataset also defines a branch. The first pending event checks the global Y coordinate of the vehicle mass center ②, and loads a dataset to terminate the run ③ if it does not exceed the minimum limit. The second pending event is based on the same condition that is always true (`T > 0`). If the run is not terminated, the second event will trigger, loading dataset I to continue the run.

When the Boolean formula used to define an event includes functions or more than one variable, the expression must be placed into the yellow field on the right-hand side of the conditional function. The yellow field on the left side must always contain the name of a variable. In this case, the expression is `ABS(YCG_TM)` ②. The absolute value is taken so the event condition will be valid for either direction of steering: `YCG_TM` might be positive or negative.

Because the effect of the first event ② triggering is to end the run ③, it is essential to clear this event immediately if was not triggered at the checkpoint (1.07 s after BOS). Therefore, dataset I is loaded at the next time step and is set to clear all pending events. Figure 25 shows the part of the **Events** screen that has data for dataset I. Indeed, the box is checked to clear all existing events ①, so the only pending event in effect after this dataset is loaded is the one for the checkpoint 2.929 s after BOS ②.



Pending events to control the future use of data

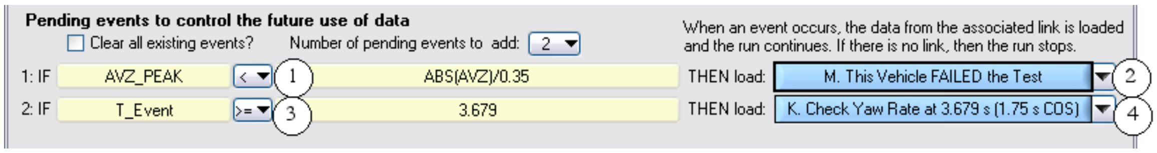
① [✓] Clear all existing events? Number of pending events to add: 1

When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

1: IF	T_Event	>= ②	2.929	THEN load:	J. Check Yaw Rate at 2.929 s (1.0 s after COS) ③

Figure 25. New pending event added by dataset I to create checkpoint at time 2.929 s after BOS.

Figure 26 shows the part of the **Events** screen that has data for dataset J. It compares the peak yaw rate to the current yaw rate. The vehicle fails if the current yaw rate is greater than 35% of the peak: `ABS(AVZ) > 0.35*AVZ_PEAK`.



Pending events to control the future use of data

[ ] Clear all existing events? Number of pending events to add: 2

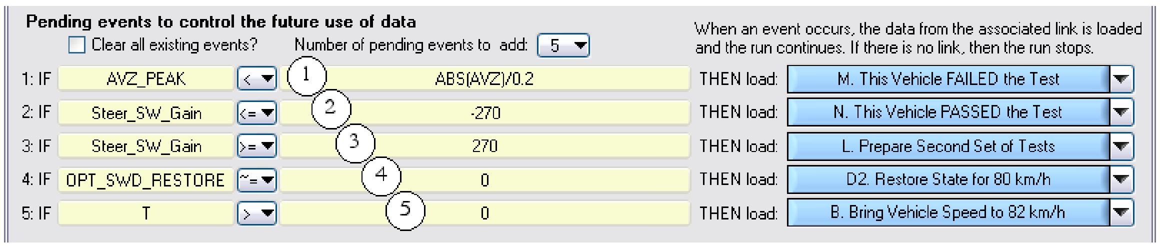
When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

1: IF	AVZ_PEAK	< ①	ABS(AVZ)/0.35	THEN load:	M. This Vehicle FAILED the Test ②
2: IF	T_Event	>= ③	3.679	THEN load:	K. Check Yaw Rate at 3.679 s (1.75 s COS) ④

Figure 26. Dataset J: checking yaw rate at 2.929 s after BOS.

As mentioned earlier, the yellow field on the left side of the conditional function must contain the name of a variable. The conditional expression was therefore rearranged to the form shown in the figure: `AVZ_PEAK < ABS(AVZ)/0.35`

If the yaw rate remains less than 35% of the peak, the run continues to the next checkpoint at 3.679 s (1.75 s after COS). This is the last point of interest in a sine with dwell maneuver. The dataset loaded at this time checks for several conditions (Figure 27):



Pending events to control the future use of data

[ ] Clear all existing events? Number of pending events to add: 5

When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

1: IF	AVZ_PEAK	< ①	ABS(AVZ)/0.2	THEN load:	M. This Vehicle FAILED the Test
2: IF	Steer_SW_Gain	<= ②	-270	THEN load:	N. This Vehicle PASSED the Test
3: IF	Steer_SW_Gain	>= ③	270	THEN load:	L. Prepare Second Set of Tests
4: IF	OPT_SWD_RESTORE	~= ④	0	THEN load:	D2. Restore State for 80 km/h
5: IF	T	> ⑤	0	THEN load:	B. Bring Vehicle Speed to 82 km/h

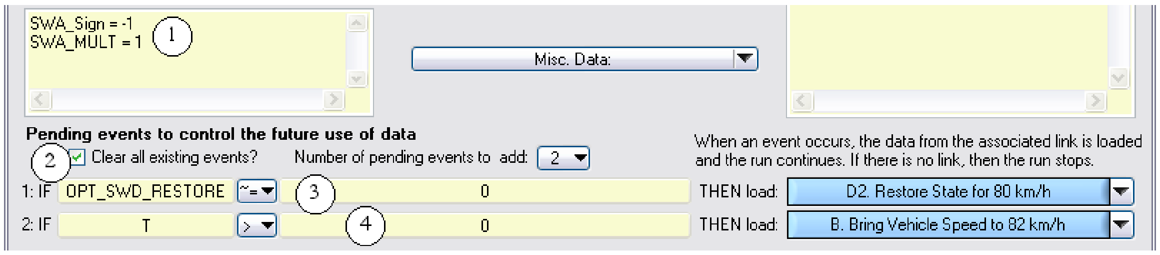
Figure 27. Dataset K: checks made at the end of a sine with dwell maneuver.

- The absolute value of yaw rate is compared with the 20% of the peak value ①; if it is more than 20% the vehicle fails the test and the run is terminated.

- If the steering wheel amplitude is less than or equal to -270° ②, the run is over and the vehicle passed the test.
- If the steering wheel amplitude is greater than or equal to 270° ③, the first series of sine with dwell tests is over and a few variables are set to prepare for the second series.
- If the parameter `OPT_SWD_RESTORE` has a non-zero value ④, an advanced simulation method is used to start the next sine with dwell test. (This method is described in the next section.)
- Otherwise ⑤, the vehicle speed will be increased to 82 km/h in preparation for the next test.

Setting up the Second Sine with Dwell Test Series

When the first series of sine with dwell tests is complete (the steering is greater than or equal to 270°), dataset L is loaded (Figure 28).


SWA_Sign = -1
SWA_MULT = 1

Misc. Data:

Pending events to control the future use of data
[✓] Clear all existing events?   Number of pending events to add: 2

When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

1: IF	OPT_SWD_RESTORE	~=	0	THEN load:	D2. Restore State for 80 km/h
2: IF	T	>	0	THEN load:	B. Bring Vehicle Speed to 82 km/h

Figure 28. Dataset L: prepare for second series of sine with dwell tests.

This dataset sets the parameter `SWA_Sign` to -1 so the steering will have the opposite sign from that used in the first series. Recall that the first amplitude should be 1.5*A*. When each test is started, the multiplier is incremented by 0.5. Hence, the steering multiplier `SWA_MULT` is set here to 1, just as it was before the run started.

Summary of Events for the Sine with Dwell Series

The two series of sine with dwell maneuvers are controlled by the **Events** datasets that have been presented. Following is a review of all the events in the **Sine with Dwell** category. References to local time mean the variable `T_Event`, which is zero at the Beginning of Steer (BOS).

A. **Wait to Start Sine with Dwell**. Add a pending event to wait for the slowly increasing steer tests to complete. Dataset B is loaded when the variable `SLOW_STEER_DONE` is no longer zero.

B. **Bring Vehicle Speed to 82 km/h.** Activate the built in speed controller with a command speed of 83 km/h. Disable writing to file. Clear any existing events, and add a pending event to load dataset C when the actual speed reaches 82 km/h.

C. **Coast Down to 80 km/h**. Switch to open-loop speed control and set the throttle to zero. Also switch to closed-loop steering, using the straight road reference line as a target path so the vehicle will be travelling in a straight line. Rest the vehicle position to ensure that it is where the closed-loop steer controller expects it to be. Add a pending event to load dataset D when the vehicle coasts down to 80 km/h.

D. Have Coasted to 80 km/h. Set up a branch using two new pending events. One will use the save/restore method described in the next section; otherwise, dataset E is loaded at the next time step.

E. Start Sine with Dwell. Enable writing to file, restart control clocks, reset the vehicle position and orientation, and start the sine with dwell maneuver. Scale the amplitude using the direction associated with the series (clockwise or counter-clockwise), the reference angle obtained from the slowly increasing steer tests, and a multiplier that is incremented by 0.5 for each test in the series. Add a pending event to load dataset F when local time reaches 0.714 s.

F. Reset Peak Yaw Rate at 0.714 s. Set the peak yaw rate variable to zero and add a pending event to load dataset G when local time reaches 1.07 s.

G. Check SWA Amplitude at 1.07 s. Set up a branch using two new pending events. Load dataset H if the steering multiplier is five or greater; otherwise, load dataset I at the next time step.

H. Check Lateral Displacement. Clear any existing events and set up a branch using two new pending events. Terminate the test (going to dataset M) if the lateral displacement is less than the specified tolerance; otherwise, set up the next checkpoint using dataset I.

I. Clear Events and Continue to 2.929 s. Clear any existing events and set up the next checkpoint using dataset J.

J. Check Yaw Rate at 2.929 s (1.0 after COS). Add two pending events. One terminates the run by loading dataset M if the absolute value of yaw rate exceeds 35% of the peak yaw rate. The other sets up the next checkpoint using dataset K.

K. Check Yaw Rate at 3.679 s (1.75 s after COS). This is the last checkpoint for the maneuver. The dataset adds five pending events:

1. Terminate the run (go to dataset M) if the absolute value of yaw rate exceeds 20% of the peak yaw rate.
2. Terminate the run (go to dataset N) if the steering wheel gain is ≤ -270°. This occurs when the vehicle has passed the full test procedure.
3. Go to dataset L if the steering wheel gain is ≥ 270° to run the second series of sine with dwell tests.
4. Go to dataset D2 to start the next sine with dwell maneuver in the current series using the method described in the next section if the parameter `OPT_SWD_RESTORE` is not zero.
5. Otherwise, go to dataset B to start the next sine with dwell maneuver in the existing series.

L. Prepare Second Set of Tests. Clear any existing events, reset variables for the new series, and load dataset B at the next time step.

M. **This Vehicle FAILED the Test**. Clear any existing events, set variables listed in the echo end file and exported to Simulink, and immediately load dataset O.

N. **This Vehicle PASSED the Test**. Clear any existing events, set variables listed in the echo end file and exported to Simulink, and immediately load dataset O.

O. **Wait 0.5 s and Stop**. Disable writing to file, reset control clocks, and add a pending event to stop the run when local time reaches 0.5 s.

Using Save and Restore to Reduce Simulation Time

VS solvers support an option to save the state of the math model and later restore it. This can save significant time when simulated repetitive test sequences such as the sine with dwell series. If you are not interested in this topic, feel free to skip to the next section.

The method of restoring a previously saved state works well when using the VS solver by itself. When other software is used, the method will not work if the external software has variables not controlled by the VS solver that define the state of the system. For example, if CarSim is run with hardware in the loop (HIL), it is almost certain that the save/restore method will not work. If a Simulink model is used that in which current calculations depends on variables from a previous time step (e.g., variables defined in Simulink with differential equations), it is likely the save/restore method will not work. On the other hand, if the Simulink model makes all calculations at each time step based only on constants and variables exported from the CarSim math model, then the method can work OK.

The example stability controller provided with CarSim 8.2 does in fact work OK with the method described in this section.

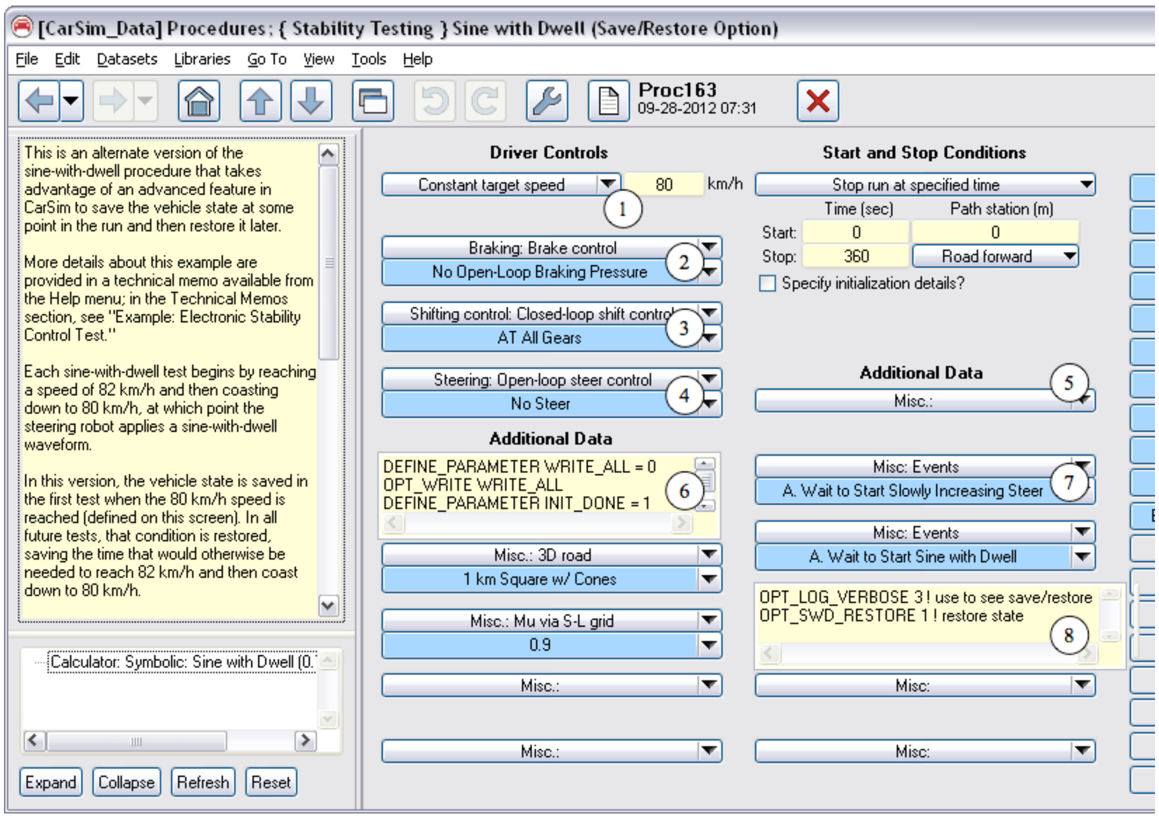
Changes Made to Use Save and Restore

All discussion in the previous section was based on a procedure in CarSim 8.2 in the category **Stability Testing** and the name **Sine with Dwell (ECE R13H, FMVSS 126)**. Figure 29 shows a very similar dataset in the same category with the name **Sine with Dwell (Save/Restore Option)**.

Comparing this dataset to the one used earlier (Figure 6, page 7) reveals only a few differences. The earlier version included three links to **Events** datasets. This one has only two links. The link used previously to bring the vehicle from an at-rest condition up to the test speed of 80 km/h is not used and the link is cleared ⑤.

Because this run will involve discontinuities in speed, there is no reason to simulate the vehicle transition from at-rest to 80 km/h. Instead, the vehicle starts at 80 km/h with the closed-loop speed controller working to maintain that speed ①. Other controls that were set during the first series of events are instead set on this screen, including braking ②, gear shifting ③, and steering ④.

The **Events** sequence to start slowly increasing steer ⑦ begins with a pending event that monitors a parameter `INIT_DONE` that was added via the VS command `DEFINE_PARAMETER`. Because `INIT_DONE` was defined in a dataset not used here, the parameter is defined on this screen and given a value of 1 ⑥, indicating that the next sequence can start at the next time step.



This is an alternate version of the sine-with-dwell procedure that takes advantage of an advanced feature in CarSim to save the vehicle state at some point in the run and then restore it later.

More details about this example are provided in a technical memo available from the Help menu; in the Technical Memos section, see "Example: Electronic Stability Control Test."

Each sine-with-dwell test begins by reaching a speed of 82 km/h and then coasting down to 80 km/h, at which point the steering robot applies a sine-with-dwell waveform.

In this version, the vehicle state is saved in the first test when the 80 km/h speed is reached (defined on this screen). In all future tests, that condition is restored, saving the time that would otherwise be needed to reach 82 km/h and then coast down to 80 km/h.

Calculator: Symbolic: Sine with Dwell (0.

Driver Controls

Constant target speed 80 km/h (1)

Braking: Brake control — No Open-Loop Braking Pressure (2)

Shifting control: Closed-loop shift control — AT All Gears (3)

Steering: Open-loop steer control — No Steer (4)

Additional Data

```
DEFINE_PARAMETER WRITE_ALL = 0
OPT_WRITE WRITE_ALL
DEFINE_PARAMETER INIT_DONE = 1
```
(6)

Misc.: 3D road — 1 km Square w/ Cones

Misc.: Mu via S-L grid — 0.9

Misc.:

Misc.:

Start and Stop Conditions

Stop run at specified time

	Time (sec)	Path station (m)
Start:	0	0
Stop:	360	Road forward

☐ Specify initialization details?

Additional Data (5)

Misc.:

Misc: Events — A. Wait to Start Slowly Increasing Steer (7)

Misc: Events — A. Wait to Start Sine with Dwell

```
OPT_LOG_VERBOSE 3 ! use to see save/restore
OPT_SWD_RESTORE 1 ! restore state
```
(8)

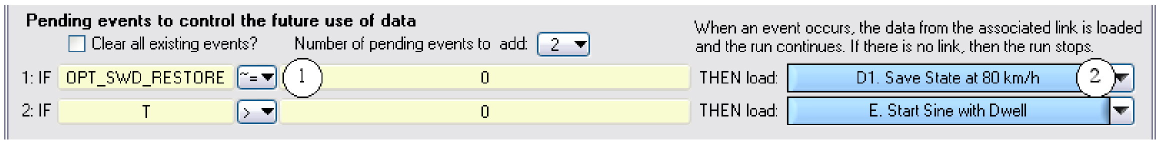
Misc:

Misc:

Figure 29. Procedures dataset for using start/restart method.

The parameter named `OPT_SWD_RESTORE` was defined in the previous version, but was set to zero. When this parameter is zero the option to run sine with dwell using the restore commands is never used. The capability is enabled on this screen by setting the value to 1 (8). Also, an option to record save and restore events in the log file is set using the built-in system parameter `OPT_LOG_VERBOSE` (8).

The Save and Restore options are used in two **Events** datasets lettered D1 and D2. These are reached with events that are triggered when `OPT_SWD_RESTORE` has a nonzero value. The first event is added with dataset **D. Have Coasted to 80 km/h** (Figure 30).



Pending events to control the future use of data

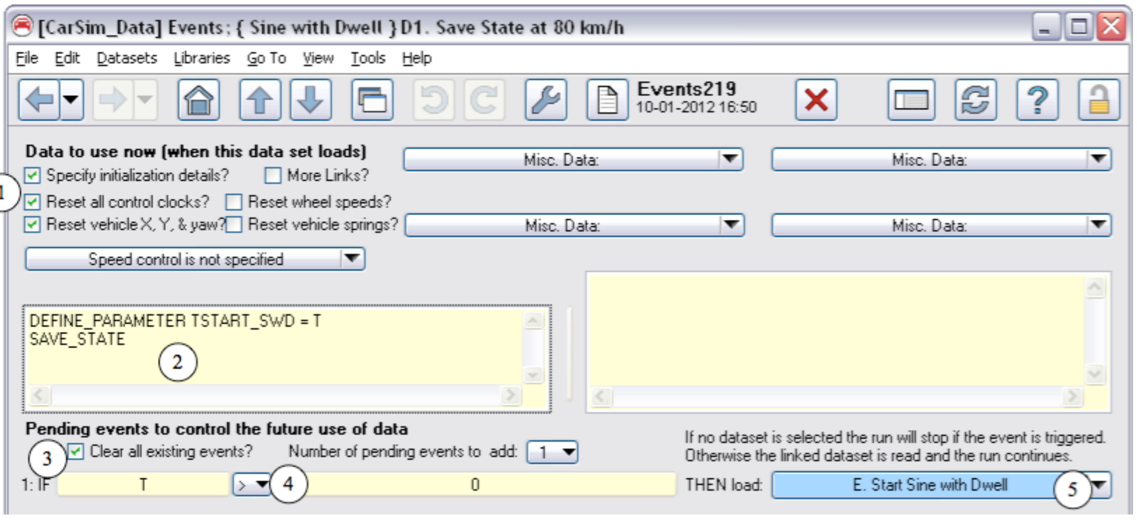
☐ Clear all existing events? Number of pending events to add: 2

When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

1: IF	OPT_SWD_RESTORE	~=	0 (1)	THEN load:	D1. Save State at 80 km/h (2)
2: IF	T	>	0	THEN load:	E. Start Sine with Dwell

Figure 30. Pending events added by events dataset D to check for save/restore option.

If `OPT_SWD_RESTORE` has a nonzero value, the dataset D1 loaded at the next time step (Figure 31). It clears any existing events (3) and resets the vehicle position and control clocks (1). It defines a new parameter `TSTART_SWD` with its value set to the current time `T`, and applies the VS command `SAVE_STATE` (2). The `SAVE_STATE` command makes a request to the VS solver to save the state of the vehicle math model at the end of the current time step. In this case, the state is saved when the vehicle has coasted down to 80 km/h and is ready to start a sine with dwell maneuver.


[CarSim_Data] Events: { Sine with Dwell } D1. Save State at 80 km/h

Events219
10-01-2012 16:50

Data to use now (when this data set loads)
- [✓] Specify initialization details?
- [ ] More Links?
- [✓] Reset all control clocks?
- [ ] Reset wheel speeds?
- [✓] Reset vehicle X, Y, & yaw?
- [ ] Reset vehicle springs?

Speed control is not specified

```
DEFINE_PARAMETER TSTART_SWD = T
SAVE_STATE
```

Pending events to control the future use of data
- [✓] Clear all existing events?

Number of pending events to add: 1

1: IF	T	>	0	THEN load:	E. Start Sine with Dwell

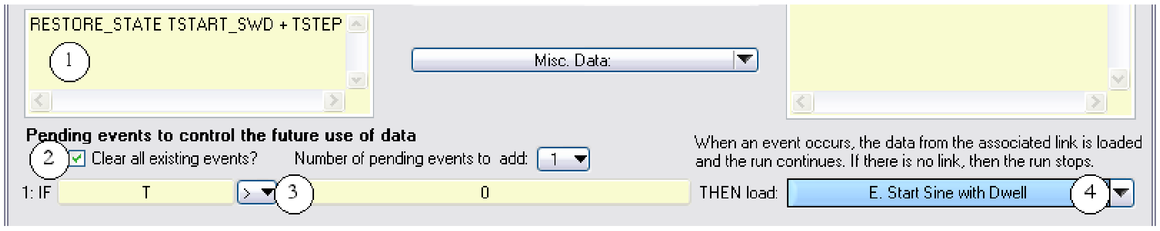
If no dataset is selected the run will stop if the event is triggered. Otherwise the linked dataset is read and the run continues.

Figure 31. Events dataset to save current state of the CarSim math model.

The pending event added with this dataset (4) defines a condition that is always true, ensuring that the dataset E is loaded at the next time step to start the first sine with dwell maneuver (5).

The first reference to `OPT_SWD_RESTORE` resulted in dataset D1 being loaded to save the state of the math model. The second reference is made to load dataset D2, to restore the state. This second reference is made at the end of each sine with dwell maneuver in dataset K (Figure 27, page 22). An event is added which will trigger if `OPT_SWD_RESTORE` is not zero and another sine with dwell test is needed in the current series.

Figure 32 shows that restoring the state of the math model is done with a single command: `RESTORE_STATE`, which specifies the time associated with the state `TSTART_SWD + TSTEP`. The `RESTORE_STATE` command requests the VS solver to restore the state of the model to a time saved at or before the specified time. Although the request to save the state was made at time `TSTART_SWD`, the action took place at the end of the time step and might be slightly later. The current time step interval is added to ensure that the VS solver will find the saved state.


```
RESTORE_STATE TSTART_SWD + TSTEP
```

Pending events to control the future use of data
- [✓] Clear all existing events?

Number of pending events to add: 1

1: IF	T	>	0	THEN load:	E. Start Sine with Dwell

When an event occurs, the data from the associated link is loaded and the run continues. If there is no link, then the run stops.

Figure 32. Events dataset to restore a saved state of the CarSim math model.

Comparison of the Simulation Methods

The motivation for using the `SAVE_STATE` and `RESTORE_STATE` commands is to save time when simulating the entire sine with dwell procedure for testing vehicles with stability control systems. Viewing the log files for runs made with the two procedures shows that the total simulation time is reduced from 211.4 s to 92.7 s. In both cases, the same portions of the testing were recorded. Transitions between tests were not recorded even though they were simulated using the lengthier method, allowing direct comparison of the plots made from the recorded motions. For example, Figure 33 shows that plots overlaid from the two runs match perfectly (within the resolution of the plots).

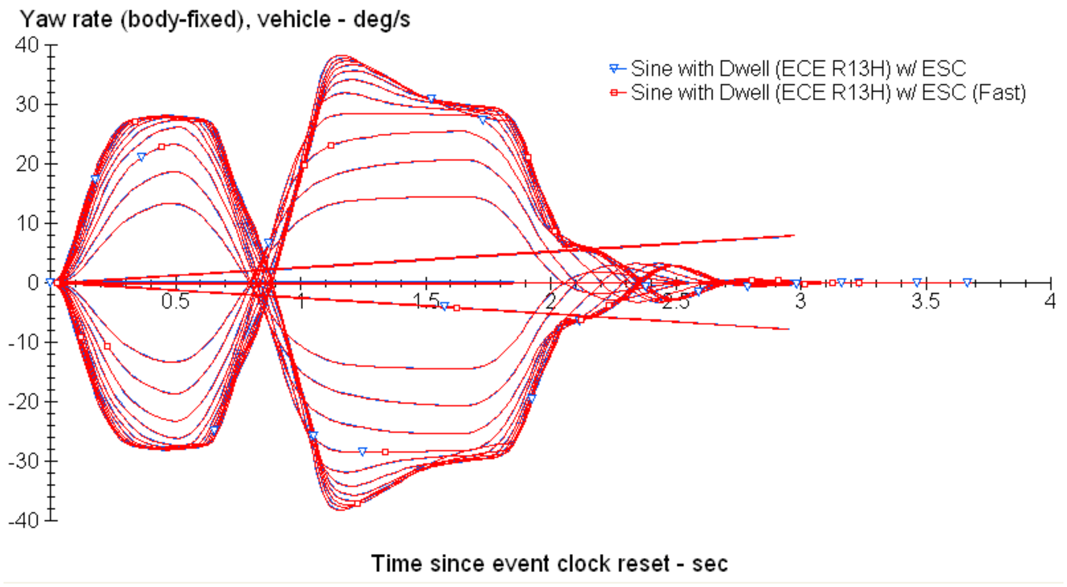
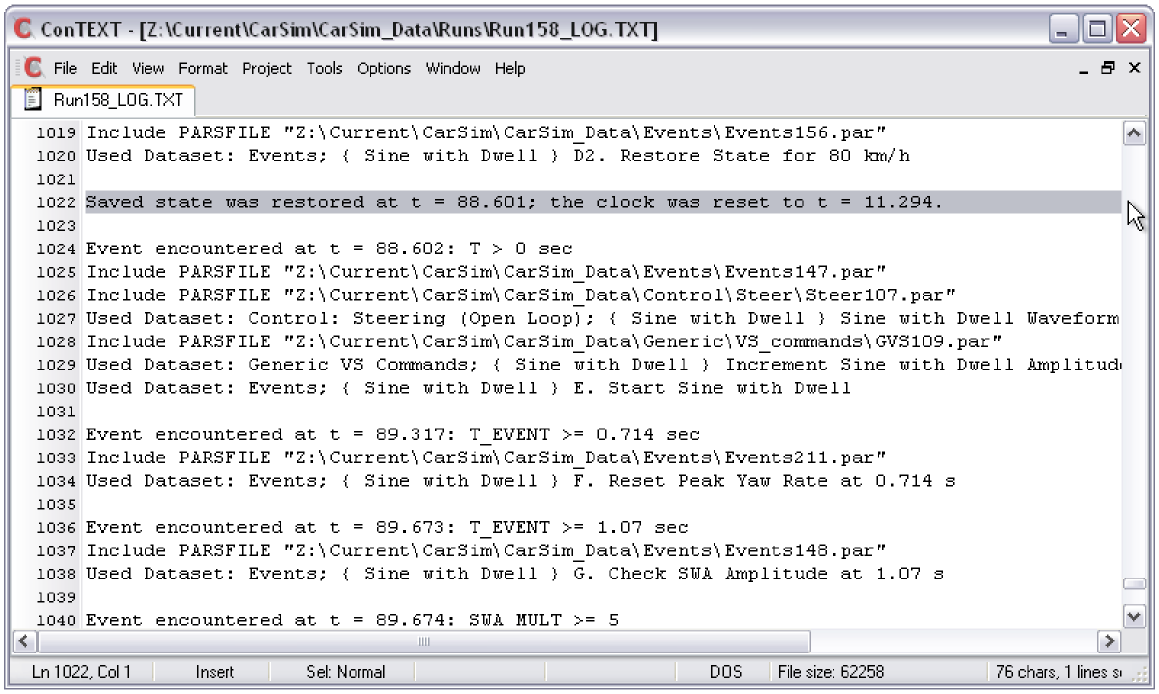


Figure 33. Comparison of recorded yaw rate time histories for the two simulation methods.

Figure 34 shows part of the log file for the run in which the save and restore commands were used. Besides tracking every event that was triggered, the file also indicates every time the math model state was saved or restored, as indicated by the highlighted line.



```
1019 Include PARSFILE "Z:\Current\CarSim\CarSim_Data\Events\Events156.par"
1020 Used Dataset: Events; { Sine with Dwell } D2. Restore State for 80 km/h
1021
1022 Saved state was restored at t = 88.601; the clock was reset to t = 11.294.
1023
1024 Event encountered at t = 88.602: T > 0 sec
1025 Include PARSFILE "Z:\Current\CarSim\CarSim_Data\Events\Events147.par"
1026 Include PARSFILE "Z:\Current\CarSim\CarSim_Data\Control\Steer\Steer107.par"
1027 Used Dataset: Control: Steering (Open Loop); { Sine with Dwell } Sine with Dwell Waveform
1028 Include PARSFILE "Z:\Current\CarSim\CarSim_Data\Generic\VS_commands\GVS109.par"
1029 Used Dataset: Generic VS Commands; { Sine with Dwell } Increment Sine with Dwell Amplitud
1030 Used Dataset: Events; { Sine with Dwell } E. Start Sine with Dwell
1031
1032 Event encountered at t = 89.317: T_EVENT >= 0.714 sec
1033 Include PARSFILE "Z:\Current\CarSim\CarSim_Data\Events\Events211.par"
1034 Used Dataset: Events; { Sine with Dwell } F. Reset Peak Yaw Rate at 0.714 s
1035
1036 Event encountered at t = 89.673: T_EVENT >= 1.07 sec
1037 Include PARSFILE "Z:\Current\CarSim\CarSim_Data\Events\Events148.par"
1038 Used Dataset: Events; { Sine with Dwell } G. Check SWA Amplitude at 1.07 s
1039
1040 Event encountered at t = 89.674: SWA_MULT >= 5
```

Figure 34. Portion of the log file for the run with save and restore commands.

Time Scales

Three major time scales have been used in this memo:

1. The simulation time `T` starts at a specified value `TSTART` (normally zero) and increases with each step as the simulation runs. This is the same as the time shown in Simulink if used. This variable is recorded in the output ERD files with the name `T_Stamp`.
2. A local "Event" time scale is used in which the clock is sometimes reset as the run continues. Local time `T_Event` is defined as (`T – T_EVENT_START`), allowing it to be reset at any time by setting the parameter `T_EVENT_START` to the current time. This is done automatically whenever the checkbox labeled **Reset all control clocks** on the **Events** screen is checked.
3. A file time (recording time) named `Time` is normally used for the horizontal axis in plots made from simulation results. This variable is not recorded directly in the file; it is defined implicitly based on the position in the file and the time step associated with the file.

Figure 35 compares the three time scales for the two simulation methods. The horizontal axis is based on the file time, `Time`. Because the same logic was used to control writing to file, both runs generated files that covered the same amount of time (86.9 s).

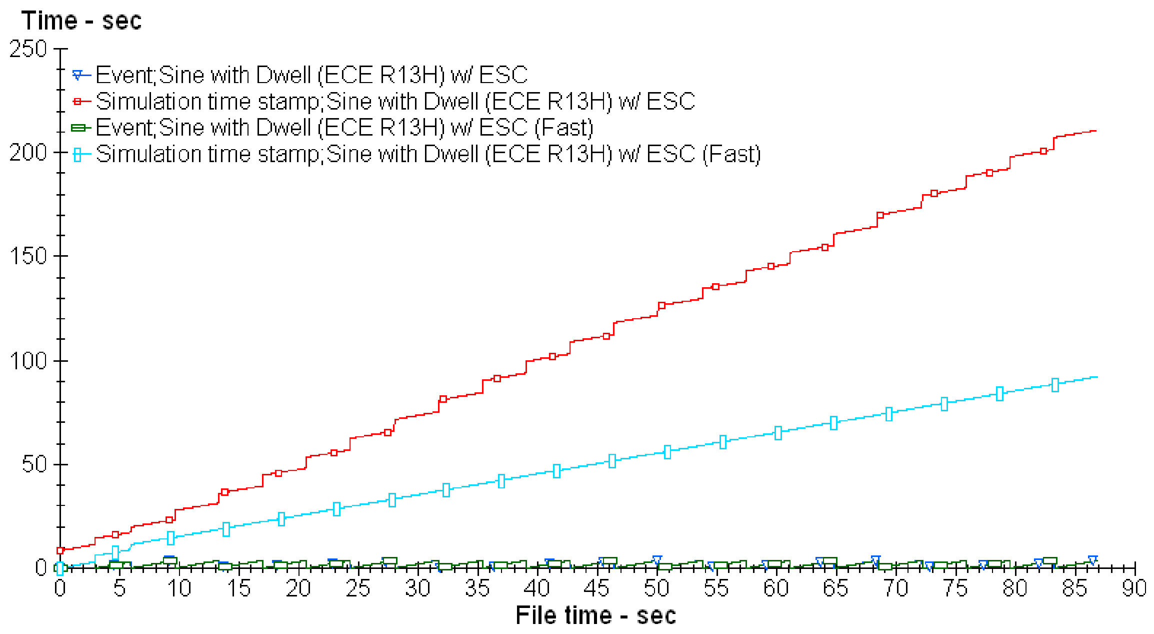


Figure 35. Comparison of three time scales used in the simulations.

The two plots whose names begin with "Event" show the local event time (relative to the start of each test). It never exceeds 3.68 s. Because both simulation methods used the same logic to reset the control clocks, both record the same local times throughout the run. The two plots show names begin with "Simulation time stamp" show the simulation time. These lines show jumps whenever recording is disabled. The red line (using the method for maintaining speed continuity) shows many jumps, corresponding to the transitions between the many tests. The blue line (fast simulation method) shows only two jumps at the very beginning: one is the transition between the two slowly increasing steer tests, and the other is before the first sine with dwell test. All subsequent transitions before the sine with dwell tests were handled using the `RESTORE_STATE` command, involving only one time step in the simulation.

Limits and Assumptions

Motor vehicle safety standards contain physical testing procedures that normally include checking electronics, conditioning the vehicle and tires, and accounting for the inevitable variations that occur in a testing environment. Since there are no performance variations in the simulation process, these procedures are not included here. Several other aspects of the standards are noted here that special attention in experimental testing, but are very easy to apply in simulation.

In determining the reference angle *A*, designated `SWA_REF` in the simulations, the standards specify six test runs, with the results processed using linear regression to relate steering wheel angle to lateral acceleration (A_y). The intent is to reduce the effects of measurement noise and test-to-test variation. Because these factors do not exist in the CarSim world, we run just two tests and simply stop when A_y reaches limits of ±0.3g.

Due to the expense of an automated instrumentation package that can measure absolute X and Y coordinates, the standards allow calculation of lateral displacement by double integrating A_y after removing vehicle roll and pitch effects from the accelerometer measurements. Although it is close, this is not exactly the same as true lateral position due to the interaction with yaw during the test. This possibly complicated data processing method was eliminated in the example by setting up the vehicle at the start of each Sine with Dwell test such that the yaw and lateral displacement were essentially zero.

Two approaches were presented for handling the transition from one test to the next. In the first, continuity in speed is maintained. The vehicle starts at rest and is controlled through throttle to bring the speed up when necessary. This approach is intended for running with a detailed controller model or real-time HIL system with physical electronic controllers. Real controller algorithms can be sensitive to abrupt changes in vehicle speed or steering wheel angle, causing them to disengage. Even though speed continuity is maintained, shortcuts are taken to reduce simulation time and complexity. The vehicle is moved abruptly between tests. The assumption is that ESC systems do not rely on measured global X and Y coordinates, nor do they rely on absolute yaw angle measurements. Therefore, there is no harm in adjusting those variable to produce nicer plots and animations.

In the second approach, continuity in speed is sacrificed to obtain simulation efficiency. The simulation begins with the vehicle running at 80 km/h, ready for the first slowly increasing steer test. The full coast down simulation in preparation for a sine with dwell test is done only once and the state of the math model is saved. The saved state is then restored as needed to start all the subsequent tests, eliminating time between the consecutive sine-with-dwell maneuvers. This will work with ESC controller models that base 100% of their logic on constants and on variables exported from the CarSim math model. For example, the ESC model provided in CarSim 8.2 works with the save and restores method. On the other hand, HIL systems with ESC hardware would probably fail if continuity in speed were not maintained.

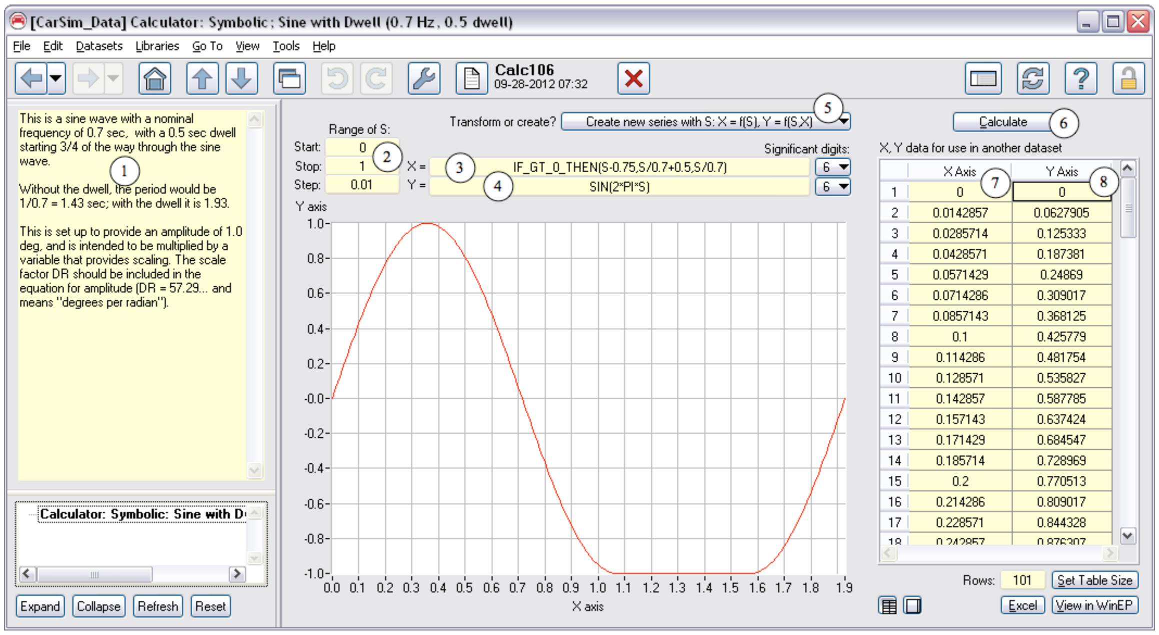
Appendix: Programming the Sine with Dwell Waveform

This appendix describes how the sine with dwell waveform is generated and will be manipulated during the series.

Creating the Sine with Dwell Waveform

The sine-with-dwell waveform is not built into CarSim. If this example were not installed, you could choose among several methods to define the waveform. You might use a spreadsheet. Or you could use the built-in CarSim calculator library. An advantage of using the built-in library is that you keep the definition in the database and can go back to make minor changes (e.g., better resolution, a variation for heavy trucks, etc.).

The waveform dataset is in fact included in the CarSim 8.2 database as a calculator example. To see it, use the **Tools** menu, select **Calculator**, and then use the **Datasets** menu to find the dataset shown in Figure 36.



[CarSim_Data] Calculator: Symbolic ; Sine with Dwell (0.7 Hz, 0.5 dwell)

Calc106
09-28-2012 07:32

This is a sine wave with a nominal frequency of 0.7 sec, with a 0.5 sec dwell starting 3/4 of the way through the sine wave.

Without the dwell, the period would be 1/0.7 = 1.43 sec; with the dwell it is 1.93.

This is set up to provide an amplitude of 1.0 deg, and is intended to be multiplied by a variable that provides scaling. The scale factor DR should be included in the equation for amplitude (DR = 57.29... and means "degrees per radian").

Calculator: Symbolic: Sine with D

Transform or create? Create new series with S: X = f(S), Y = f(S,X)

Range of S:
Start: 0
Stop: 1
Step: 0.01

X = IF_GT_0_THEN(S-0.75,S/0.7+0.5,S/0.7)
Y = SIN(2*PI*S)

Significant digits: 6, 6

X, Y data for use in another dataset

	X Axis	Y Axis
1	0	0
2	0.0142857	0.0627905
3	0.0285714	0.125333
4	0.0428571	0.187381
5	0.0571429	0.24869
6	0.0714286	0.309017
7	0.0857143	0.368125
8	0.1	0.425779
9	0.114286	0.481754
10	0.128571	0.535827
11	0.142857	0.587785
12	0.157143	0.637424
13	0.171429	0.684547
14	0.185714	0.728969
15	0.2	0.770513
16	0.214286	0.809017
17	0.228571	0.844328
18	0.242857	0.876307

Rows: 101

Figure 36. Sine with dwell waveform, generated with calculator tool.

To generate this waveform, the calculator tool was set to create a new series ⑤. In this mode, the values shown for X (first column ⑦) and Y (second column ⑧) were calculated as functions of an independent variable S. (Incidentally, the variable S on this screen has no relationship to the station S used in road descriptions.) S is automatically updated using a starting value, stopping value, and step size. As shown in the figure, in this dataset S goes from 0 to 1 with steps of 0.01 ②.

The formula used to calculate Y is: `Y = SIN(2*PI*S)` ④. This simply produces one period of a sine wave.

The formula used to calculate X is more complicated ③:

```
X = IF_GT_0_THEN(S-0.75,S/0.7+0.5,S/0.7)
```

If there were no dwell, the calculation would simply be `X = S/0.7` (mapping one period of the sine wave to a frequency of 0.7 Hz). To account for the dwell, we want the first ¾ of the sine wave to be normal, with `X = S/0.7`. For that last ¼, we want to add the dwell time of 0.5, such that `X = S/0.7 + 0.5`. To include both, we use the VS function `IF_GT_0_THEN` which applies an if-then condition: if the first argument is greater than zero it returns the second argument; otherwise it returns the third argument.

The notes field ① is used to document the purpose of this dataset.

Note More details about the Calculator Tool are provided in the **Help** document for that screen: **Help** -> **Tools** -> **Calculator Screen (Symbolic)**. The detailed reference material for VS Commands and functions such as `IF_GT_0_THEN` is provided in the VS Commands Reference Manual.

Transforming the Sine with Dwell Waveform

VS configurable functions used for driver controls all include offsets and gains that allow then to be transformed easily by adjusting just a few parameters. This subsection shows how to find the keywords needed to transform the sine with dwell waveform to start at the current simulation time, and to be scaled according to the ESC testing specification.

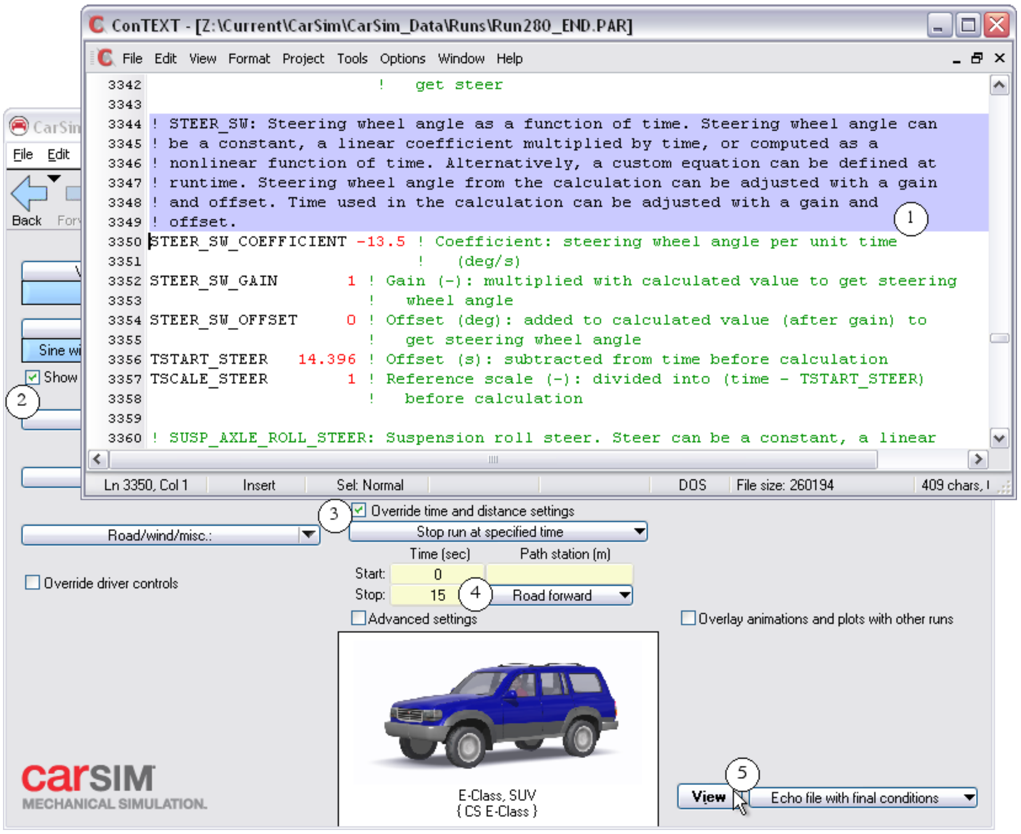
As with all configurable functions in CarSim, additional information about keywords is available in the echo file produced with any run in which the function is used. To see an echo file (Figure 37) showing all of the open-loop steering function options, you can make a special run that stops when the function is active and all transformation options are available.

1. Go to a **Run Control** dataset with a sine-with-dwell run (e.g., Figure 4, page 6).
2. Change the stop time to 15 seconds. To do this, check the box to show more options (②, Figure 37), check the box to override time and distance settings ③, choose the option to stop at a specified time, and enter a stop time of 15 ④.
3. Make the run.
4. Use the drop-down list in the lower-right part of the screen to select **Echo file with final conditions** and click the adjacent **View** button ⑤. This brings up the echo file created at the end of the run.

The highlighted part of the file (in the figure) describes the available options for specifying steering wheel angle as a function of time ①. The parameters related to the function are shown just below this section, with their values at the time the run ended at 15 s.

For the figure shown, the run ended during the second slowly increasing steer test. The echo file provides the following information:

- Line 3350 indicates that steering wheel angle was specified to be calculated using a linear coefficient (`STEER_SW_COEFFICIENT`) with a value of `-13.5`.


ConTEXT - [Z:\Current\CarSim\CarSim_Data\Runs\Run280_END.PAR]
File Edit View Format Project Tools Options Window Help
Ln 3350, Col 1 | Insert | Sel: Normal | DOS | File size: 260194 | 409 chars
1
2
3
4
5
Back
Show
Road/wind/misc.:
Override driver controls
Override time and distance settings
Stop run at specified time
Time (sec)
Path station (m)
Start:
Stop:
Road forward
Advanced settings
Overlay animations and plots with other runs
E-Class, SUV
{ CS E-Class }
View
Echo file with final conditions
carSIM
MECHANICAL SIMULATION

```
3342                          !   get steer
3343
3344 ! STEER_SW: Steering wheel angle as a function of time. Steering wheel angle can
3345 ! be a constant, a linear coefficient multiplied by time, or computed as a
3346 ! nonlinear function of time. Alternatively, a custom equation can be defined at
3347 ! runtime. Steering wheel angle from the calculation can be adjusted with a gain
3348 ! and offset. Time used in the calculation can be adjusted with a gain and
3349 ! offset.
3350 STEER_SW_COEFFICIENT -13.5 ! Coefficient: steering wheel angle per unit time
3351                             !    (deg/s)
3352 STEER_SW_GAIN        1 ! Gain (-): multiplied with calculated value to get steering
3353                        !   wheel angle
3354 STEER_SW_OFFSET      0 ! Offset (deg): added to calculated value (after gain) to
3355                        !   get steering wheel angle
3356 TSTART_STEER   14.396 ! Offset (s): subtracted from time before calculation
3357 TSCALE_STEER         1 ! Reference scale (-): divided into (time - TSTART_STEER)
3358                        !   before calculation
3359
3360 ! SUSP_AXLE_ROLL_STEER: Suspension roll steer. Steer can be a constant, a linear
```

Settings shown: Override time and distance settings [✓]; Stop run at specified time; Start: 0; Stop: 15; Road forward; Advanced settings [ ]; Override driver controls [ ]; Overlay animations and plots with other runs [ ]; Echo file with final conditions.

Figure 37. Echo file for a shortened run that shows options for scaling steering wheel angle.

- Line 3352 indicates that the calculated steering wheel angle was then multiplied by a gain (`STEER_SW_GAIN`) of one (i.e., the gain had no effect).
- Line 3354 indicates that the calculated steering wheel angle was added to an offset (`STEER_SW_OFFSET`) of zero (i.e., the offset had no effect).
- Line 3356 indicates that the time multiplied by the linear coefficient was adjusted by subtracting an offset (`TSTART_STEER`) of 14.396 from the simulation time.
- Line 3357 indicates that the time multiplied by the linear coefficient was scaled by a factor (`TSCALE_STEER`) of one (i.e., the time scaling option had no effect).

Thus, when the simulation time was 14.396, the steering was zero; as the simulation proceeded, the steering wheel angle (degrees) was calculated as:

```
Steer_SW = -13.5•(T – 14.396)
```

All four scaling and offset parameters are included in calculations whenever the steering wheel angle is not a constant. (Full details for using VS configurable functions are provided in the VS Solvers Manual.) The example echo file shows that one of them was used during the slowly increasing steer test: `TSTART_STEER` had a nonzero value of 14.396. This parameter is set whenever the box is checked on the **Events** screen to **Reset all control clocks** (e.g., 1, Figure 13, page 13). Whenever a dataset to start the sine with dwell control is loaded, the function can be is configured such that “time” used in the calculations is zero.