Processor Utilization and motion sequence

Part I: Processor Utilization

Objective

- While your motor is critical to your application, it is not the only element in your design.

- Your motor control objectives may be constrained by other aspects of your design, and in particular, the requirements of your application code.

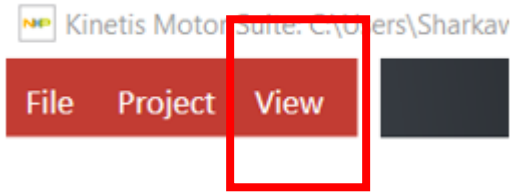
- In this lab, examine CPU utilization of the KMS firmware. Change execution rates and operating modes to determine the processor capacity available for your application code. Observe the tradeoff between processor usage and motor control operation.

Procedure

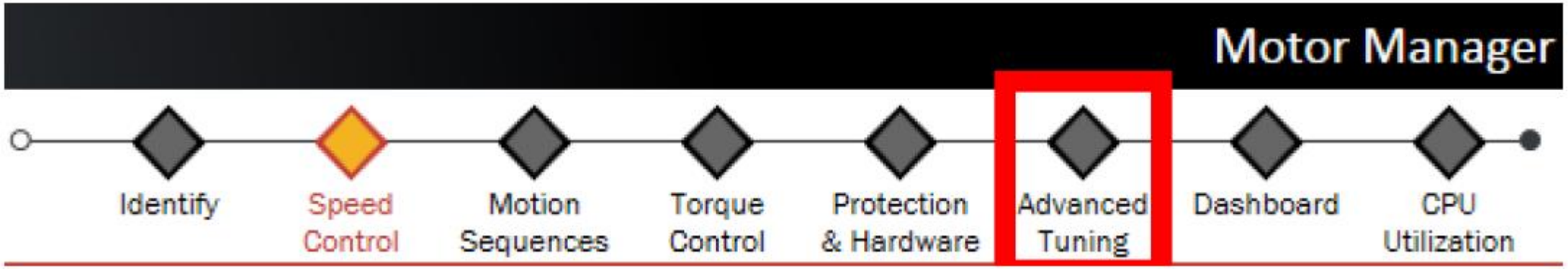
1. Understand usage in the existing configuration

- Launch KMS by double-clicking the Desktop icon.

- Click on “View” then select “Go to the Motor Manager”.



- Navigate to the Advanced Tuning page of Motor Manager

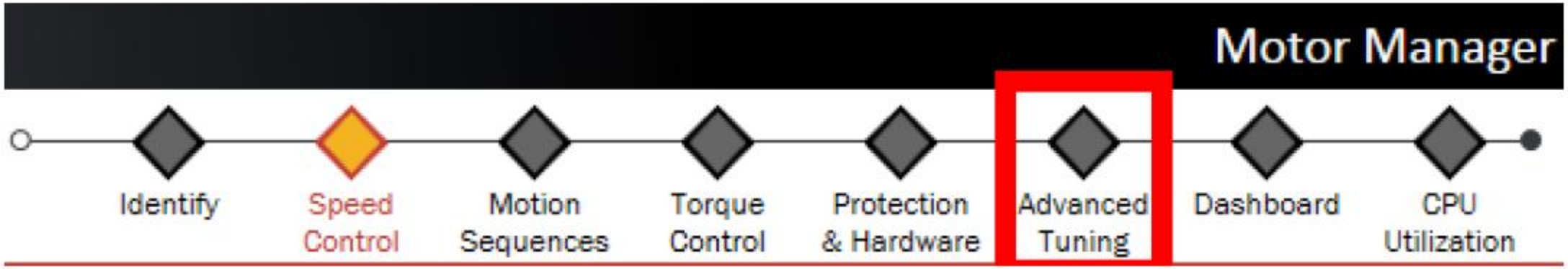


- Locate the System Frequencies section

Procedure

1. Understand usage in the existing configuration

- Navigate to the Advanced Tuning page of Motor Manager



- Locate the System Frequencies section

System Frequencies

KMS firmware relies on three operating frequencies: the PWM switching frequency, a Fast interrupt service routine (ISR) execution frequency, and a Slow ISR execution frequency. Define the relationships between these frequencies below.

PWM Frequency	10	[kHz]
PWM / Fast ISR	1	
Fast ISR / Slow ISR	10	

- The three values in this section describe core execution frequencies of the KMS firmware, which is architected as a dual interrupt service routine (ISR) system to separate motor control operation from application-focused code.

- These frequencies determine CPU usage.

Variable	Description
PWM Frequency	Rate at which Pulse Width Modulation occurs
PWM / Fast ISR	Ratio between PWM switching frequency and the rate at which the core motor control code is executed; the Fast ISR is also known as the PWM ISR
Fast ISR / Slow ISR	Ratio between the rate at which core motor control code and slower application code is executed

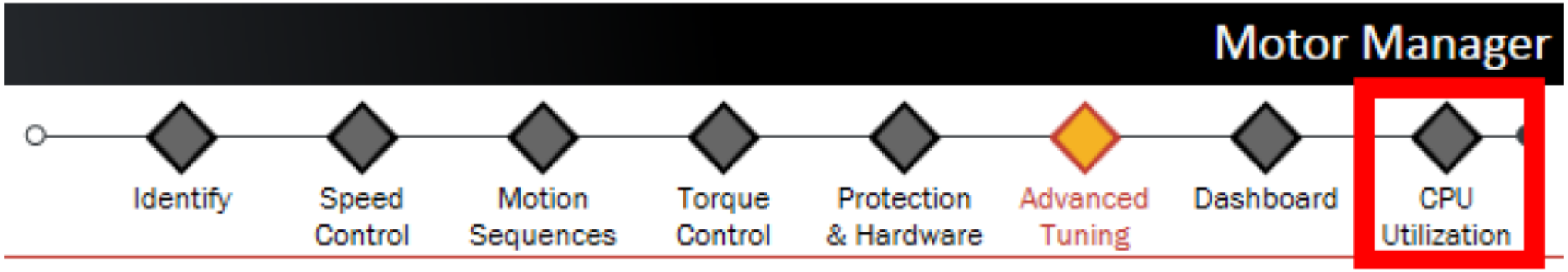
- The values for these variables depend on the values that KMS identifies during measurement of the motor’s electrical characteristics.

- For example, if a low inductance has been measured, KMS attempts to operate the motor at a faster switching rate than the default values.

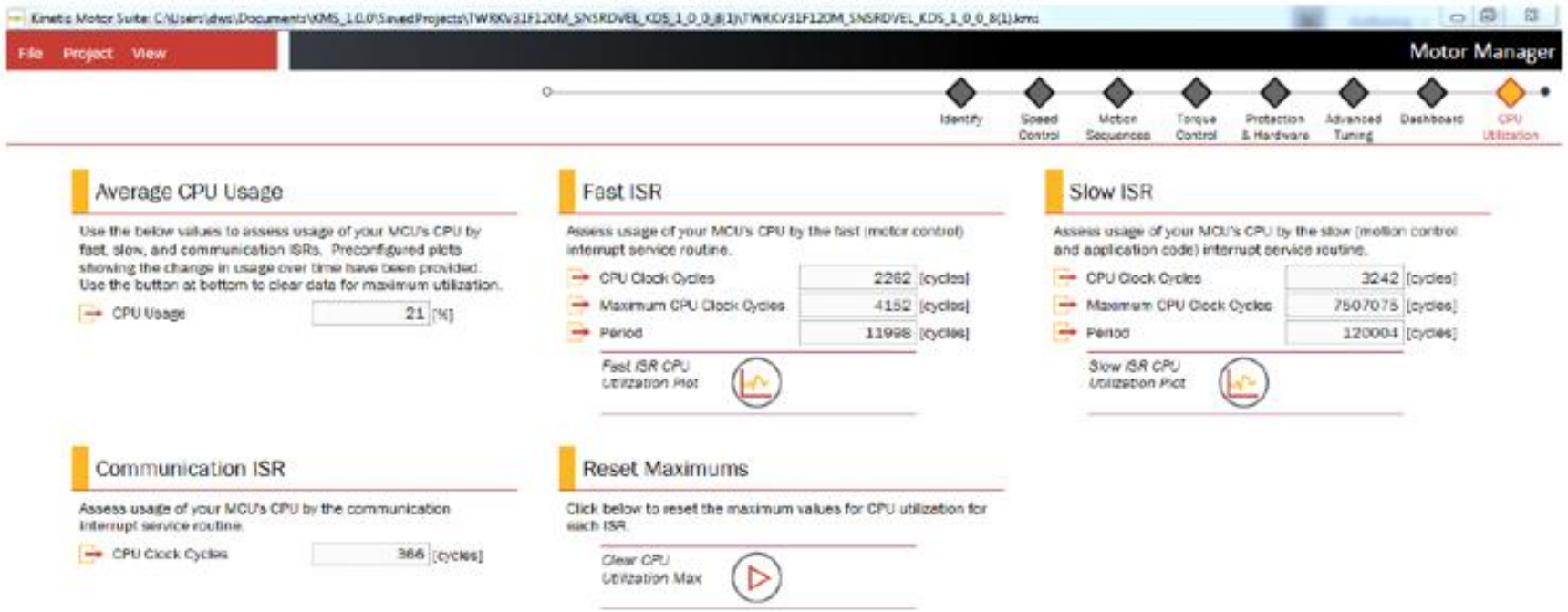
- You may also manually configure these values.

2. Assess CPU utilization using default configuration

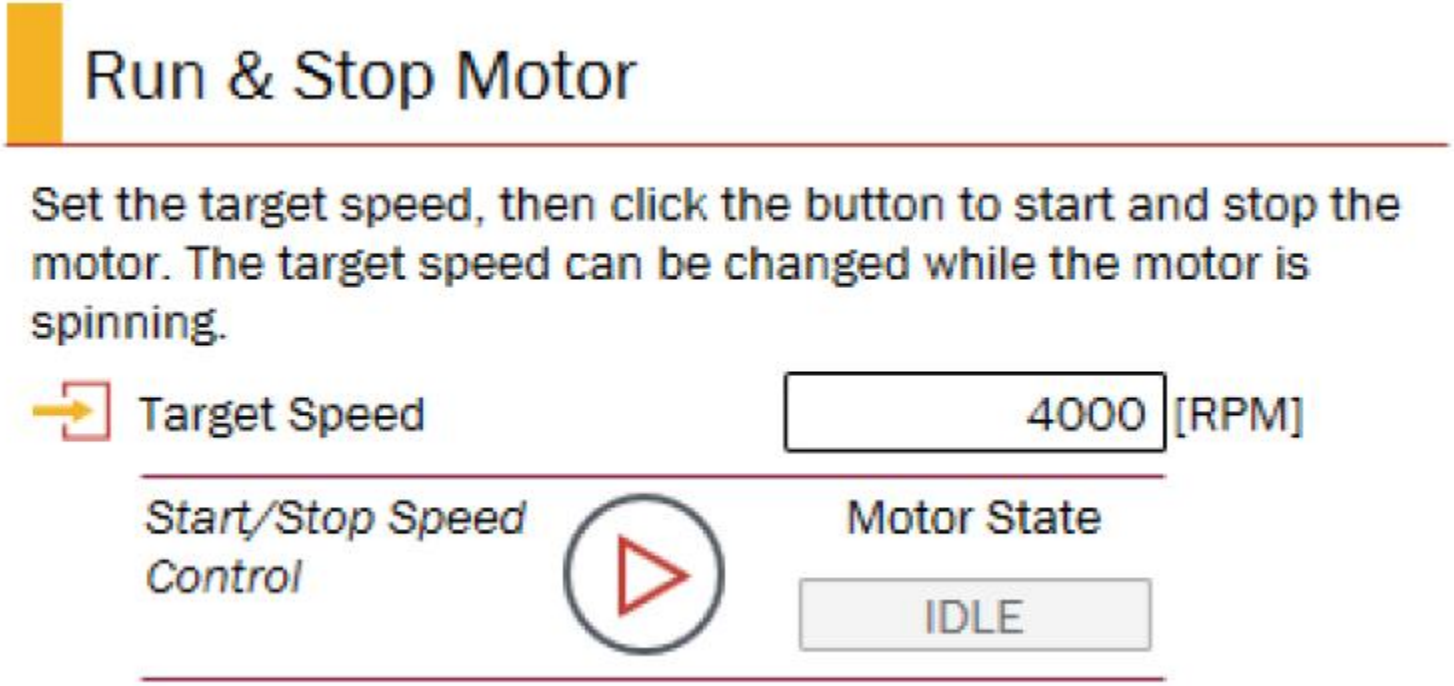
- Given the existing values in the System Configuration section, how much processor capacity is available for your code?
- To find out, start by navigating to the CPU Utilization page of Motor Manager.



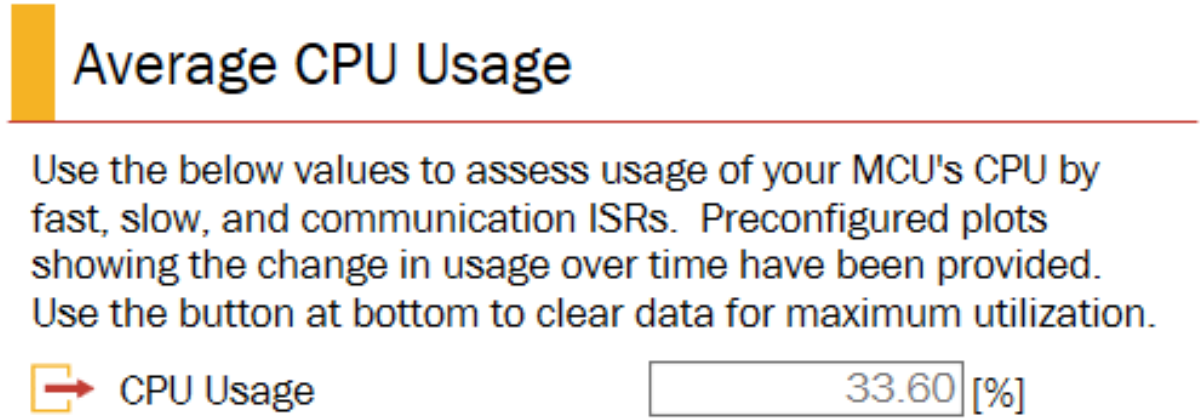
- Note the different information available:
  - Average percent usage of CPU
  - Fast ISR cycles
  - Slow ISR cycles
  - Communication cycles



- Now put the motor back into operation to see how these values change in real-time.
- Go to the Speed Control page, enter the motor's rated speed as Target Speed, and click to Start the motor.



- Return to the CPU Utilization page and observe CPU usage, both in terms of percentage and maximum cycles for the Fast ISR (motor control).
- Typical usage for operation at or below the motor's rated speed is shown next.



KV3x CPU usage running Linix 45ZWN24-40 in sensorless velocity mode at 4000 rpm

- Click to Stop Motor.

3. Increase values of execution rates and observe CPU impact

Certain motors may require faster operating frequencies to achieve good performance.

In particular, low inductance motors require usage of higher frequencies for PWM and the Fast (motor control) ISR.

If KMS' motor identification routine reveals your motor to be low inductance it will automatically adjust these values. But this has an effect on CPU usage.

- To simulate the impact on available CPU of using a low inductance motor, manually change these frequencies. Return to the System Frequencies section of the Advanced Tuning page in Motor Manager.

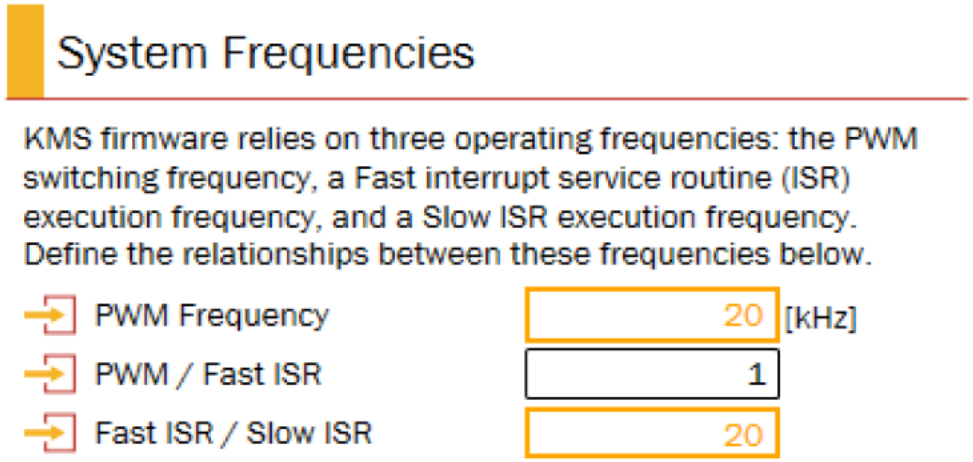
System Frequencies

KMS firmware relies on three operating frequencies: the PWM switching frequency, a Fast interrupt service routine (ISR) execution frequency, and a Slow ISR execution frequency. Define the relationships between these frequencies below.

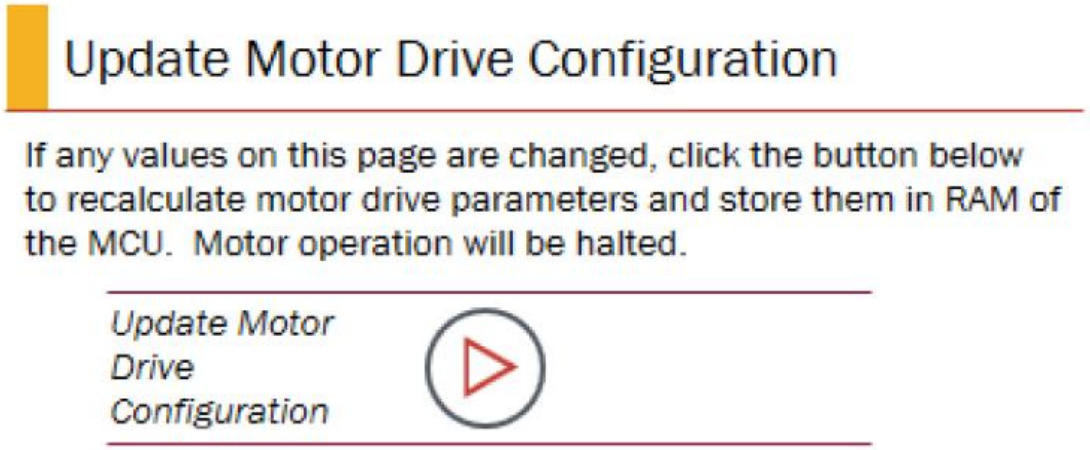
PWM Frequency	10	[kHz]
PWM / Fast ISR	1	
Fast ISR / Slow ISR	10	

- Increase the rate at which KMS code executes by changing the values to those listed below.

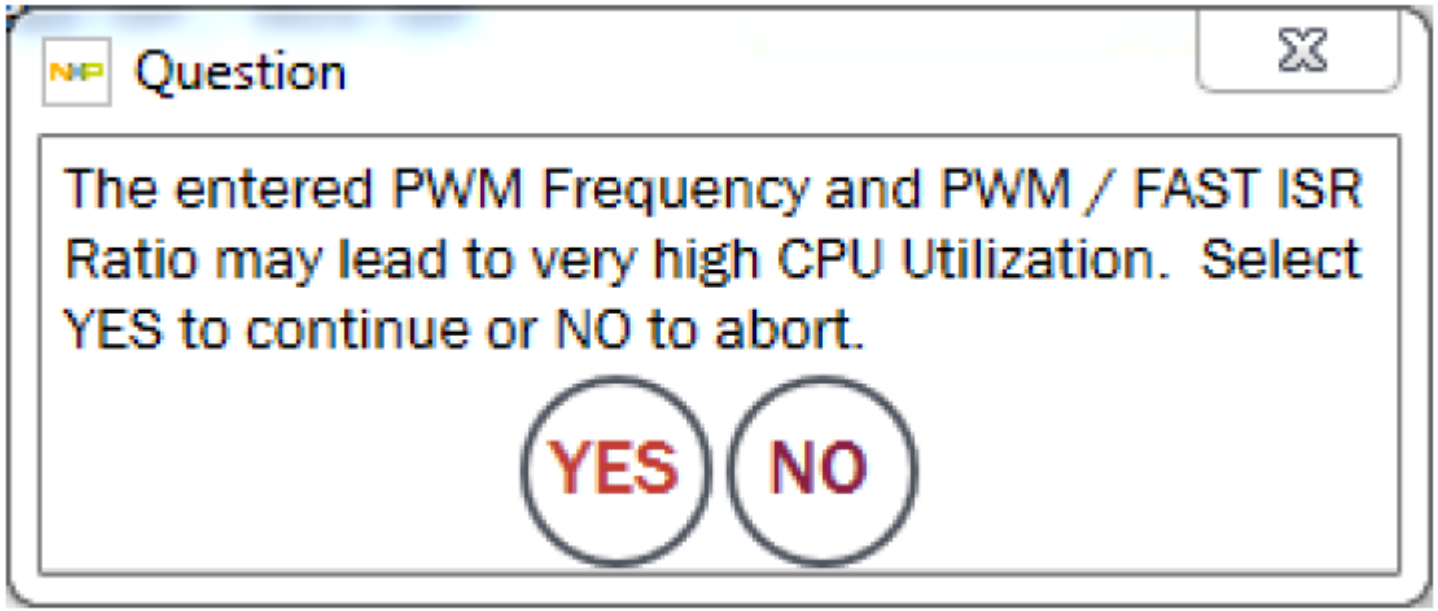
Variable	Current value	Change to
PWM Frequency	10	20
PWM / Fast ISR	1	1
Fast ISR / Slow ISR	10	20



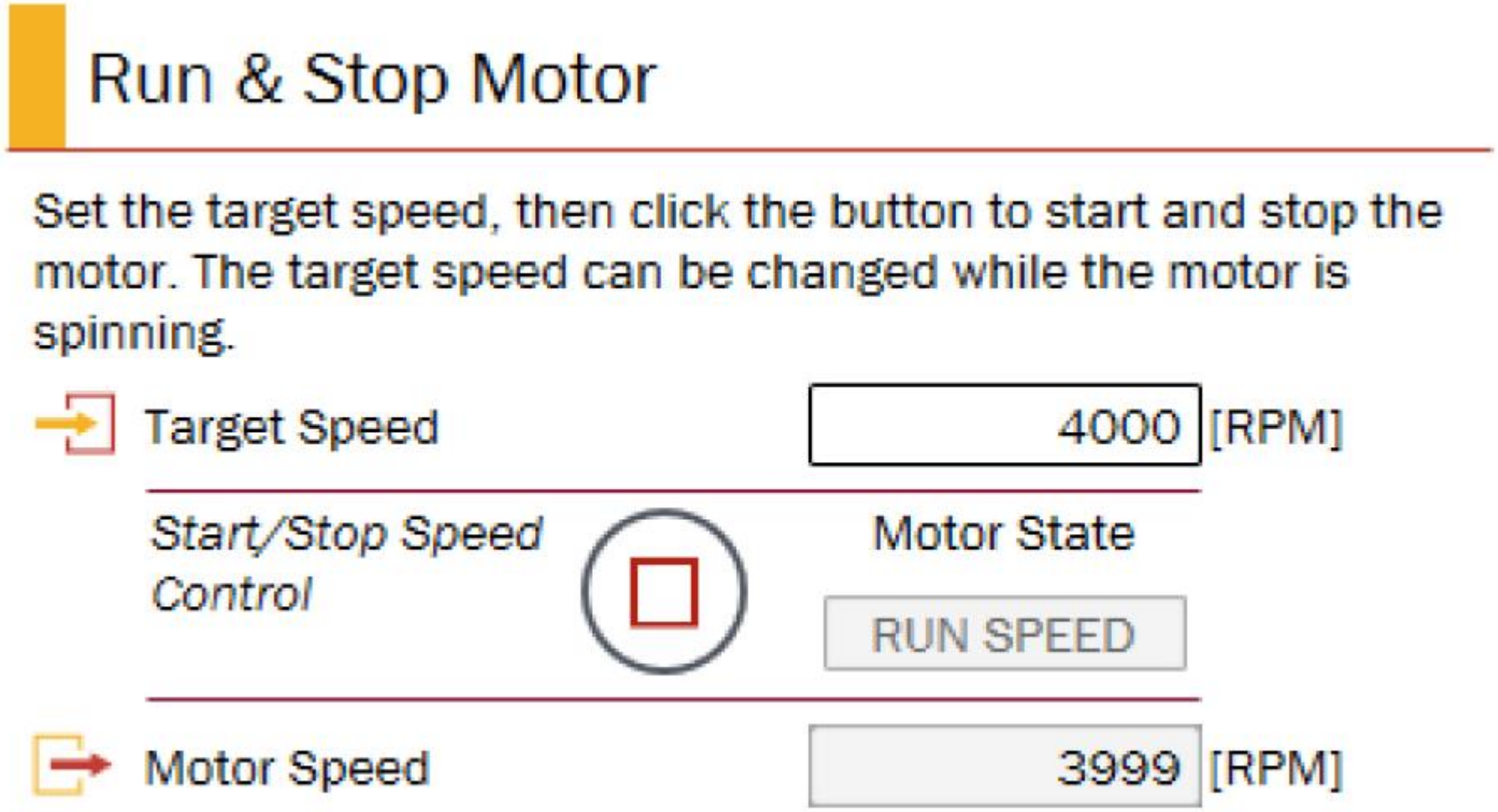
- Changes are highlighted in orange. This indicates updated values that have not yet been sent to the MCU. Send these values to the MCU by clicking the button to Update Motor Drive Configuration.



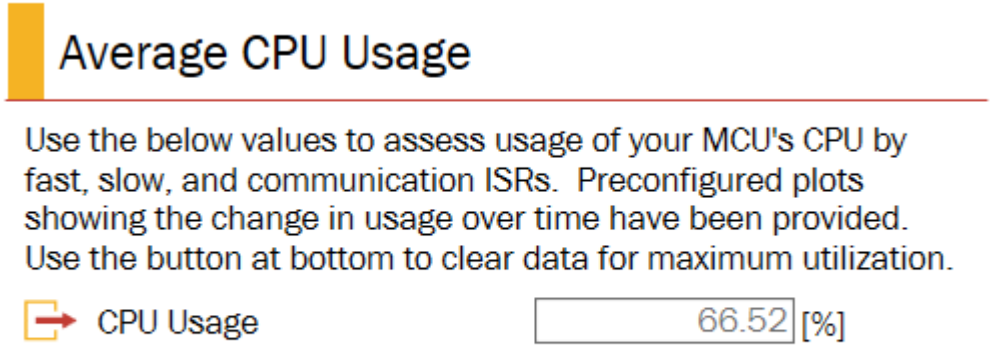
- You may receive a notification that these values may lead to high CPU utilization. Click Yes to accept.



- When values have updated and orange outlines disappear, go to the Speed Control page, enter Rated Speed as Target Speed and click to Start Motor.



- Return to the CPU Utilization page and observe the increase in usage.



- Stop the motor from the Speed Control page.

4. Decrease values of execution rates and observe CPU impact

Just as increasing motor control frequency results in higher CPU usage, decreasing the operating frequencies leads to lower CPU usage.

If your system is not especially dynamic but requires a substantial amount of application-focused code, you may want to assess how slow you can run the motor control code while still achieving acceptable motor performance.

This experiment enables you to maximize CPU available for your application code.

- Return to System Frequencies on the Advanced Tuning page. Change the execution rates in accordance with next table.

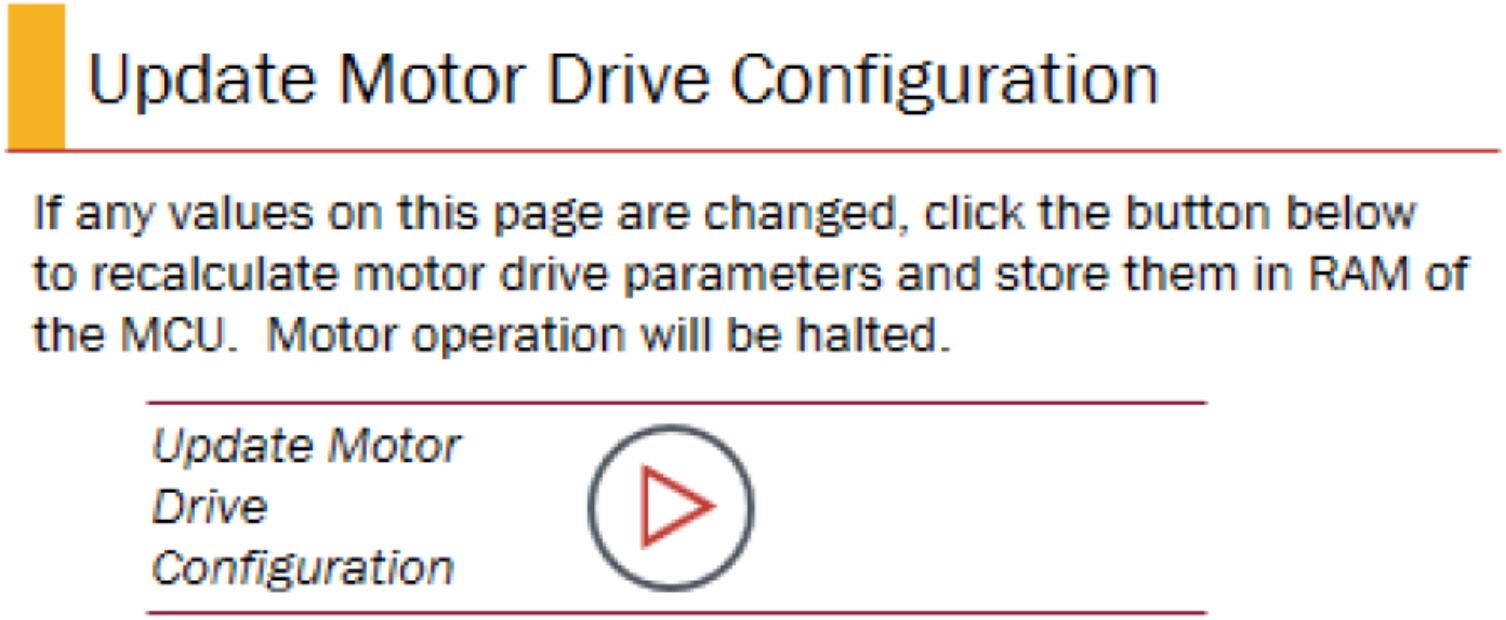
Variable	Current value	Change to
PWM Frequency	20	5
PWM / Fast ISR	1	1
Fast ISR / Slow ISR	20	5

System Frequencies

KMS firmware relies on three operating frequencies: the PWM switching frequency, a Fast interrupt service routine (ISR) execution frequency, and a Slow ISR execution frequency. Define the relationships between these frequencies below.

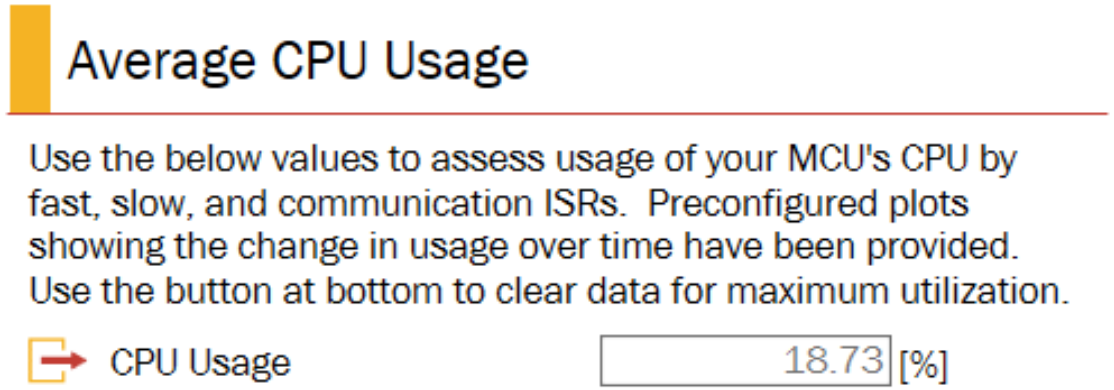
PWM Frequency	5	[kHz]
PWM / Fast ISR	1	
Fast ISR / Slow ISR	5	

- Click to Update Motor Drive Configuration.

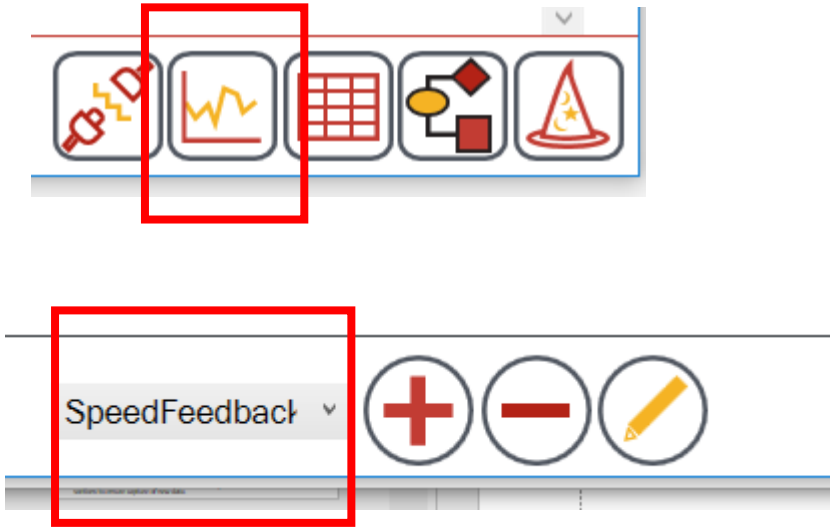


- When values have updated and orange outlines disappear, go to the Speed Control page, enter Rated Speed as Target Speed and click to Start Speed Control.

- Your motor may not be able to reach Rated Speed with this configuration. If it does, note the reduced usage on the CPU Utilization page . You may also notice increased audible noise due to the lower frequency of operation.



- If your motor is successfully running at Rated Speed, activate the Software Oscilloscope, select the Speed Feedback Plot, and begin sampling.

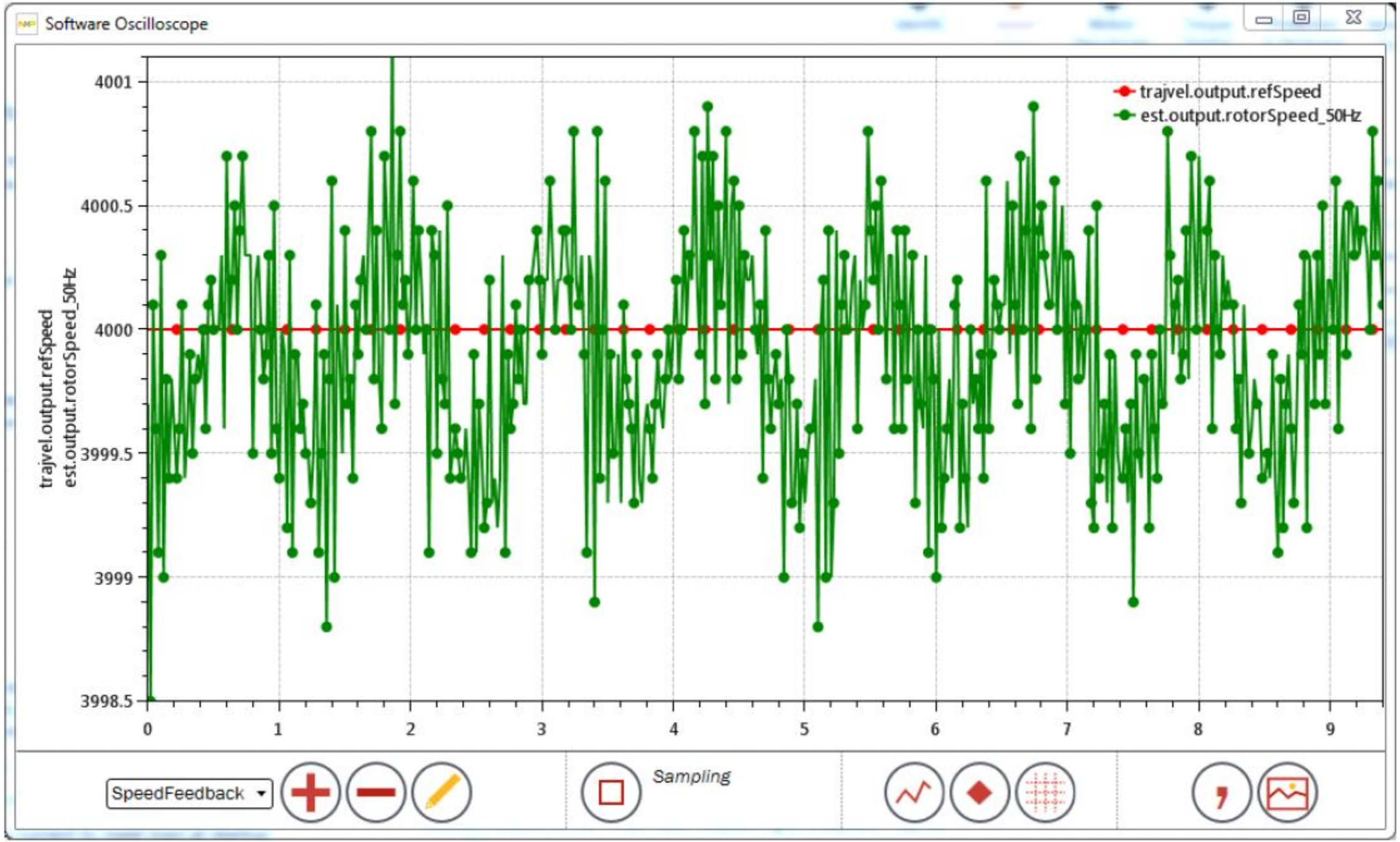


- Note that there may be degradation of control performance while running at the lower execution rates.

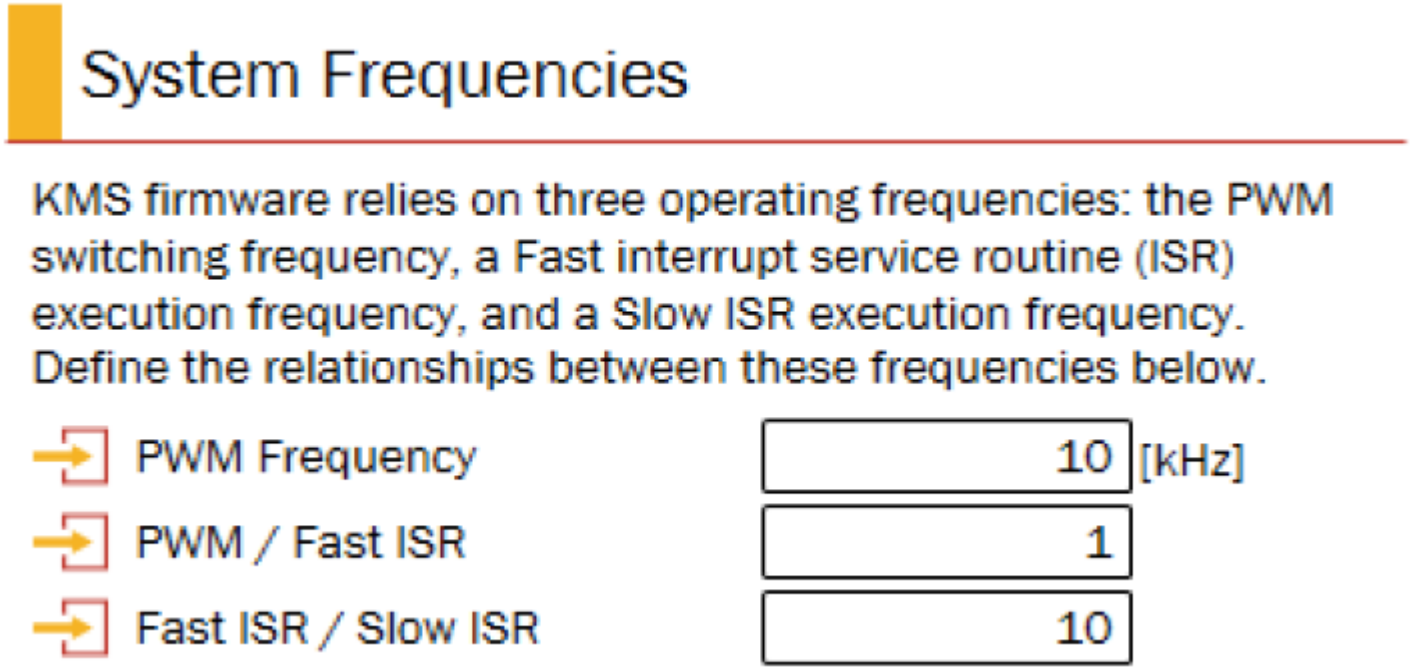
- This is most likely to occur when running in sensorless velocity mode.

- These are the types of design tradeoffs that must be evaluated during the design cycle.

- Reducing the execution rate may free up the processor, but control performance may suffer.


Software Oscilloscope
trajvel.output.refSpeed
est.output.rotorSpeed_50Hz
trajvel.output.refSpeed
est.output.rotorSpeed_50Hz
4001
4000.5
4000
3999.5
3999
3998.5
0
1
2
3
4
5
6
7
8
9
SpeedFeedback
Sampling

- Stop sampling and on Speed Control page, click to Stop Motor.
- Change the System Frequencies back to their default values and Update Motor Drive Configuration.



5. Change motor operating mode and observe CPU impact

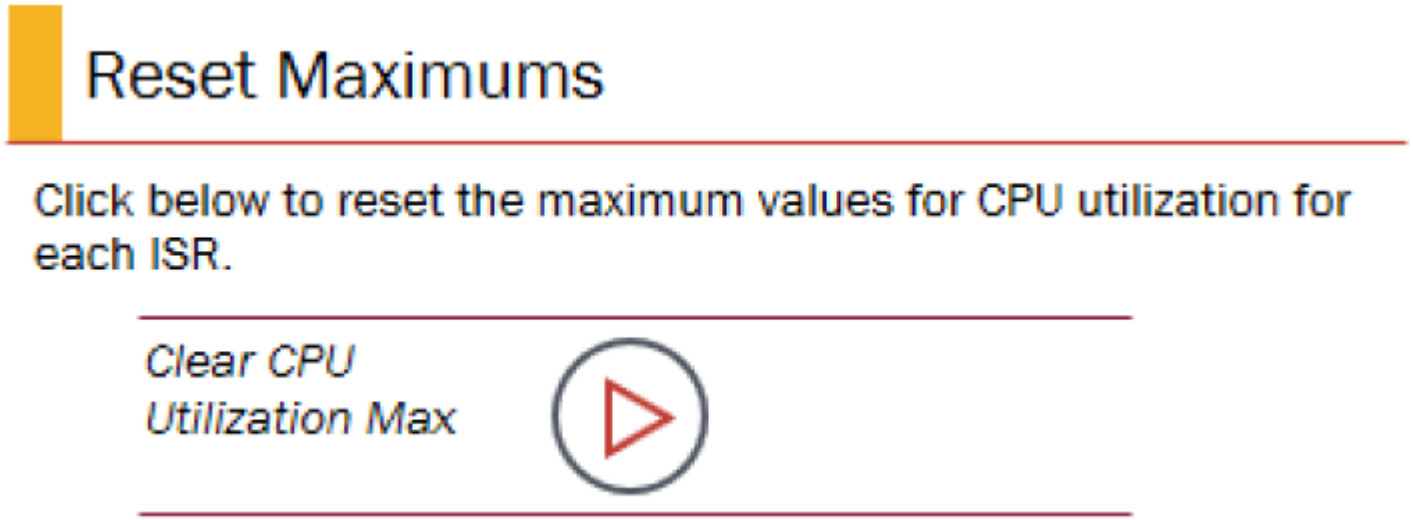
KMS seamlessly enables different modes of motor operation. Indeed, several different modes are behind the Motor Tuner wizard process: motor measurement, spinning to rated speed, and simulating an application constitute three slightly different arrangements of core KMS firmware modules.

Since the underlying software is implemented differently for different modes of operation, usage of the CPU may change based on mode. In this section, explore two such scenarios.

- On the Speed Control page, enter your motor’s rated speed as Target Speed and click to Start Speed Control.

- Navigate to the CPU Utilization page. Find the Reset Maximums section and click to Clear CPU Utilization Max.

- This will clear the values for Maximum CPU Clock Cycles in the Fast ISR and Slow ISR sections to ensure capture of new data.



- As the motor runs at Rated Speed, observe the CPU usage percentage and the Maximum CPU Clock Cycles for the Fast ISR.

- This is your baseline for comparison: normal motor operation at Speed.

Average CPU Usage

Use the below values to assess usage of your MCU's CPU by fast, slow, and communication ISRs. Preconfigured plots showing the change in usage over time have been provided. Use the button at bottom to clear data for maximum utilization.

CPU Usage 34.00 [%]

Fast ISR

Assess usage of your MCU's CPU by the fast (motor control) interrupt service routine.

CPU Clock Cycles 3751 [cycles]

Maximum CPU Clock Cycles 3983 [cycles]

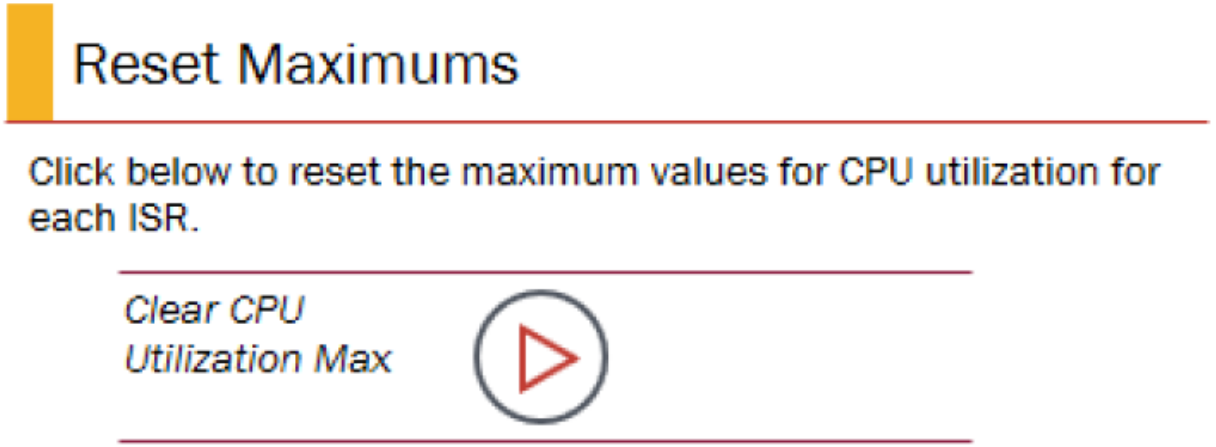
Period 12007 [cycles]

Fast ISR CPU Utilization Plot

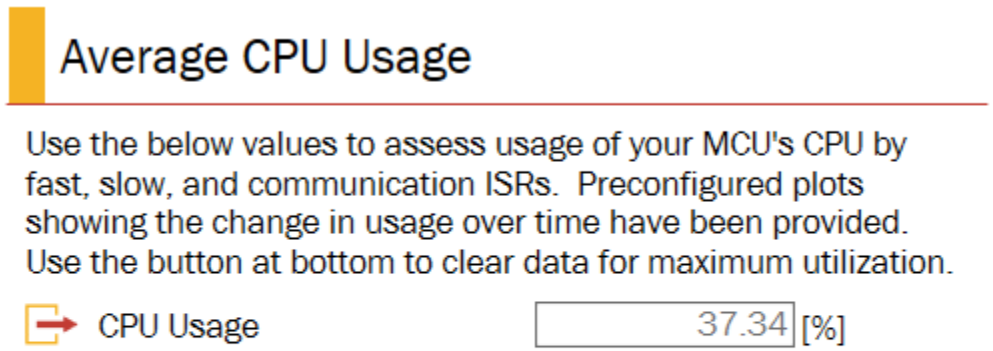
- Return to the Speed Control page and click to Stop the motor.

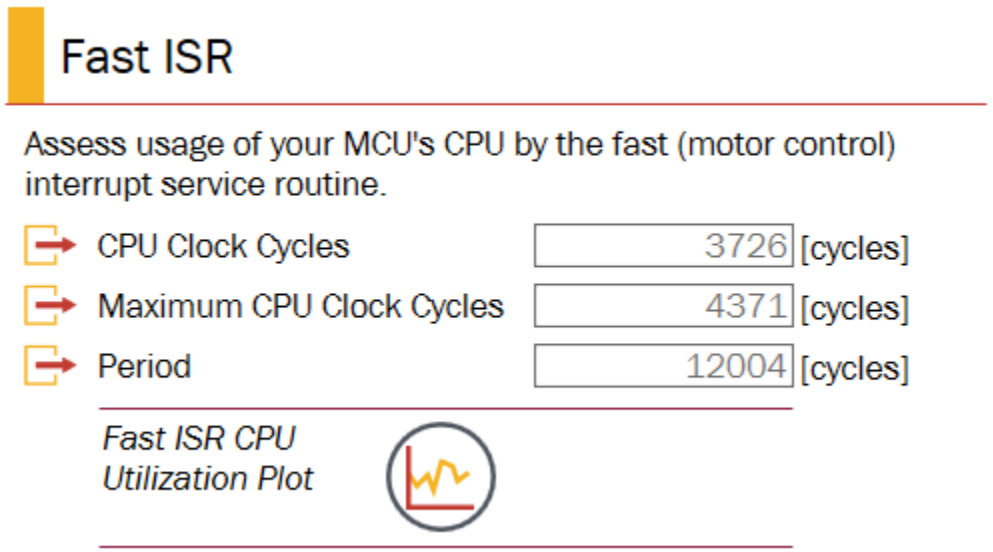
- Now assess CPU usage while running an application motion sequence. Navigate to the Motion Sequences page and click to Start Motion Sequence.

- As your motor begins to perform the washing machine motion sequence, return to the CPU Utilization page. Click to Clear CPU Utilization Max.



- Observe the values update for CPU Usage and Fast ISR Maximum CPU Clock Cycles.
- Note that running an application motion sequence may add to the CPU burden.
- This effect is likely to be more pronounced with more complex motion sequences.





- When the motion sequence completes, return to the Speed Control page.

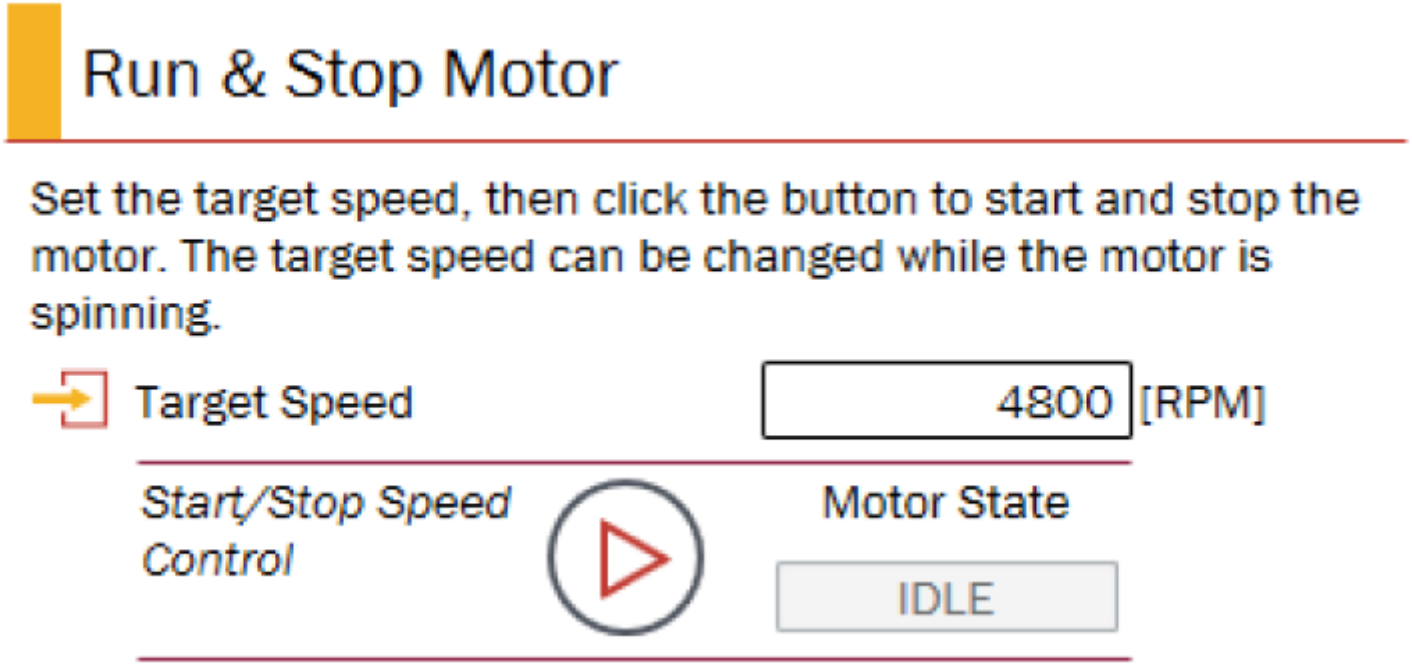
6. Engage field weakening and observe CPU impact

CPU usage may also be affected by attempting to run beyond the motor's normal operating range.

By default, KMS allows "field weakening," which manipulates the current signals to achieve greater speed for the motor at the cost of torque.

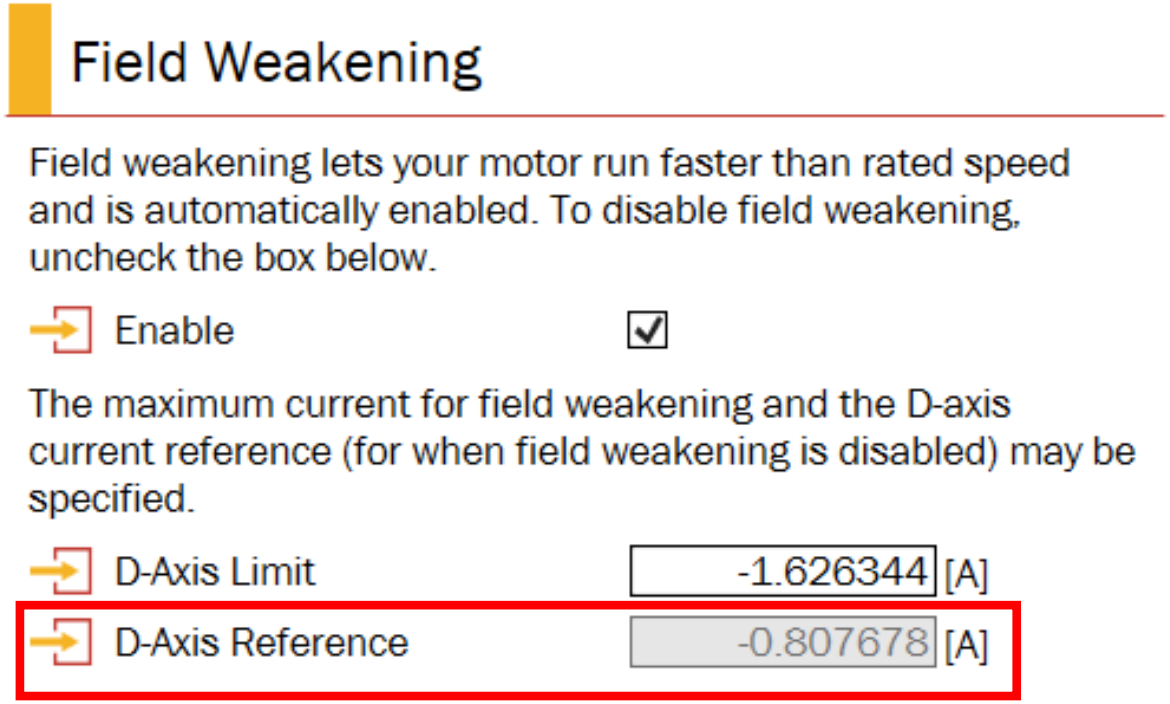
However this requires additional computation and so may require additional CPU cycles.

- To observe this effect, enter a speed 20% greater than rated speed into the Target Speed field and click to Start Speed Control.

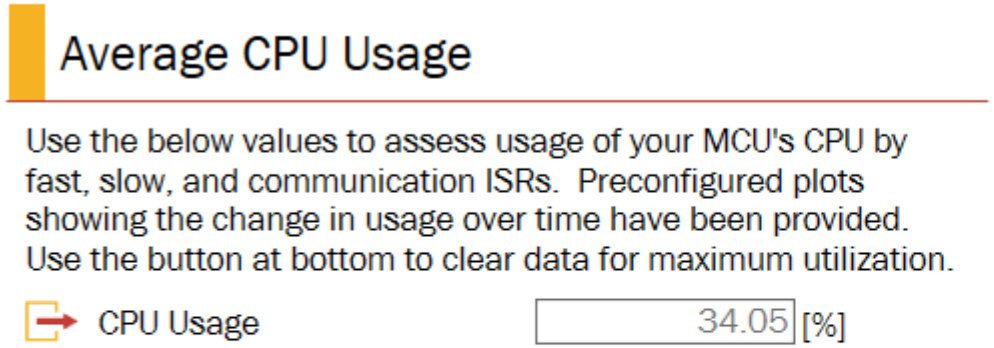


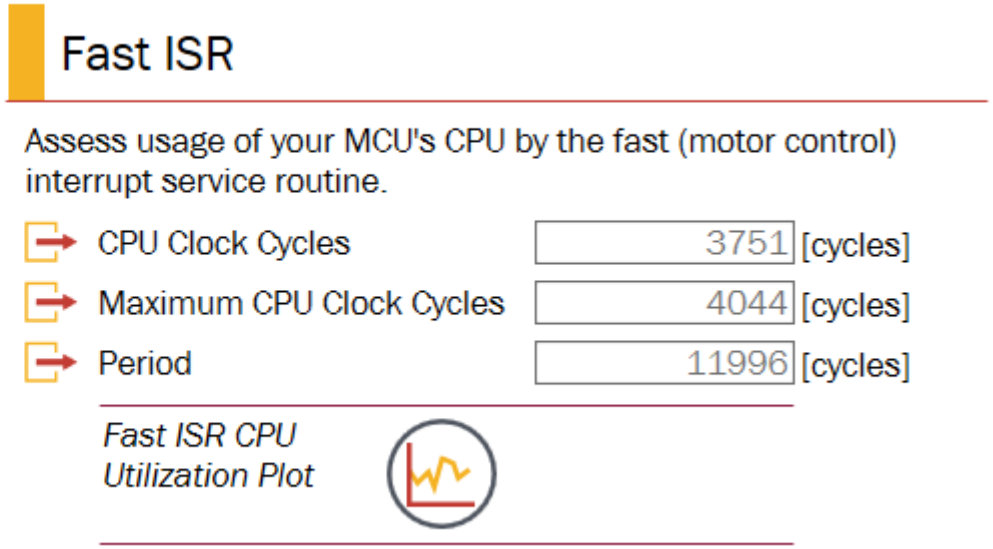
- As the motor surpasses rated speed, scroll down to the Field Weakening step and observe the value for D-Axis Reference. Normally, this value should be zero, because applying current on this axis reduces torque.

- However, in field weakening, reduction in torque is acceptable to achieve increased speed, so this value must become nonzero (and in fact, go negative).



- After D-Axis Reference has gone negative, return to the CPU Utilization page.
- Click to Clear CPU Utilization Max to reset values, then observe usage statistics and in particular, the Fast ISR Maximum CPU Clock Cycles.





- For the motors specified by Kinetis development platforms, the utilization may not differ from utilization at normal operating speeds.
- This is because these motors are not constructed to optimize field weakening operation.
- However, for motors designed to operate in deep field weakening, such as washing machine motors, the CPU usage may increase substantially.

- Stop motor from Speed Control page.

Summary

- In this part of the lab, you performed the following steps to observe the tradeoff between processor utilization and motor control operation:
  - Viewed CPU utilization statistics in normal operation
  - Changed system frequencies that might need to be switched for your motor
  - Spun the motor in different operating modes

Part II: Build a simple motion sequence

Objective

- Now that you have an idea of valid motor control operation, move on to building out an application.

- Motion Sequence Builder allows you to define the speeds at which you want your application to run, as well as the conditions under which it runs at each speed.

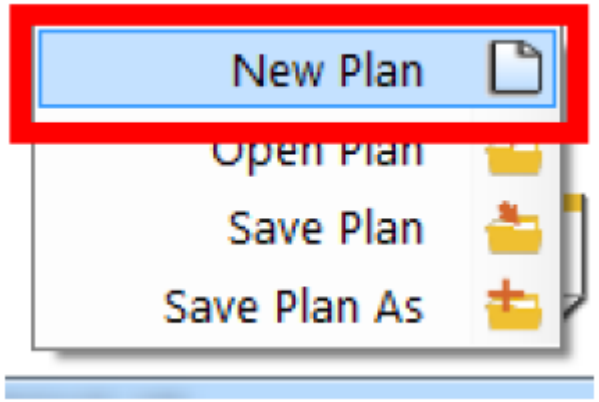
- Use Motion Sequence Builder to create a simple ceiling fan consisting of three speeds and a “button” to cycle through the speeds.

Procedure

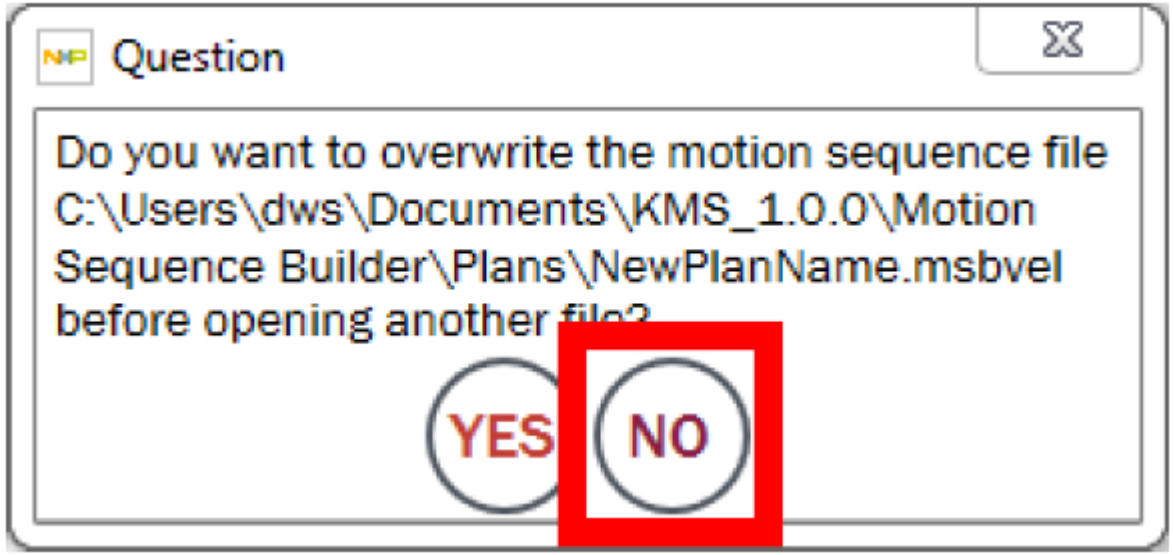
1. Set up and run desired speeds

- Bring Motion Sequence Builder to the front by clicking the diagram icon at bottom right.

- Start a new motion sequence by left-clicking on the file folder at bottom left then selecting New Plan.



- If prompted, click No to avoid overwriting the open motion sequence.



- Motion Sequence Builder refreshes and display the default configuration: two states enumerated where one is the Initial State of zero speed.

States

At what speeds do you want your motor to run?

	Speed [rpm]	Name
	0	InitialState
▶	2000	State1

- On the States page, click the Add button twice to add two additional states to the default configuration.

States

At what speeds do you want your motor to run?

	Speed [rpm]	Name
	0	InitialState
	2000	State1
	2000	State2
▶	2000	State3

- Click in the cell for Speed of State1 and type to change value to 1000. Press Enter.

States

At what speeds do you want your motor to run?

	Speed [rpm]	Name
	0	InitialState
▶	1000	State1
	2000	State2
	2000	State3

- Click in cell for Name of State1 and type to change value to Low_Speed. Press Enter.

States

At what speeds do you want your motor to run?

	Speed [rpm]	Name
	0	InitialState
▶	1000	Low_Speed
	2000	State2
	2000	State3

- Change State2 and State3 according to the table.

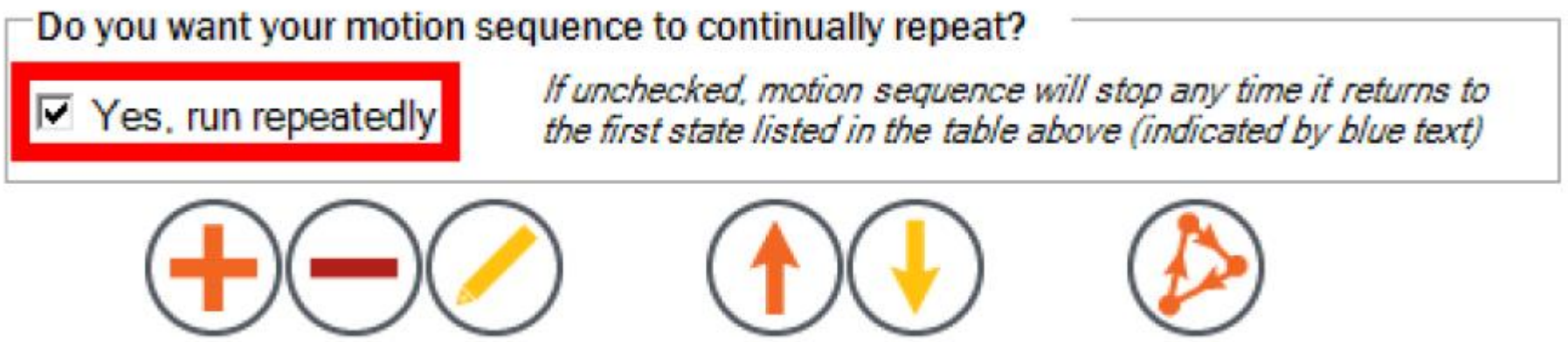
Speed [rpm]	Name
2500	Medium_Speed
4000	High_Speed

States

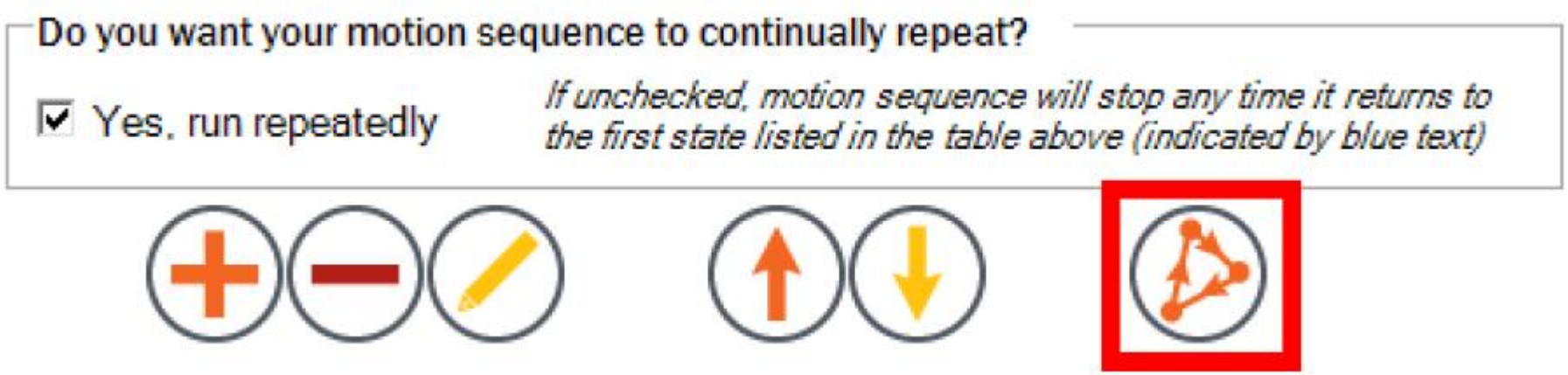
At what speeds do you want your motor to run?

	Speed [rpm]	Name
	0	**InitialState**
	1000	Low_Speed
	2500	Medium_Speed
▶	4000	High_Speed

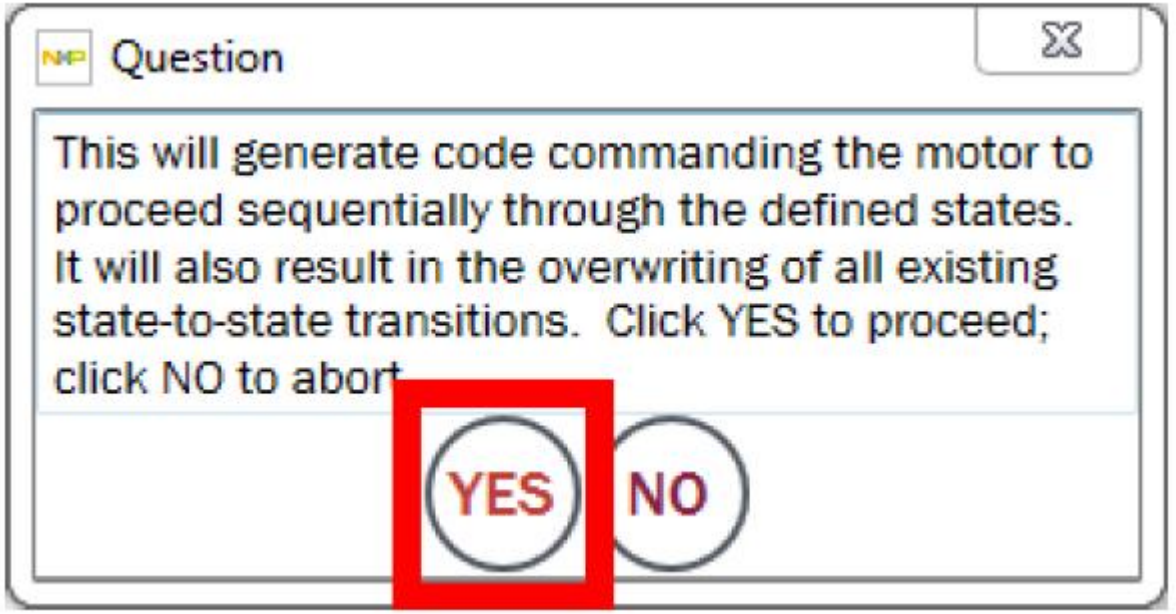
- Check the box in order for your motion sequence to continually run instead of halting upon returning to the InitialState.



- Click the Connect States button to connect the defined states in a simple loop (InitialState *TO* Low_Speed *TO* Medium_Speed *TO* High_Speed *TO* InitialState).



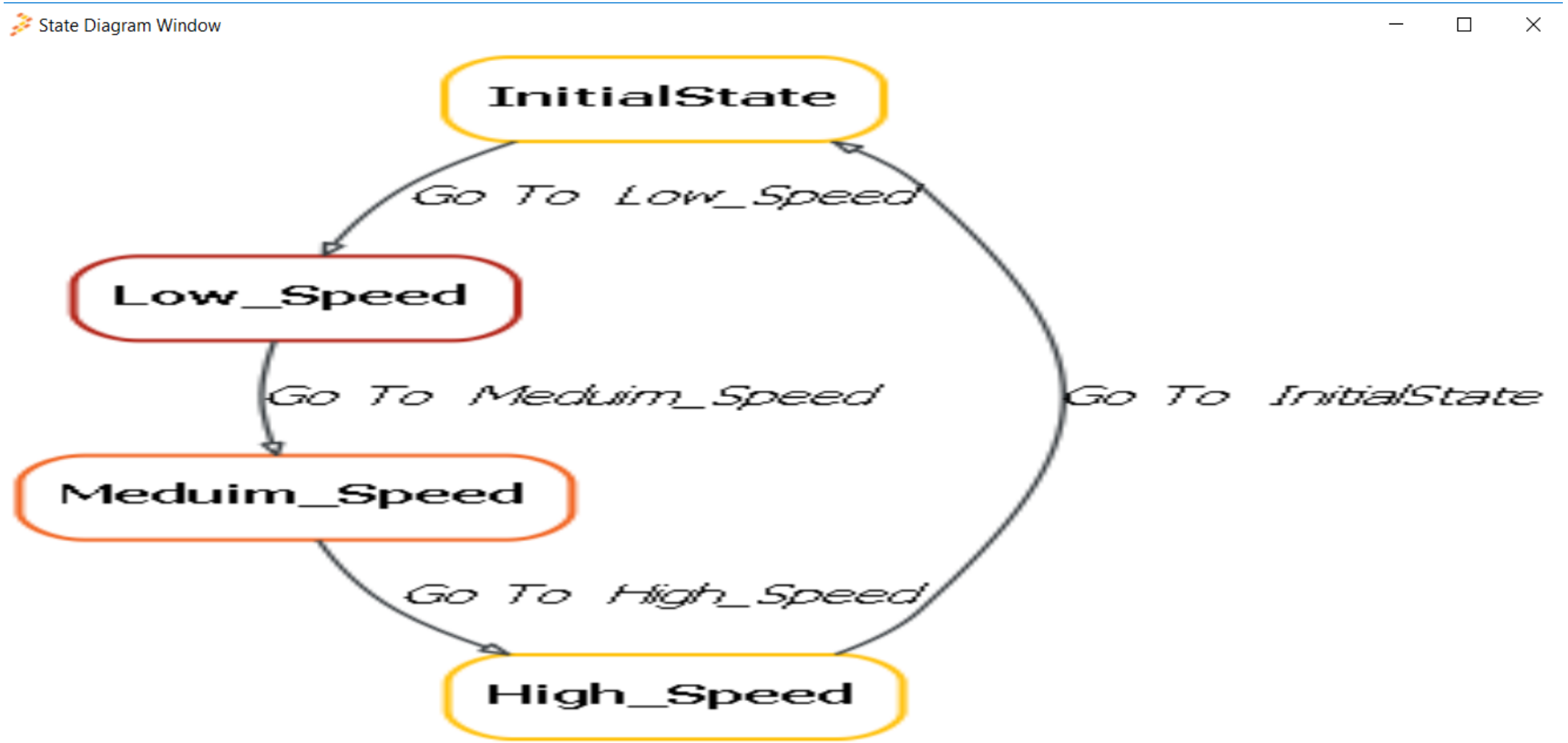
- Click Yes to accept that clicking the Connect States button overwrites any existing state-to-state transitions.



NOTE:

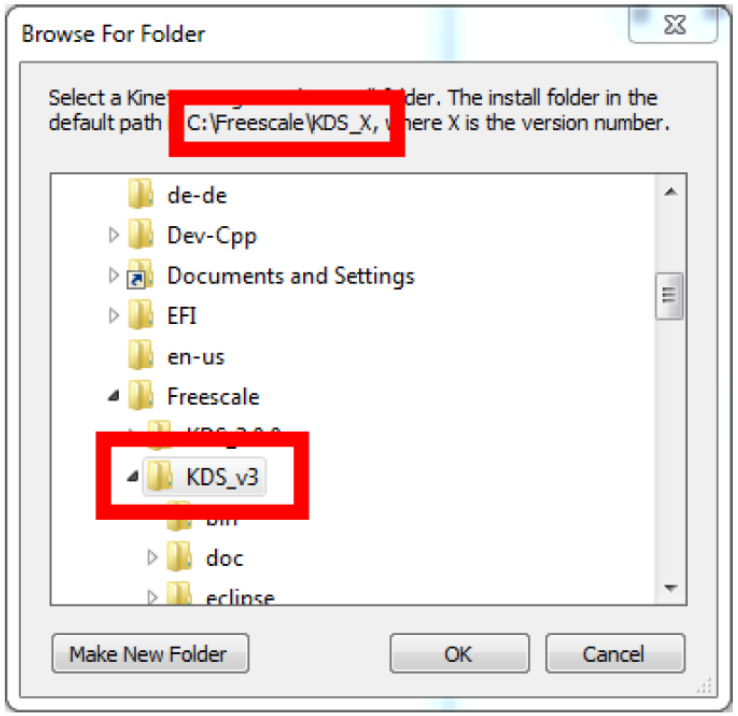
The Connect States button is most useful at the beginning of a simple design when interested in validation of motor operation at specified speeds. After you have specified numerous states and transitions among those states, be careful when using the Connect States button to avoid overwriting any desired behavior.

- Observe in your system's default image viewer a diagram showing the relationships between the states created by the Connect States button.



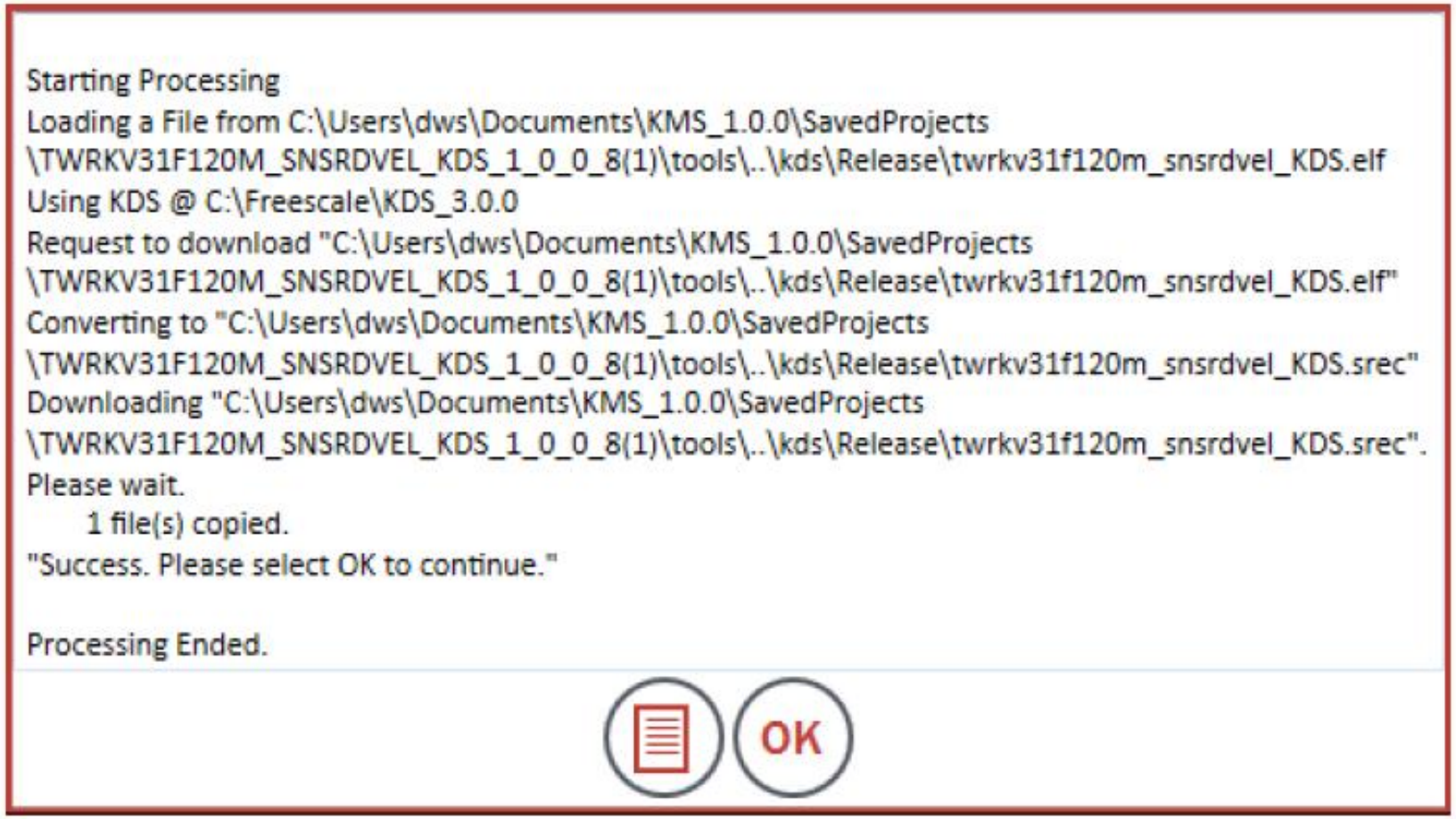
- Exit the image viewer. Click the Run on Target (bullseye) button to compile your KMS reference project with your new motion sequence.

- If you have not previously specified your Kinetis Design Studio path, KMS prompts for its location.



- A command line-style window appears. Wait while your reference project is compiled with the relevant Kinetis Software Development Kit libraries, then click OK.

- The result of compilation is then downloaded to the MCU.

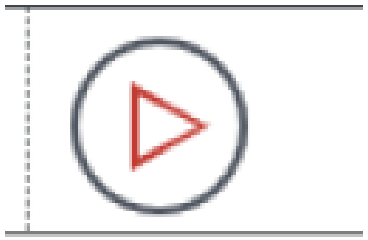


- Click OK to proceed.
- Wait until the command window disappears and return to the KMS Main Window to wait for KMS to reconnect to the MCU (the command window operation requires temporary usage of the communication port). Connection is indicated by green arrows on the communication indicator.

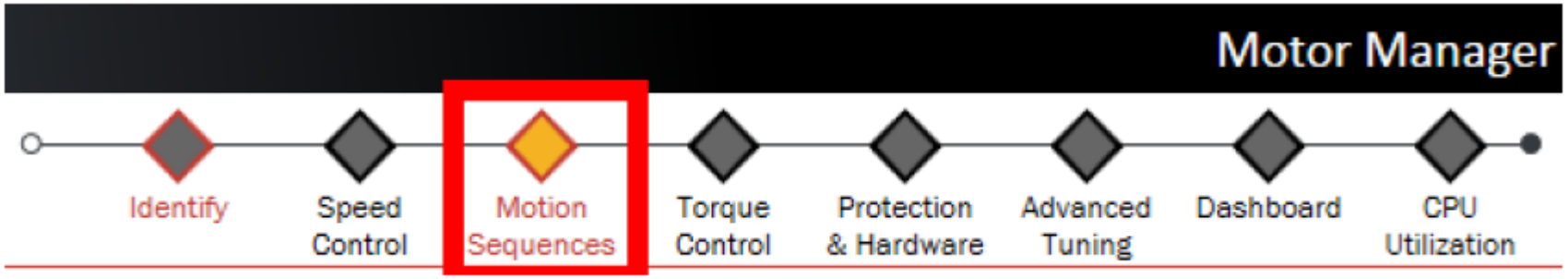
- Activate Software Oscilloscope.

- Select SpeedFeedback plot from dropdown menu at bottom left of Software Oscilloscope window.

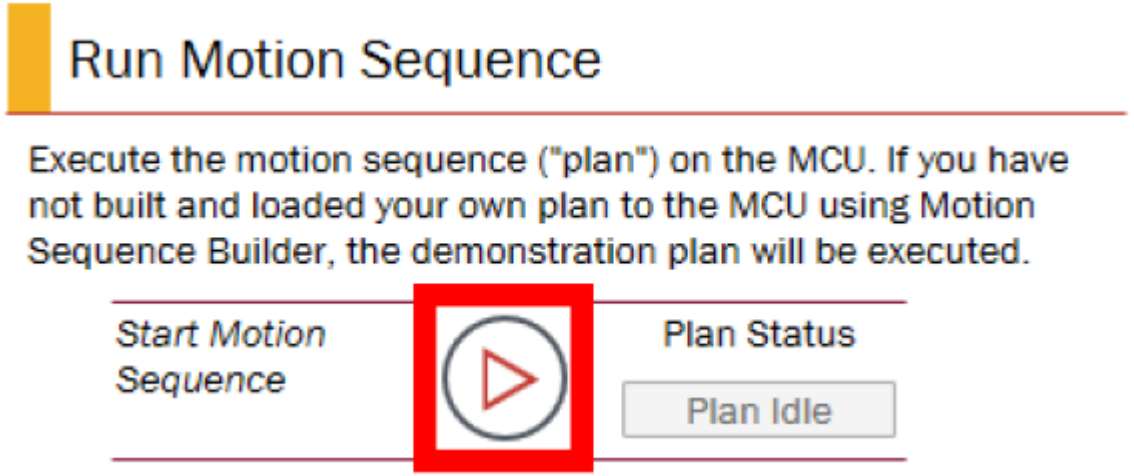
- Click Run button to begin sampling.



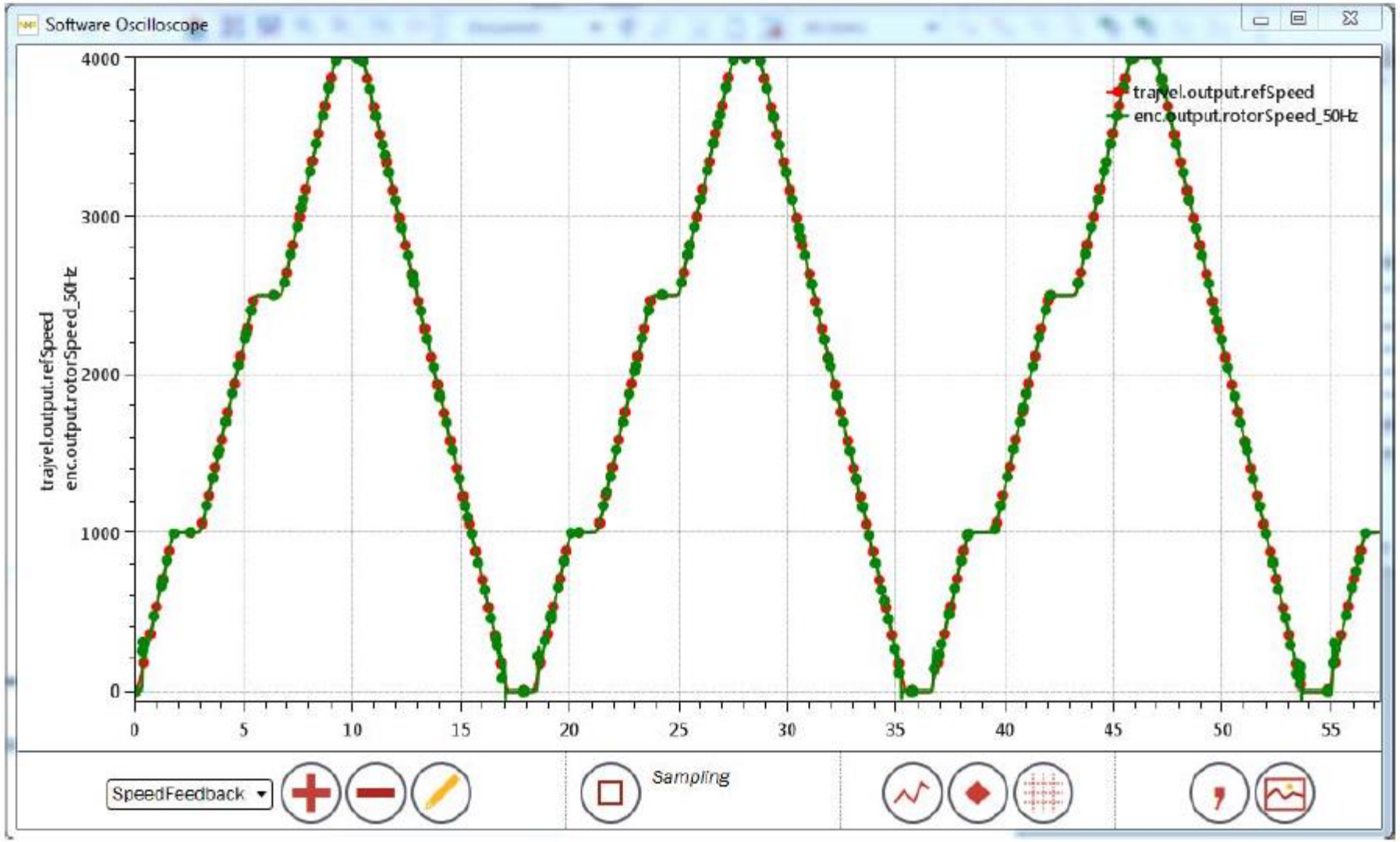
- In the KMS Main Window, navigate to the Motion Sequences page in Motor Manager.



- Find the Run Motion Sequence step and click to Start Motion Sequence. This runs the current motion sequence loaded to the MCU.

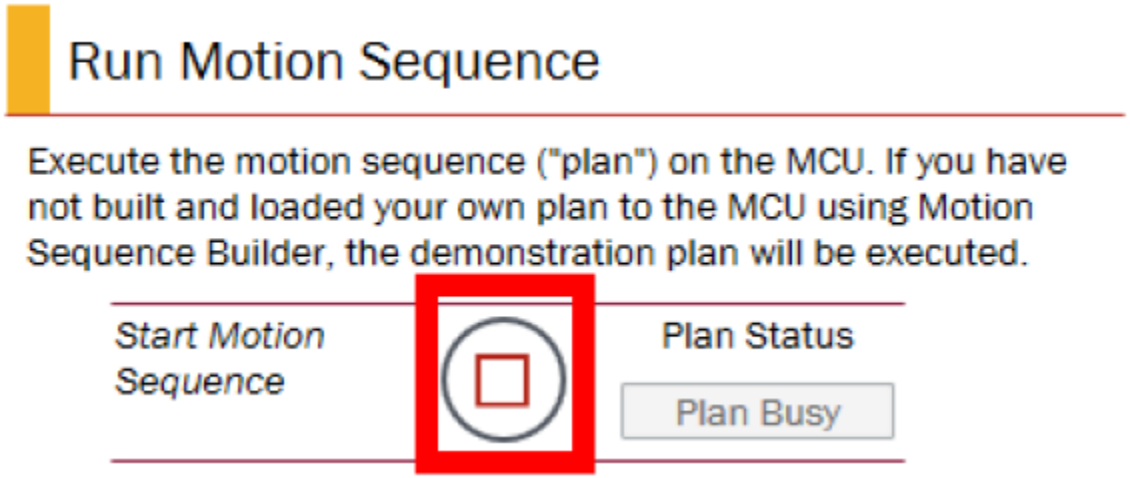


- Return to the Software Oscilloscope and watch the motor cycle through the speeds you defined.



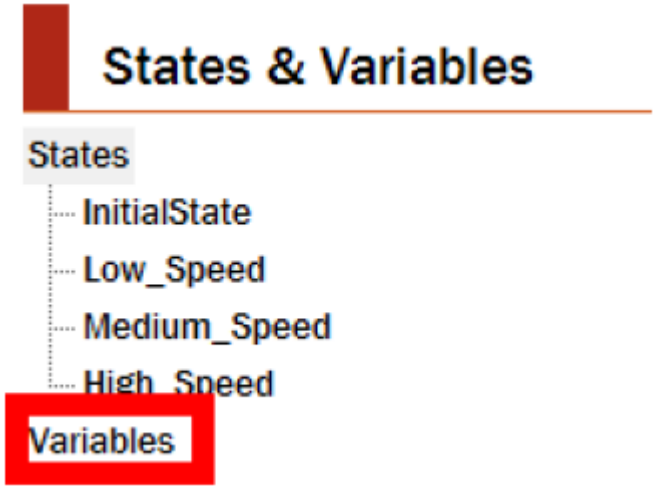
- Click Stop button to stop sampling in the Software Oscilloscope window.

- On Motion Sequences page, click Stop button to stop the motion sequence.

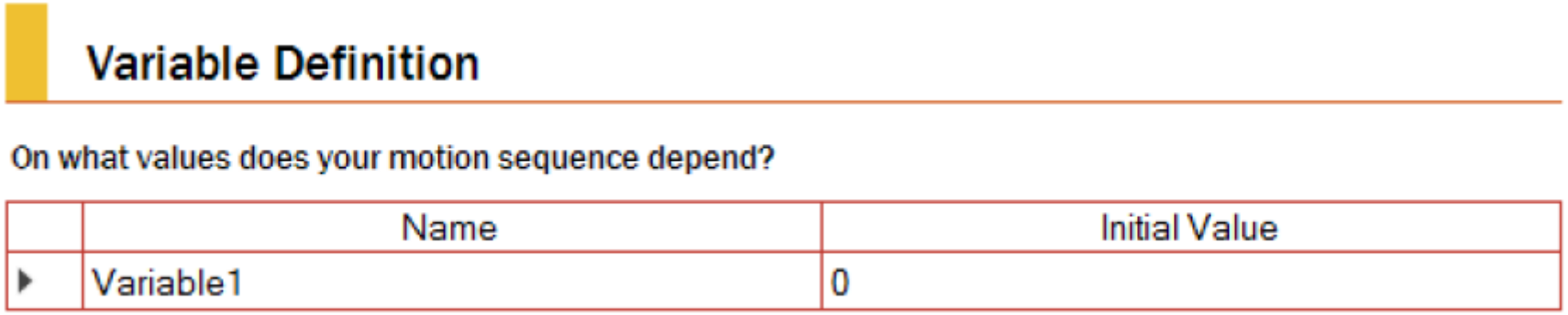


2. Make the motion conditional

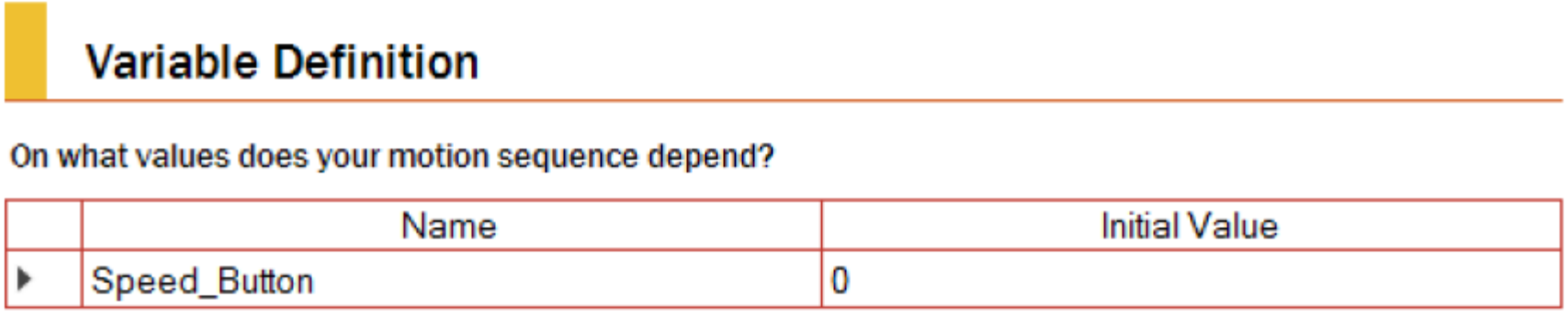
- Return to Motion Sequence Builder. Click on Variables at top left.



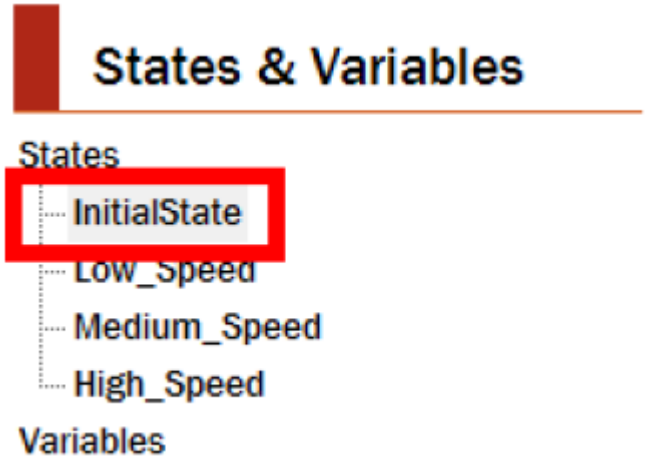
- Click Add button to add a variable .



- Click on Variable1 cell and rename by typing in Speed_Button and pressing enter.
- Note the Initial Value of the Speed_Button variable is 0.



- Click on InitialState at top left to see further detail about the InitialState



On the next page and corresponding pages for other states, you can define:

- the speed that defines the state
- the minimum time the motor remains in this state
- conditions for KMS to evaluate while in this state
- the actions resulting from a satisfied condition
- the motion limits for any transition to another state

InitialState

Basic state information

Speed: 0 [rpm] Minimum Time: 1000 [ms]

Define if-then relationships

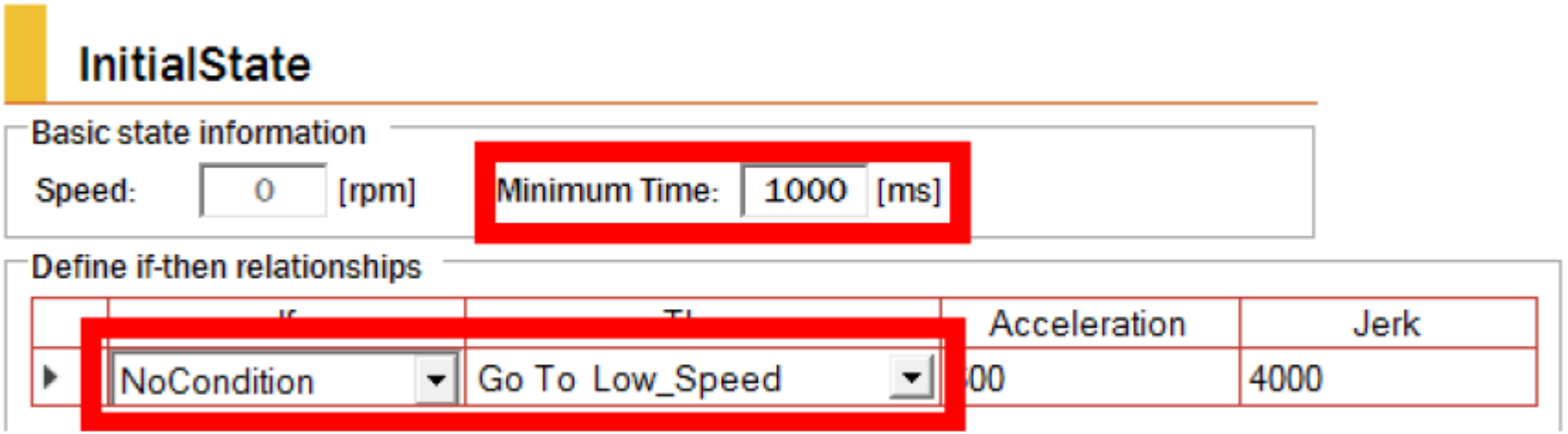
	If	Then	Acceleration	Jerk
▶	NoCondition	Go To Low_Sp...	600	4000

Define global if... statements

NoCondition

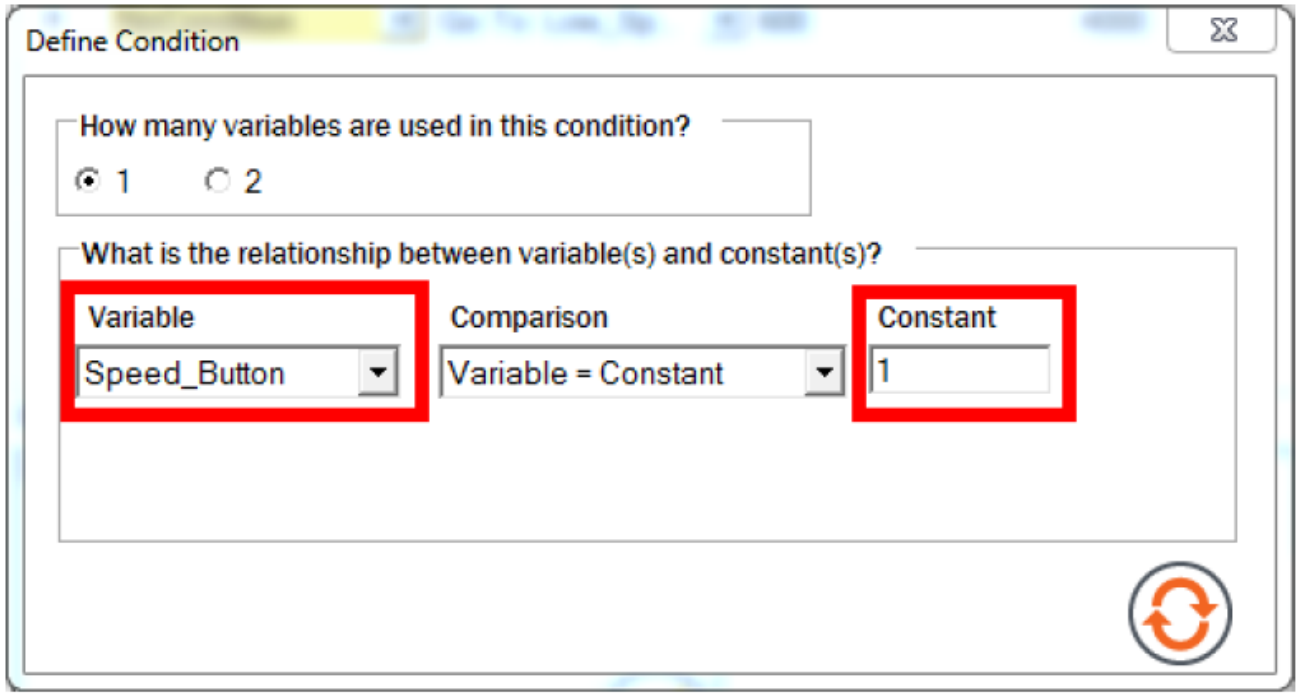
Define global then... statements

- Note that when in the InitialState, the motor is configured to go to the state Low_Speed under NoCondition.
- That is, the motor goes to Low_Speed as soon as the Minimum Time for the InitialState has elapsed, without any other condition having to be satisfied.



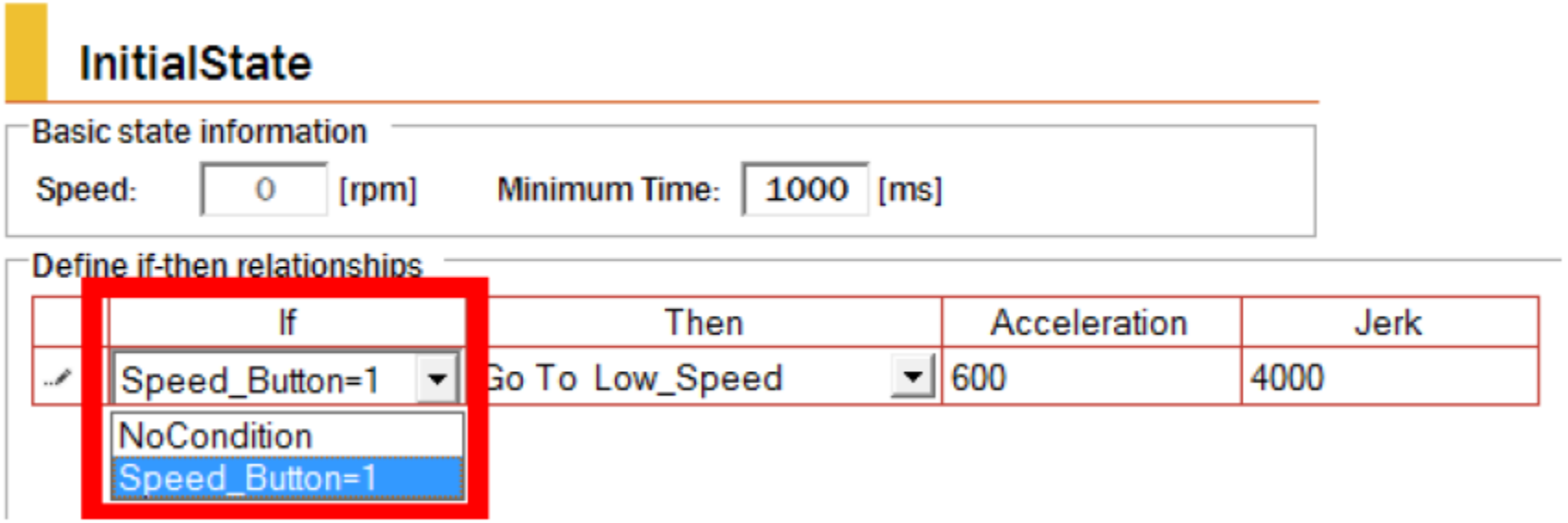
- To make this transition from InitialState to Low_Speed conditional, click the Add button to add a global if statement.

- In the dialog box that appears, make sure the variable Speed_Button is selected, then type in the Constant field a value of 1. This defines the if statement:
  - If Speed_Button equals 1



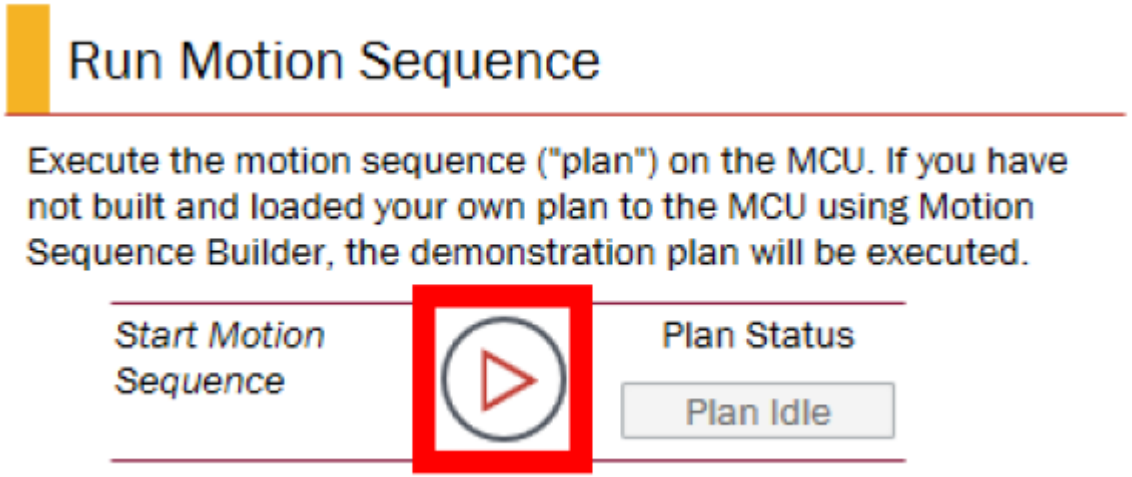
- Click the Update button to create this if statement.

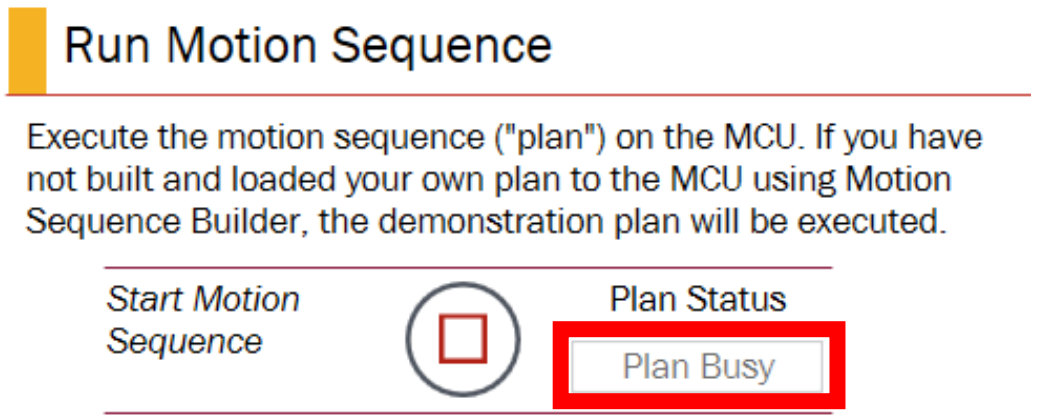
- Navigate to the If column in the If-Then Relationship table and click to expand the dropdown menu in the cell currently showing NoCondition.
- Change the value to Speed_Button=1.



- Click the Run on Target (bullseye) button to compile and download your updated motion sequence to the MCU.

- Wait for the command line build and download windows to complete.
- Follow the prompts and wait for KMS to reconnect.
- Activate the SpeedFeedback plot in the Software Oscilloscope and click the Run button to begin sampling.
- Navigate to the Motion Sequences page of Motor Manager, and click Run button to Start Example Motion Sequence.





- Observe that the motion sequence status changes to busy, and the motor may turn briefly to initialize, but otherwise there is no motion.

- This is because the motor now goes from the InitialState to Low_Speed only when Speed_Button=1.

- Return to Motion Sequence Builder and click on Variables to recall that Speed_Button by default is equal to 0 .

Variable Definition

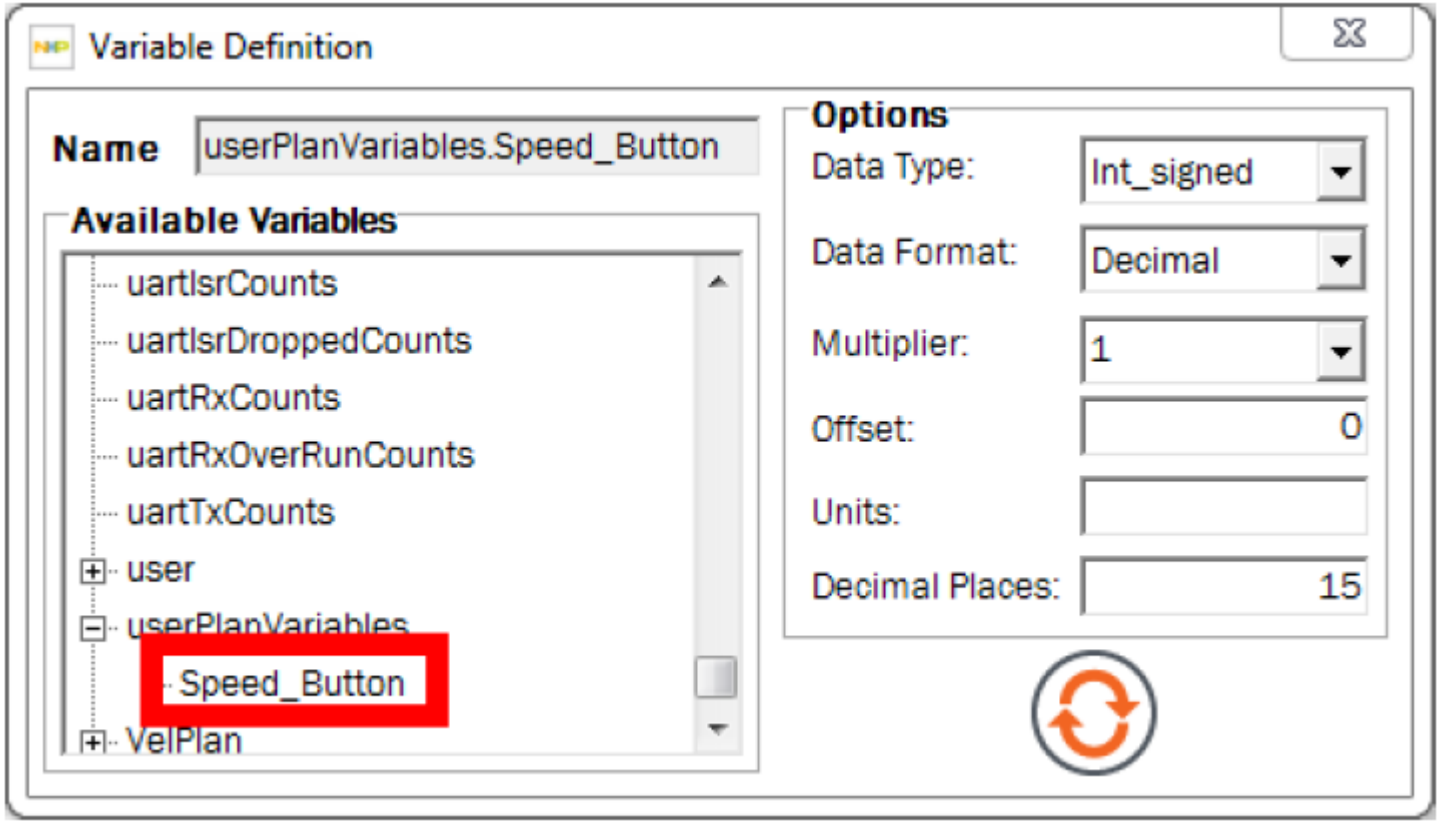
On what values does your motion sequence depend?

	Name	Initial Value
▶	Speed_Button	0

- To make the motion sequence spin the motor, Speed_Button should be made to equal one.
- Click to activate the Watch Window using the icon at the bottom right.
- The Watch Window gives you access to parsed variables on the MCU.

- Click the Add button to add a variable to the Watch Window.

- Scroll to the bottom of the Available Variables list, click the plus box to expand the group userPlanVariables, then click to select Speed_Button .



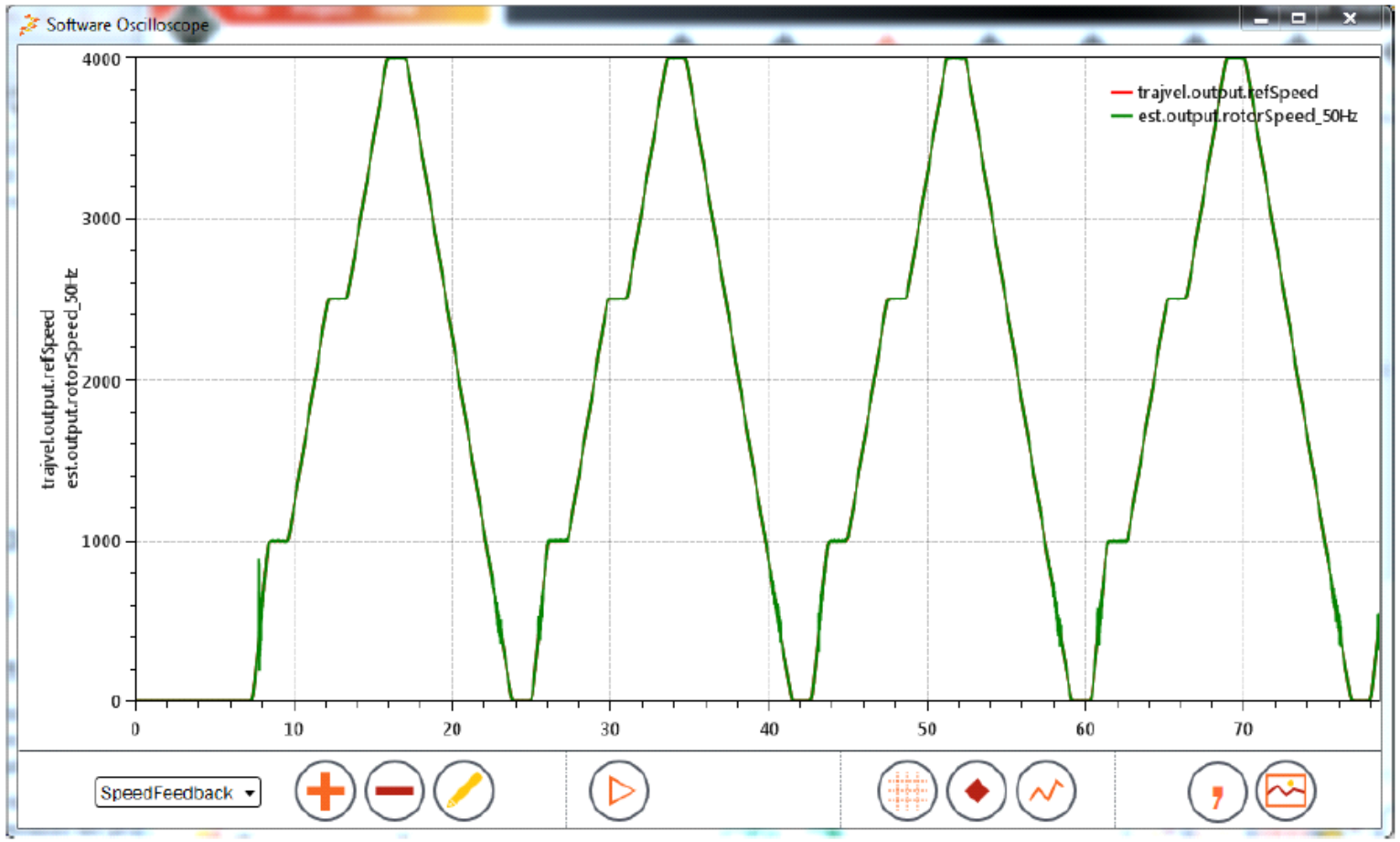
- Click the Update button to add Speed_Button to the Watch Window.

- At the bottom of the Watch Window, click the Run button to begin sampling.

- Click in the Value cell for userPlanVariables.Speed_Button and type in 1.
- Observe the motor now spinning and completing the motion sequence.
- The condition Speed_Button=1 is now fulfilled, so the motor can cycle through the desired states as it did previously.
- Click to Stop Motion Sequence and Stop Sampling.

Watch Window

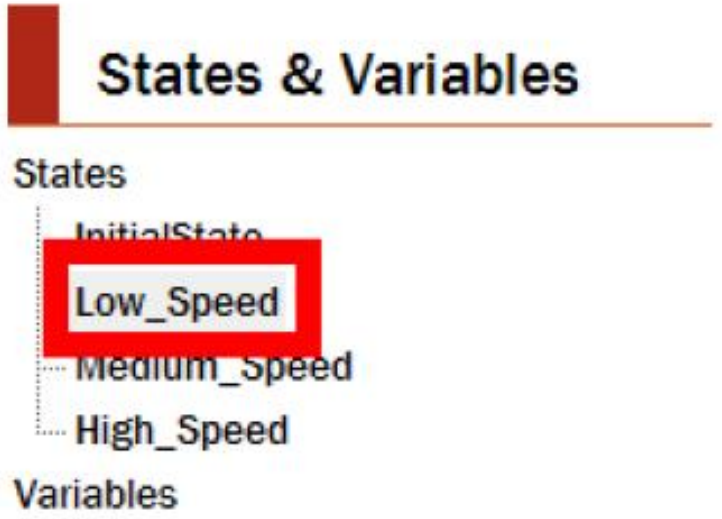
	Name	Data Type	Multiplier	Value
	user.command.resetFault	Int_signed		
▶	userPlanVariables.Speed_Button	Int_signed		1



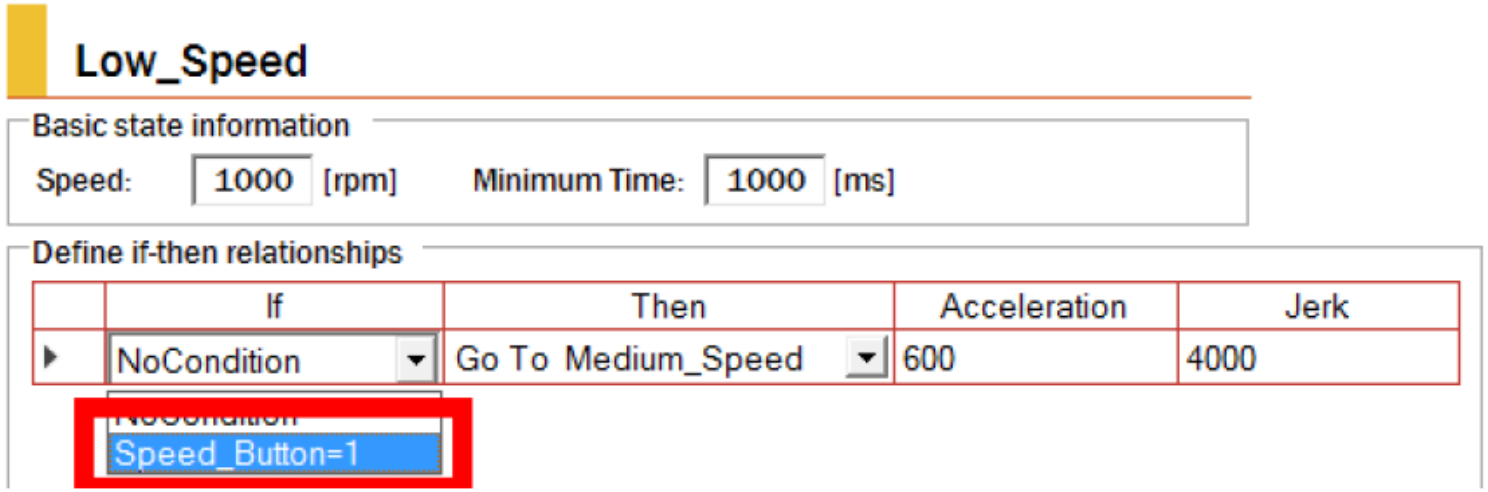
- However, a ceiling fan should be able to remain in each state according to the user's desired fan speed.

- In other words, the transitions between speeds should all be conditional based on button behavior.

- Navigate to Motion Sequence Builder and click on the Low_Speed state.



- Click on cell in If-Then table currently showing NoCondition and select the dropdown option Speed_Button=1.



- Repeat for the Medium_Speed and High_Speed states.

Medium_Speed

Basic state information

Speed: 2500 [rpm] Minimum Time: 1000 [ms]

Define if-then relationships

	If	Then	Acceleration	Jerk
▶	NoCondition	Go To High_Speed	600	4000

Speed_Button=1

High_Speed

Basic state information

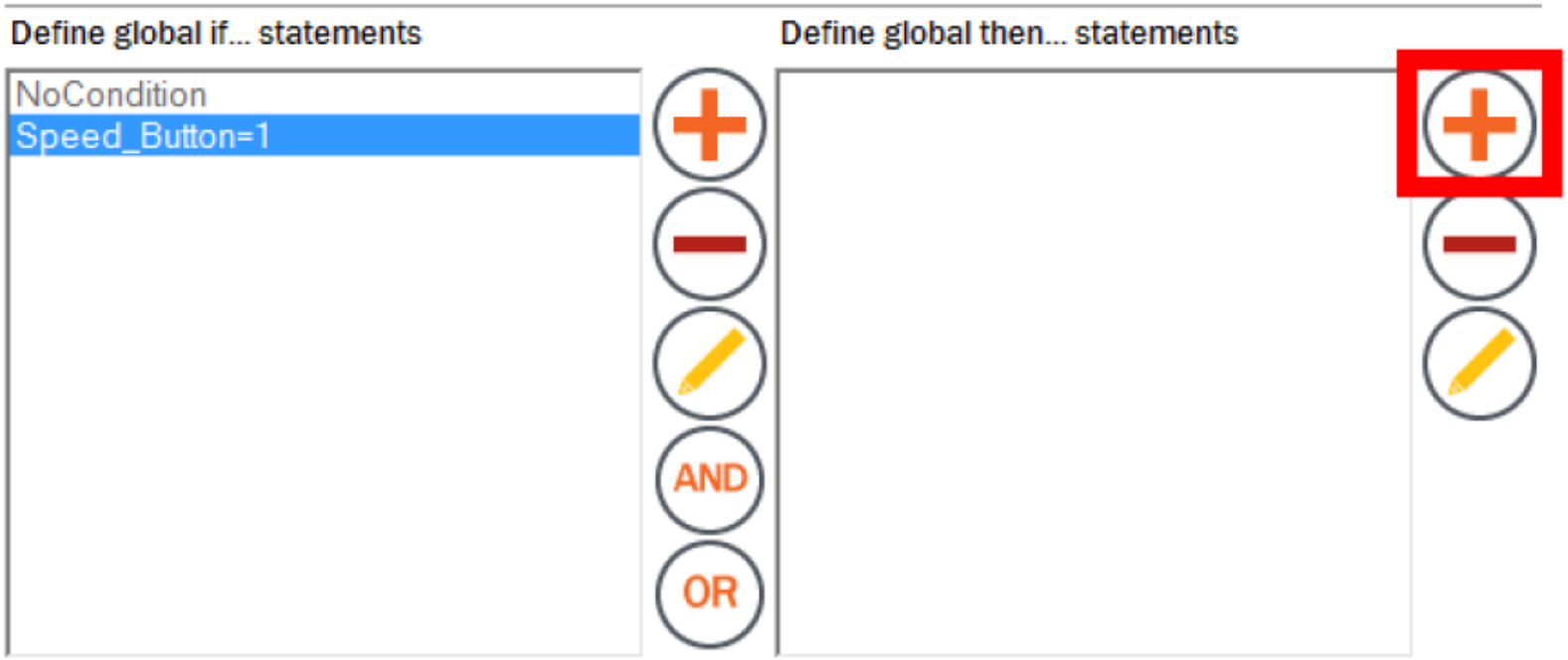
Speed: 4000 [rpm] Minimum Time: 1000 [ms]

Define if-then relationships

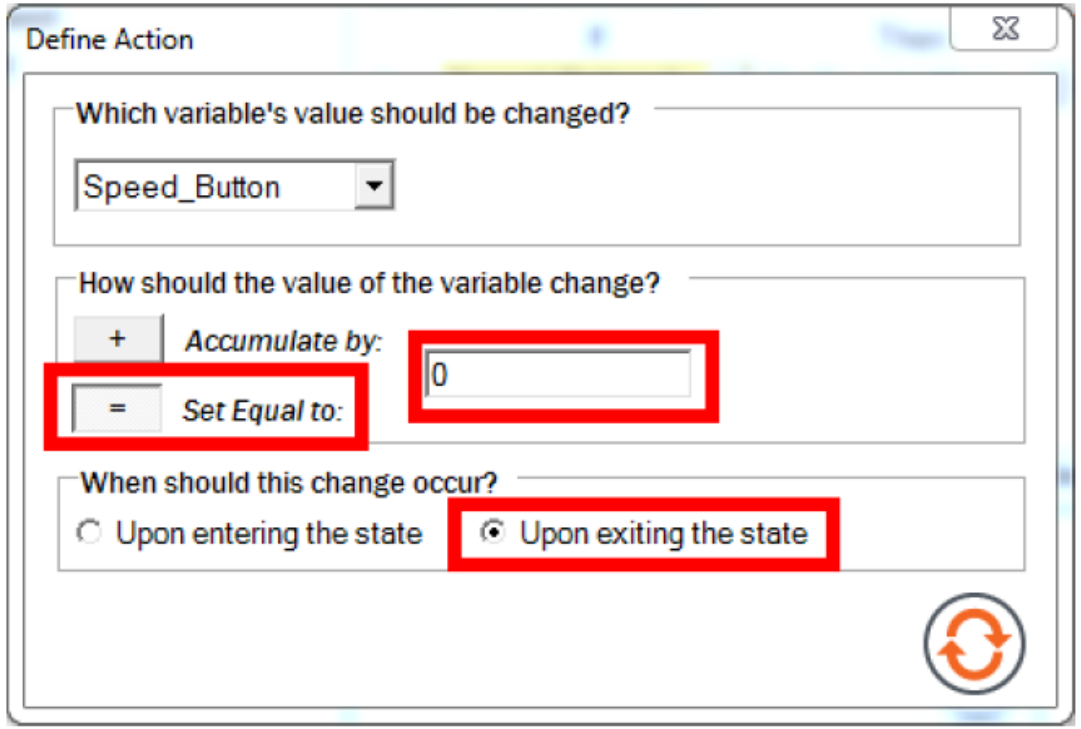
	If	Then	Acceleration	Jerk
▶	NoCondition	Go To InitialState	600	4000

Speed_Button=1

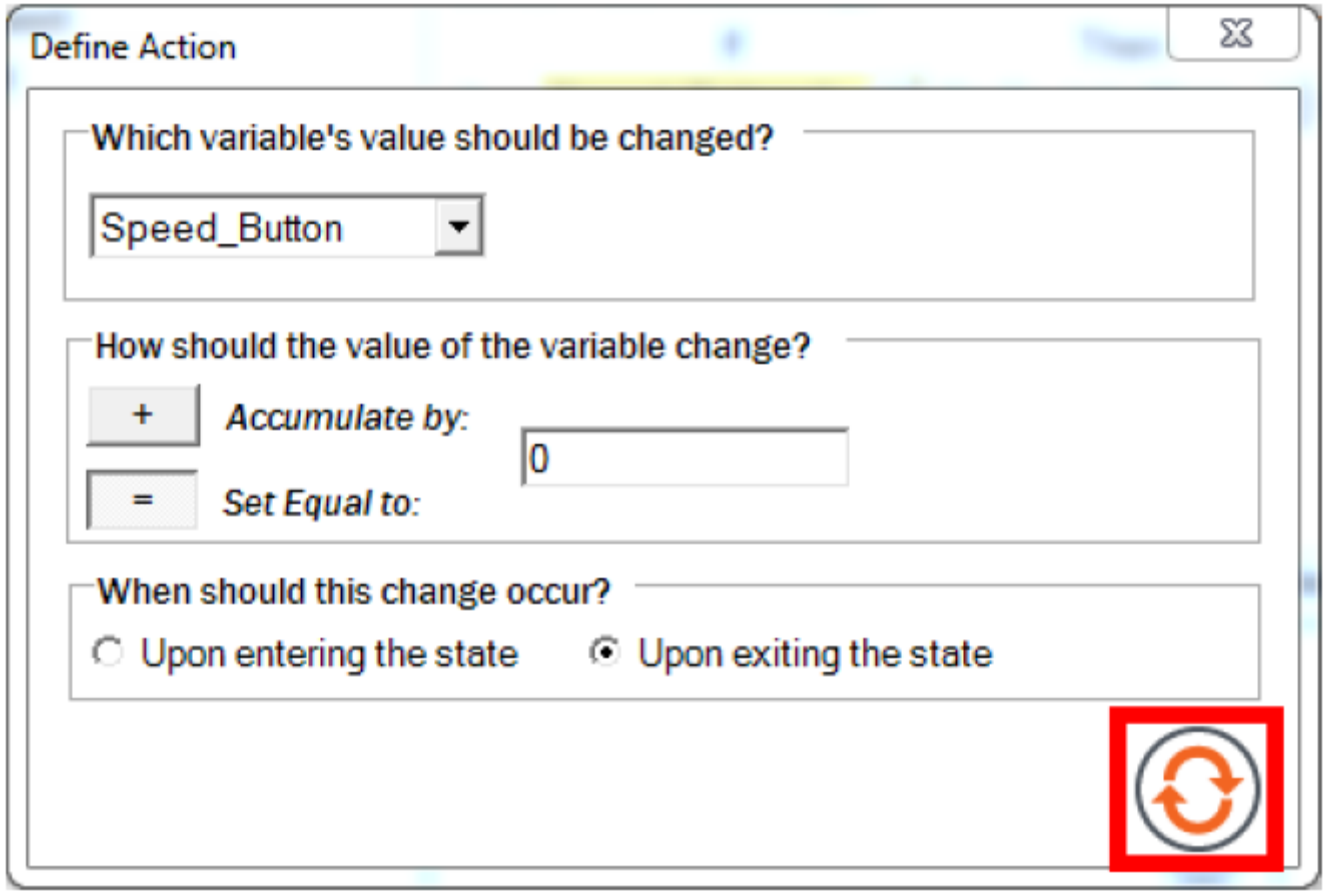
- This means that the motor advances to the next state only if Speed_Button=1.
- But if Speed_Button is not reset after each successful transition, the motor cycles through all states as before (Speed_Button is perpetually equal to one).
- To reset the Speed_Button, click the Add button next to the global then... statements list.



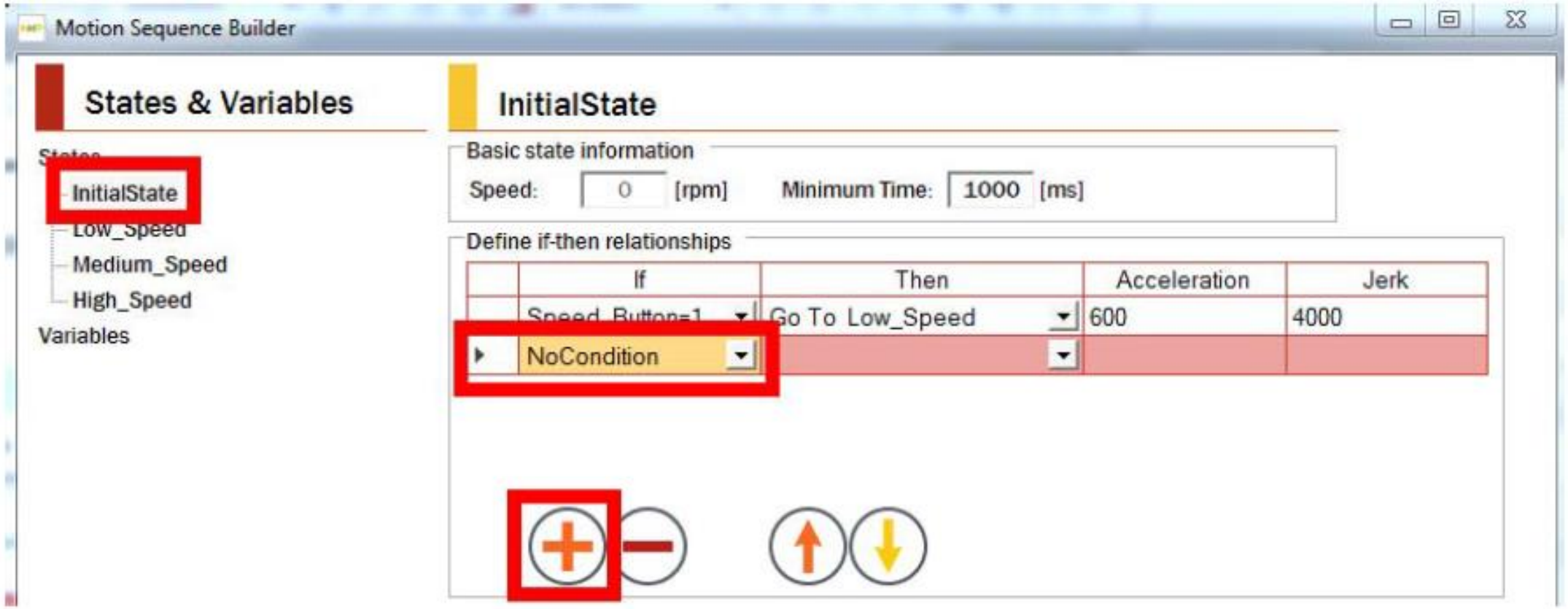
- To specify the reset of the Speed_Button, perform the following configuration steps:
  - click on the “Set equal to:” button
  - enter a value of 0 in the blank field
  - select the radio button Upon exiting the state



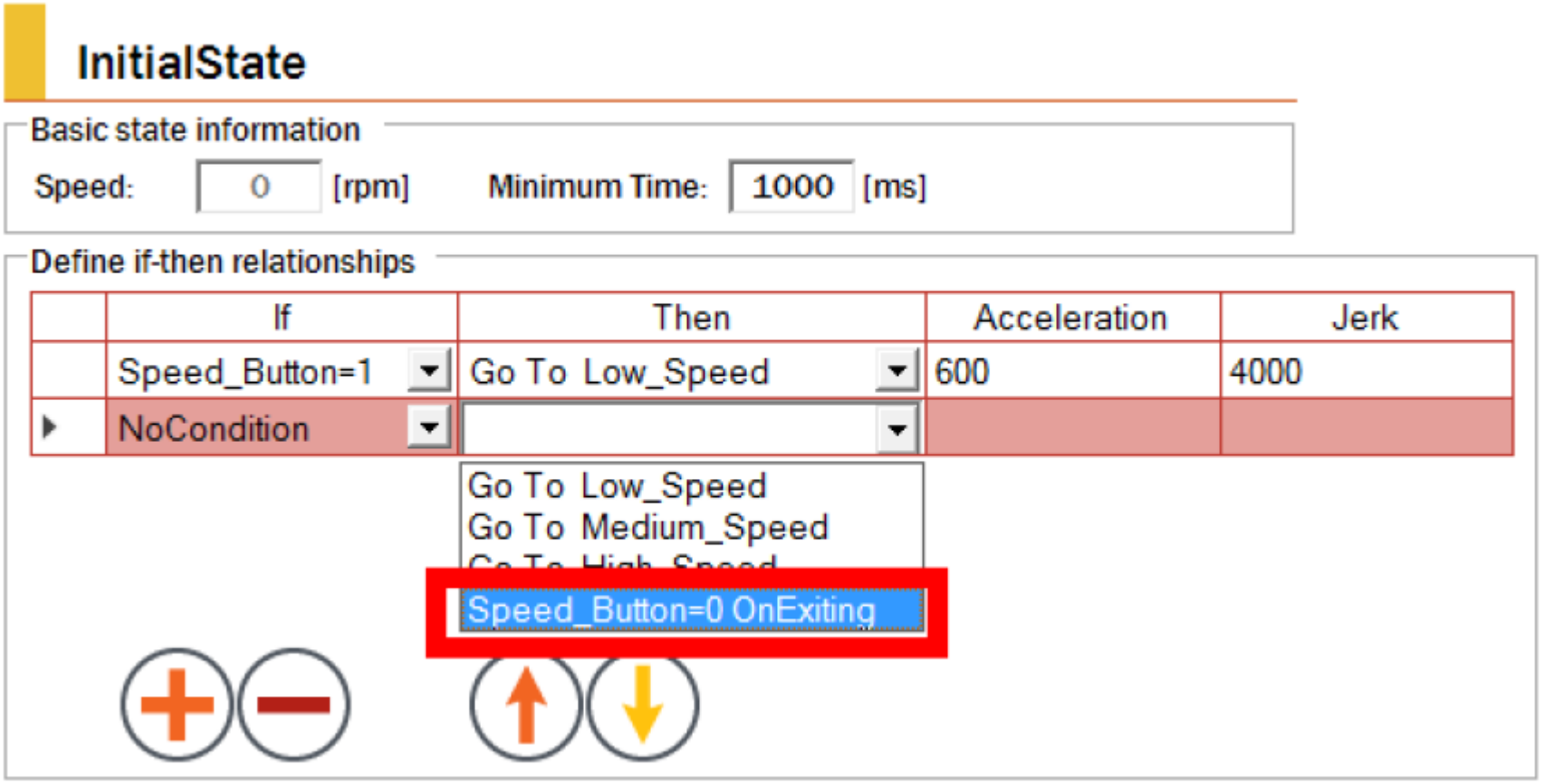
- Click the Update button to create this reset action .



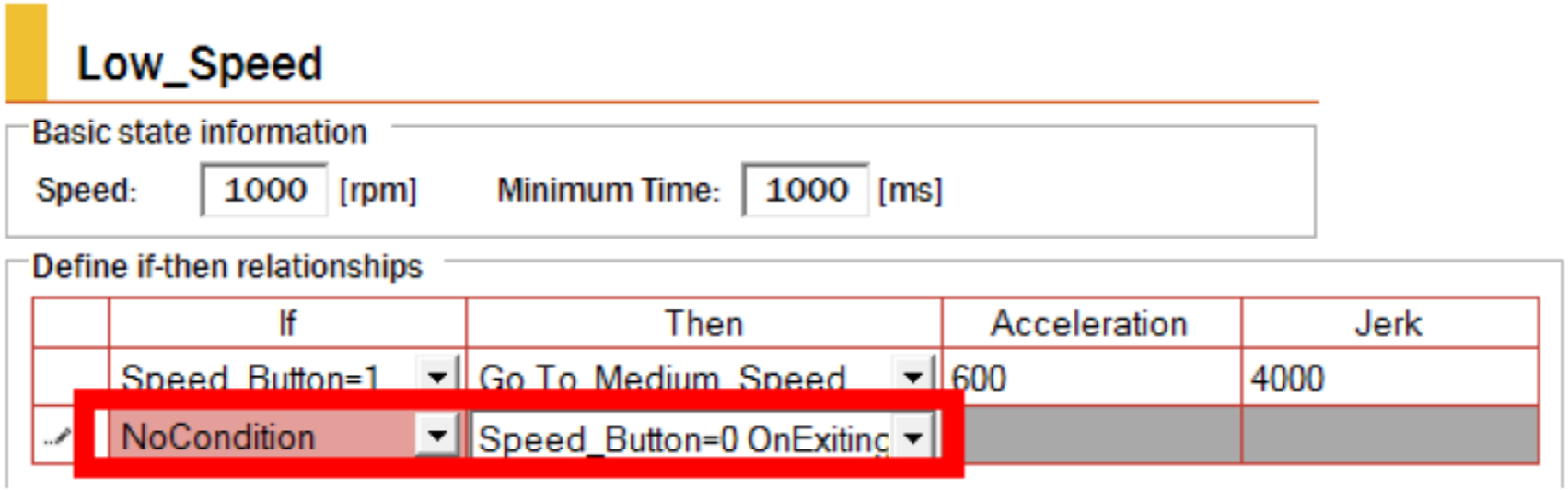
- Click on InitialState, then click to add new row to if-then table.



- In the new row, click on the dropdown arrow in the “Then” cell and select Speed_Button=0 OnExiting.



- Repeat for all other states. After leaving a state, Motion Sequence Builder may reorder the if-then table to reflect its internal order of evaluation.



Medium_Speed

Basic state information

Speed: 2500 [rpm] Minimum Time: 1000 [ms]

Define if-then relationships

	If	Then	Acceleration	Jerk
	Speed_Button=1	Go To_High_Speed	600	4000
	NoCondition	Speed_Button=0 OnExiting		

High_Speed

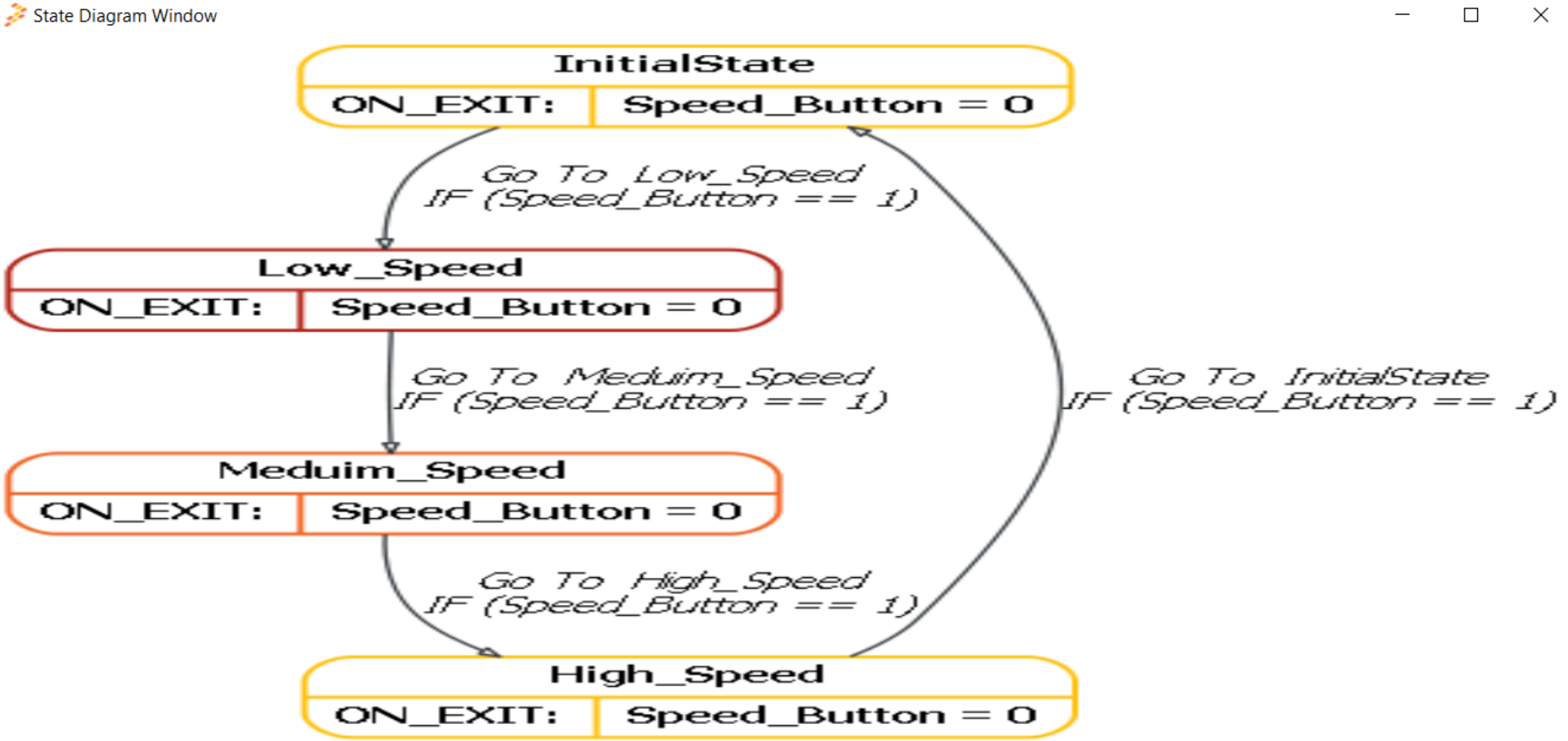
Basic state information

Speed: 4000 [rpm] Minimum Time: 1000 [ms]

Define if-then relationships

	If	Then	Acceleration	Jerk
	Speed_Button=1	Go To_InitialState	600	4000
	NoCondition	Speed_Button=0 OnExiting		

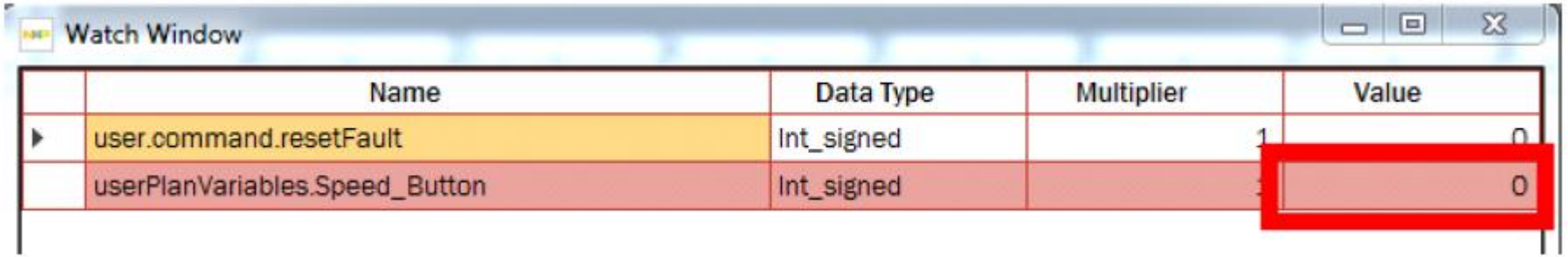
- Click the State Diagram button to see the representation of the conditional transitions to different speeds and the resetting of the Speed_Button variable.


State Diagram Window
InitialState
ON_EXIT:
Speed_Button = 0
Go To Low_Speed
IF (Speed_Button == 1)
Low_Speed
ON_EXIT:
Speed_Button = 0
Go To Meduim_Speed
IF (Speed_Button == 1)
Meduim_Speed
ON_EXIT:
Speed_Button = 0
Go To High_Speed
IF (Speed_Button == 1)
High_Speed
ON_EXIT:
Speed_Button = 0
Go To InitialState
IF (Speed_Button == 1)

- Click bullseye button to compile then download to MCU.
- Follow the command window directions then wait for KMS to reconnect to the MCU.

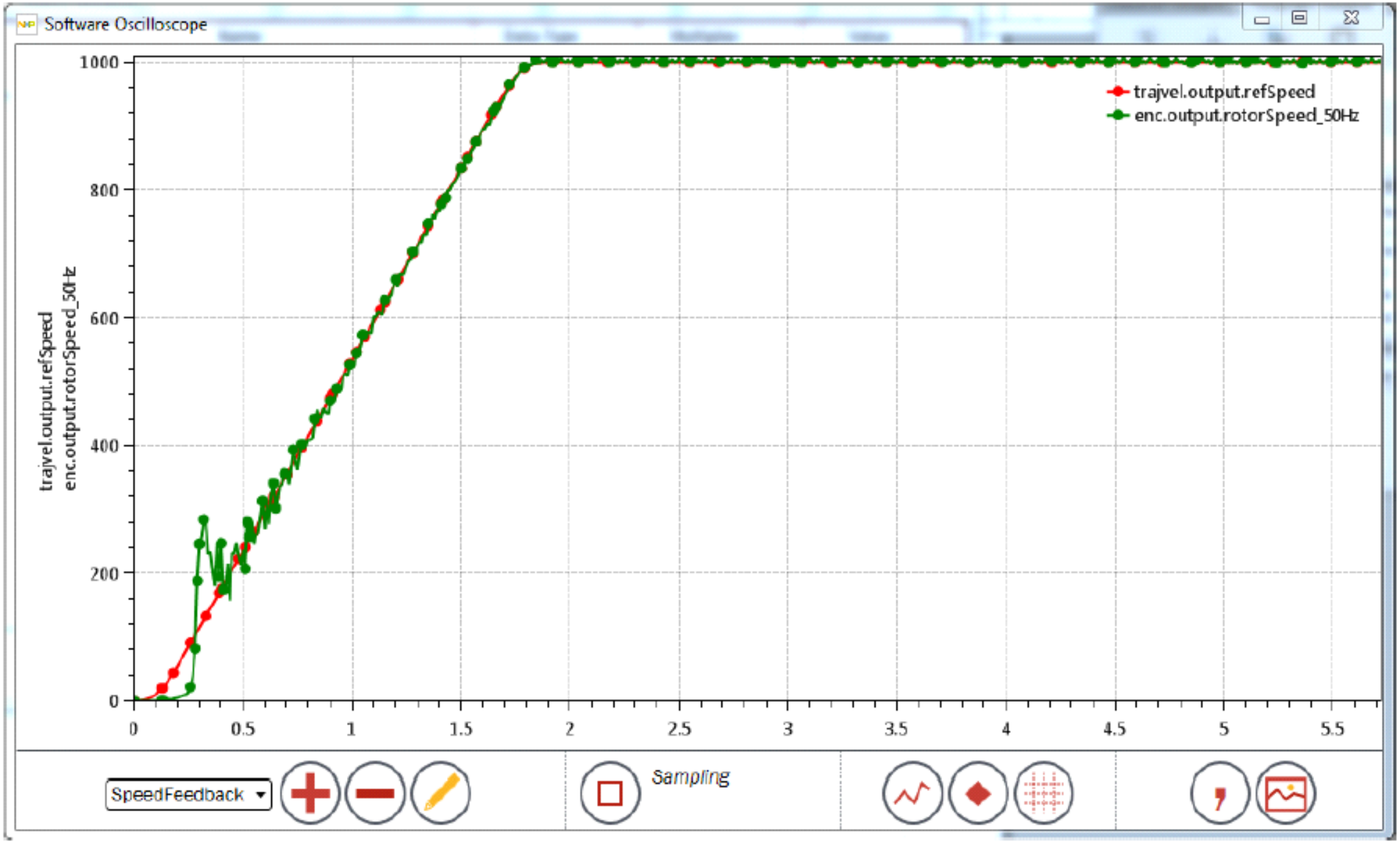
- Activate Software Oscilloscope, select SpeedFeedback plot, and click Run button to begin sampling.

- Navigate to the Motion Sequences page and click Run button to Start Example Motion Sequence.
- Activate Watch Window and click Run button to begin sampling. Note the value for userPlanVariables.Speed_Button.

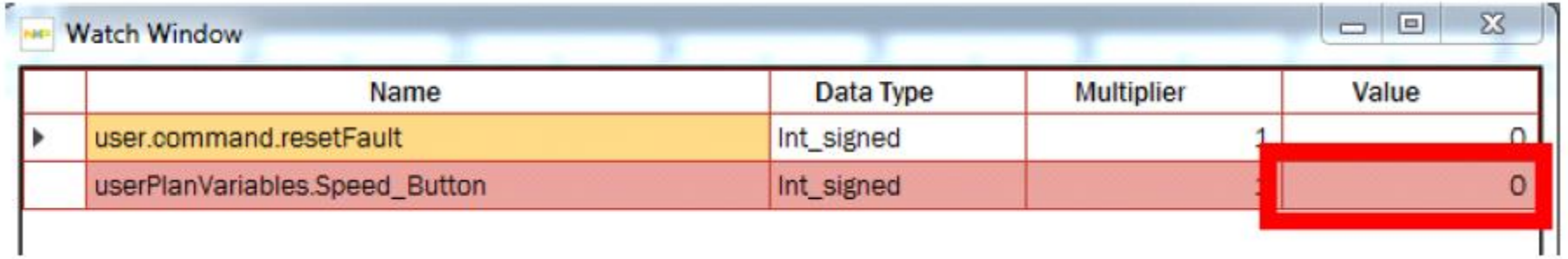


- In the Value cell for the variable userPlanVariables.Speed_Button, type in a new value of 1.

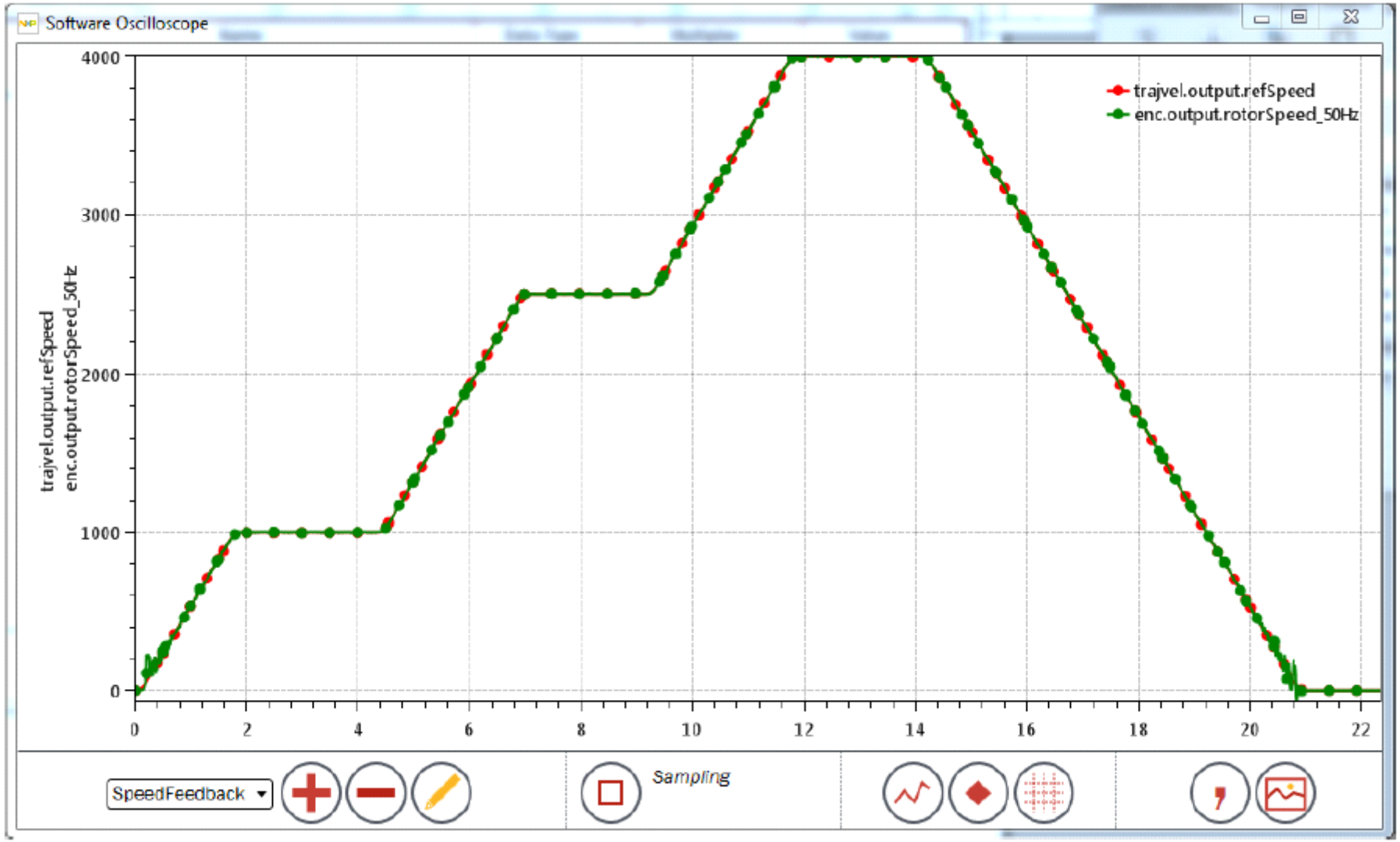
- Observe that the motor spins to next state.


Software Oscilloscope
trajvel.output.refSpeed
enc.output.rotorSpeed_50Hz
trajvel.output.refSpeed
enc.output.rotorSpeed_50Hz
1000
800
600
400
200
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
SpeedFeedback
Sampling

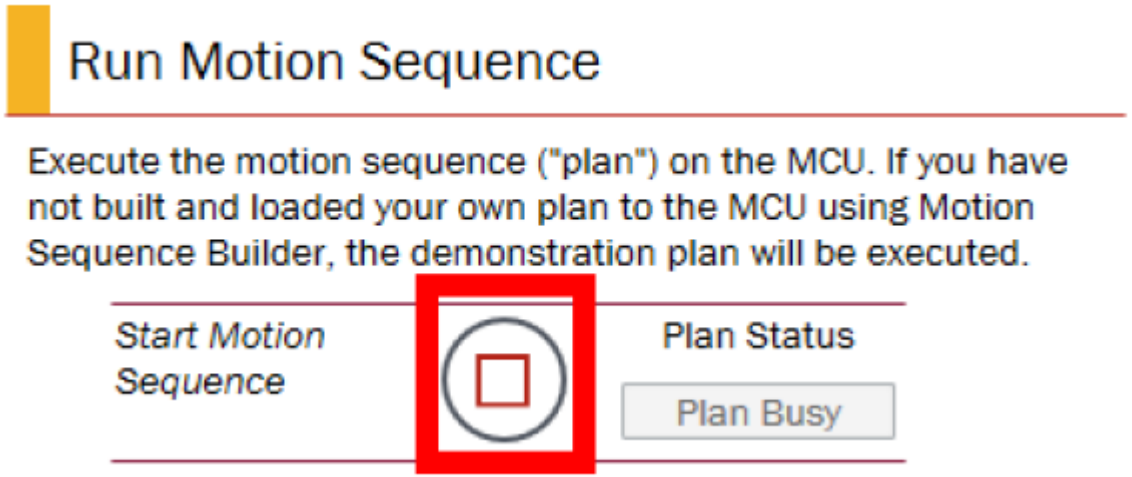
- Observe also that userPlanVariables.Speed_Button automatically returns to a value of 0.



- Continue to type in a value of 1 for userPlanVariables.Speed_Button to cycle through the states of your motion sequence .


Software Oscilloscope
trajvel.output.refSpeed
enc.output.rotorSpeed_50Hz
trajvel.output.refSpeed
enc.output.rotorSpeed_50Hz
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
18
20
22
SpeedFeedback
Sampling

- Click to stop sampling in SpeedFeedback plot and click to stop motion sequence.



Summary

In this lab, you have used Motion Sequence Builder to construct a simple motion sequence approximating the behavior of a ceiling fan.