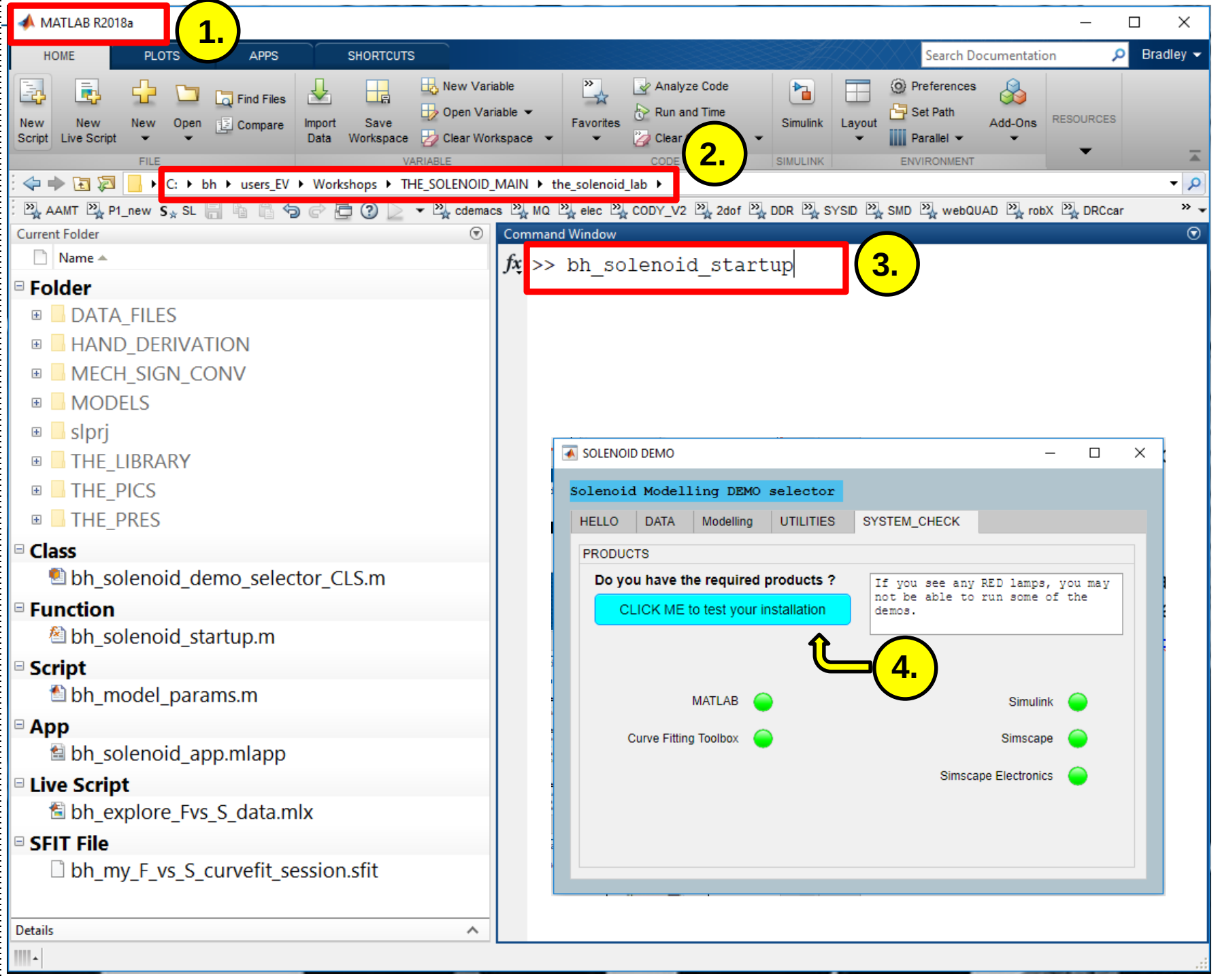
Getting Ready:
For the Lab

1. Launch MATLAB **R2018a**

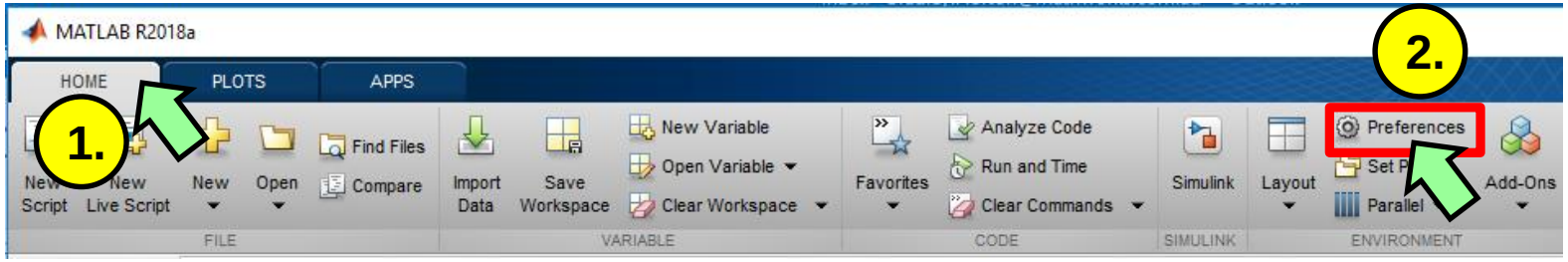
2. In MATLAB, navigate to the folder called **`the_solenoid_lab`**

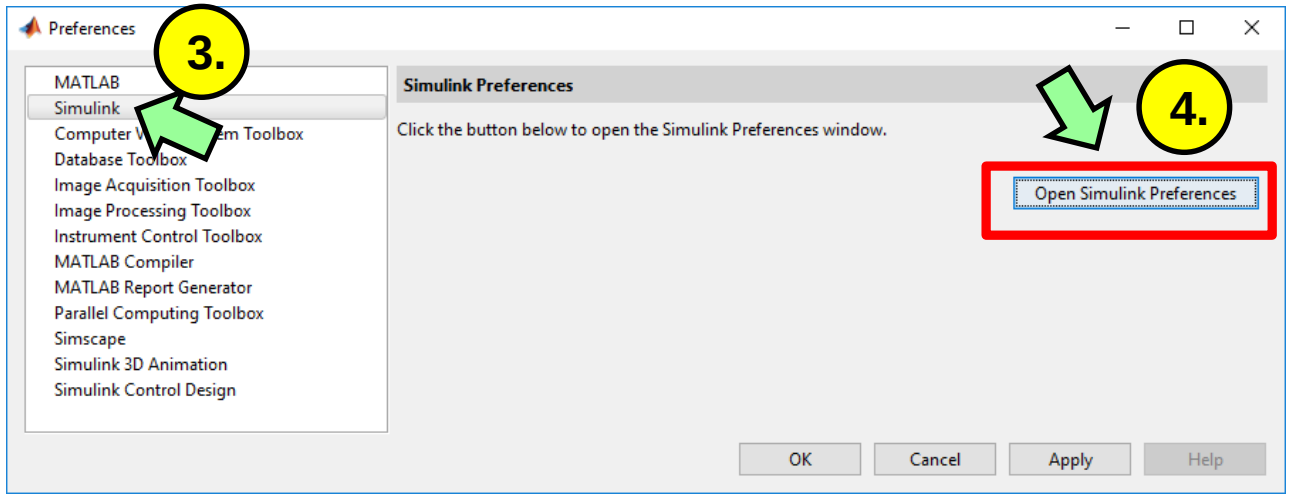
3. In the MATLAB command window, run the command:
   - **`>> bh_solenoid_startup`**

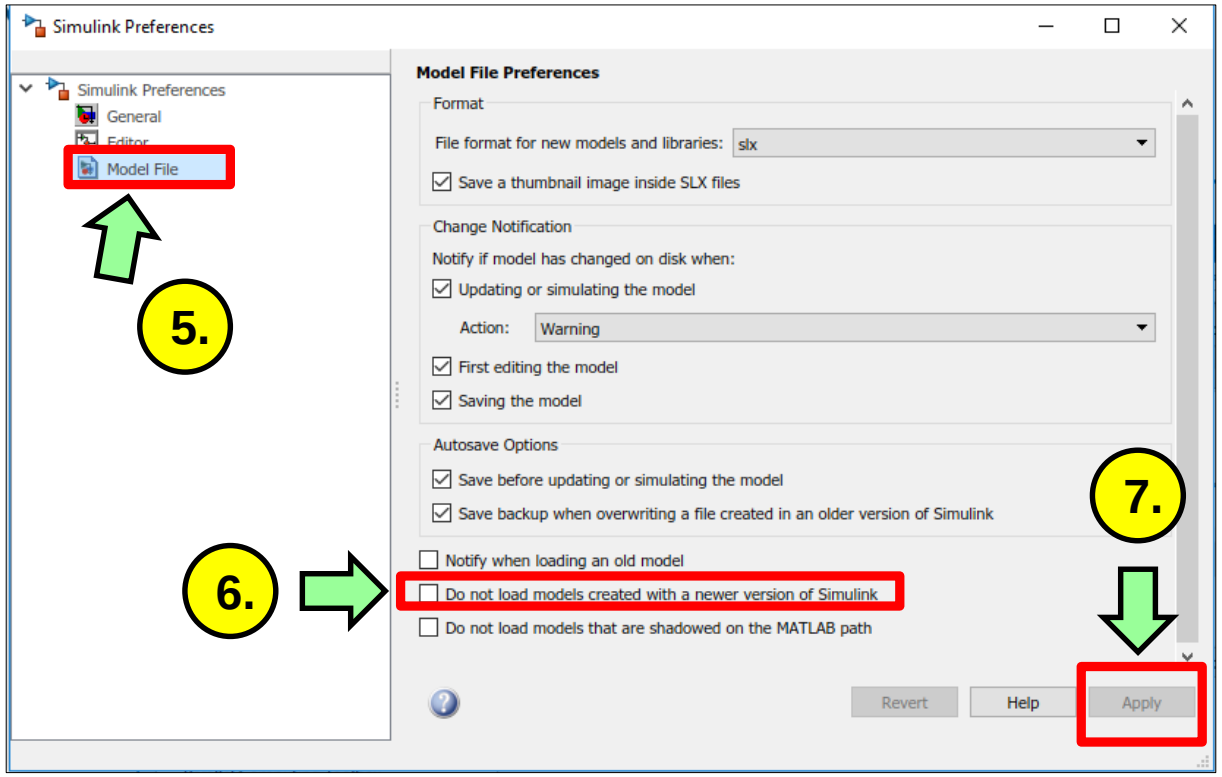
4. Check your MATLAB installation
   - You should see ALL green lamps



ATTENTION: If your Release of MATLAB is OLDER than R2018a you'll need to do the following steps:



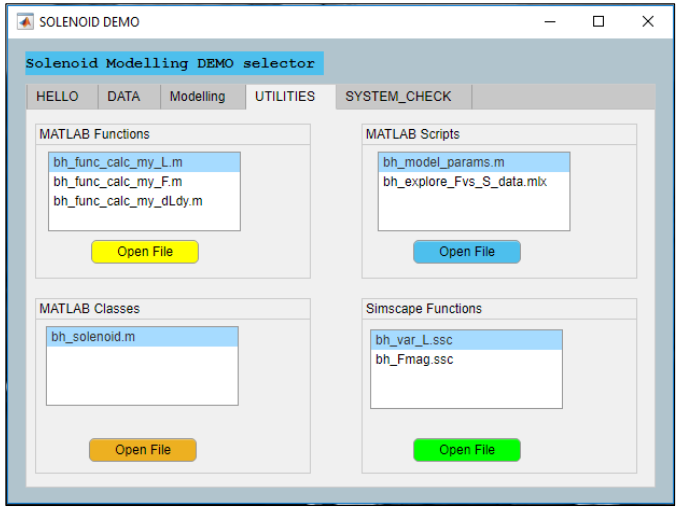


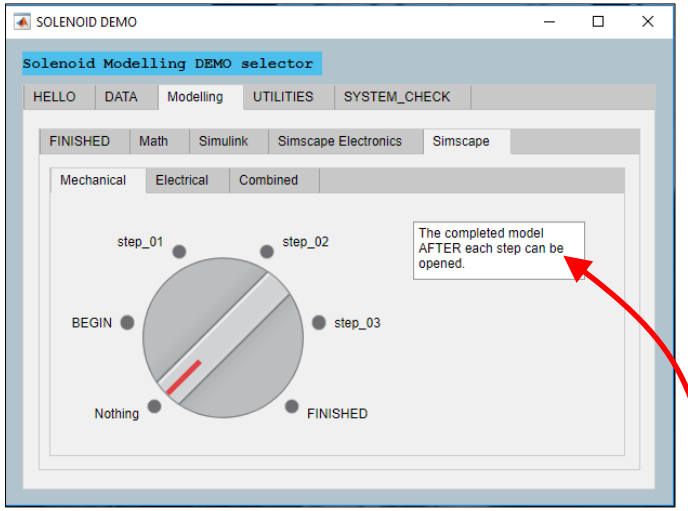


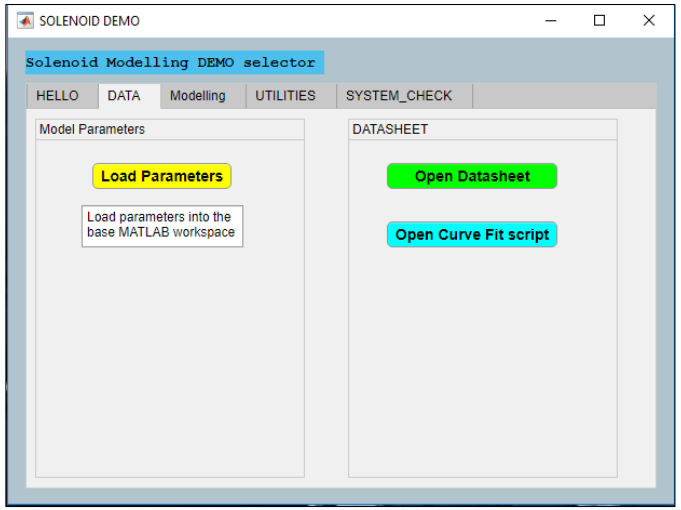
Before we get started:

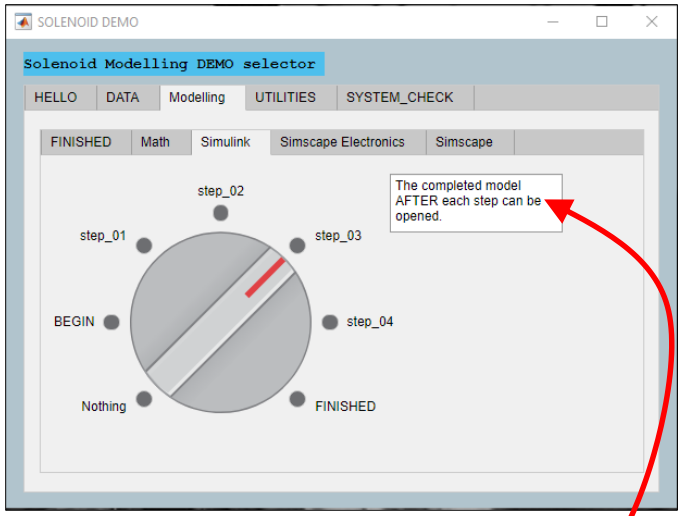
The **DEMO selector app**

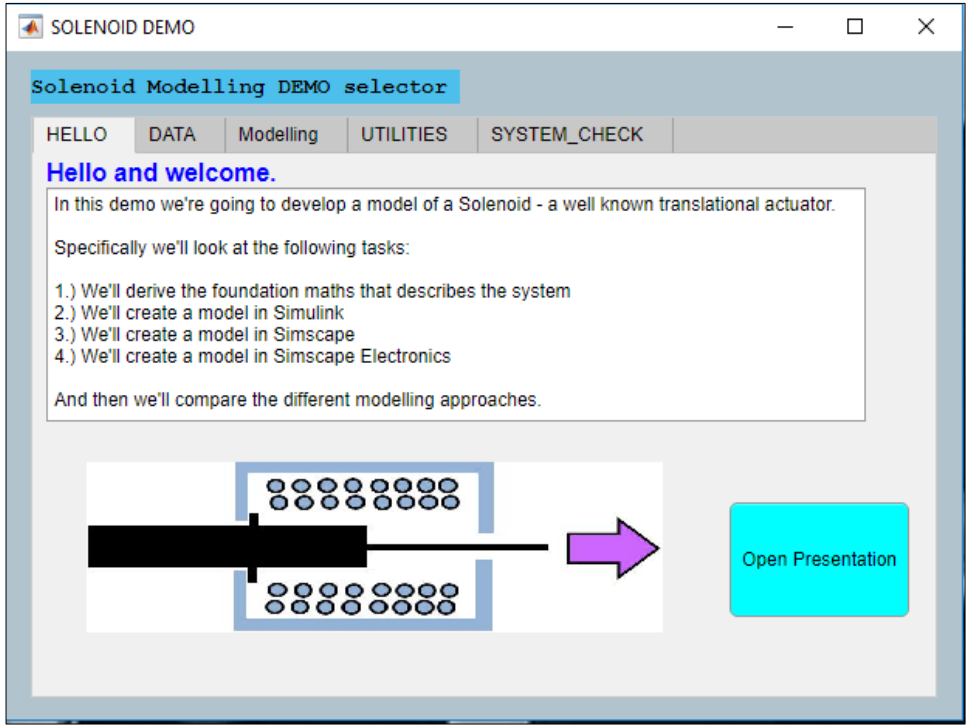
- Keep the Demo Selector app open
  - We'll use it often
  - If it disappears ….. Then relaunch it using steps of SLIDE_#1





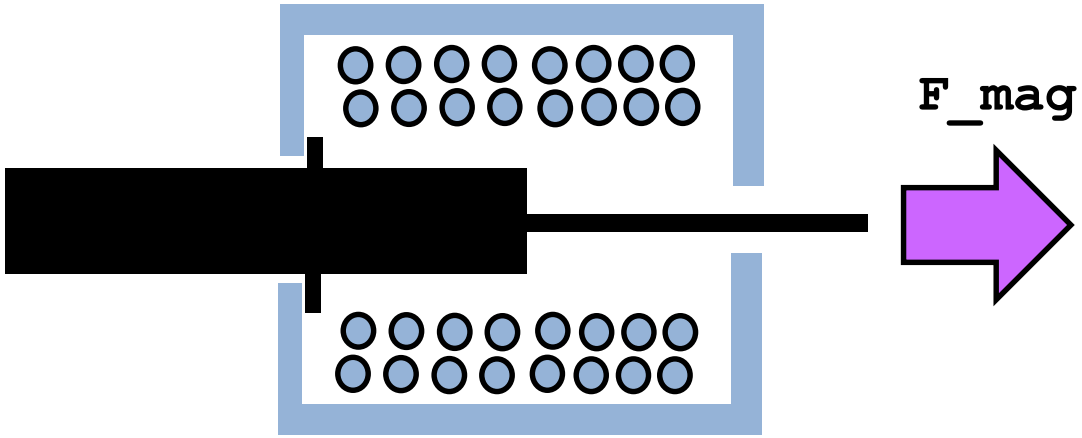


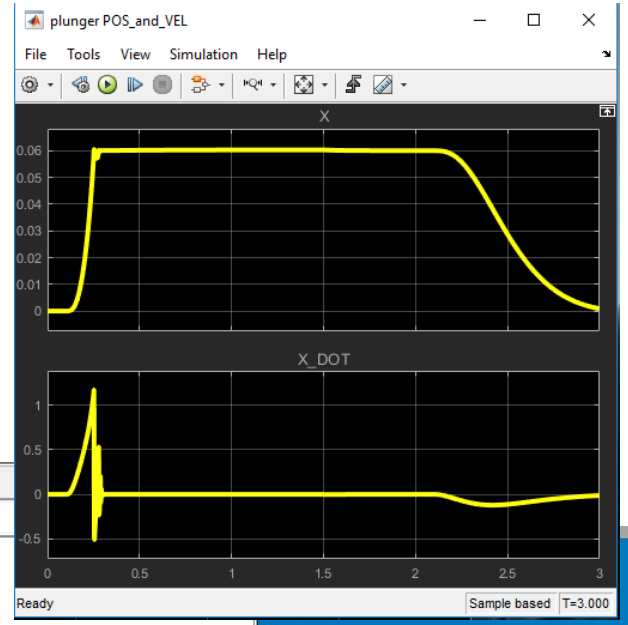




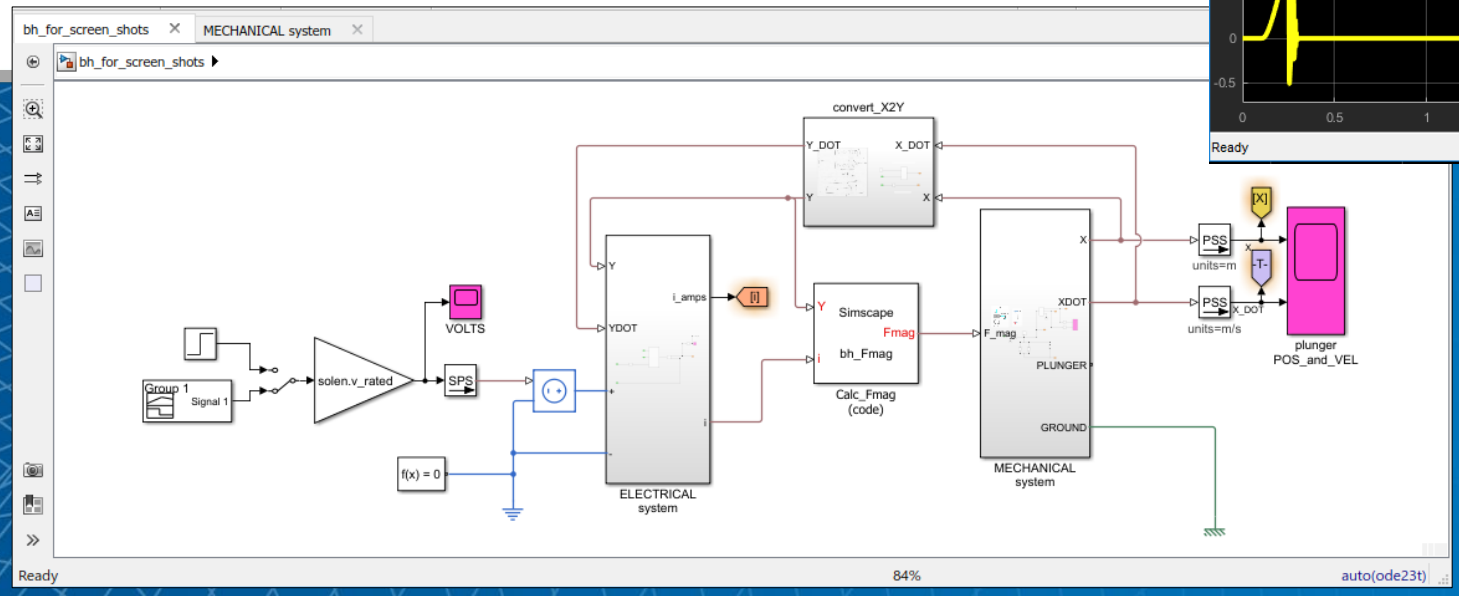
ATTENTION:
When we start to create **MODELS** ….. We'll do this together. IFFFF you miss a step AND don't want to ask me for help … then you can quickly catch up by opening a finished model at the end of each step`

Solenoid modelling





Bradley Horton
Engineer



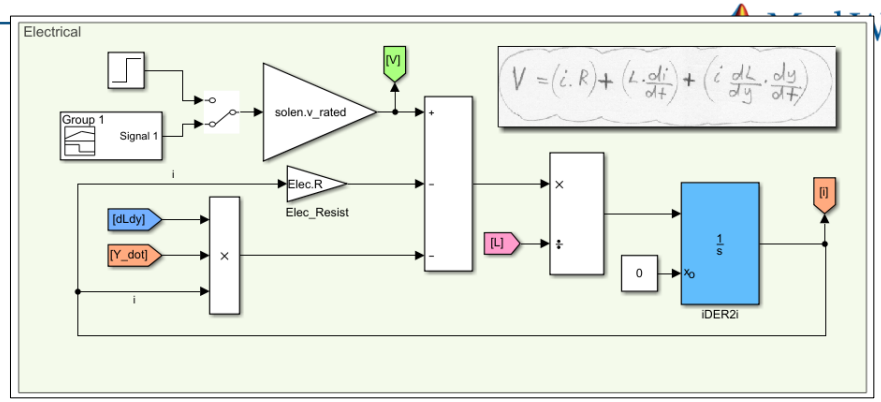
Todays objectives:

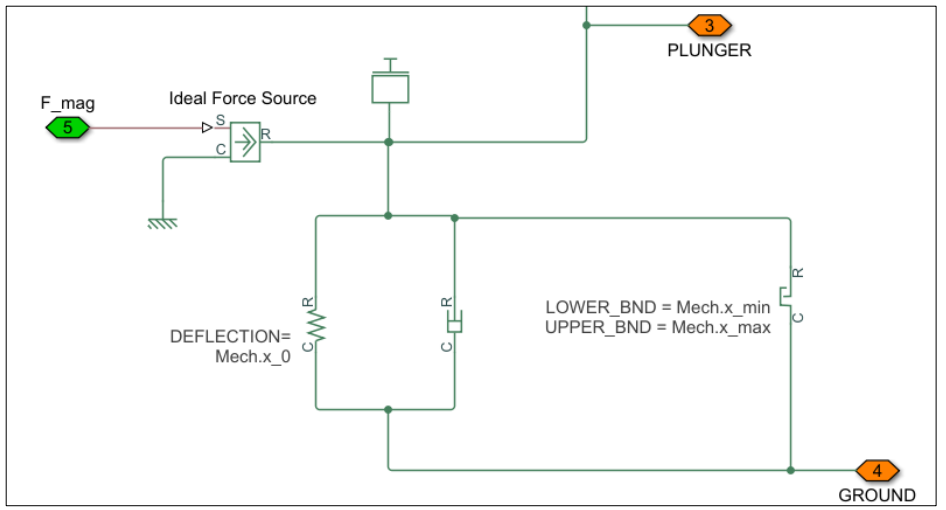
Model a solenoid

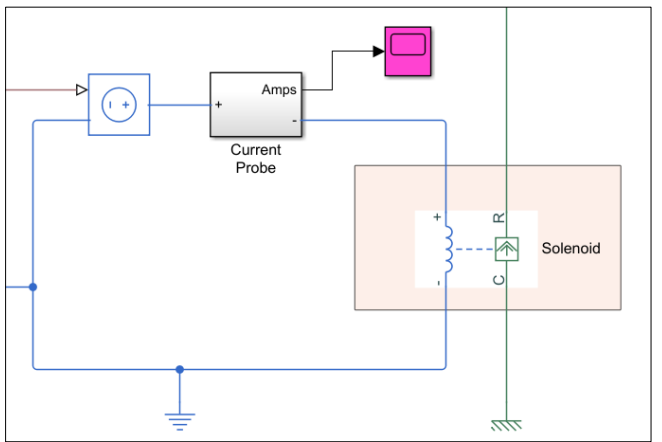
1. Implement in **Simulink**
   - We take ALL of the system mathematics (ODEs) and implement

2. Implement in **Simscape**
   - We take some of the system mathematics and we implement
   - Other parts are represented by “physical” blocks (eg: resistors, springs, etc)

3. Implement in **Simscape Electronics**
   - This is an “out of the box” model
   - We configure it using our system parameter VALUES







The device that we'll model today:

- The solenoid:

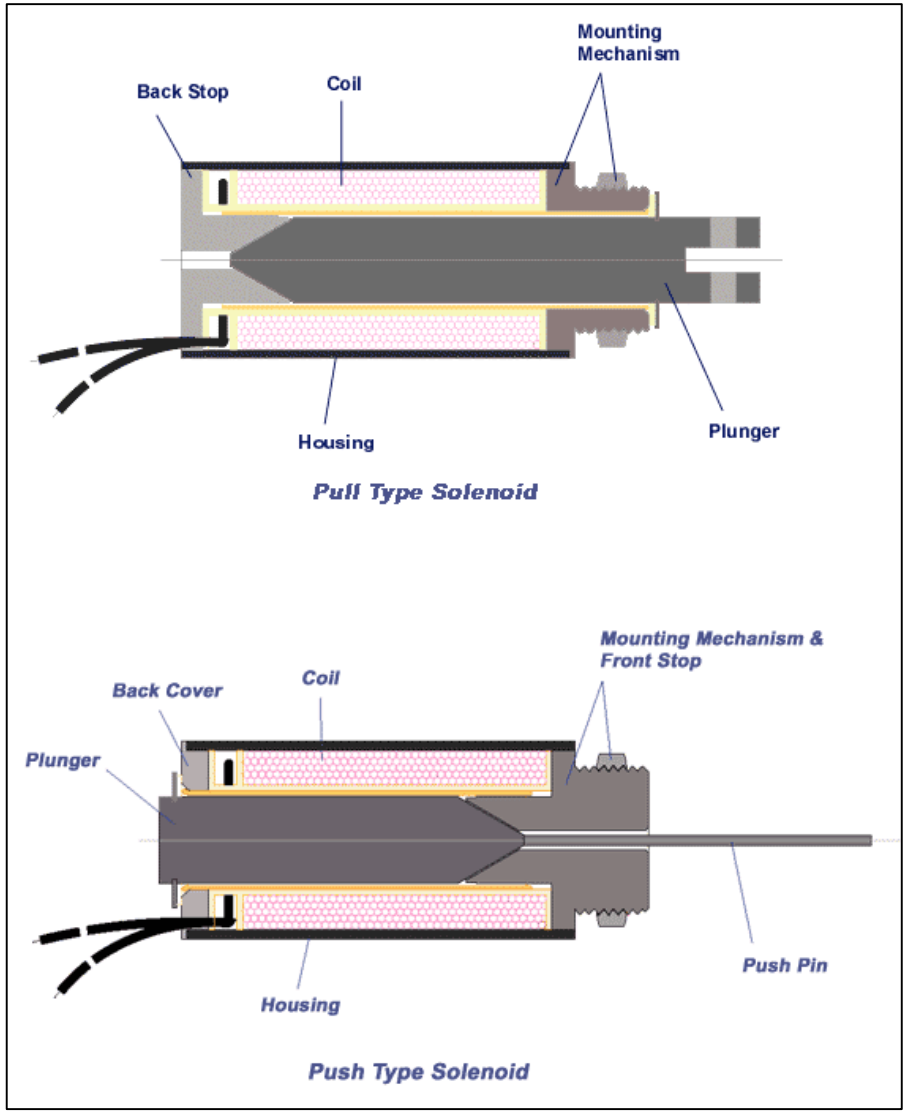


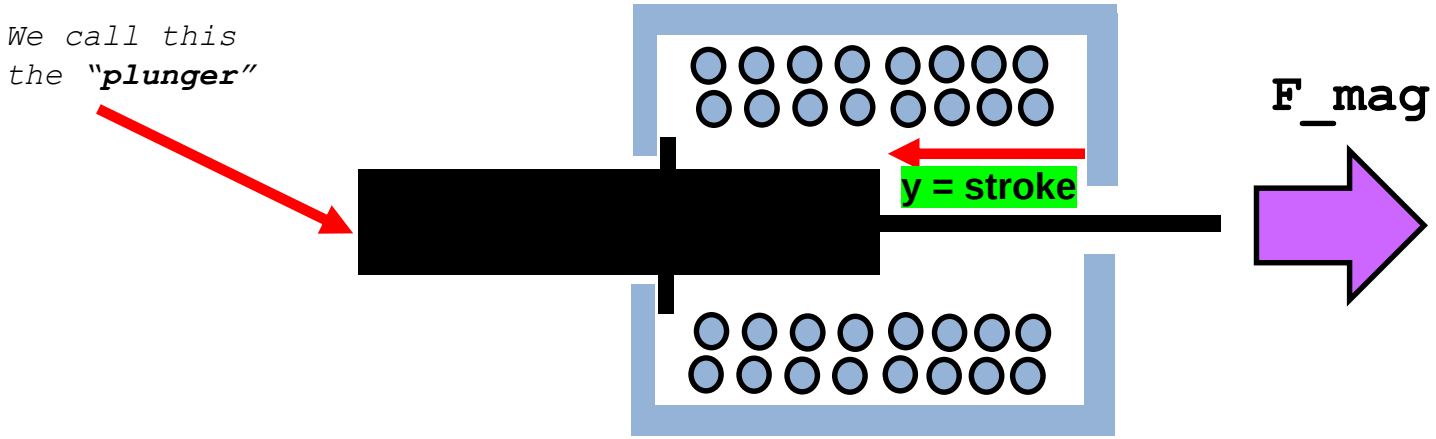
Figure: from REF_01

Today, we'll focus on the **PUSH** type solenoid

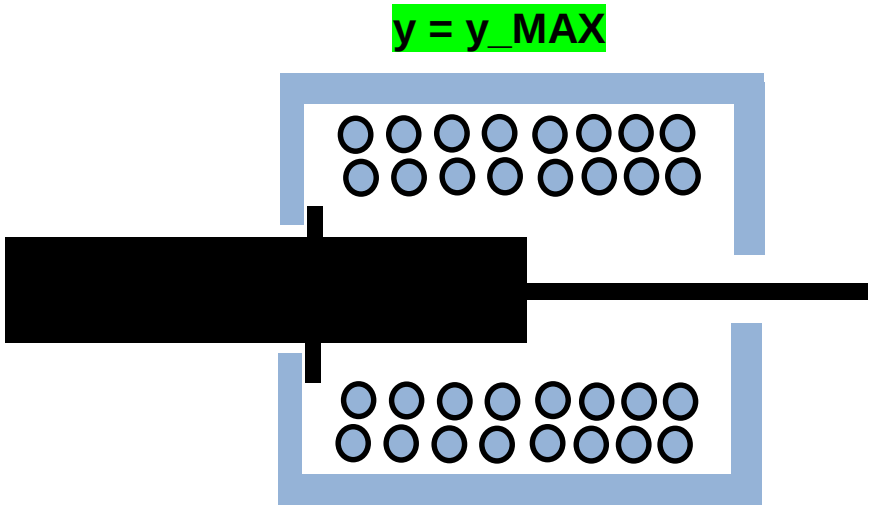
Watch a short 1 minute video

REF_01: http://www.solenoidcity.com/solenoid/manual/construction/construction.htm

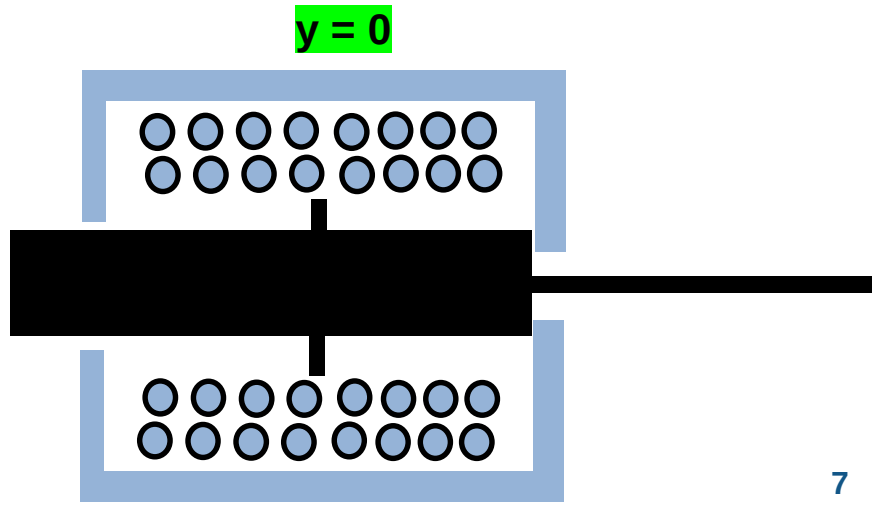
The Magnetic force and STROKE length:



STROKE at its MAXIMUM value



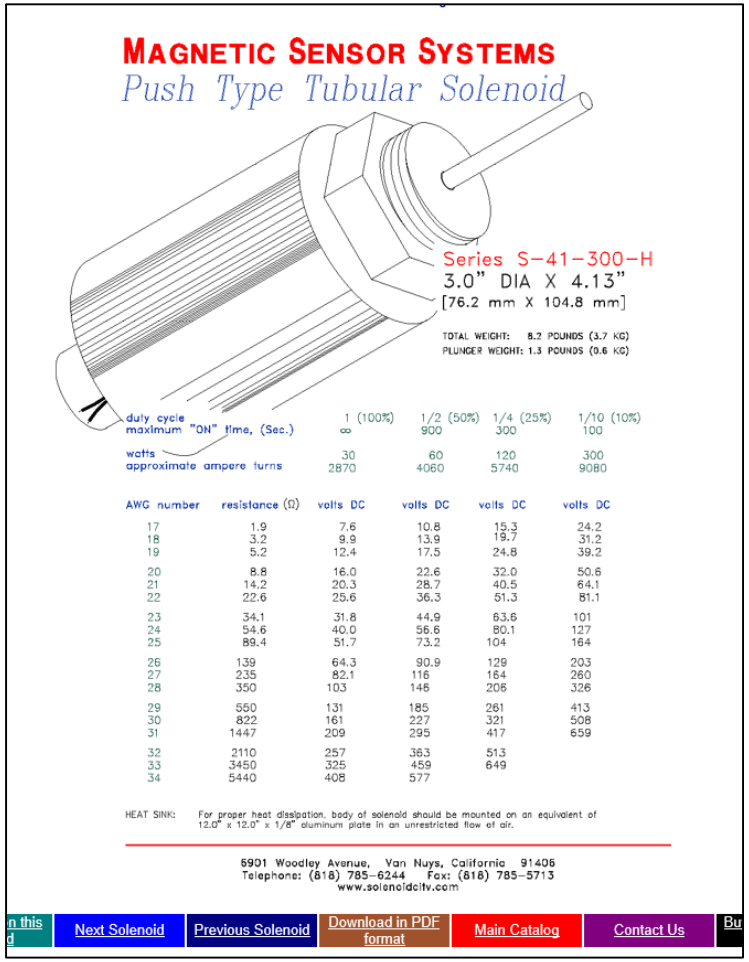
STROKE = ZERO



The device that we'll model today:

- The solenoid:

Figure: from REF_01



MAGNETIC SENSOR SYSTEMS

Push Type Tubular Solenoid

Series S-41-300-H
3.0" DIA X 4.13"
[76.2 mm X 104.8 mm]

TOTAL WEIGHT: 8.2 POUNDS (3.7 KG)
PLUNGER WEIGHT: 1.3 POUNDS (0.6 KG)

duty cycle maximum "ON" time, (Sec.)	1 (100%) ∞	1/2 (50%) 900	1/4 (25%) 300	1/10 (10%) 100
watts	30	60	120	300
approximate ampere turns	2870	4060	5740	9080

AWG number	resistance (Ω)	volts DC	volts DC	volts DC	volts DC
17	1.9	7.6	10.8	15.3	24.2
18	3.2	9.9	13.9	19.7	31.2
19	5.2	12.4	17.5	24.8	39.2
20	8.8	16.0	22.6	32.0	50.6
21	14.2	20.3	28.7	40.5	64.1
22	22.6	25.6	36.3	51.3	81.1
23	34.1	31.8	44.9	63.6	101
24	54.6	40.0	56.6	80.1	127
25	89.4	51.7	73.2	104	164
26	139	64.3	90.9	129	203
27	235	82.1	116	164	260
28	350	103	146	206	326
29	550	131	185	261	413
30	822	161	227	321	508
31	1447	209	295	417	659
32	2110	257	363	513	
33	3450	325	459	649	
34	5440	408	577		

HEAT SINK: For proper heat dissipation, body of solenoid should be mounted on an equivalent of 12.0" x 12.0" x 1/8" aluminum plate in an unrestricted flow of air.

6901 Woodley Avenue, Van Nuys, California 91406
Telephone: (818) 785-6244 Fax: (818) 785-5713
www.solenoidcity.com

n this Next Solenoid Previous Solenoid Download in PDF format Main Catalog Contact Us Bu

Magnetic Force Versus STROKE

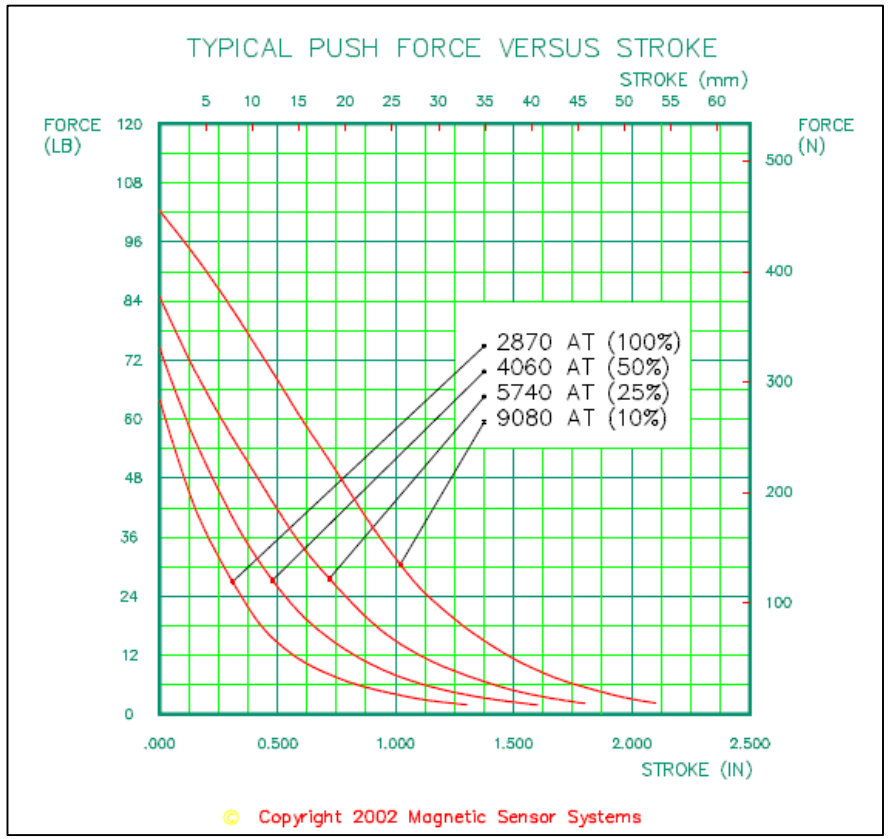
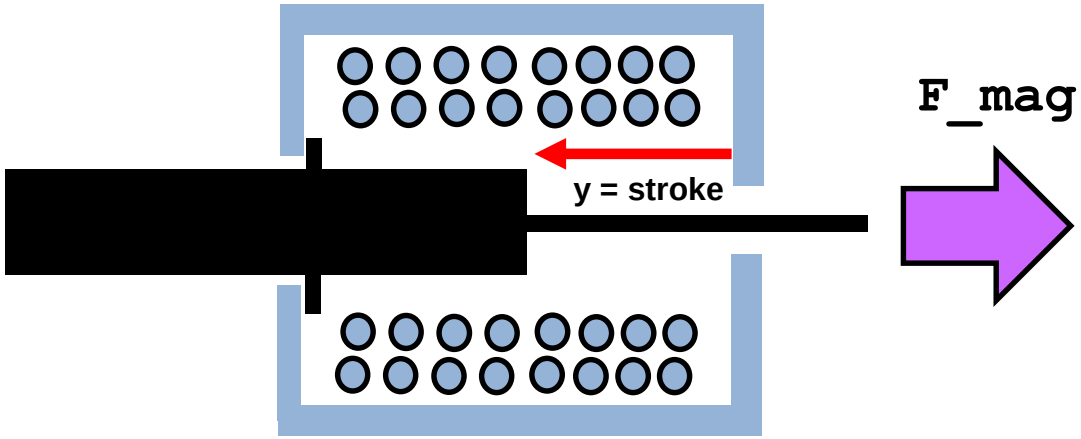


Figure: from REF_02

REF_01: http://solenoidcity.com/solenoid/tubular/S-41-300hp1.htm
REF_02: http://solenoidcity.com/solenoid/tubular/s-41-300h.pdf

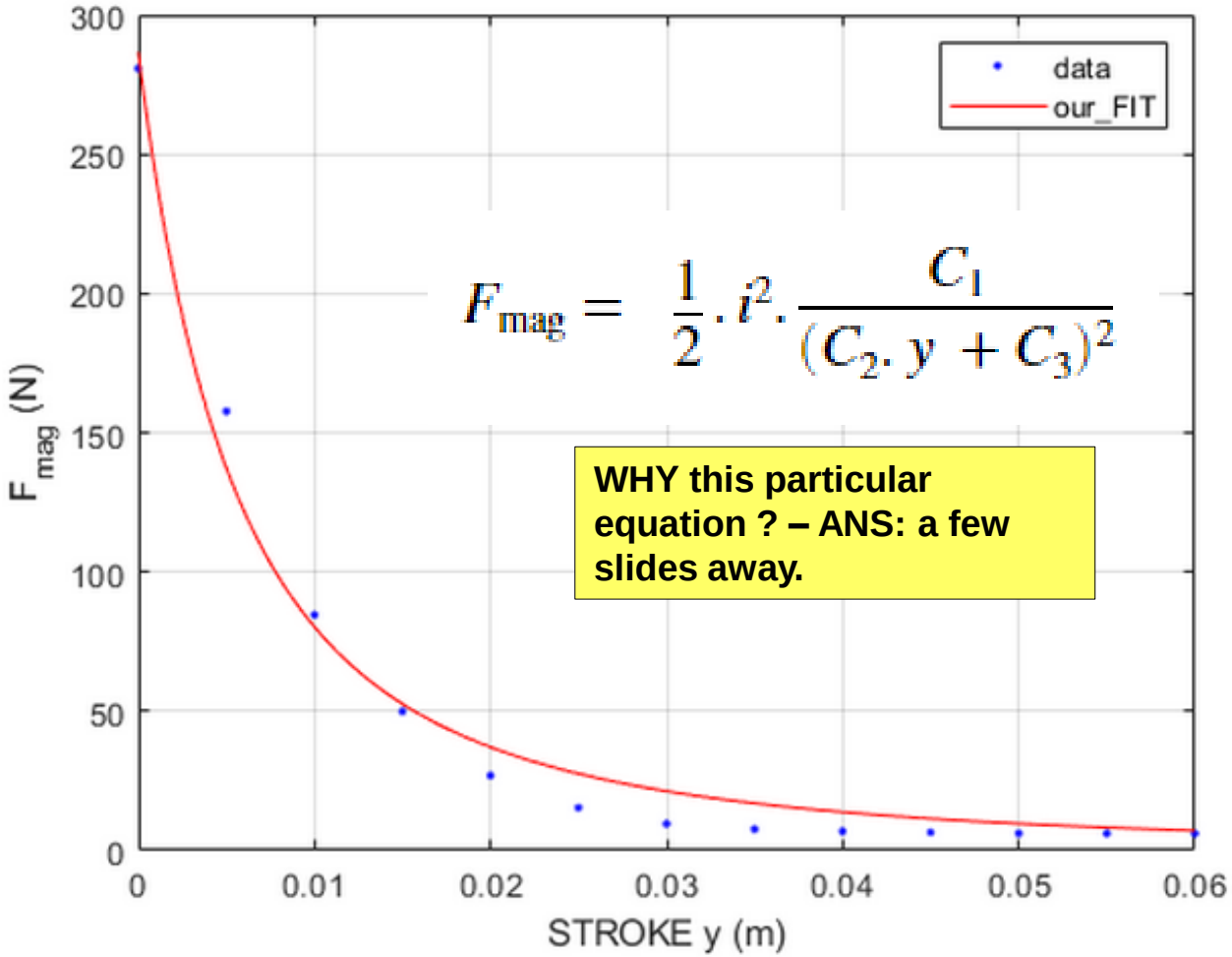
The data sheet: F_mag_FORCE vs STROKE_length



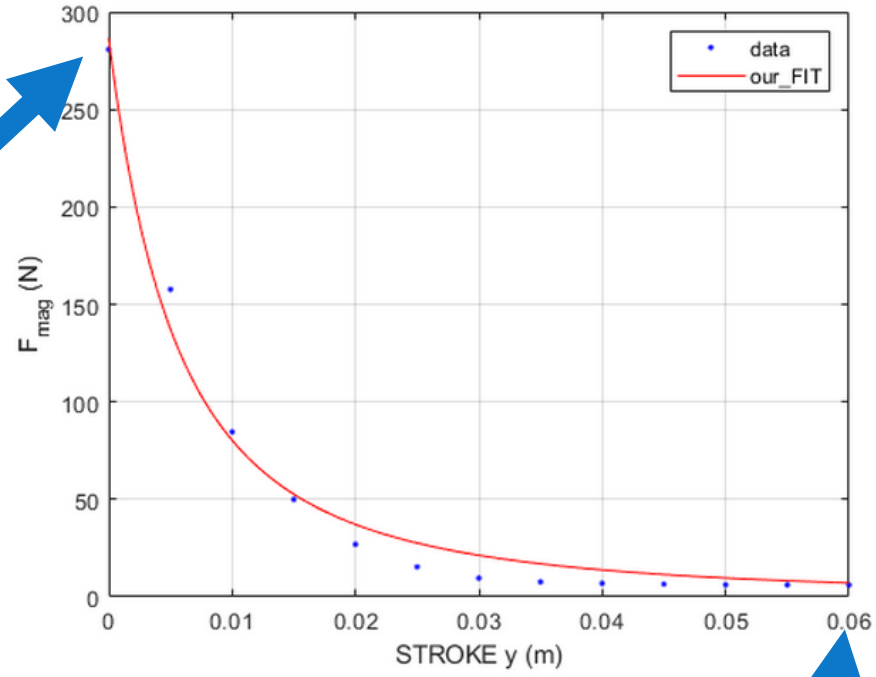

```
my_sol_OBJ =
  bh_solenoid with properties:

              mass: 0.6000
           damping: 5.3229
         stiffness: 17
      volts_rated: 7.6000
    current_rated: 4
       resistance: 1.9000
  F_mag_datasheet: [13x1 double]
   y_m_datasheet: [13x1 double]
     MAX_y_airgap: 0.0600
               C1: 20.4094
               C2: 67.0984
               C3: 0.7548
```

off data sheet

bh_explore_Fvs_S_data.mlx



F_mag_FORCE vs STROKE_length



STROKE = ZERO

STROKE at its MAXIMUM value

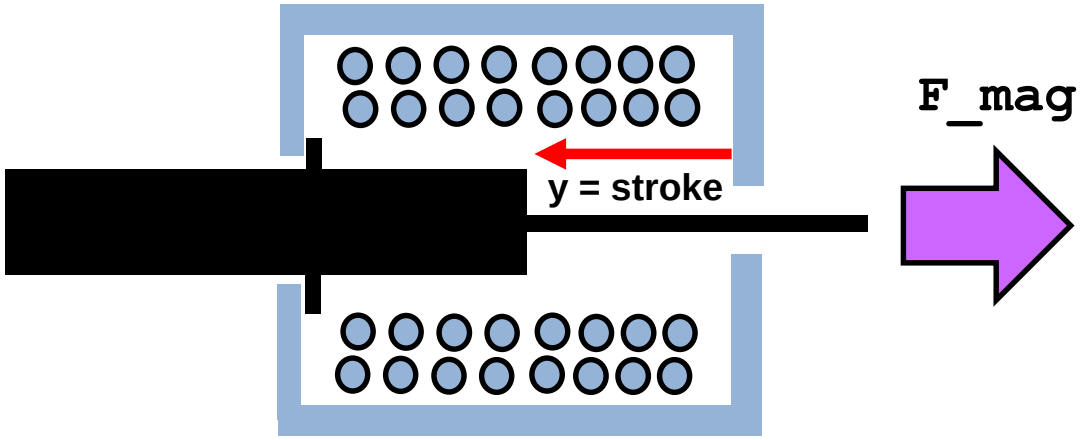
y = 0

y = y_MAX

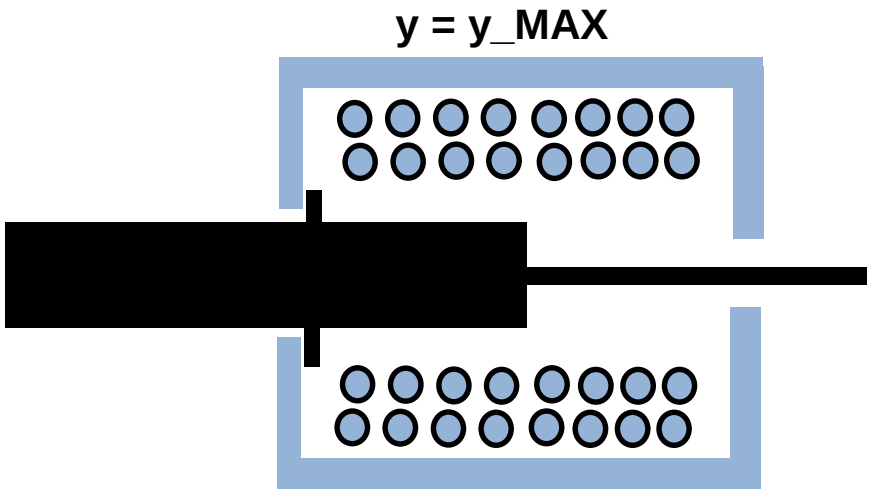
y = stroke

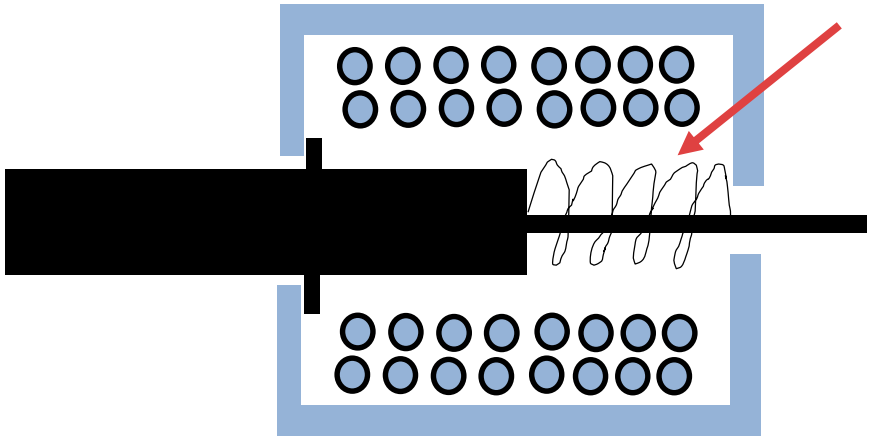
F_mag

The system that we'll model:



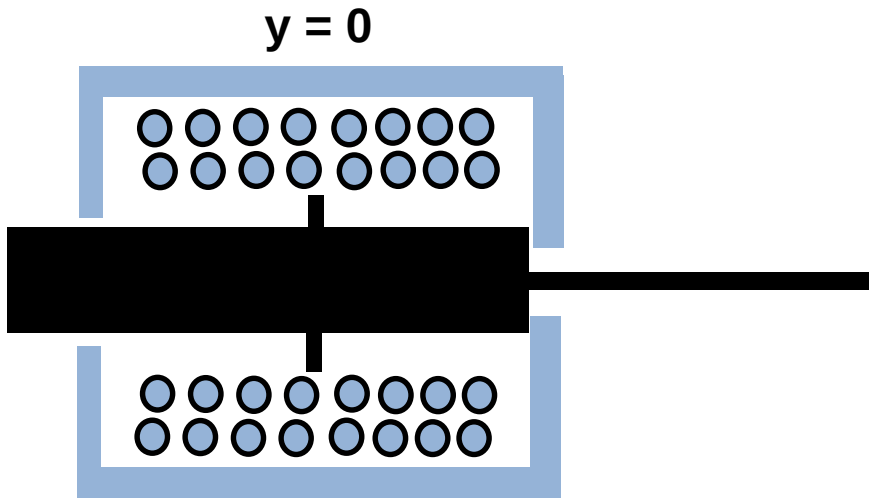
F_mag





a spring and damper

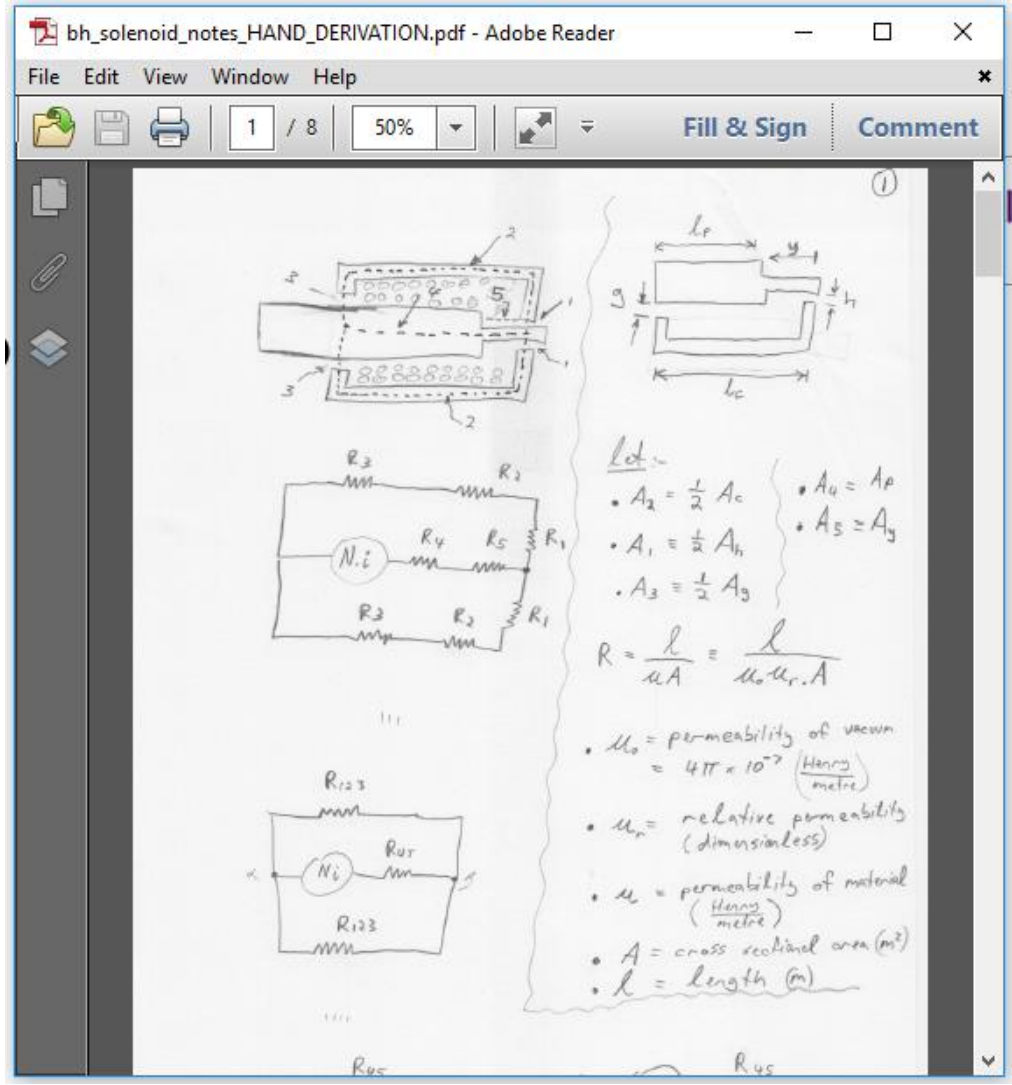
At MAX stroke, spring is in UNstretched position



The **mathematical** model:

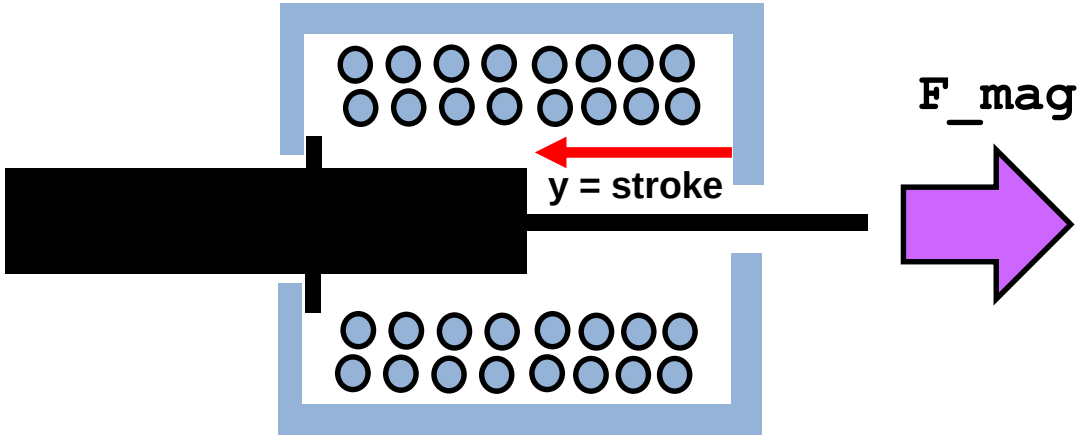
REF_01: bh_solenoid_notes_HAND_DERIVATION.pdf

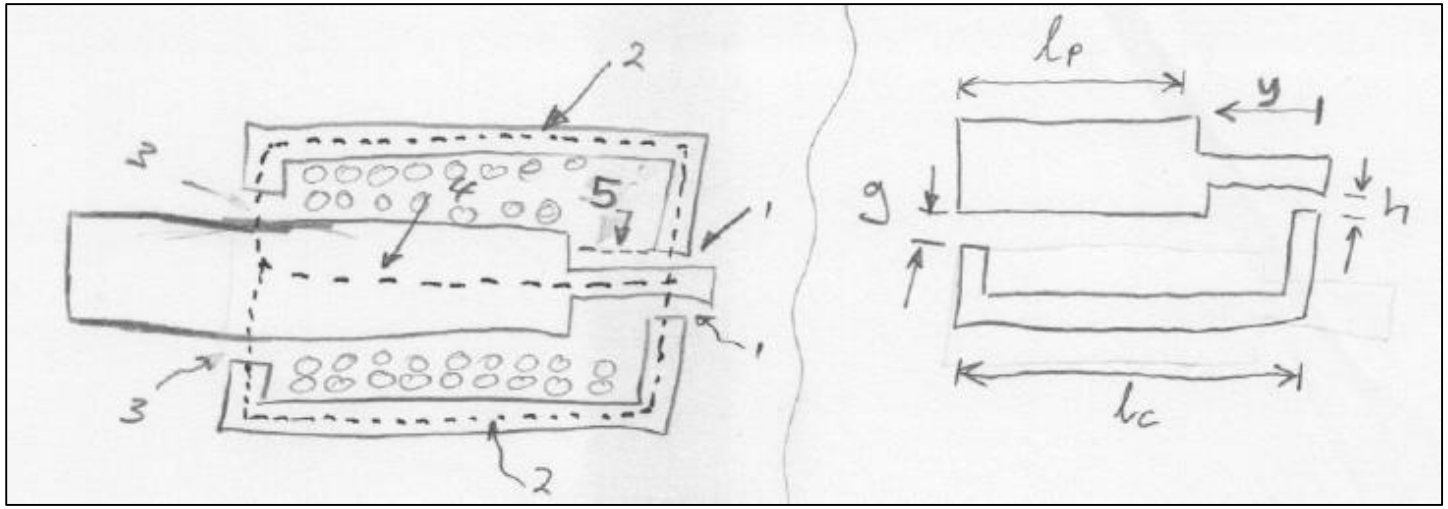
- A **Magnetic** system
  - Magnetic Force versus airgap(aka STROKE)
  - Inductance versus airgap
  - *REF_01: Pages 1-5*
- An **Electrical** system
  - Inductance versus airgap
  - Kirchoff circuit law
  - *REF_01: Page 7*
- A **Mechanical** system
  - Newton's law
  - *REF_01: Page 6*



The mathematical model:

- A Magnetic system
  - Magnetic Force versus airgap(aka STROKE)
  - Inductance versus airgap





$$F = \frac{\frac{1}{2} \cdot i^2 \cdot C_1}{(C_2 \cdot y + C_3)^2}$$

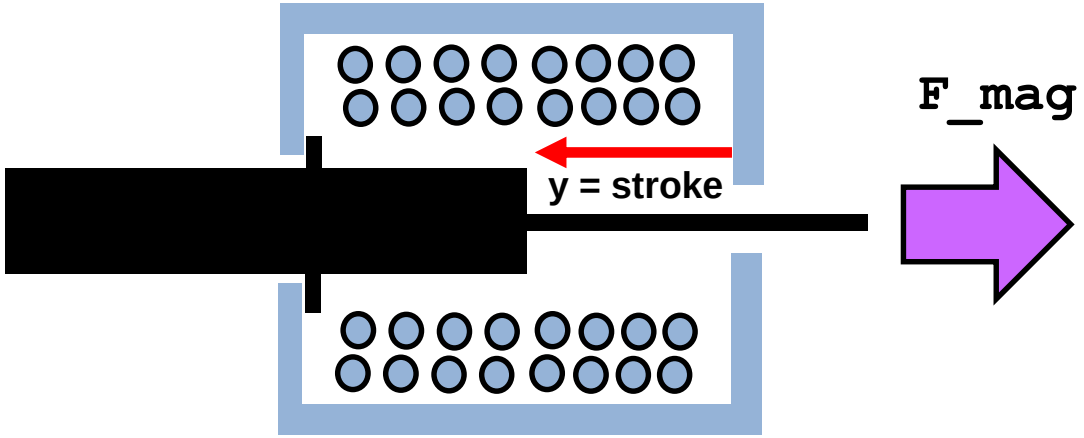
$$L = \frac{\left(\frac{C_1}{C_2}\right)}{C_2 \cdot y + C_3}$$

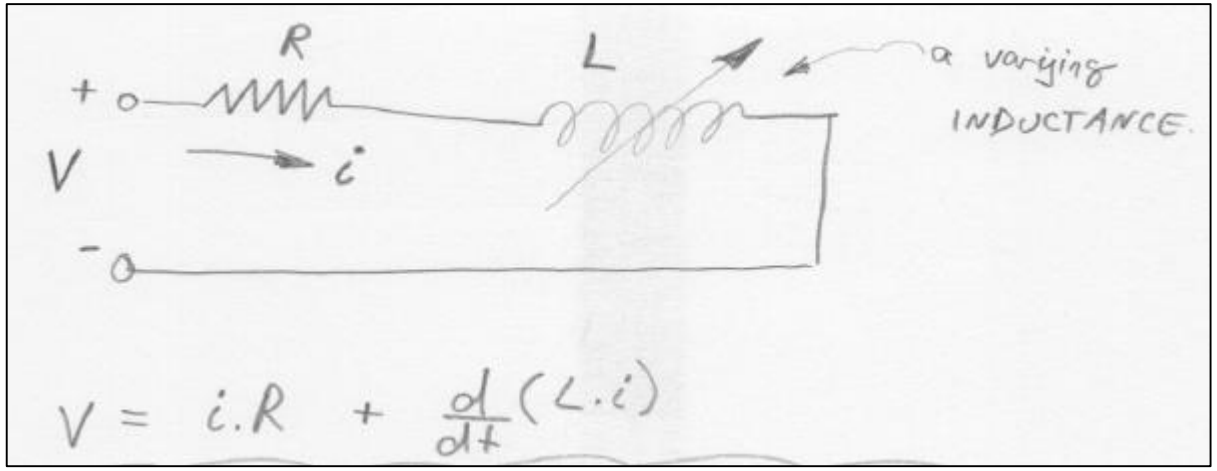
Note:

1. *We used data from the manufacturer's data sheet … and we did a Curve fit on this data to determine the coefficients C1, C2, C3.*
2. *We've implemented these expressions for Fmag and L into MATLAB functions, see <**bh_func_calc_my_F.m**> and <**bh_func_calc_my_L.m**>*

The mathematical model:

- An Electrical system
  - Inductance versus airgap
  - Kirchoff circuit law





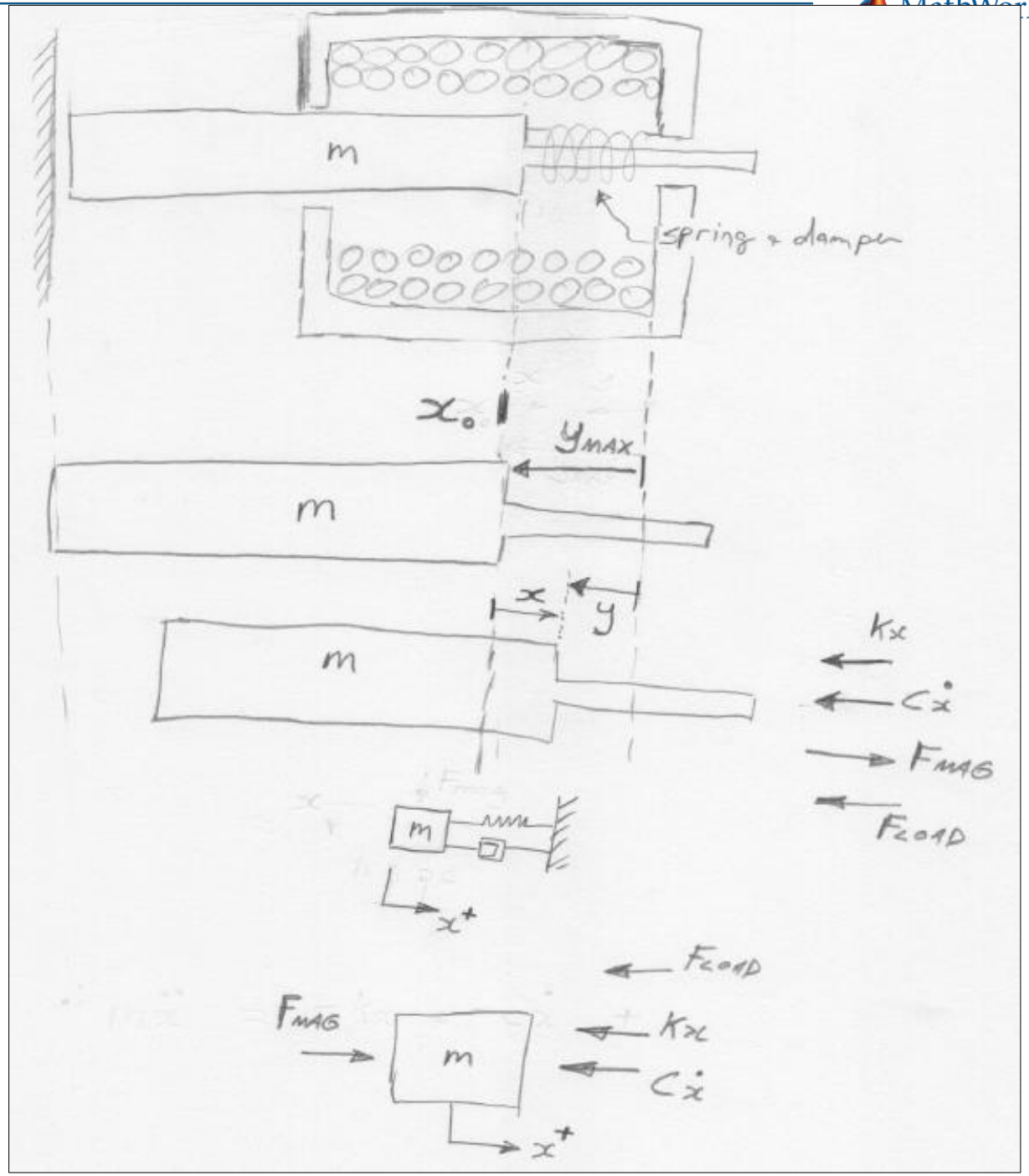
$$V = (i.R) + \left(L.\frac{di}{dt}\right) + \left(i\,\frac{dL}{dy}.\frac{dy}{dt}\right)$$

Note:
1. *We've implemented these expressions for dL/dy into a MATLAB function, see <**bh_func_calc_my_dLdy.m**>*

The mathematical model:

- A Mechanical system
  - Newton's law

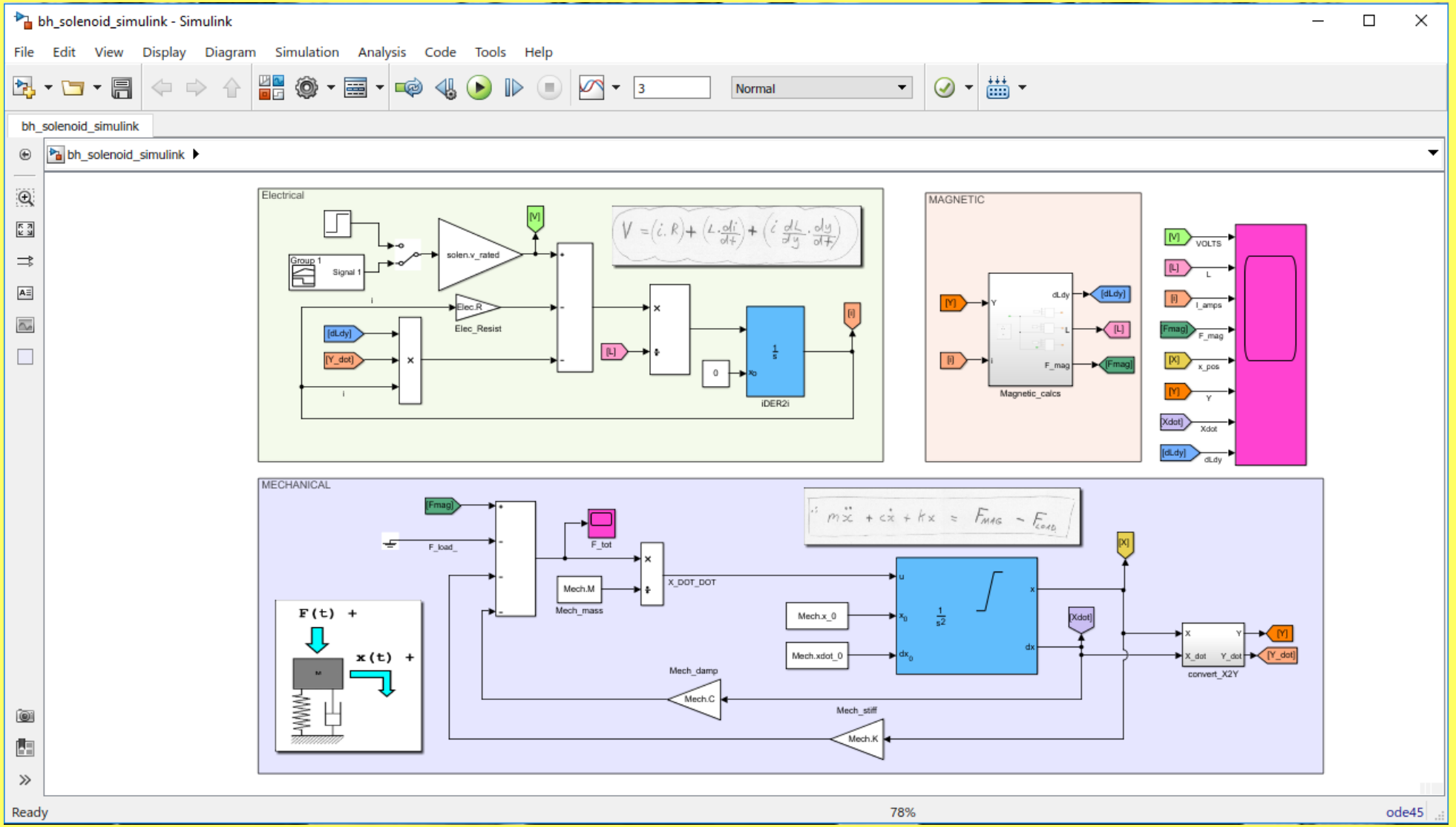
$$\therefore m\ddot{x} + c\dot{x} + kx = F_{MAG} - F_{LOAD}$$



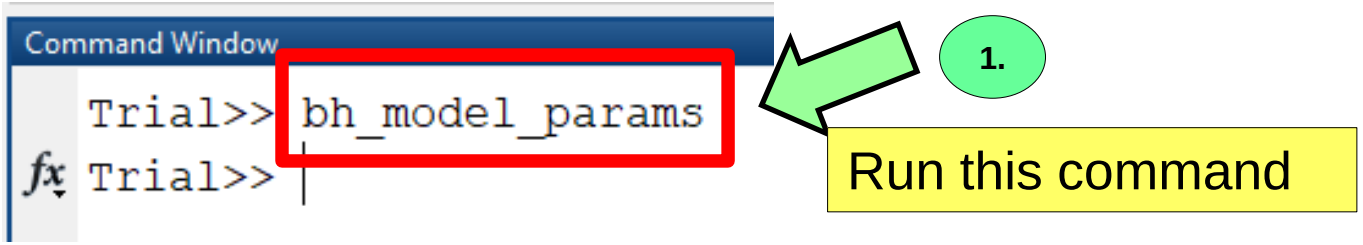
And let's say that the position x(t) of the plunger is constrained according to the limits:

- x_min = 0;
- x_max = MAX_airgap = y_max

Model: Simulink



Load parameters into the MATLAB base workspace



2. … and now your workspace should contain these parameters

Workspace

Name	Value	Size	Class
Coeffs	*1x1 struct*	1x1	struct
Elec	*1x1 struct*	1x1	struct
Mech	*1x1 struct*	1x1	struct
my_sol_OBJ	*1x1 bh_solenoid*	1x1	bh_solenoid
solen	*1x1 struct*	1x1	struct
tmp_F_mag	*13x1 double*	13x1	double
tmp_y_m	*13x1 double*	13x1	double

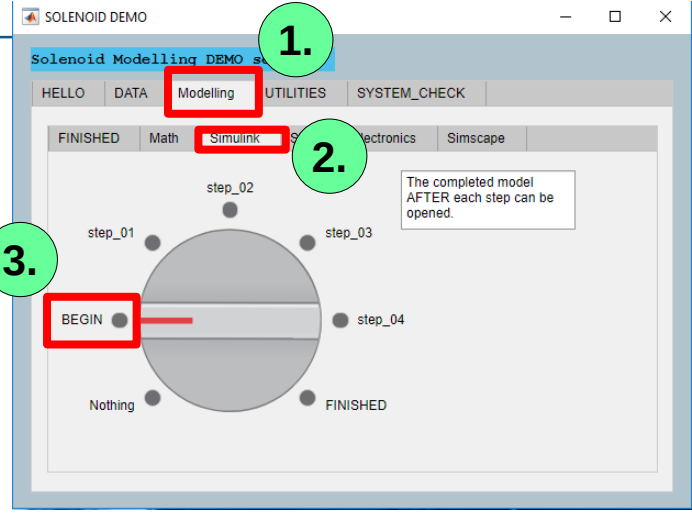
```
Trial>> Elec

Elec =

  struct with fields:

    R: 1.9000
```

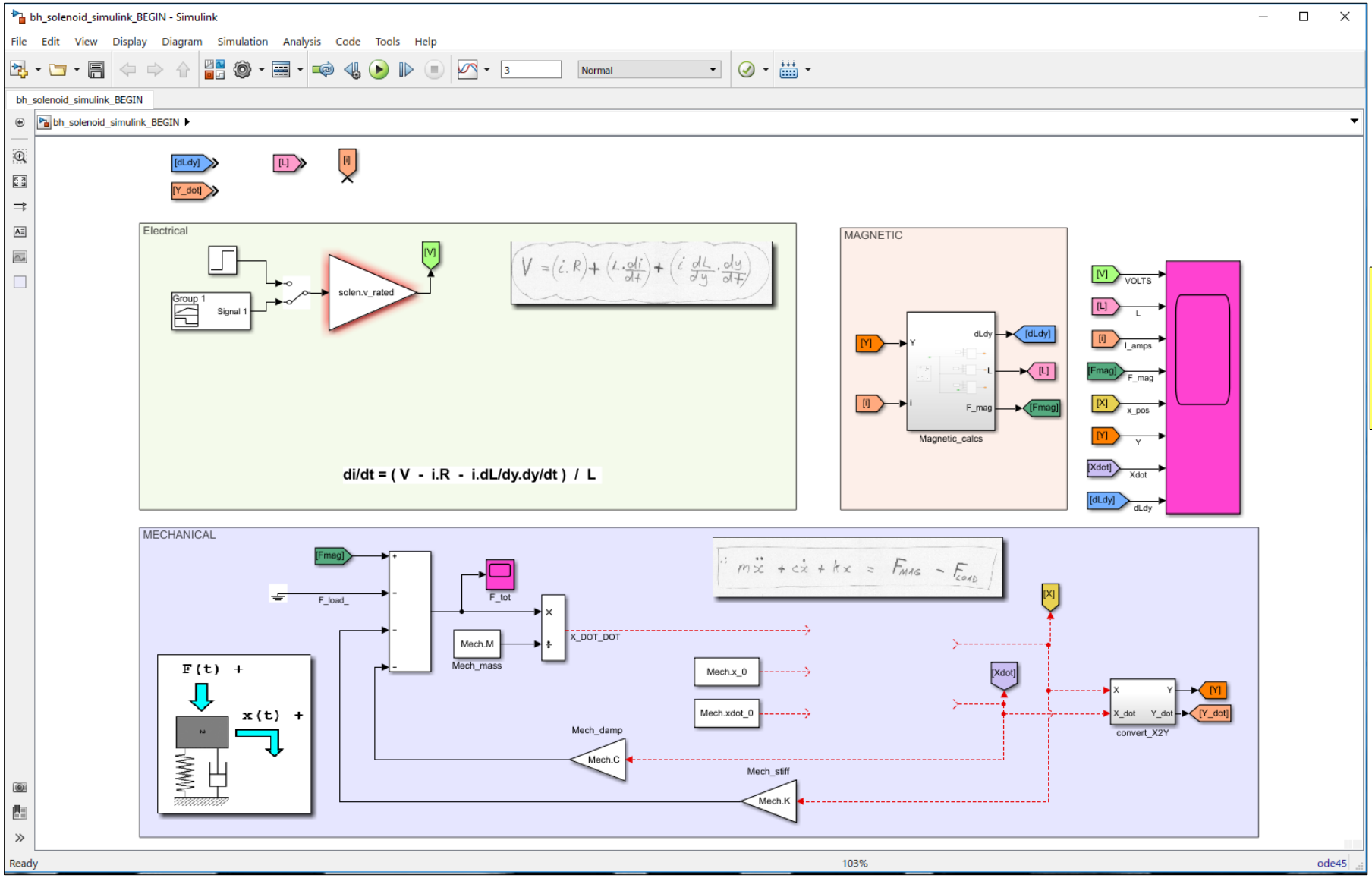
```
Trial>> Mech

Mech =

  struct with fields:

                K: 17
                C: 5.3229
                M: 0.6000
              x_0: 0
           xdot_0: 0
            x_min: 0
            x_max: 0.0600
    MAX_y_airgap: 0.0600
```

```
Trial>> solen

solen =

  struct with fields:

     v_rated: 7.6000
     i_rated: 4
    F_mag_12: [2×1 double]
      y_m_12: [2×1 double]
       F_mag: [13×1 double]
         y_m: [13×1 double]
```

```
Trial>> Coeffs

Coeffs =

  struct with fields:

    C123: [20.4094 67.0984 0.7548]
```

Simulink model: **BEGIN**

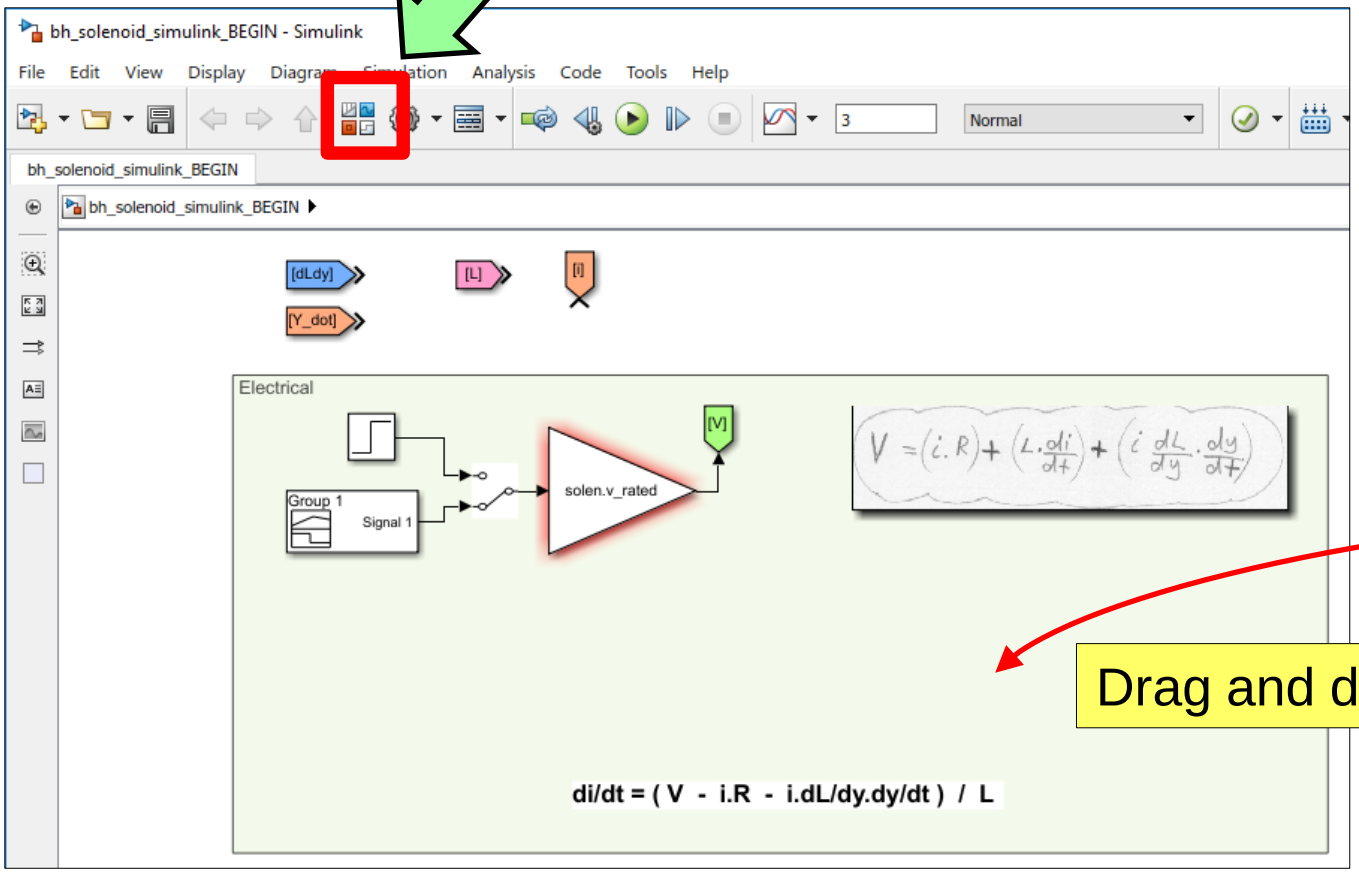




We'll start with a model that looks like this

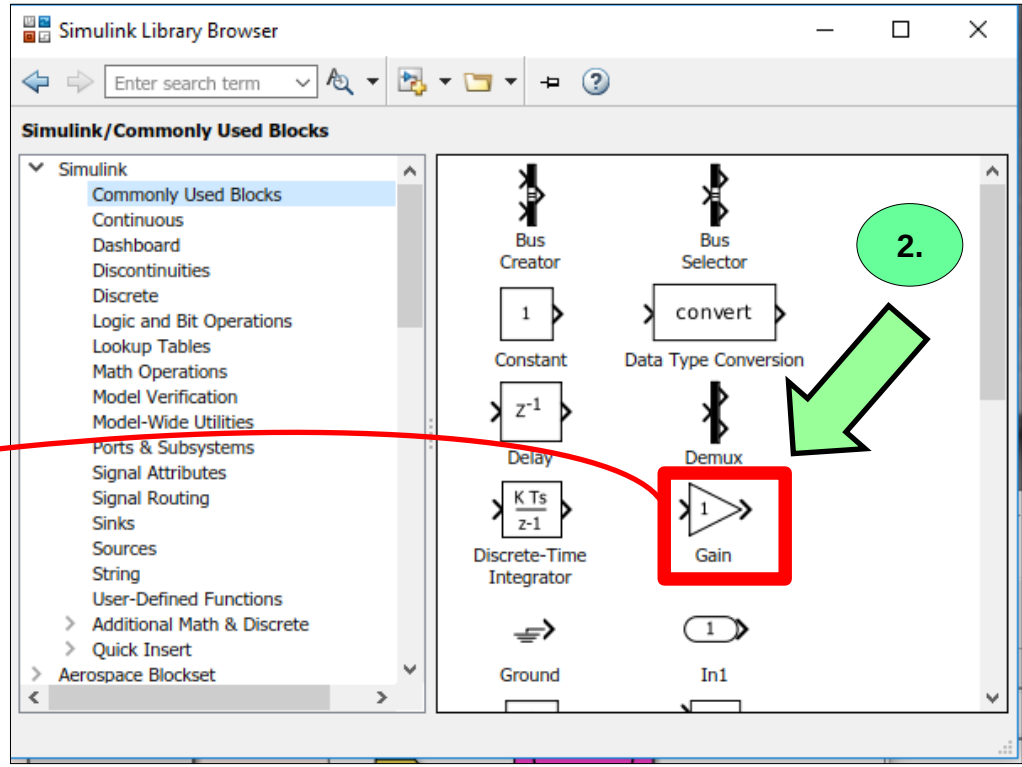
Simulink model: **BEGIN**

1. Launch the Simulink Library Browser

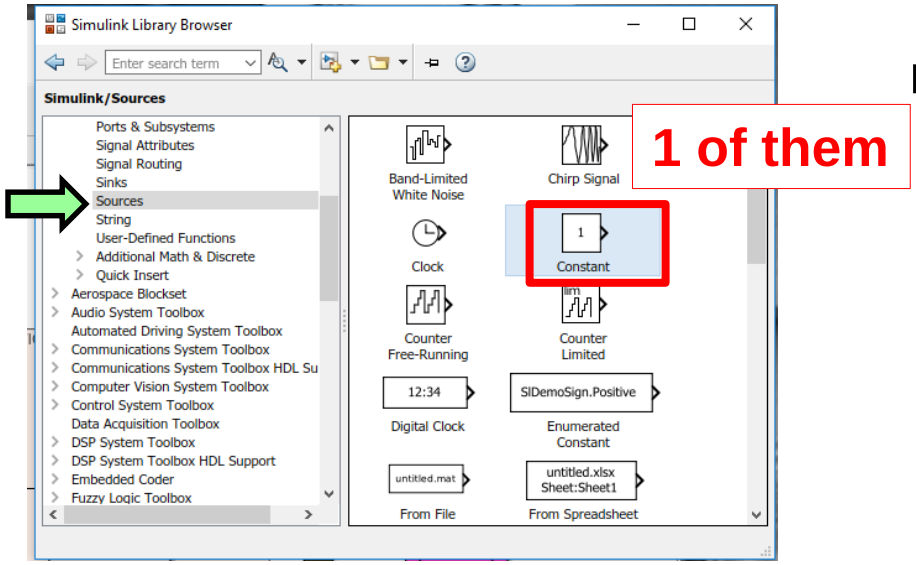


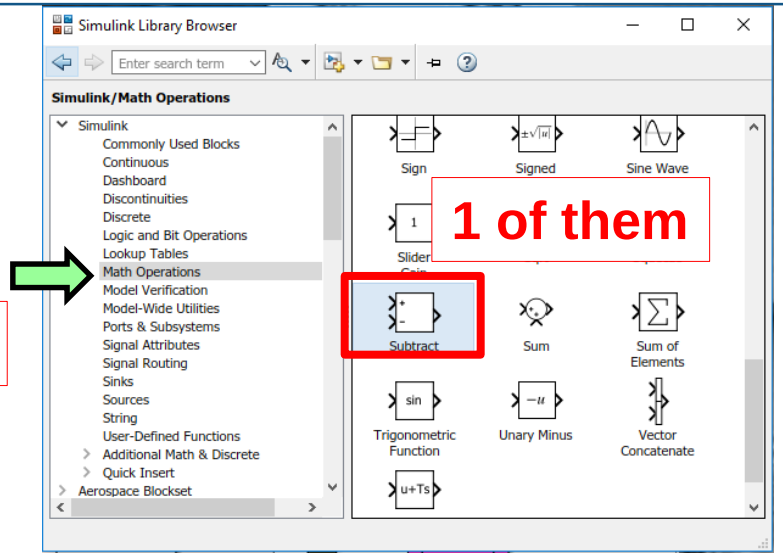
Drag and drop

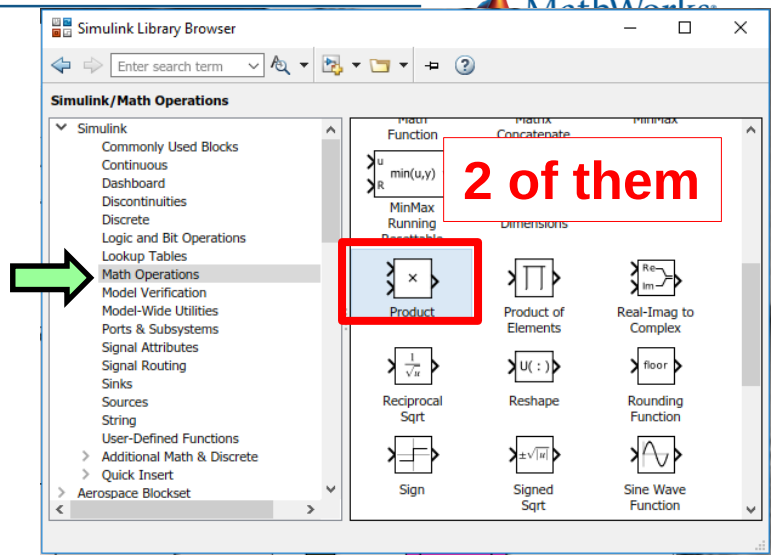
The Simulink Library Browser

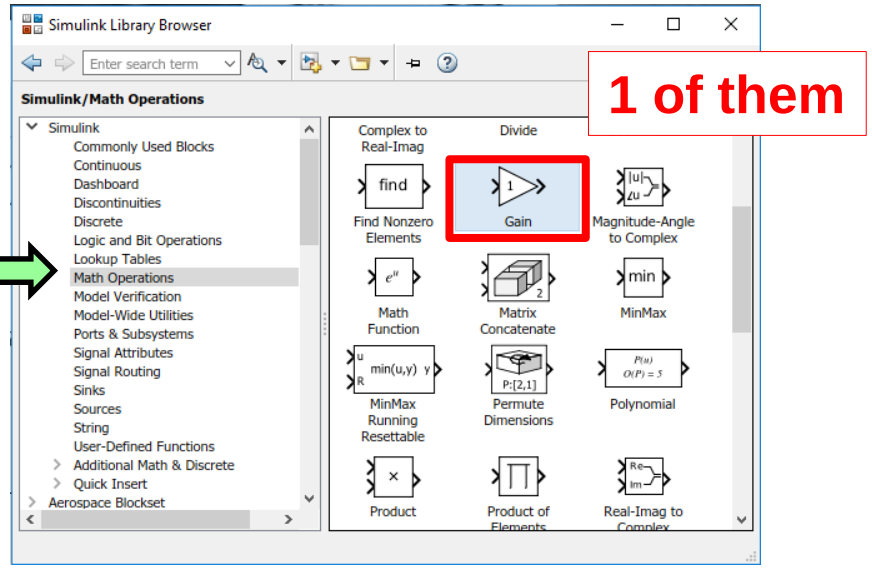


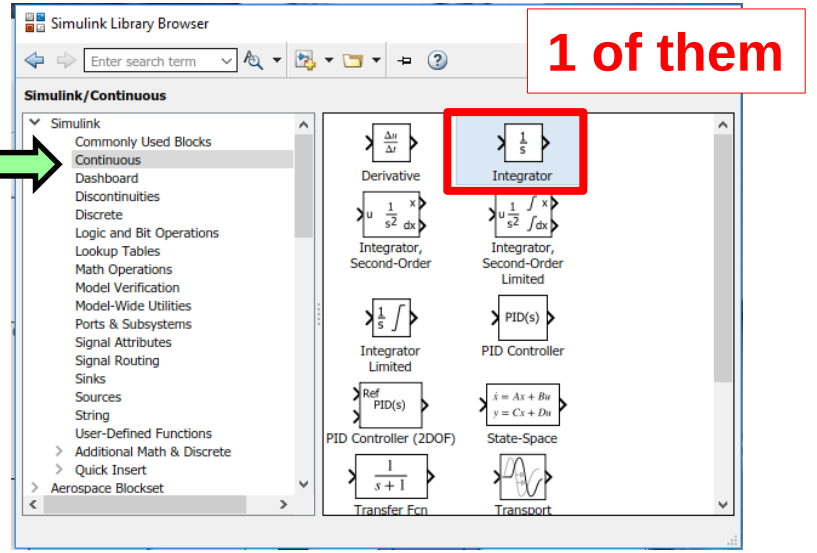
Get some blocks



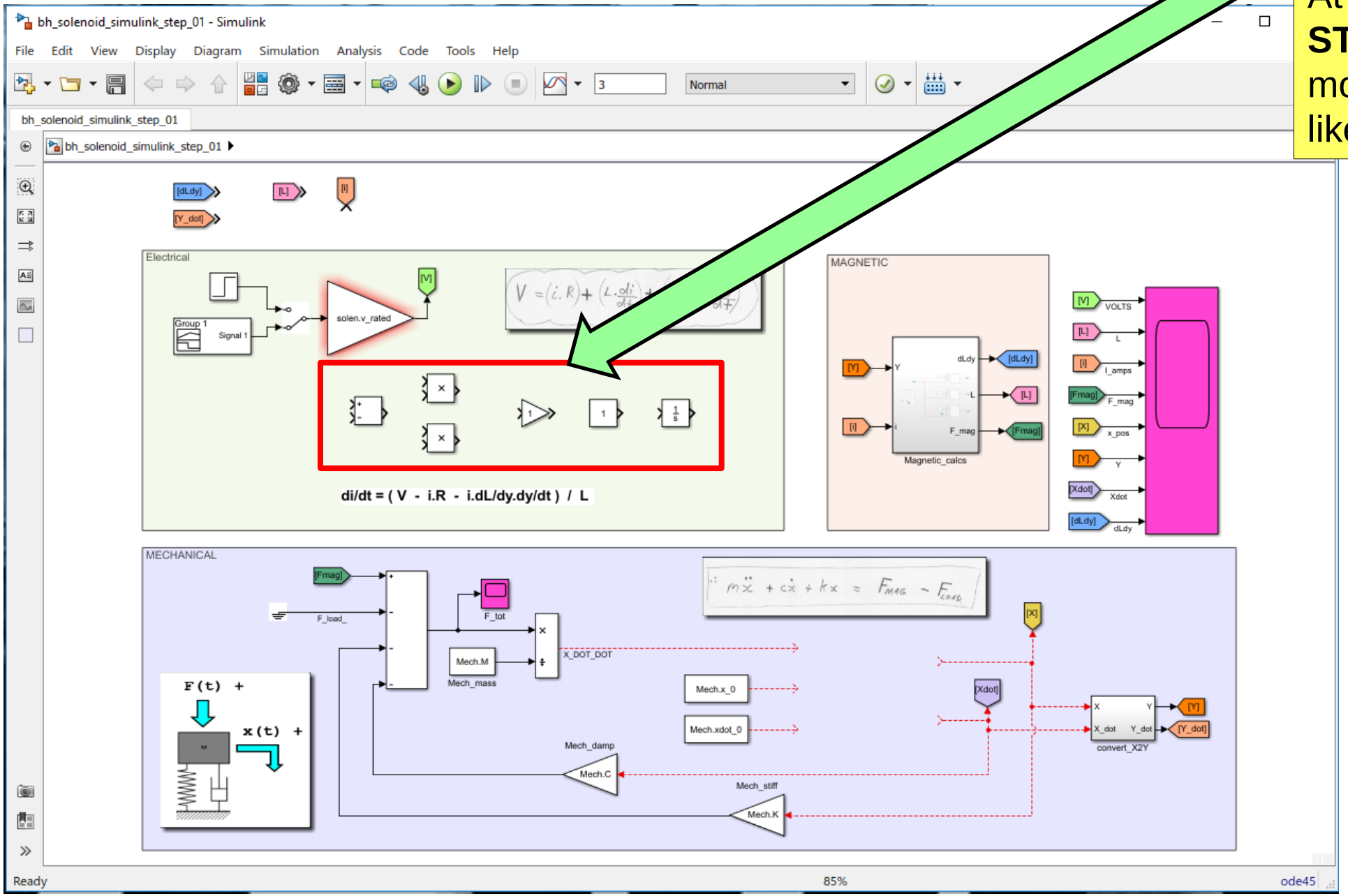








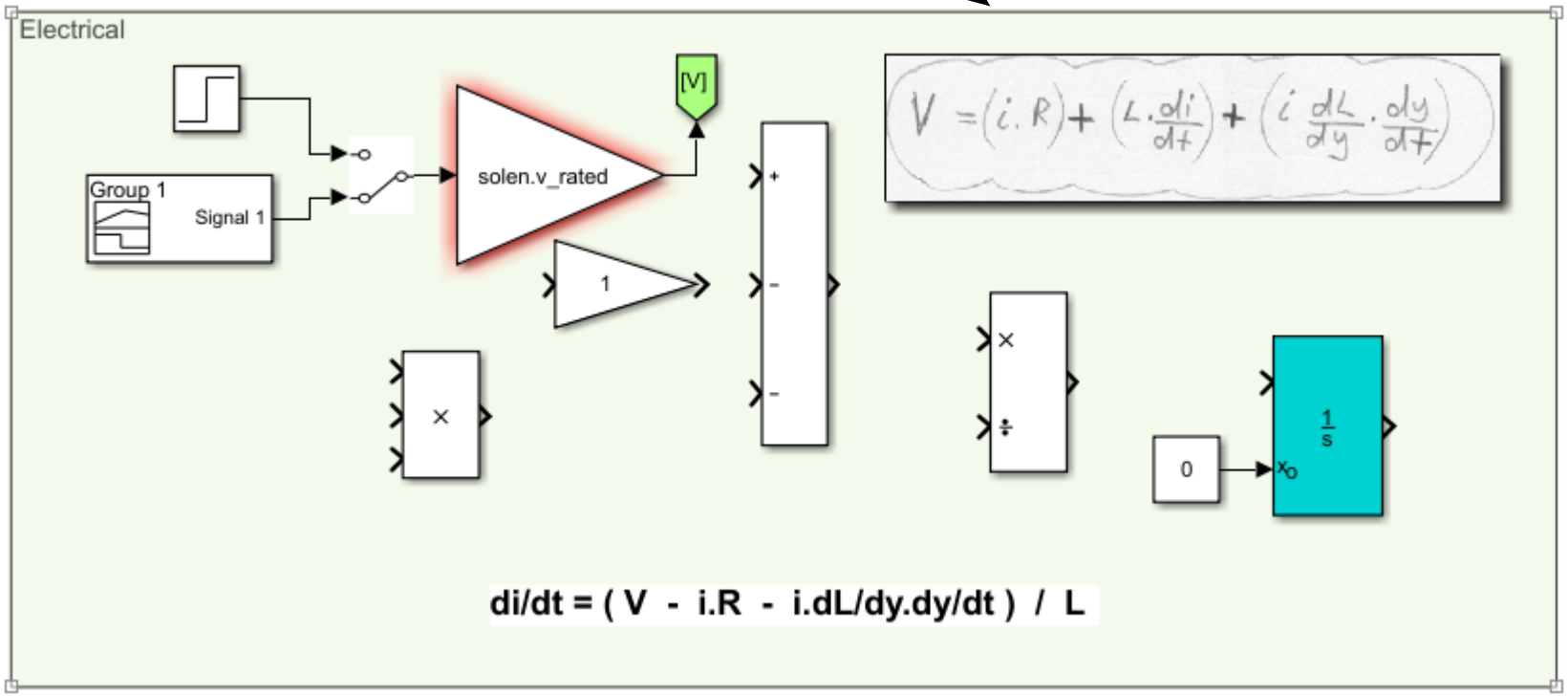
Simulink model: **step_01**



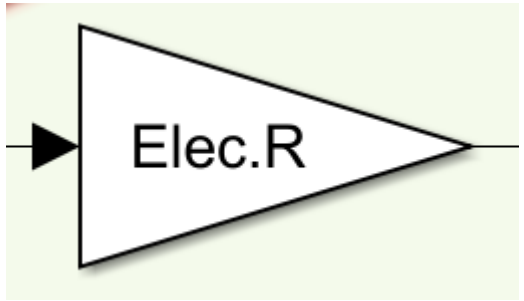
At the end of **STEP_01** your model should look like this

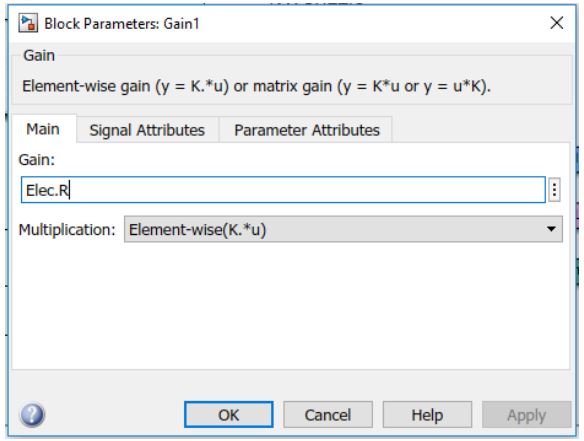
Simulink model: **step_02**

At the end of **STEP_02** your model should look like this

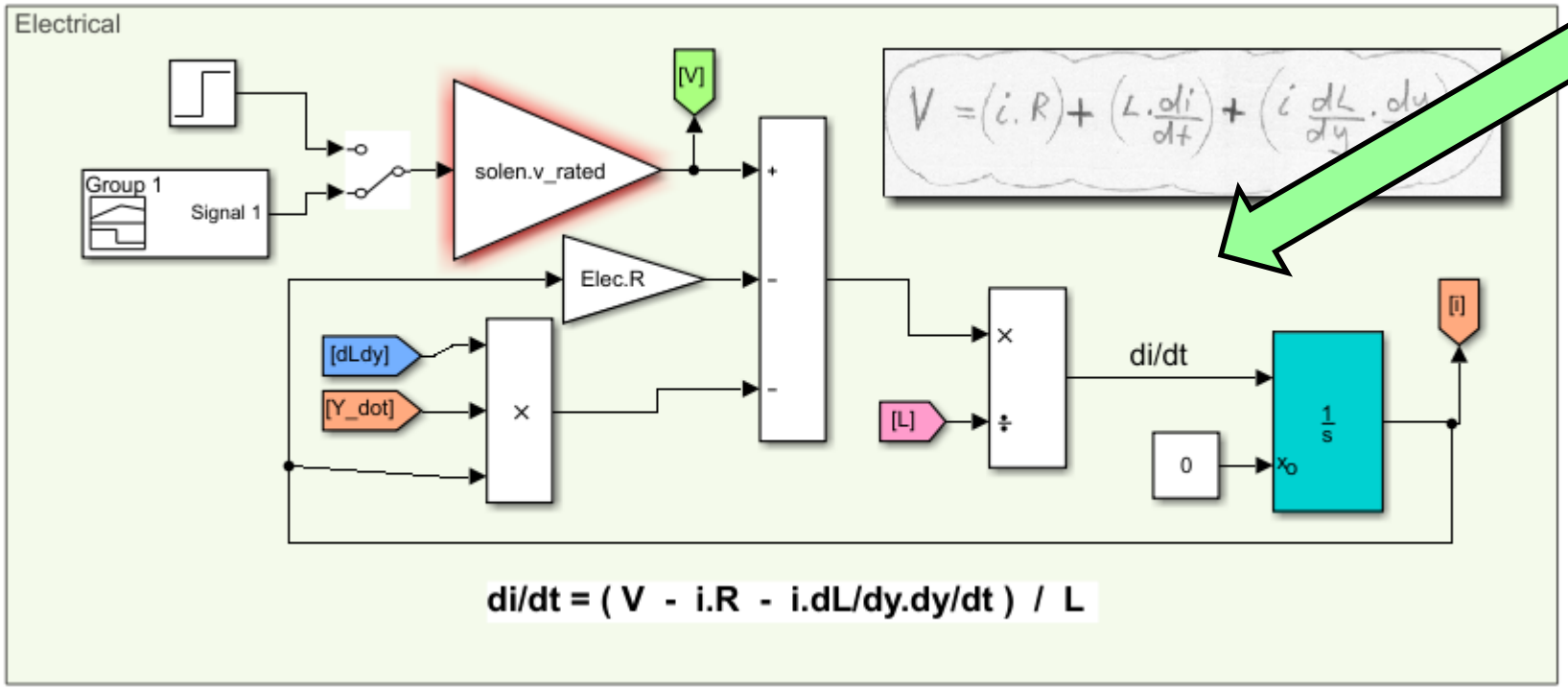


Simulink model: **step_03**



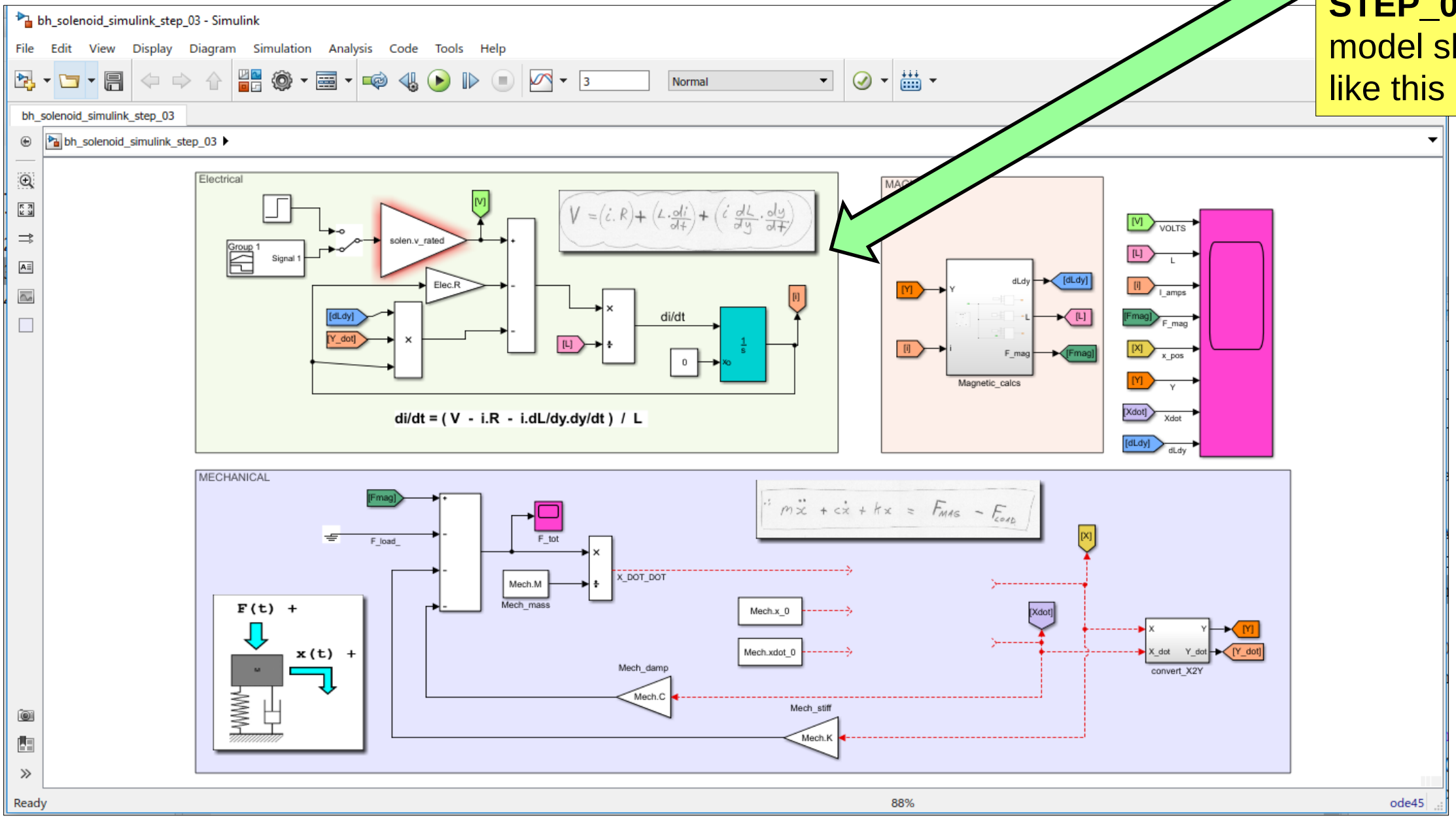


At the end of **STEP_03** your model should look like this

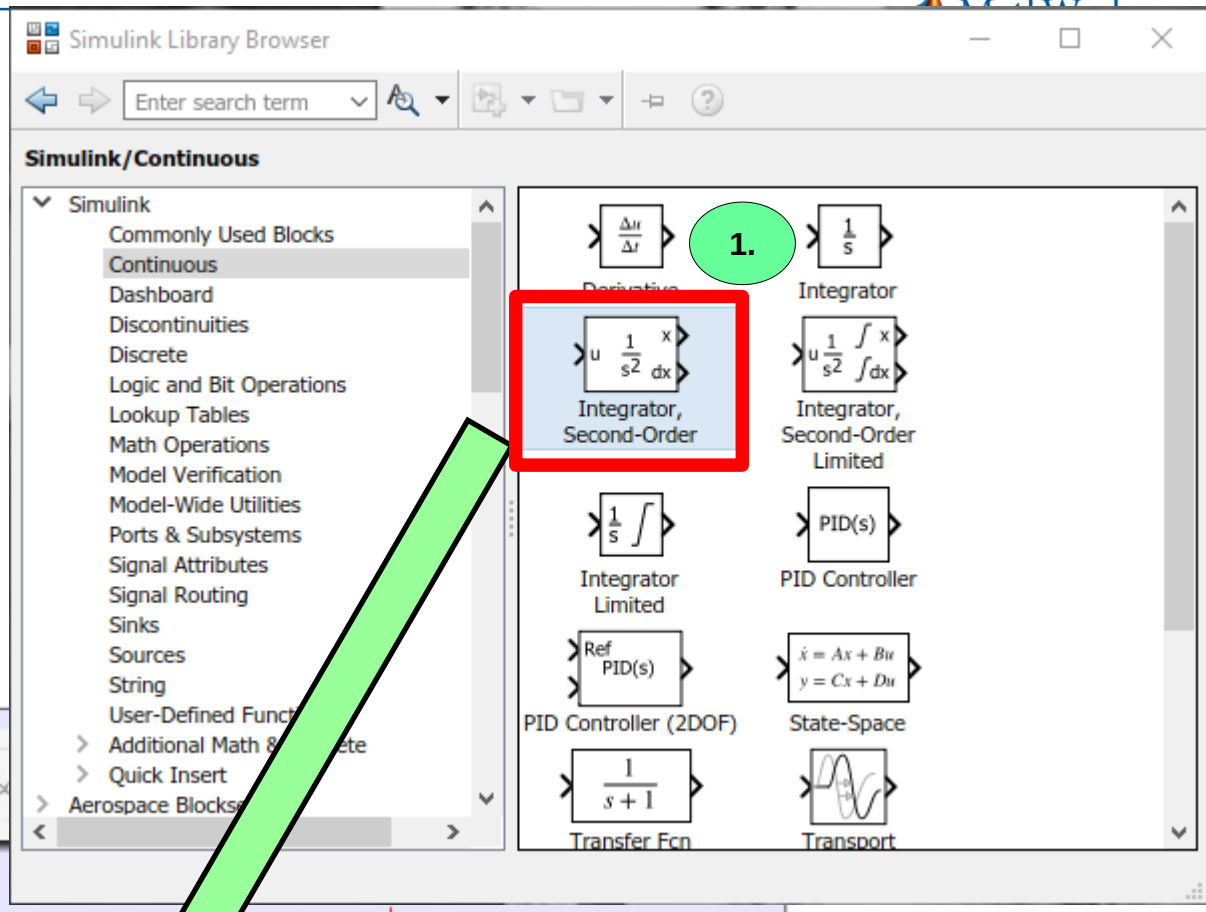


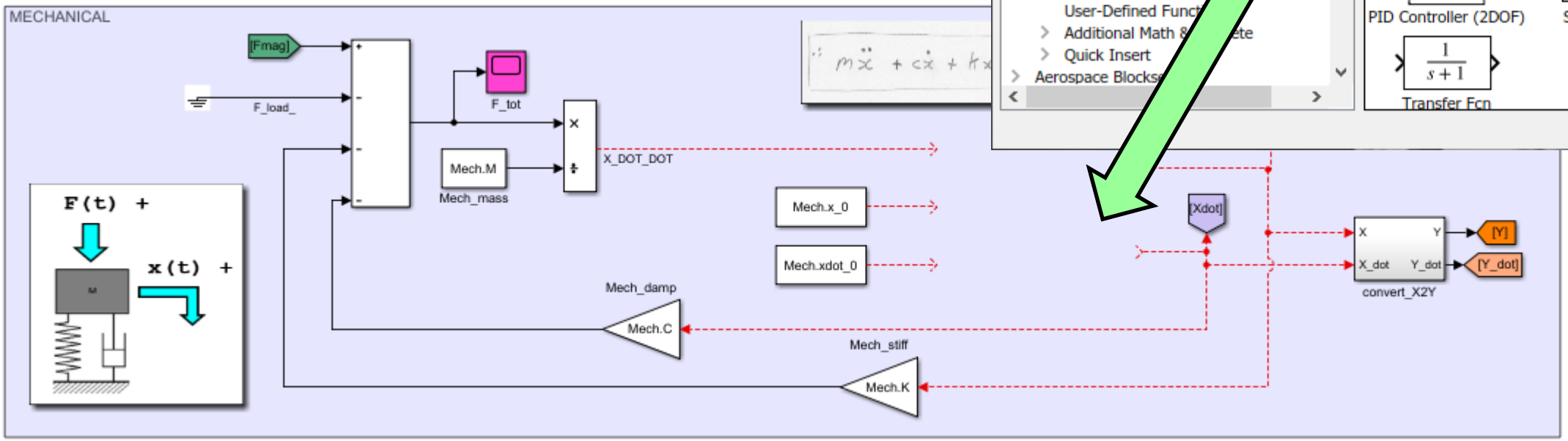
Simulink model: **step_03**

At the end of **STEP_03** your model should look like this



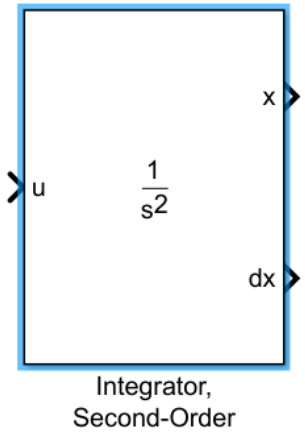
Simulink model: **step_04**



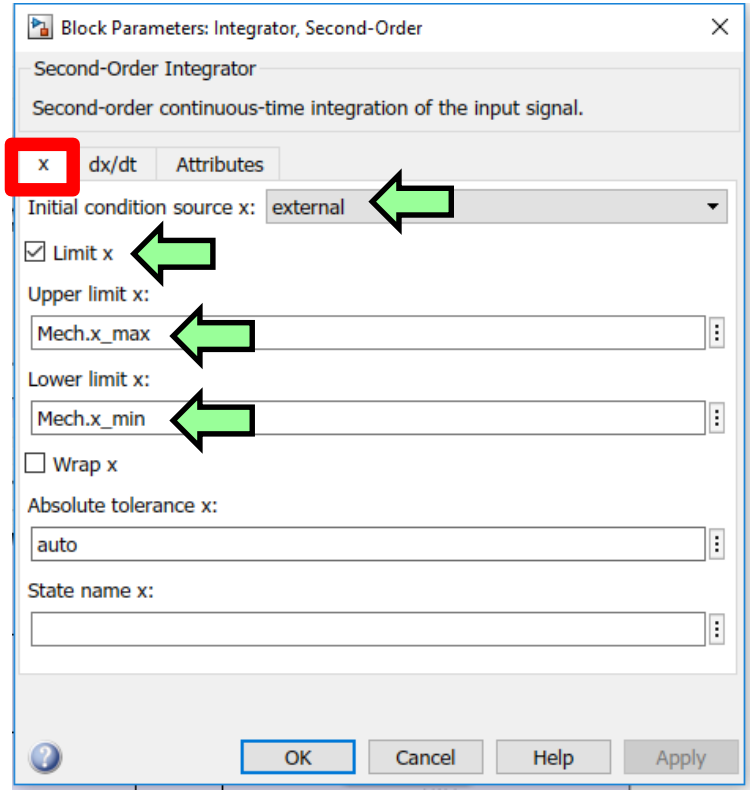


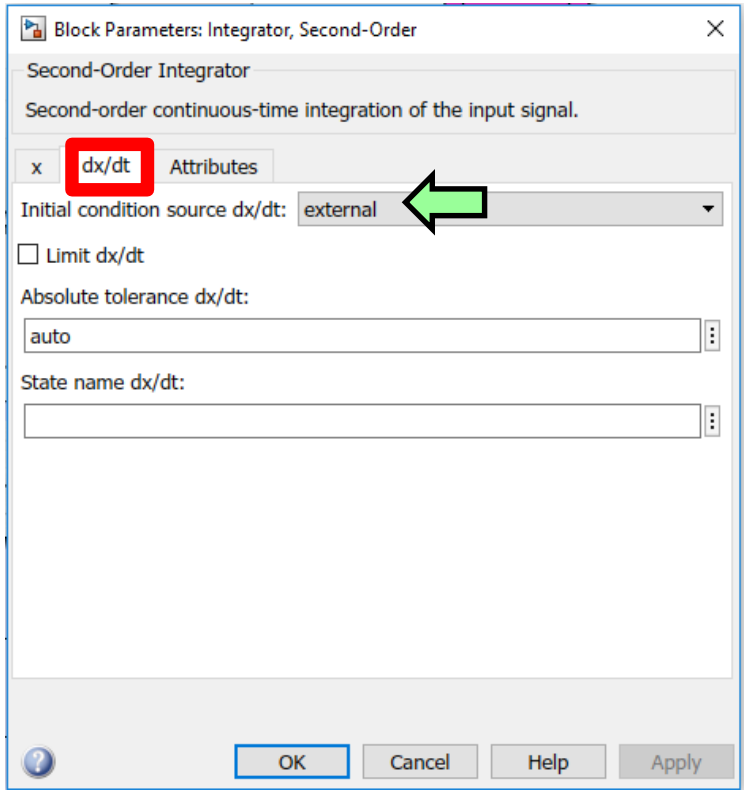
Simulink model: **step_04**

It starts like this

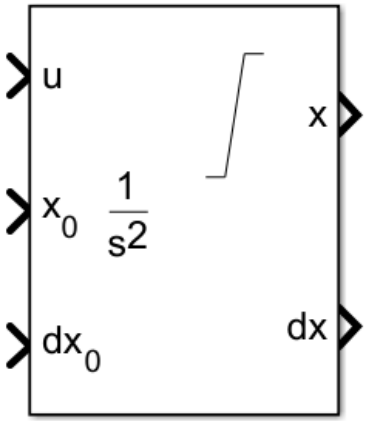


Integrator, Second-Order





It ends up like this

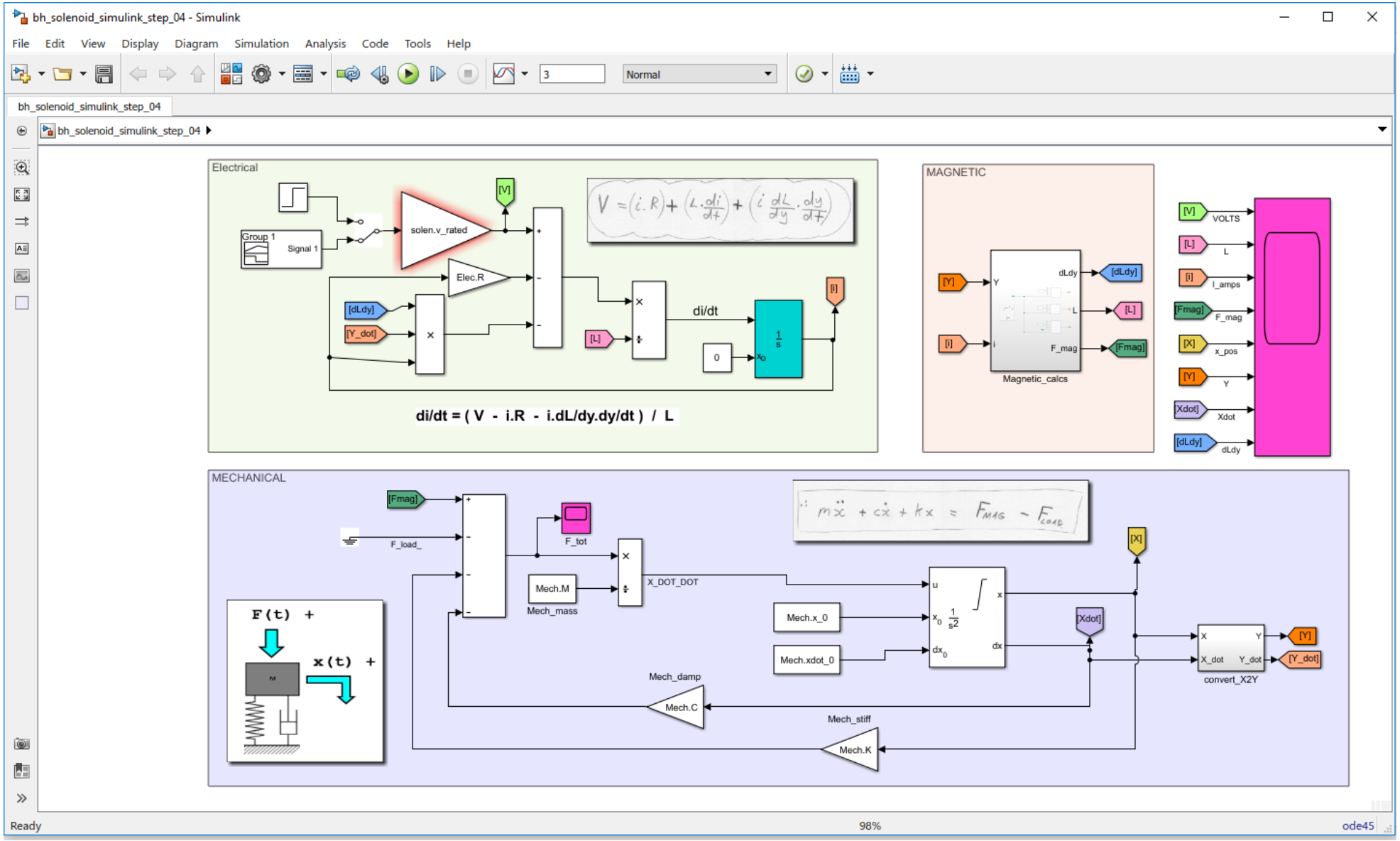


So ? – so we've configured the block so that the **VELOCITY** is reset to it's IC whenever the **POSITION** hits the MAX or MIN limits.

So ? – so this is "kind of" like modelling HARD stops at the ends of the **POSITION** limits

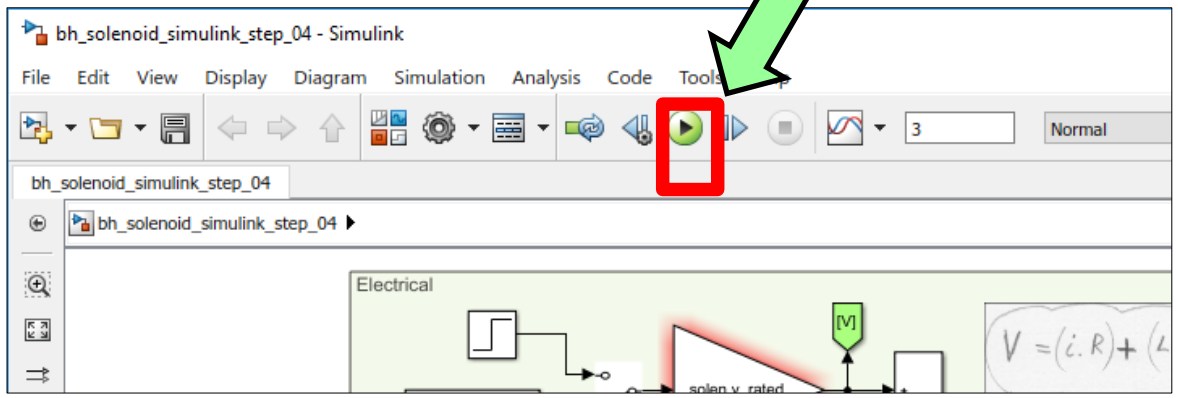
Simulink model: **step_04**

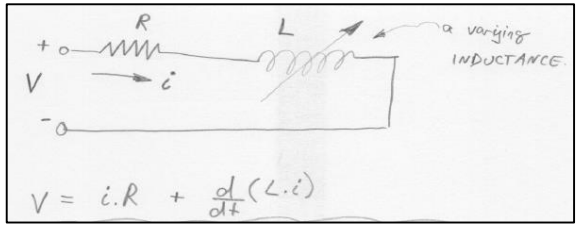
At the end of **STEP_04** your model should look like this

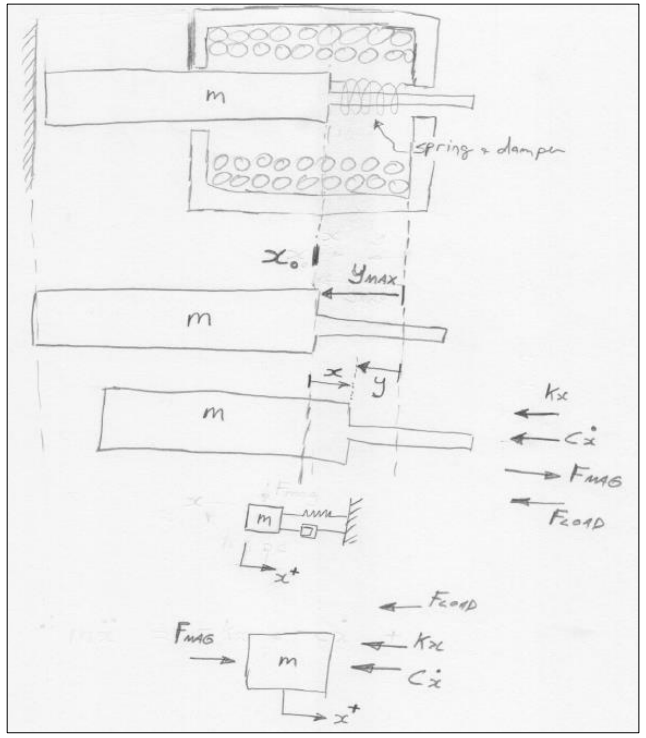


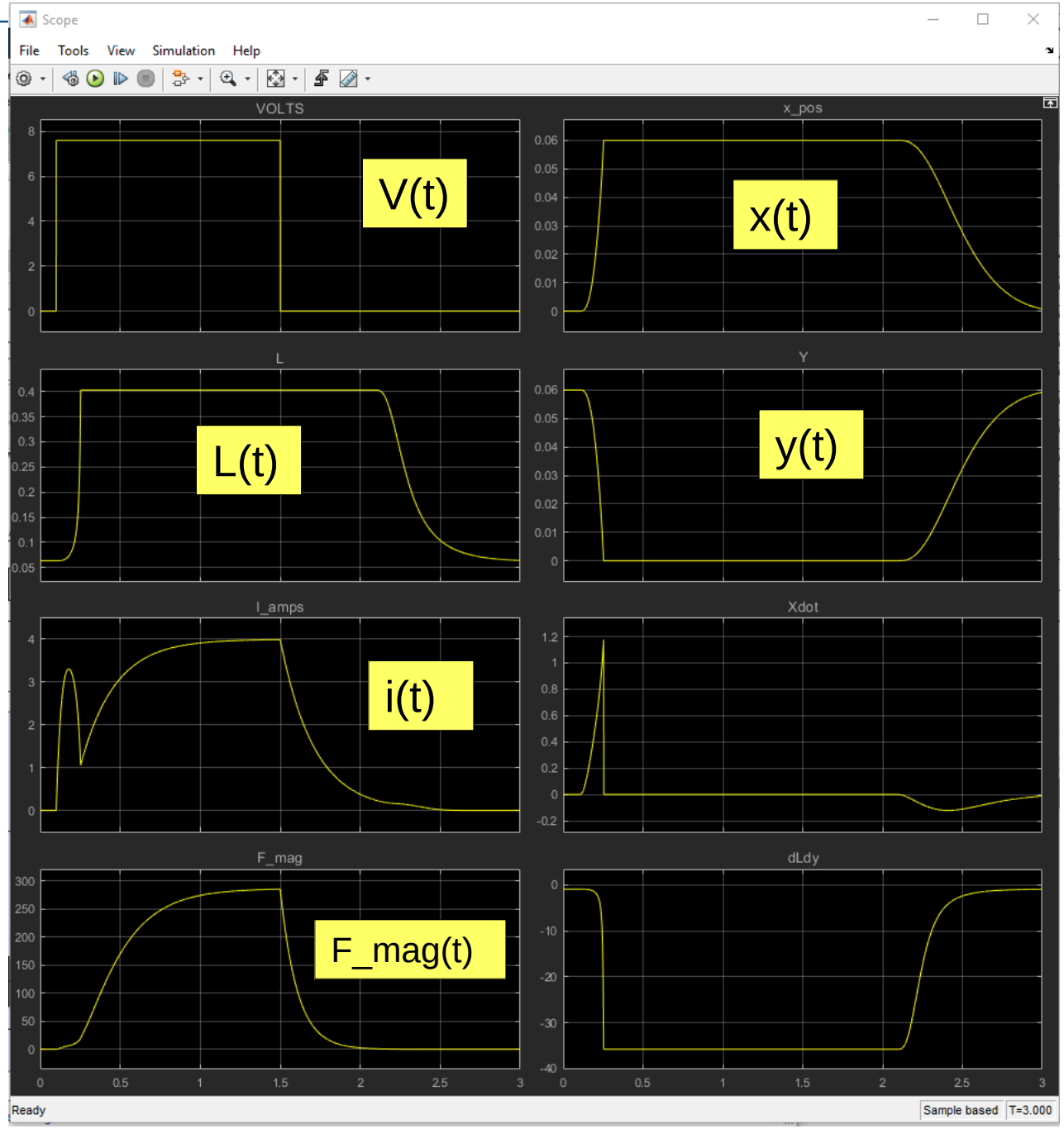
Simulink model

RUN your simulation

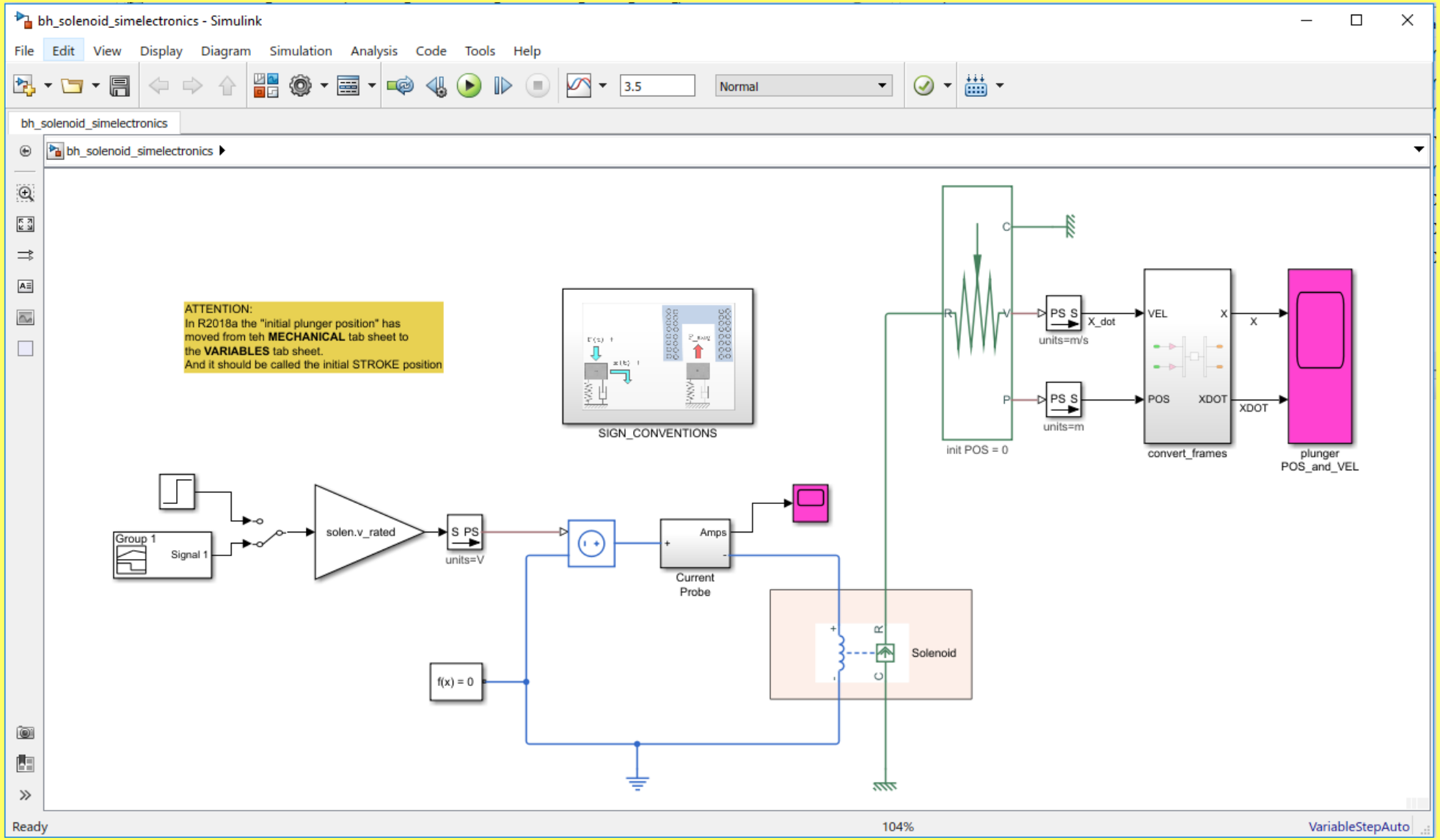




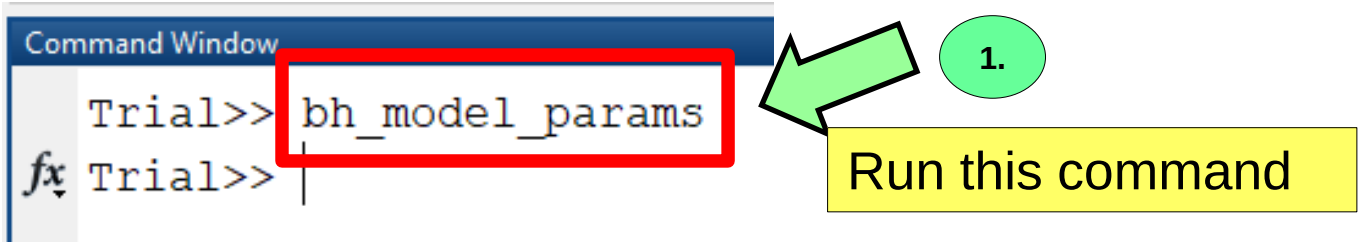




Model: Simscape Electronics



Load parameters into the MATLAB base workspace



2. … and now your workspace should contain these parameters

Workspace

Name	Value	Size	Class
Coeffs	*1x1 struct*	1x1	struct
Elec	*1x1 struct*	1x1	struct
Mech	*1x1 struct*	1x1	struct
my_sol_OBJ	*1x1 bh_solenoid*	1x1	bh_solenoid
solen	*1x1 struct*	1x1	struct
tmp_F_mag	*13x1 double*	13x1	double
tmp_y_m	*13x1 double*	13x1	double

```
Trial>> Elec

Elec =

  struct with fields:

    R: 1.9000
```

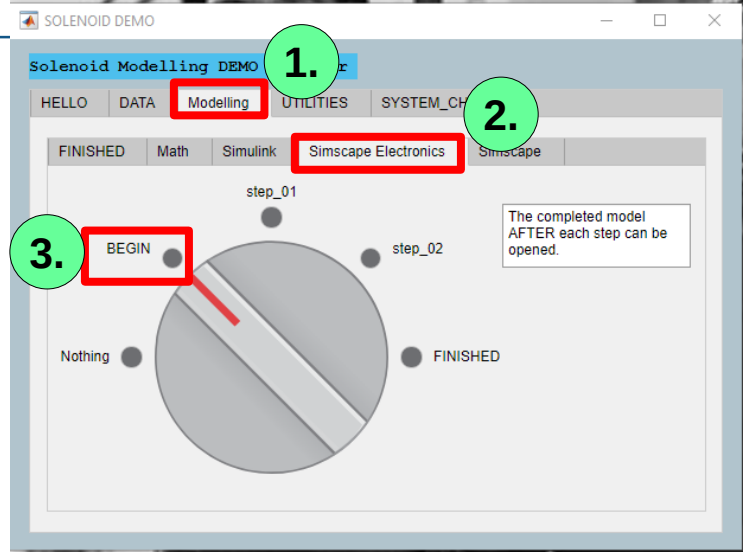
```
Trial>> Mech

Mech =

  struct with fields:

                K: 17
                C: 5.3229
                M: 0.6000
              x_0: 0
           xdot_0: 0
            x_min: 0
            x_max: 0.0600
    MAX_y_airgap: 0.0600
```

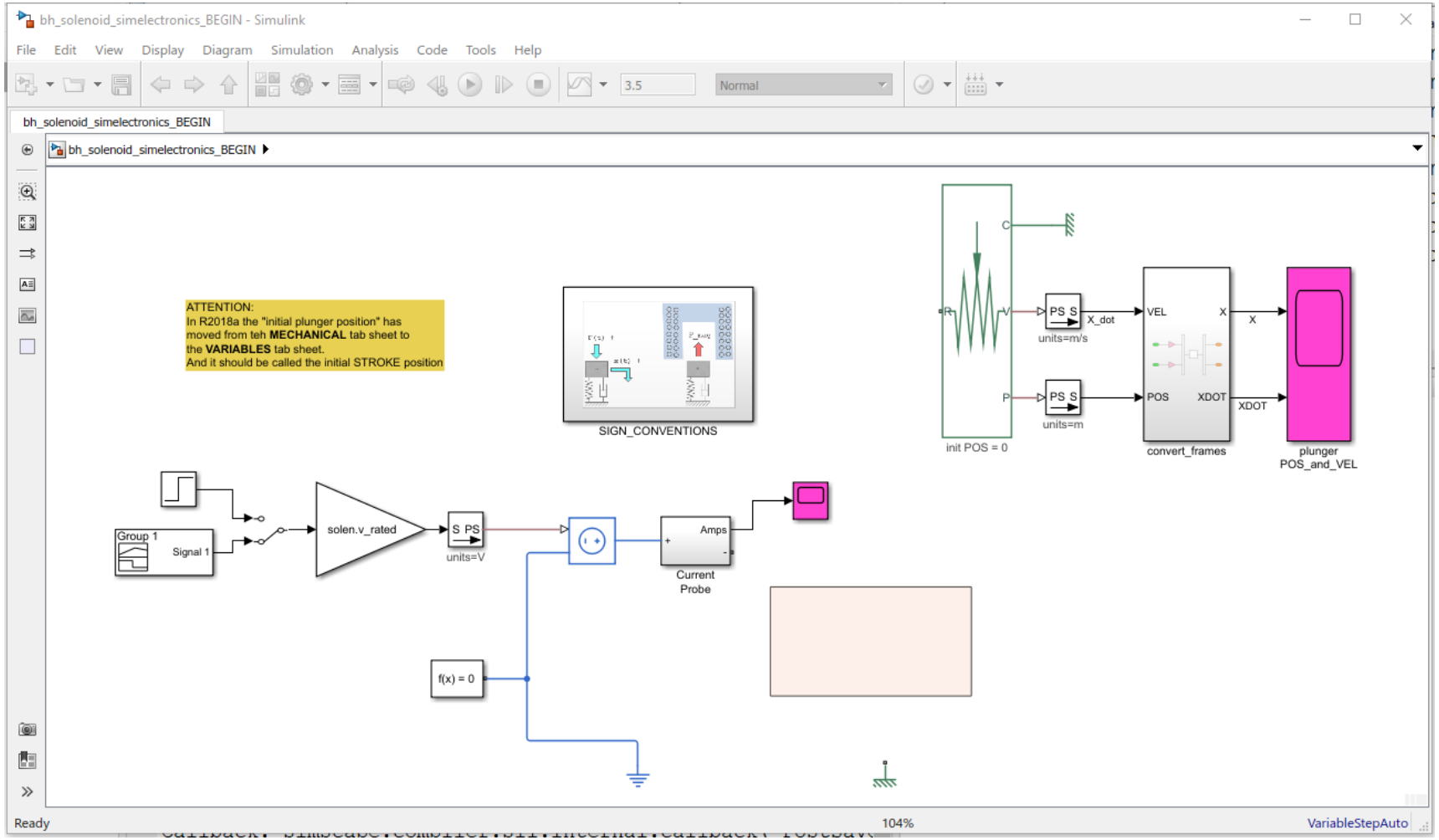
```
Trial>> solen

solen =

  struct with fields:

     v_rated: 7.6000
     i_rated: 4
    F_mag_12: [2×1 double]
      y_m_12: [2×1 double]
       F_mag: [13×1 double]
         y_m: [13×1 double]
```

```
Trial>> Coeffs

Coeffs =

  struct with fields:

    C123: [20.4094 67.0984 0.7548]
```

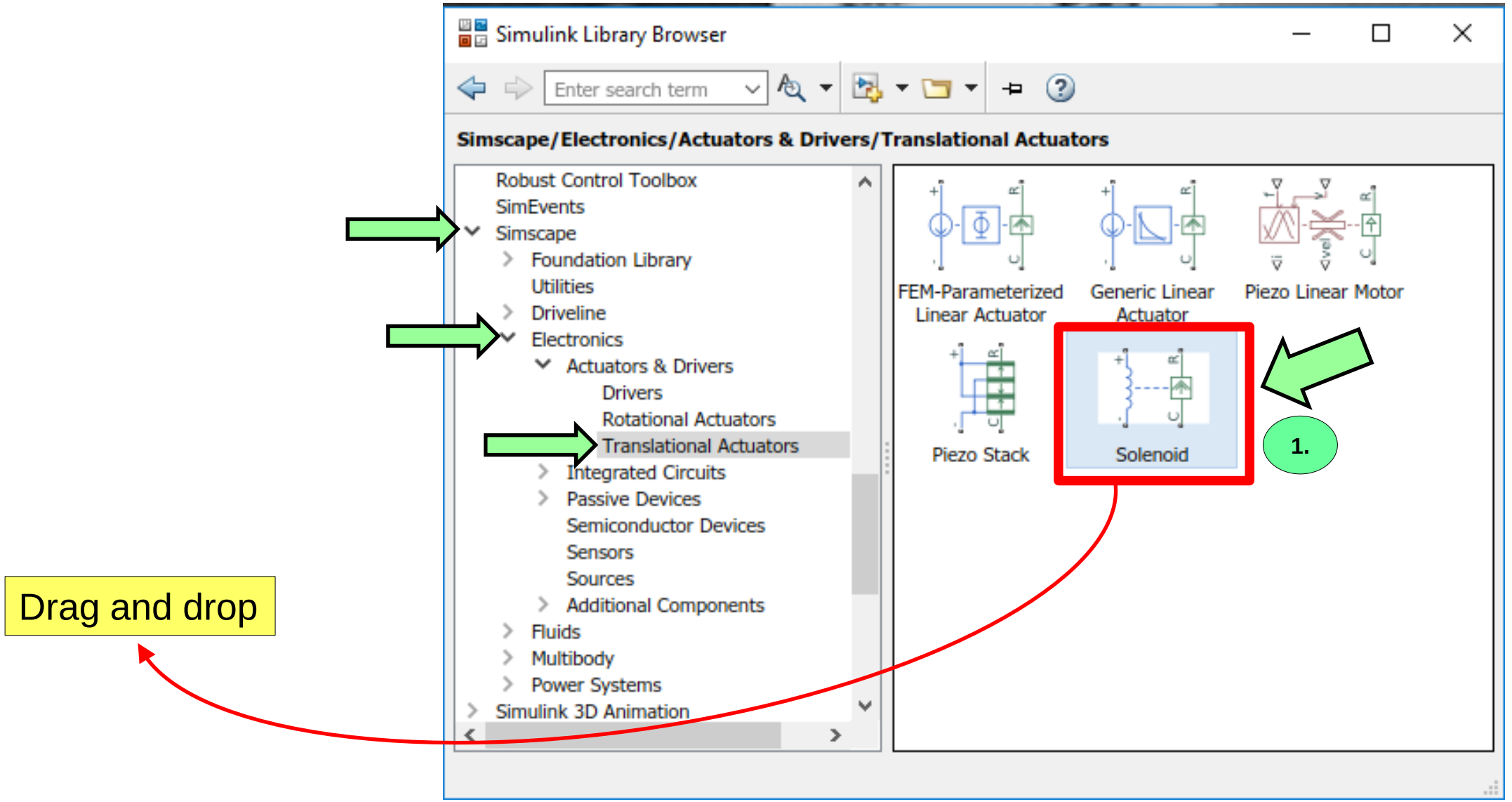
Simscape Electronics: **BEGIN**

We'll start with a model that looks like this

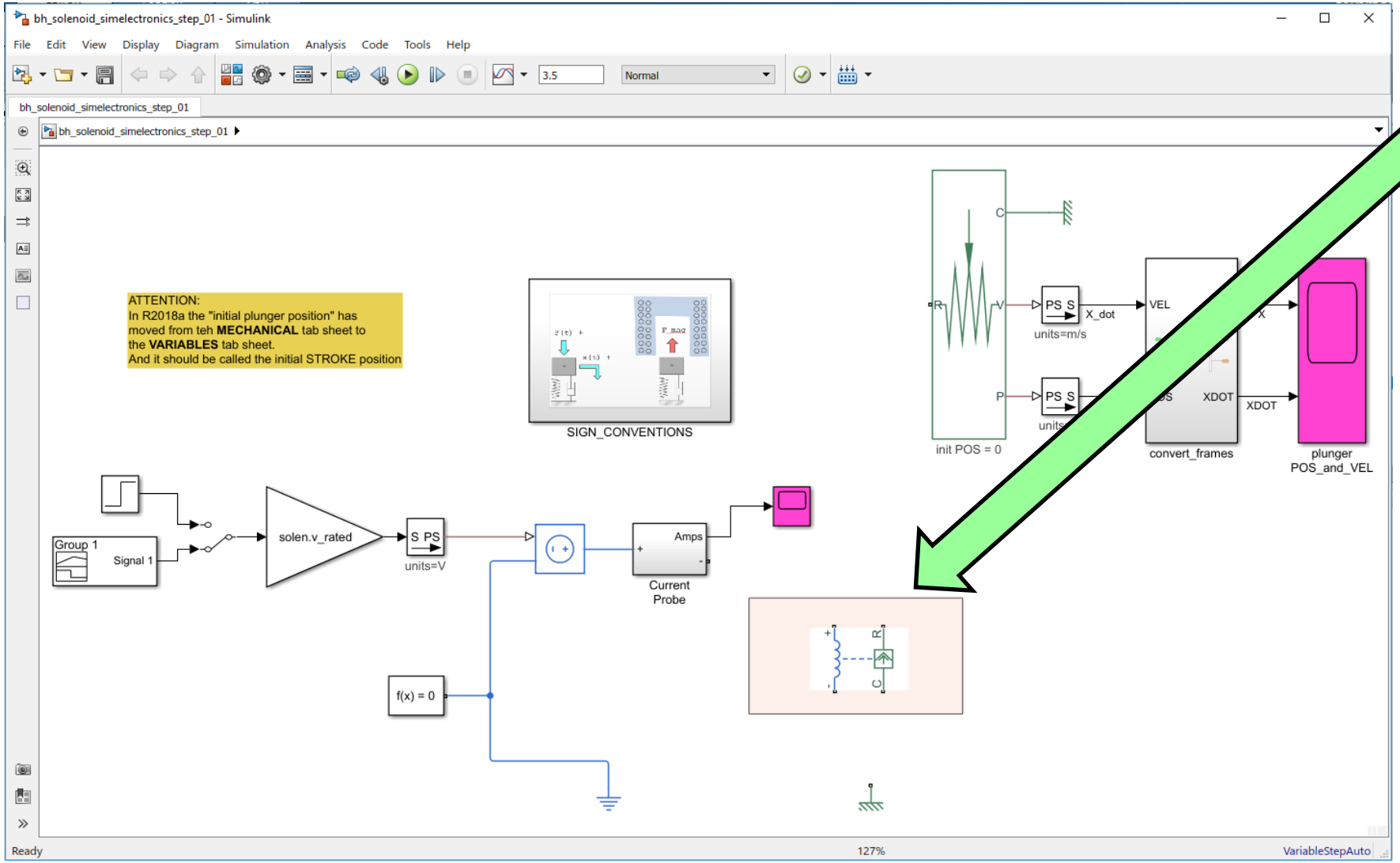




Simscape Electronics: **BEGIN**

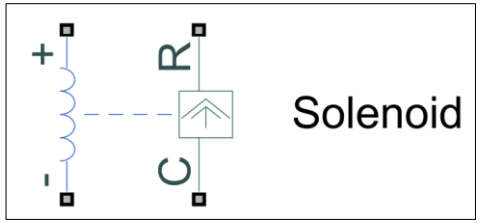


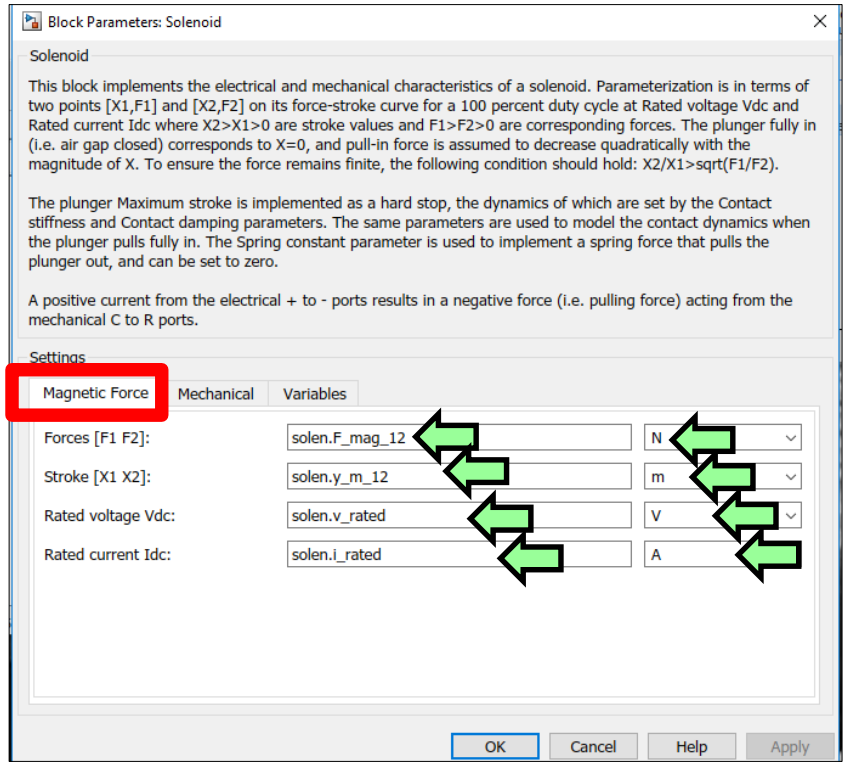
Simscape Electronics: **step_01**

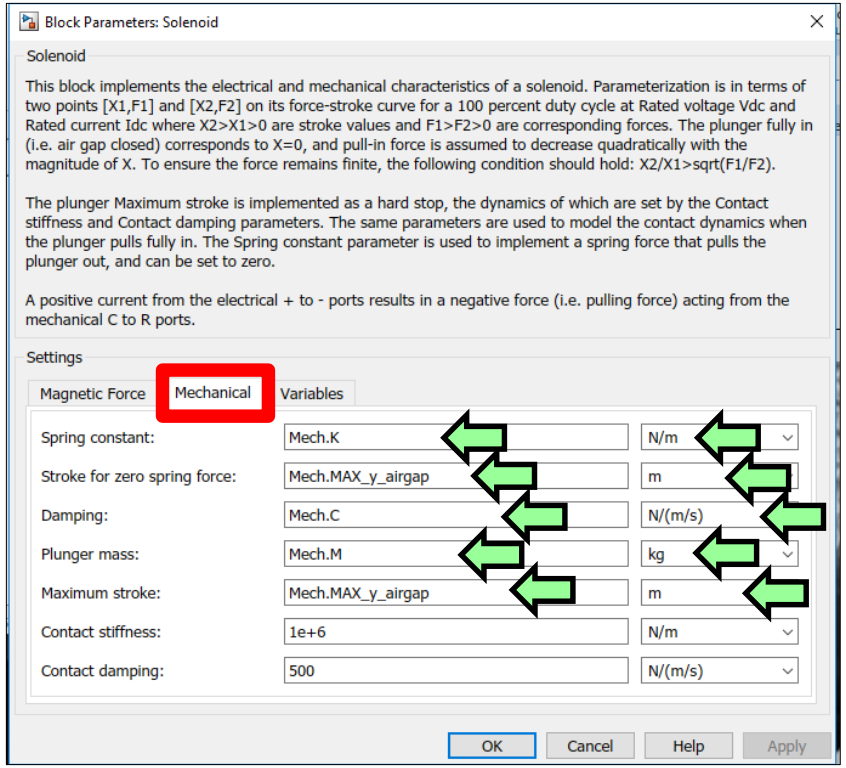


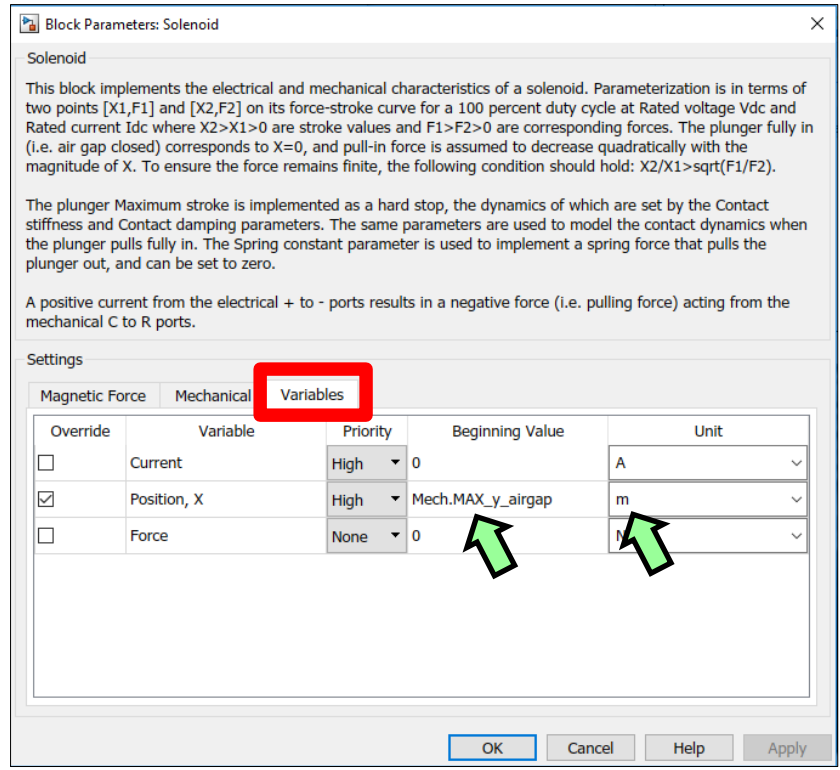
At the end of **STEP_01** your model should look like this

Simscape Electronics: **step_02**

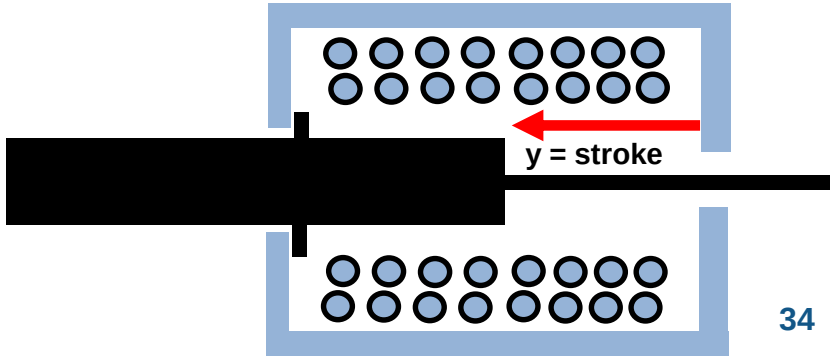




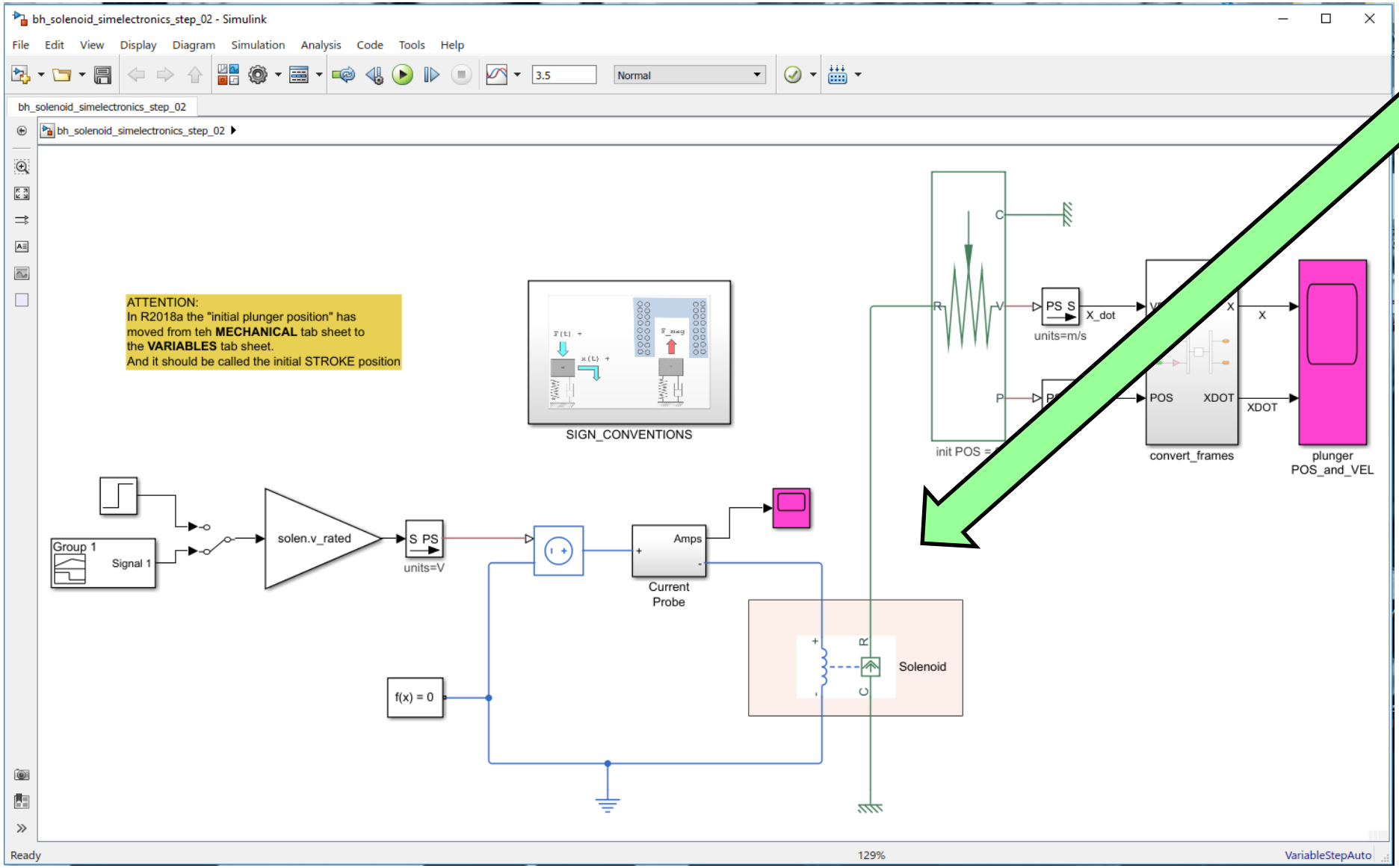




NOTE:
The parameter GUI(shown above) uses the symbol **X** to denote STROKE value …. While we have used the symbol **y** to denote STROKE value



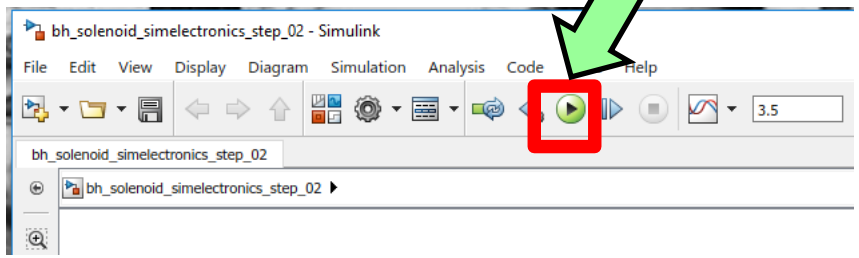
Simscape Electronics: **step_02**

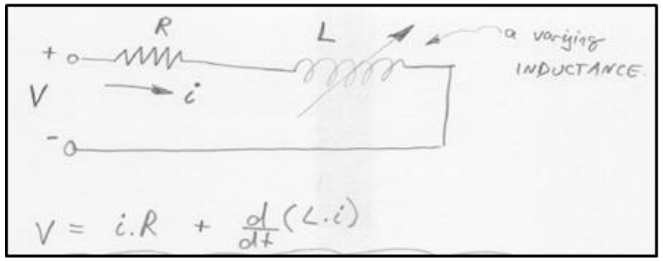


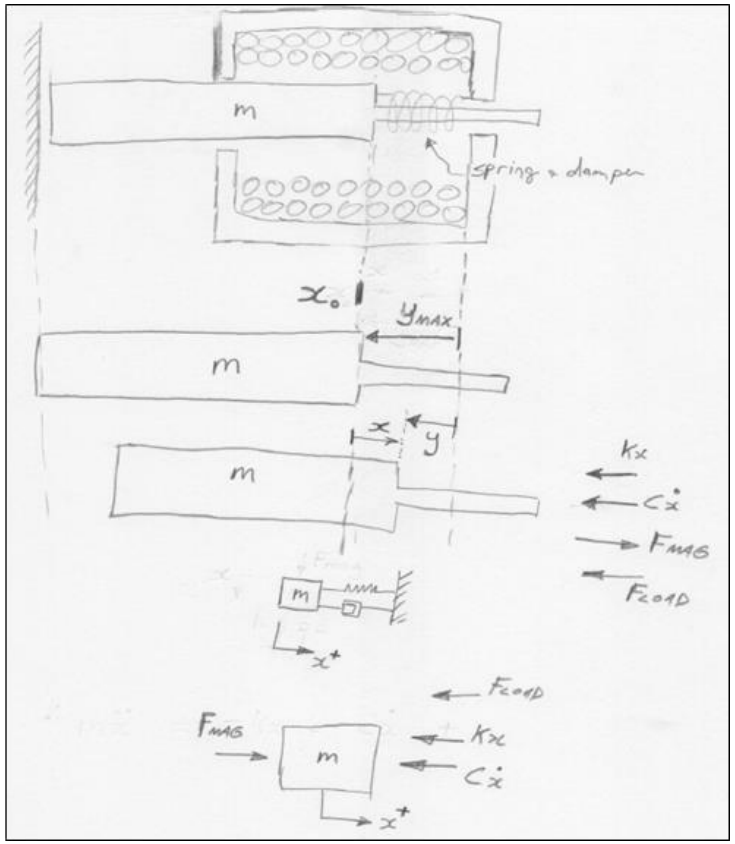
At the end of **STEP_02** your model should look like this

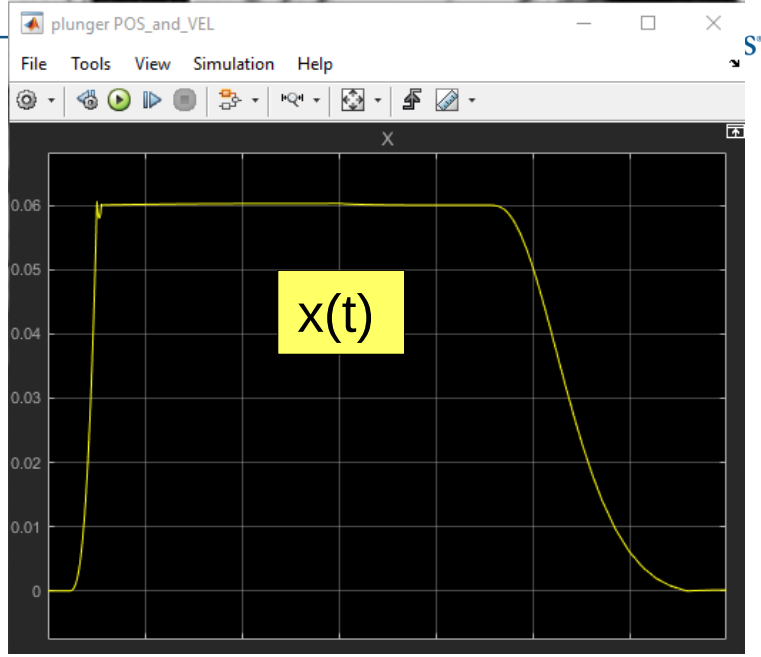
Simscape Electronics:

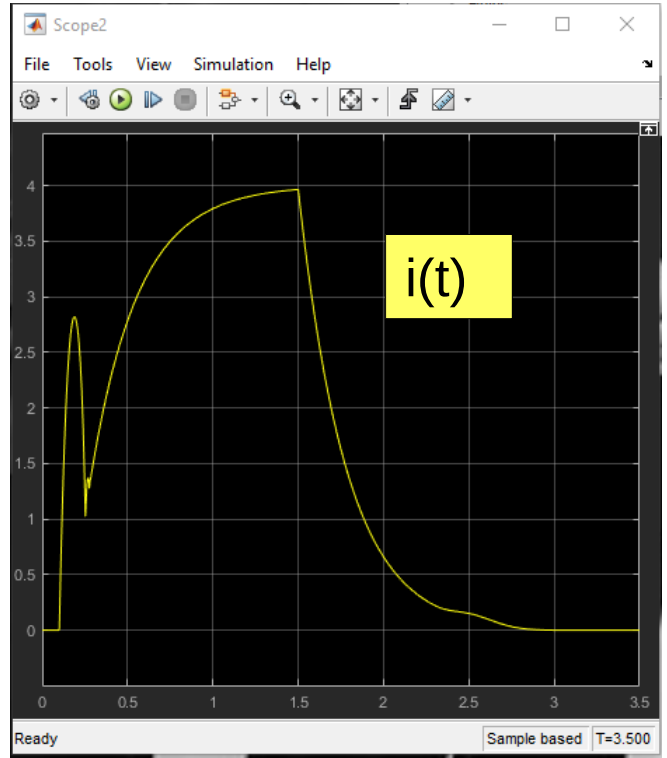
RUN your simulation



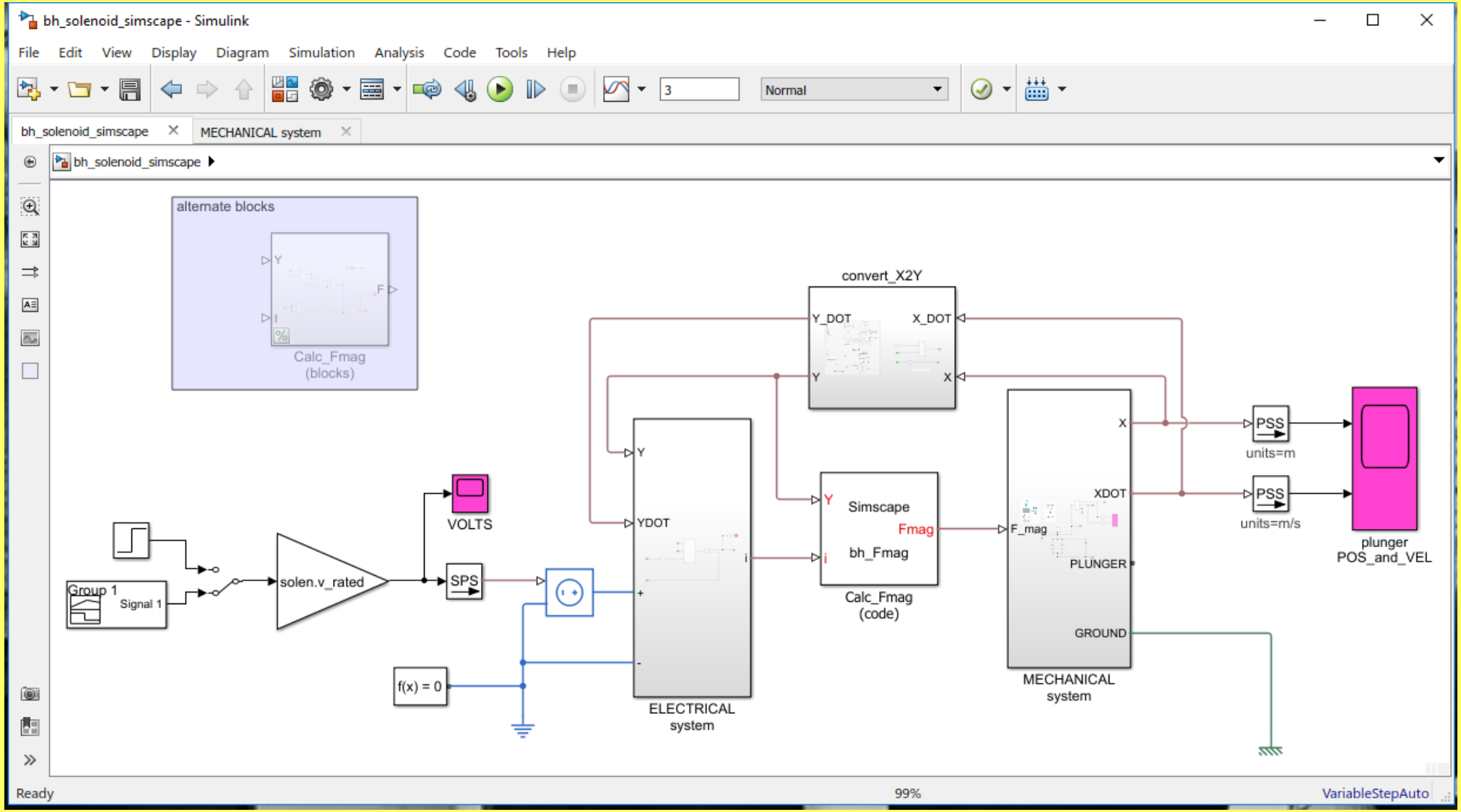




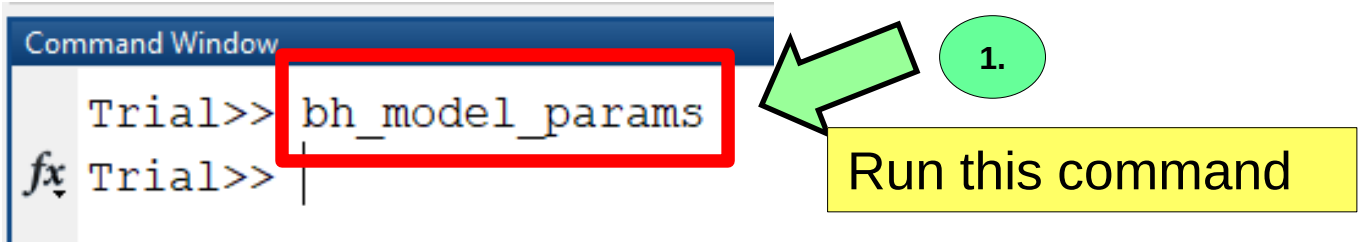




Model: Simscape



Load parameters into the MATLAB base workspace



2. … and now your workspace should contain these parameters

Workspace

Name	Value	Size	Class
Coeffs	*1x1 struct*	1x1	struct
Elec	*1x1 struct*	1x1	struct
Mech	*1x1 struct*	1x1	struct
my_sol_OBJ	*1x1 bh_solenoid*	1x1	bh_solenoid
solen	*1x1 struct*	1x1	struct
tmp_F_mag	*13x1 double*	13x1	double
tmp_y_m	*13x1 double*	13x1	double

```
Trial>> Elec

Elec =

  struct with fields:

    R: 1.9000
```

```
Trial>> Mech

Mech =

  struct with fields:

                K: 17
                C: 5.3229
                M: 0.6000
              x_0: 0
           xdot_0: 0
            x_min: 0
            x_max: 0.0600
    MAX_y_airgap: 0.0600
```

```
Trial>> solen

solen =

  struct with fields:

     v_rated: 7.6000
     i_rated: 4
    F_mag_12: [2×1 double]
      y_m_12: [2×1 double]
       F_mag: [13×1 double]
         y_m: [13×1 double]
```

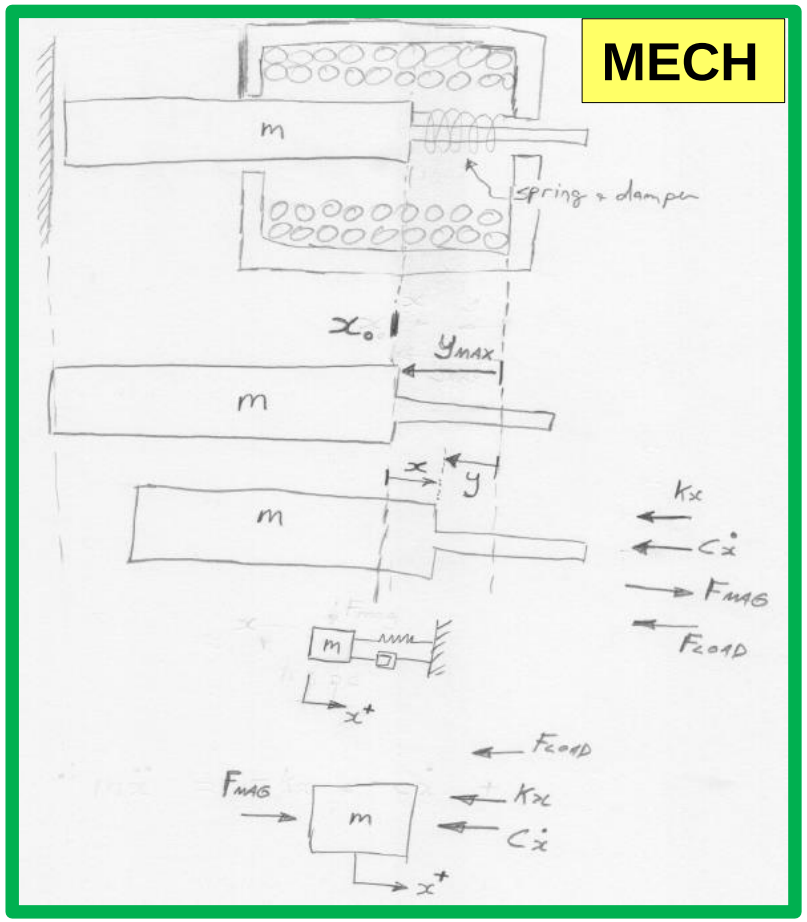
```
Trial>> Coeffs

Coeffs =

  struct with fields:

    C123: [20.4094 67.0984 0.7548]
```

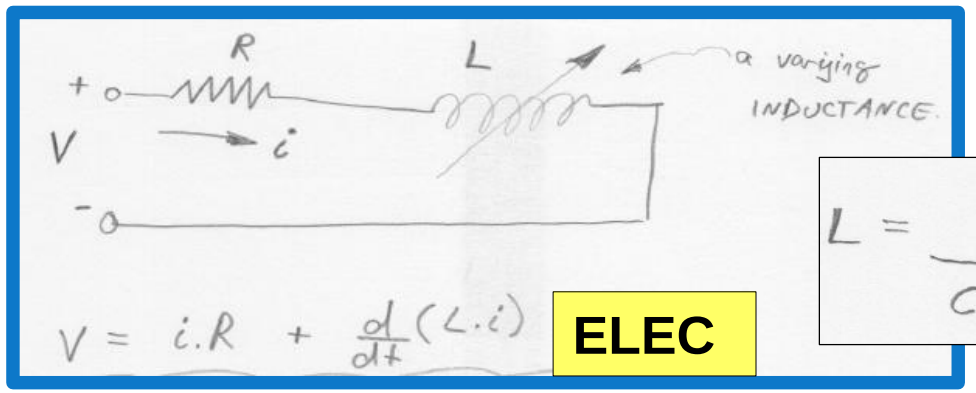
Simscape: modelling approach

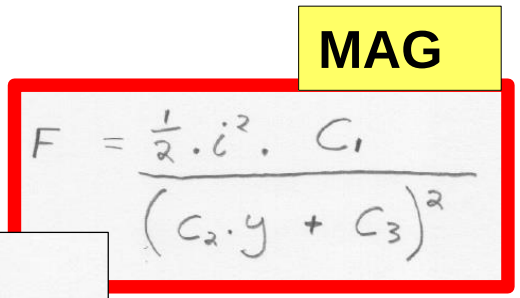
To model this complete system we'll divide and conquer

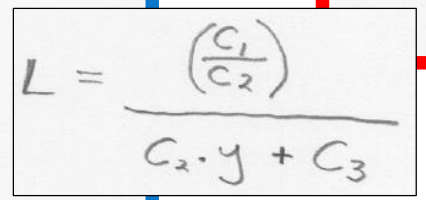
- **MECHANICAL** system
  - We'll develop the mechanical system and test it
- **ELECTRICAL** system
  - We'll develop the electrical system and test it

We'll then join these 2 main systems together along with the **MAGNETIC** system

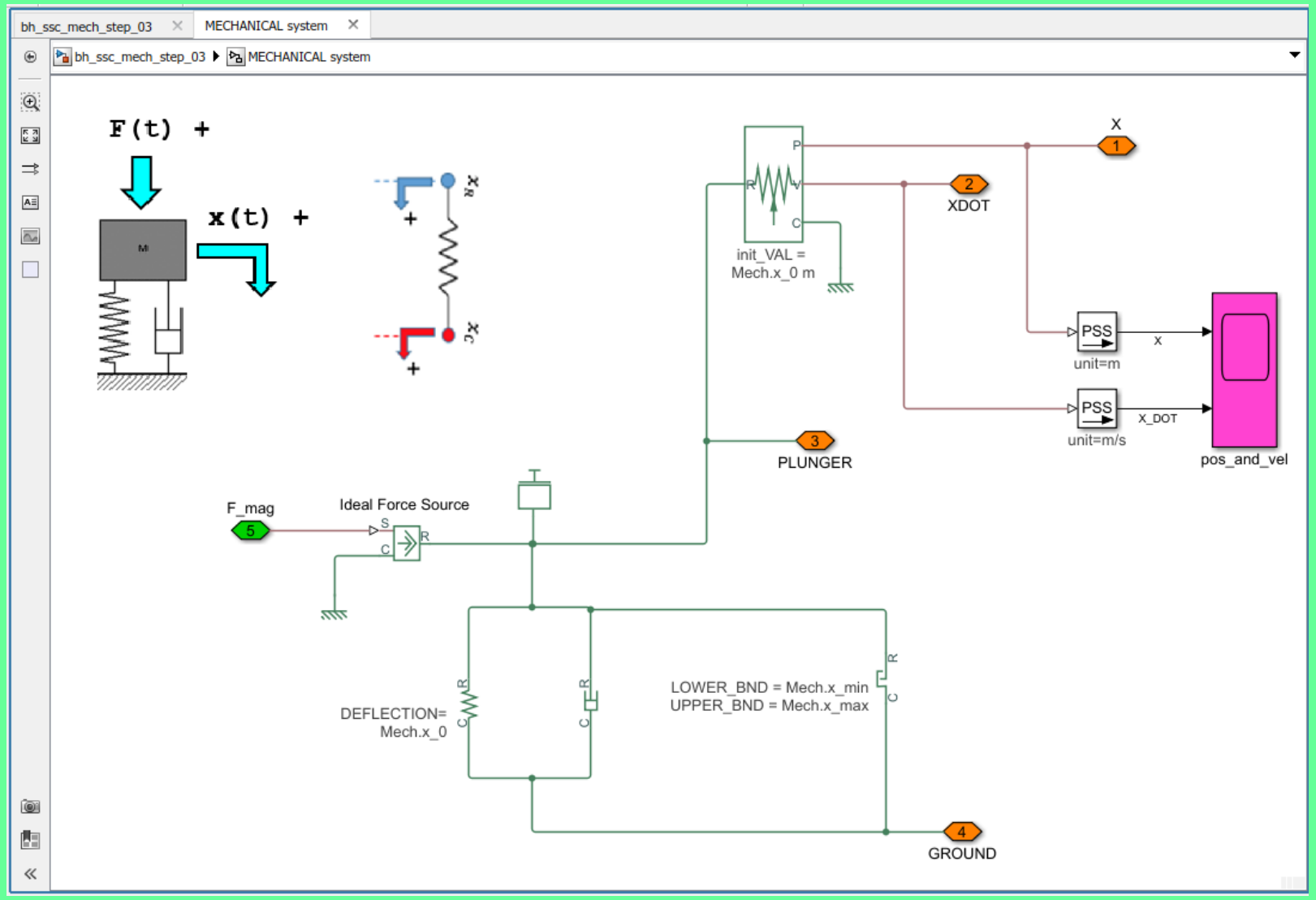




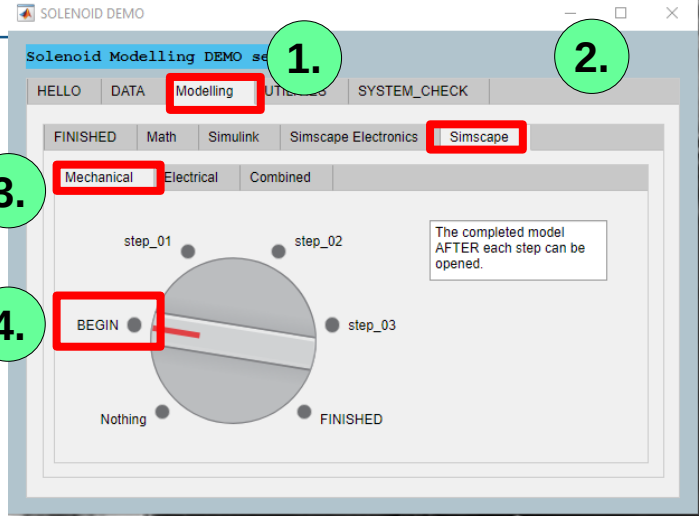


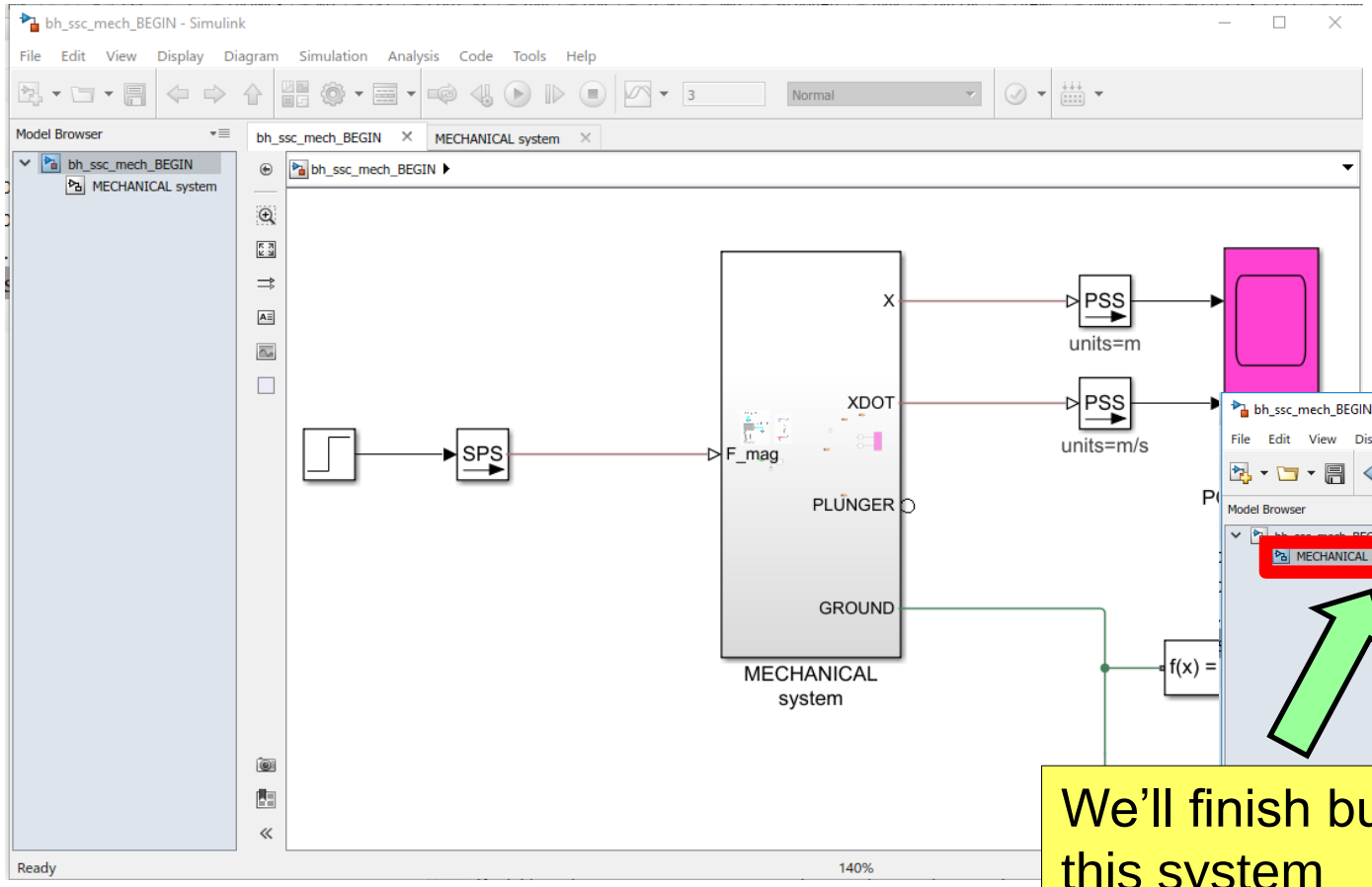


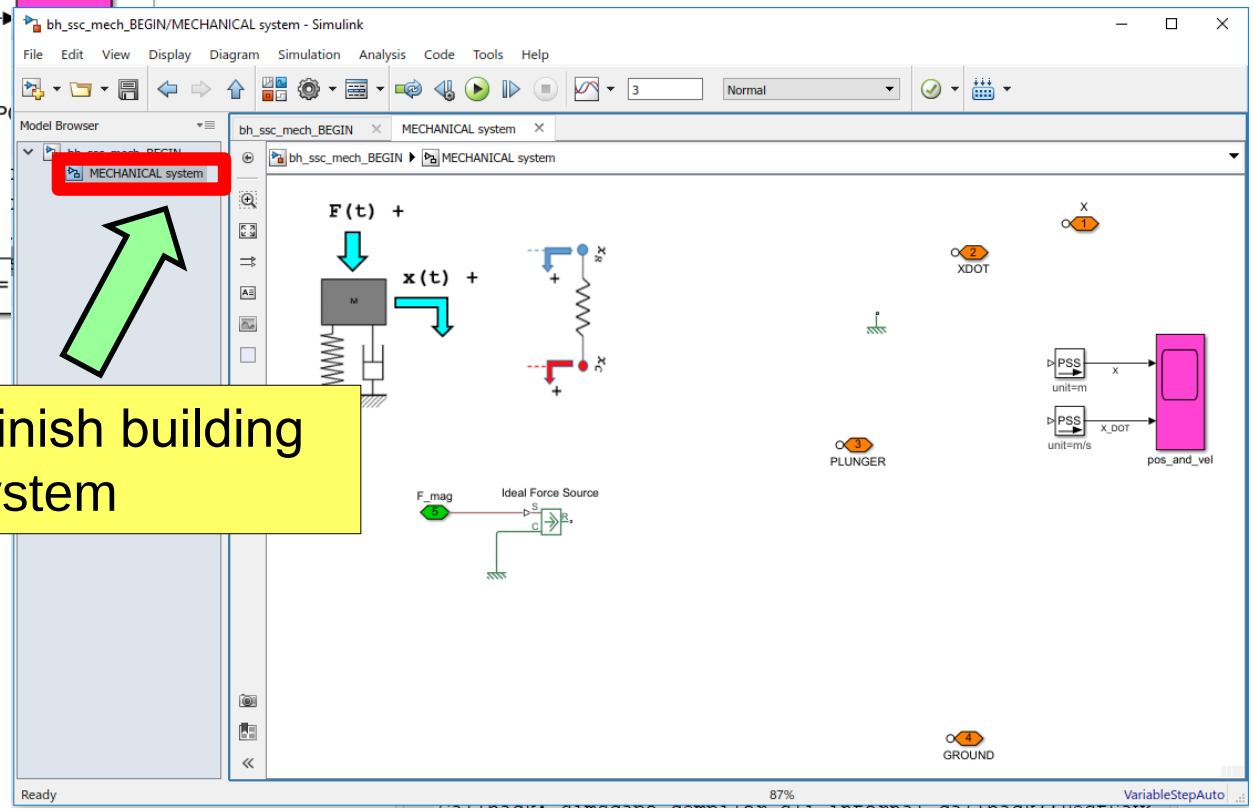
Let's start building the **MECHANICAL** system



MECHANICAL: **BEGIN**

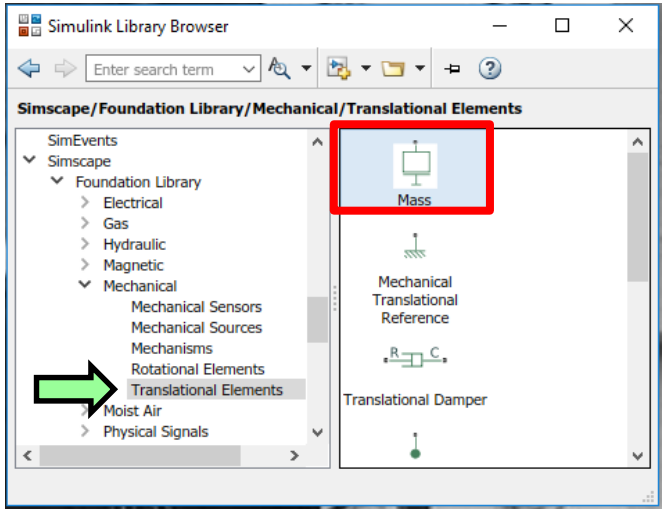


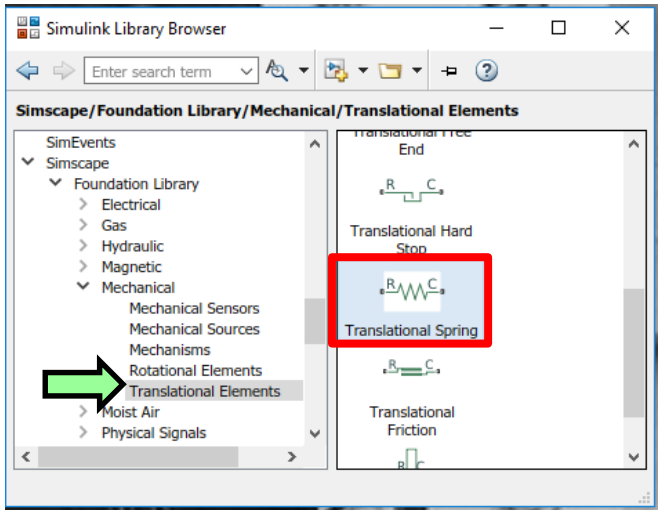


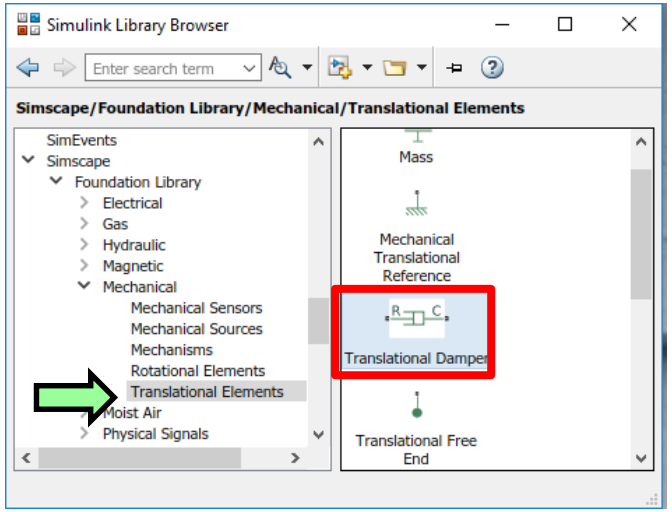


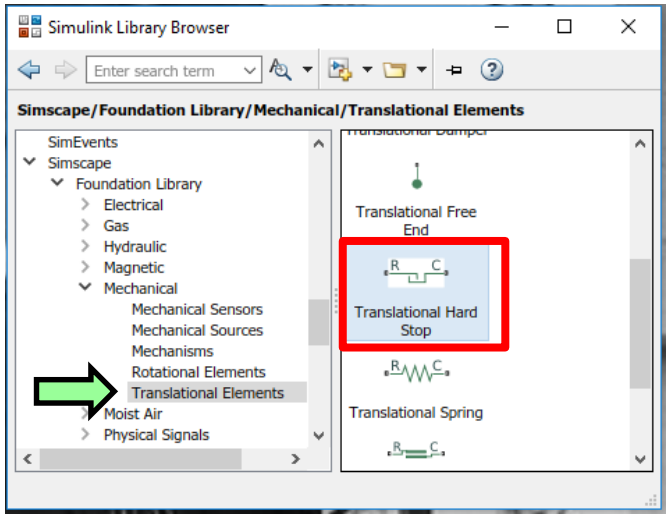
We'll finish building this system

MECHANICAL: **BEGIN**

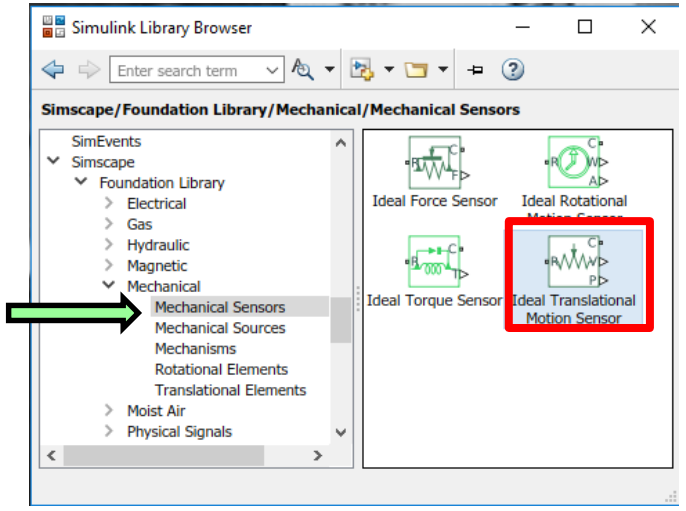




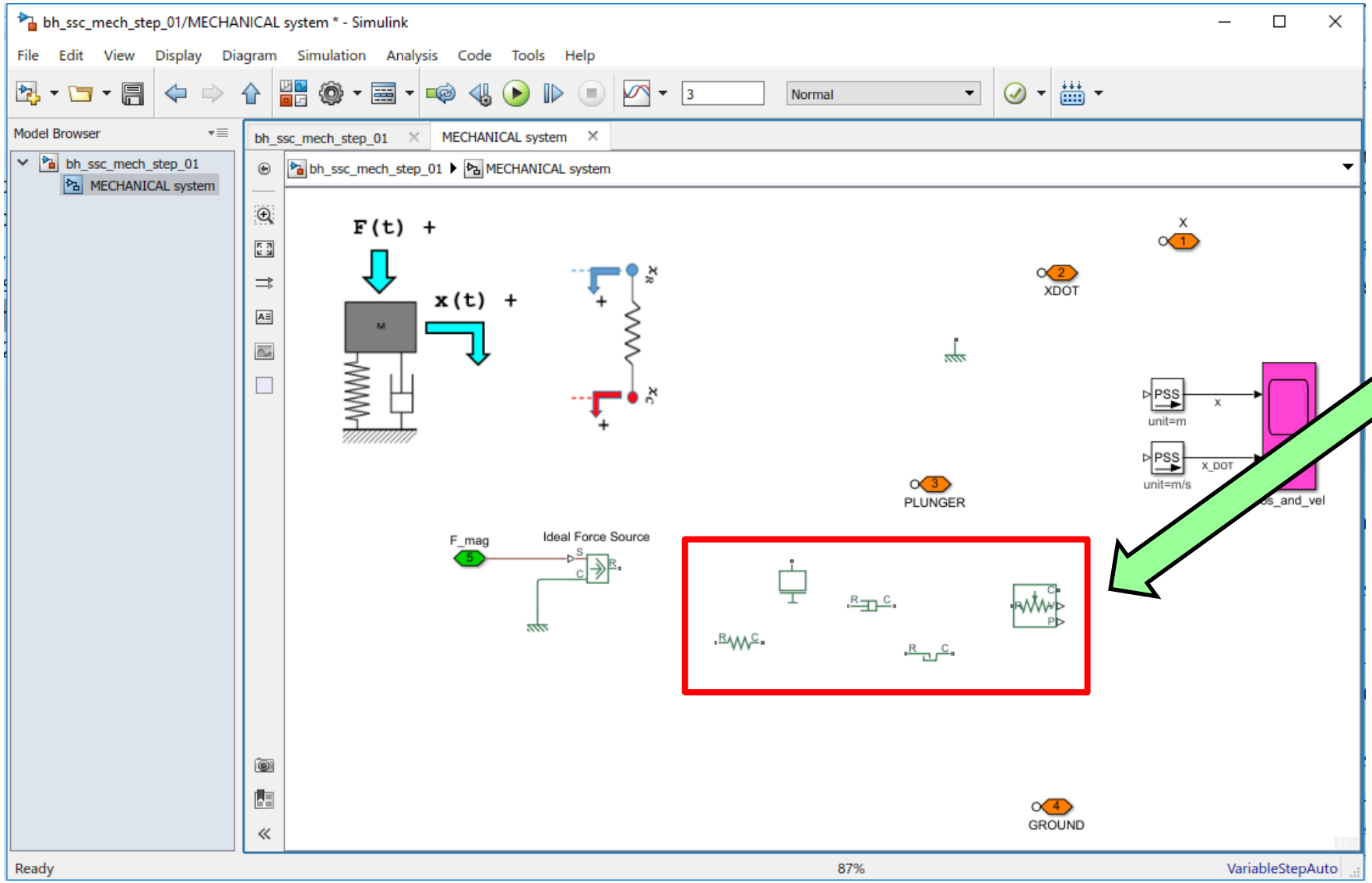




1 of each



MECHANICAL: **step_01**



So at the end of **step_01** your model should look like this

TIP: rotating blocks

1.

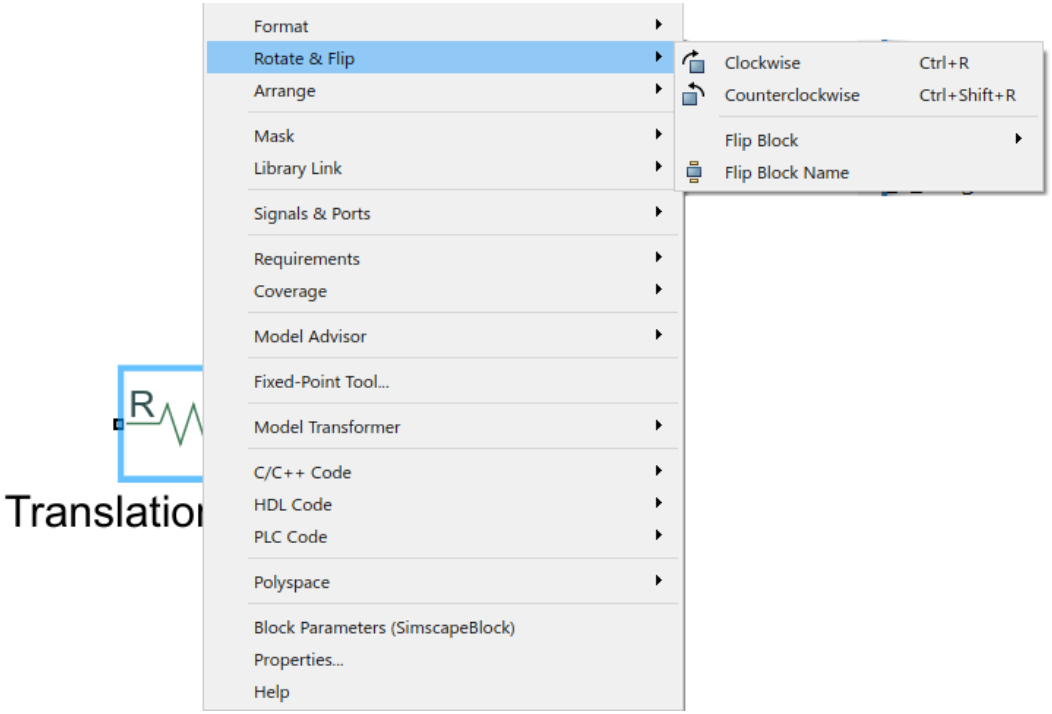
Select the block

2.

Ctrl+R

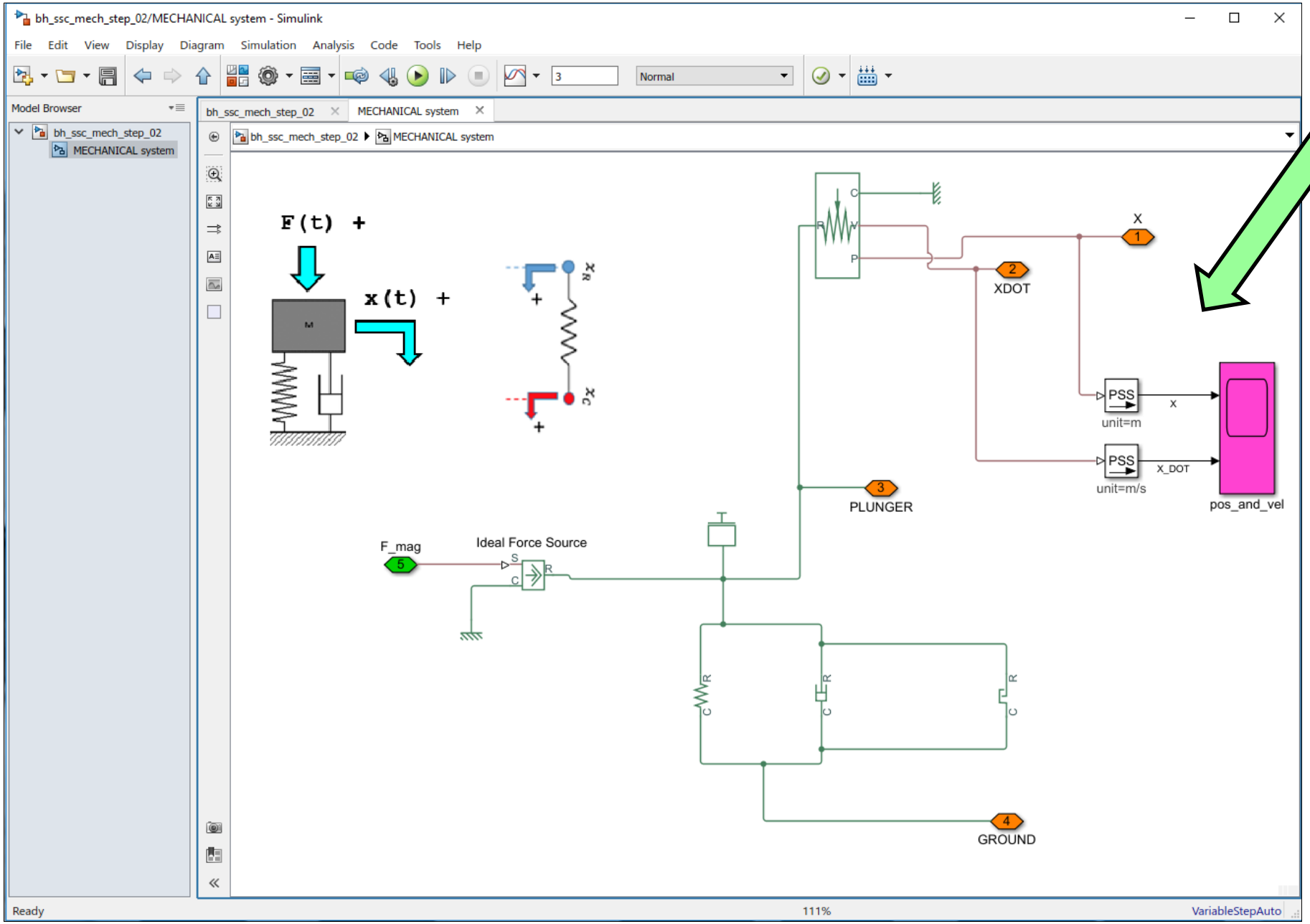
And you get this

Alternately, right click on the block and bring up a context menu

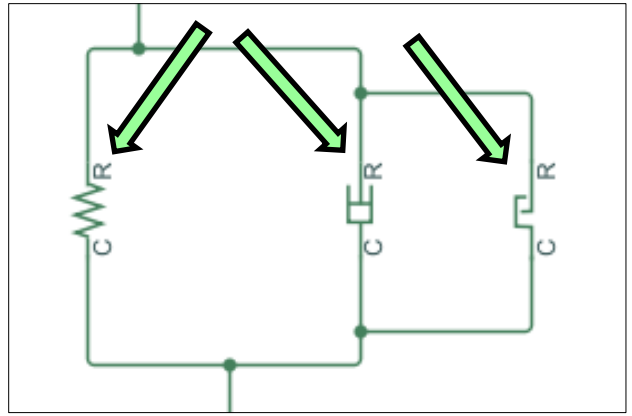


MECHANICAL: **step_02**

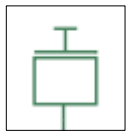
So at the end of **step_02** your model should look like this

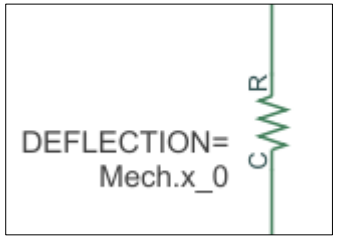


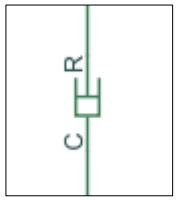
Make sure these 3 blocks have the **R-C** orientation as shown

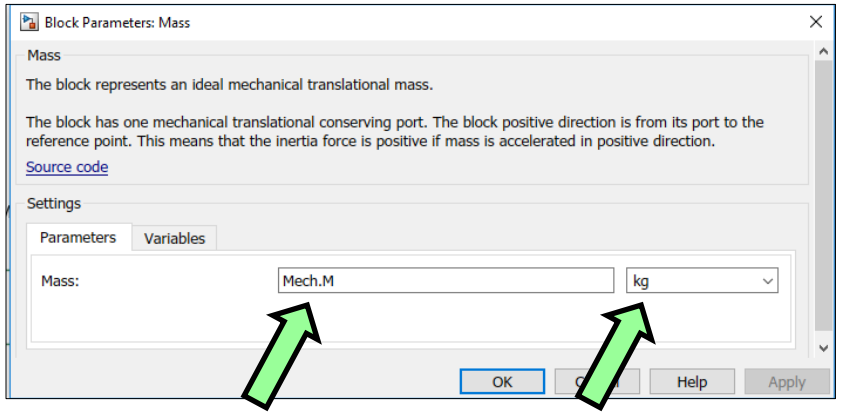


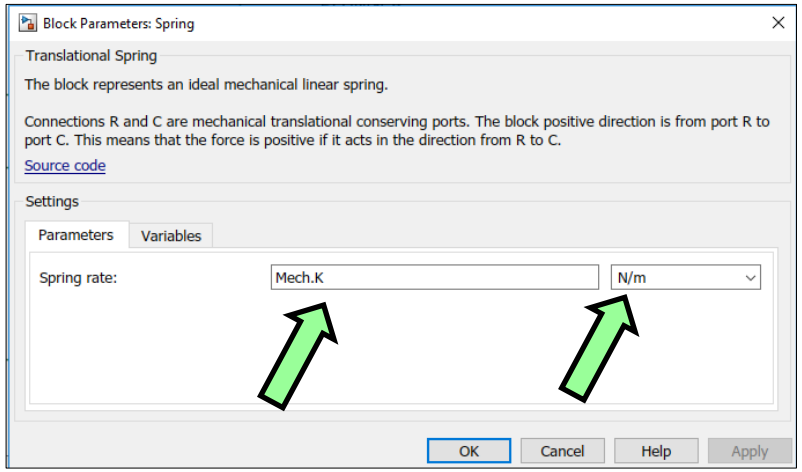
MECHANICAL: **step_03**

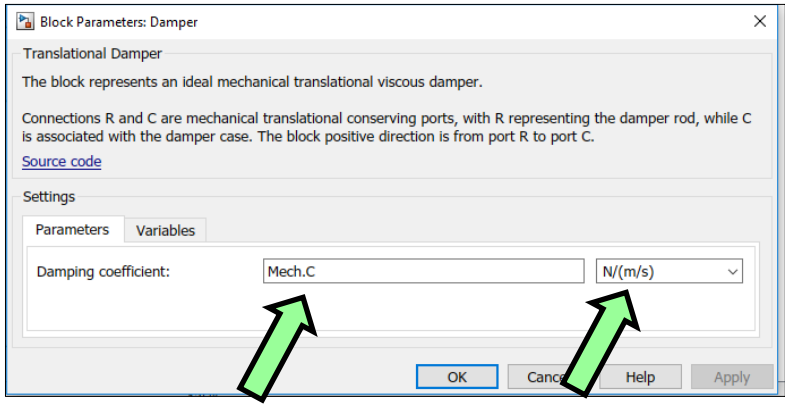


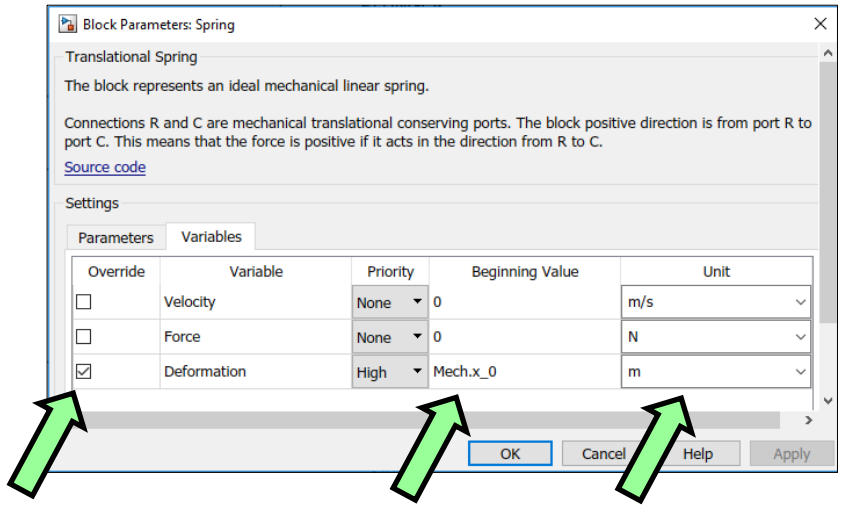




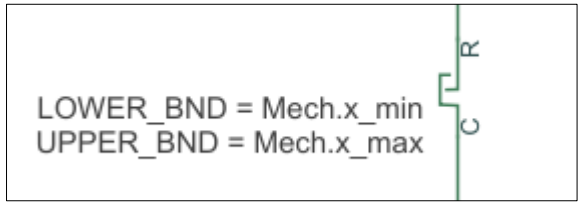


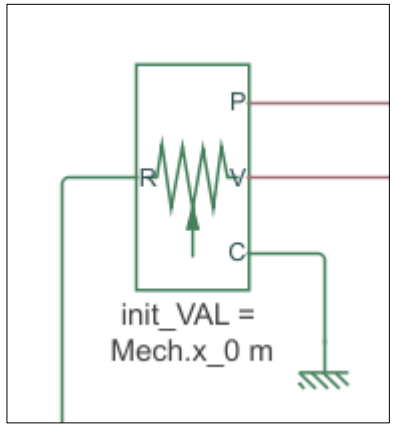


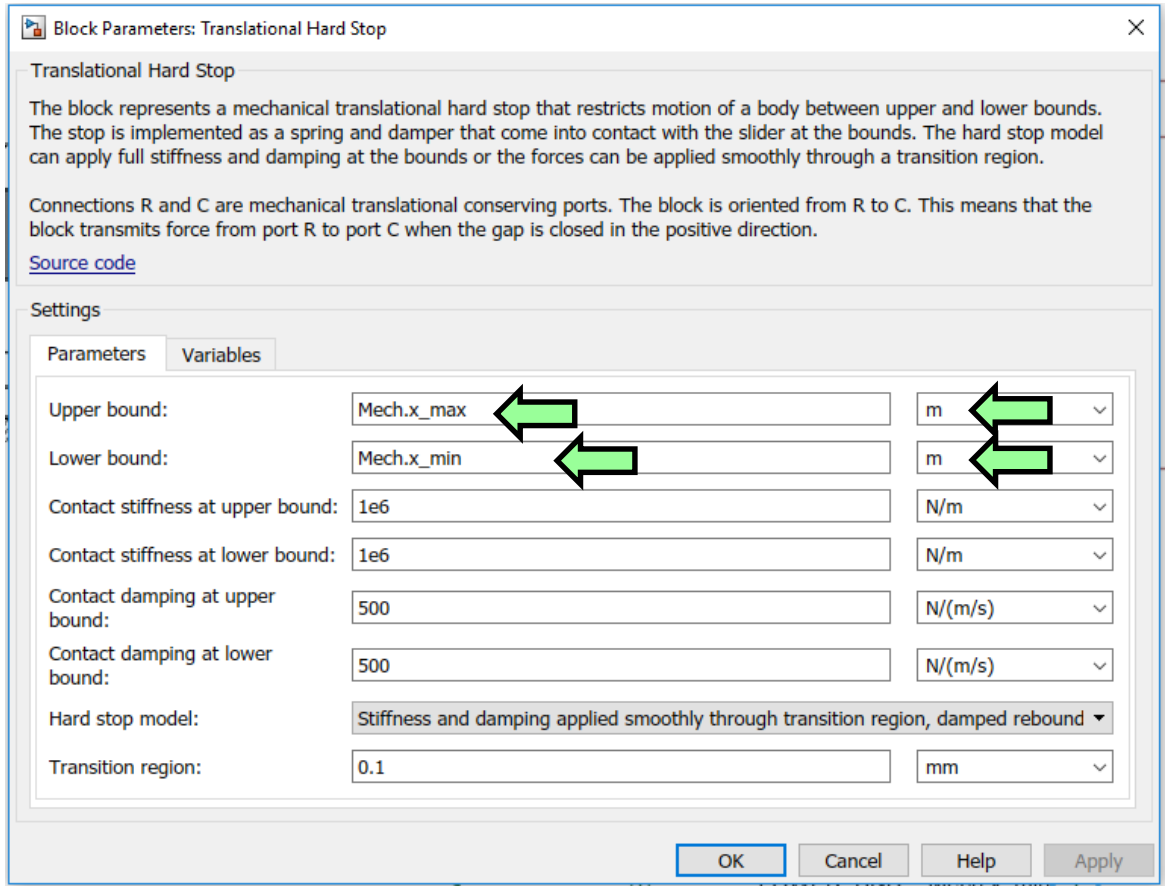


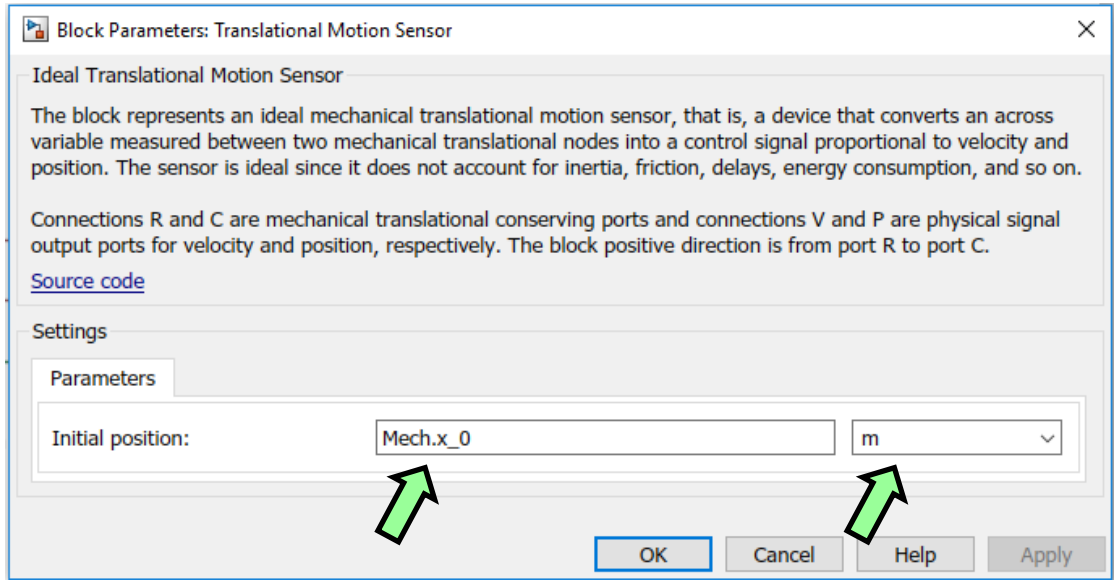


MECHANICAL: **step_03**



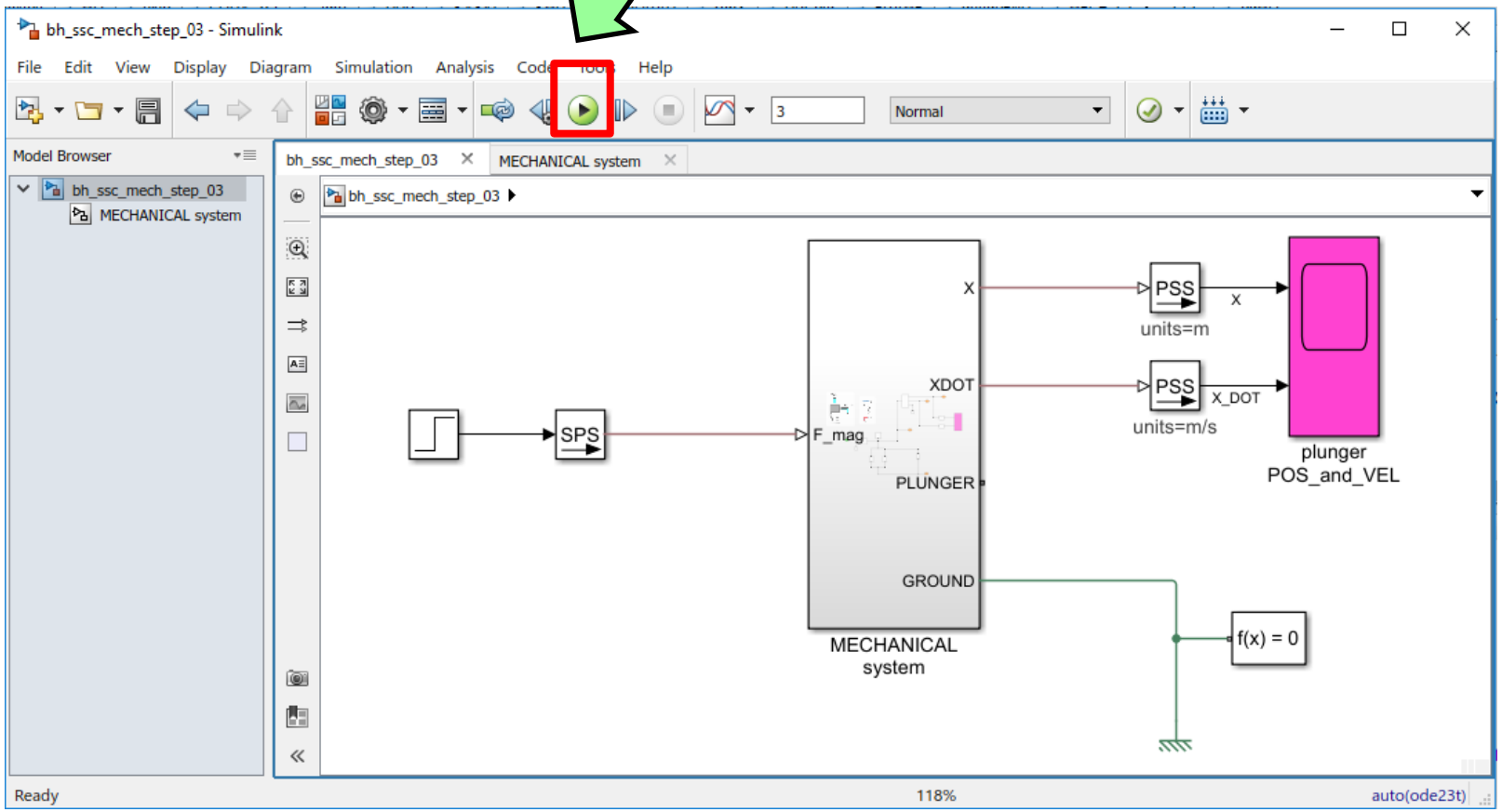


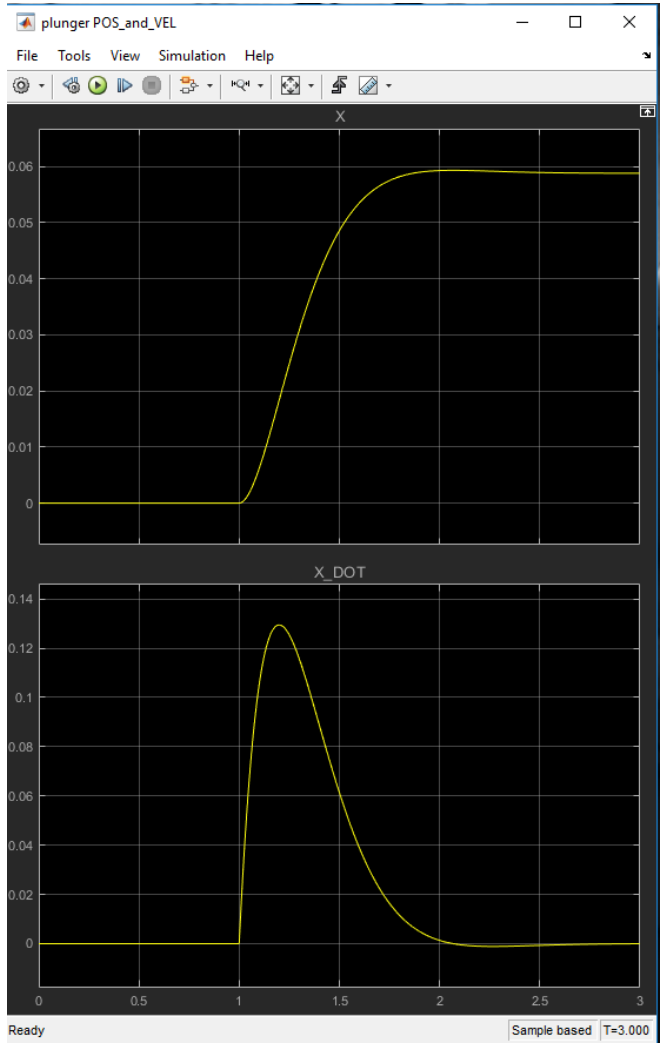




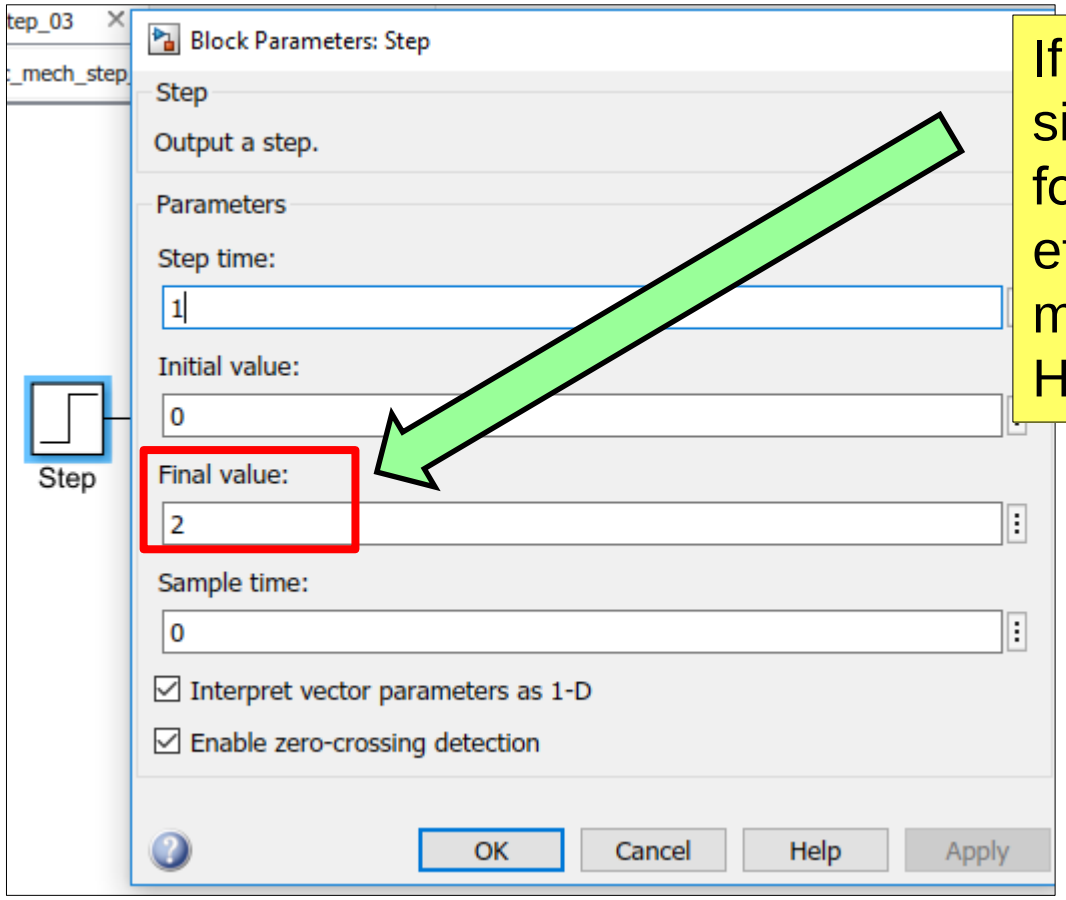
MECHANICAL: **step_03**

Run your simulation

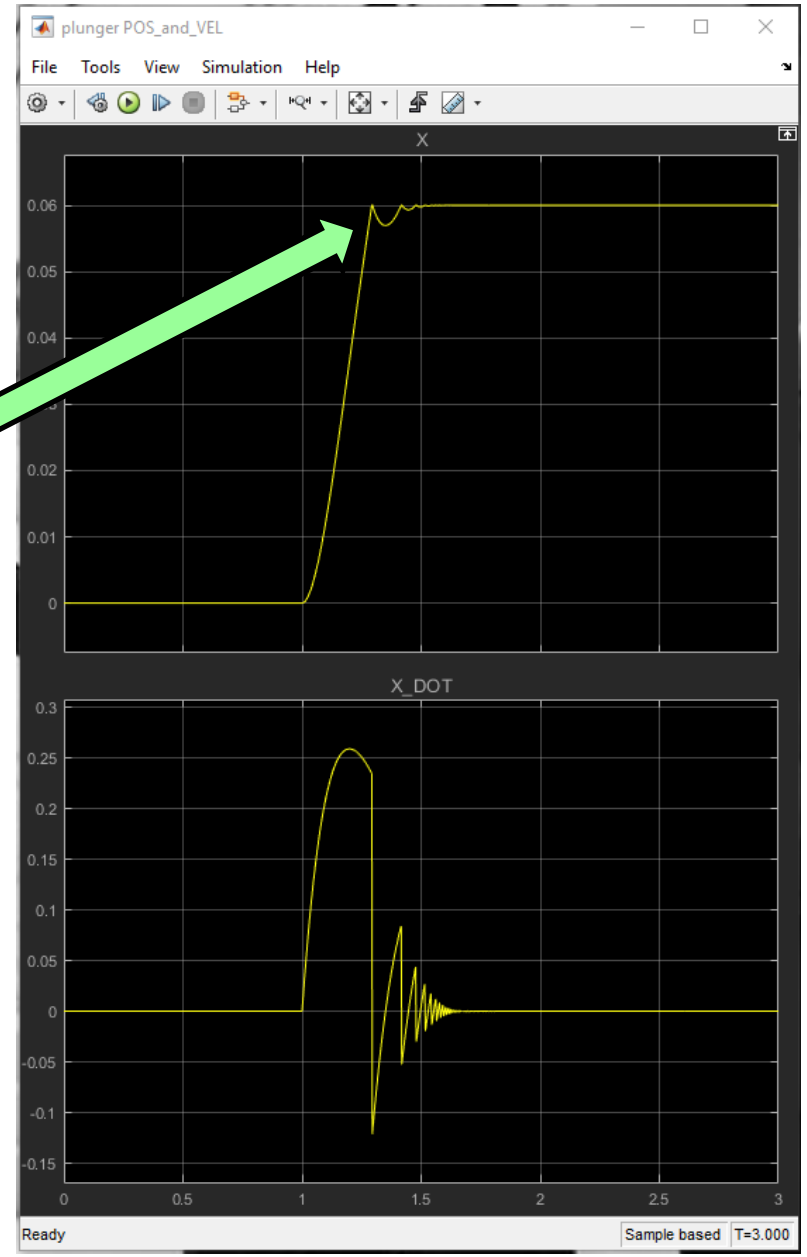




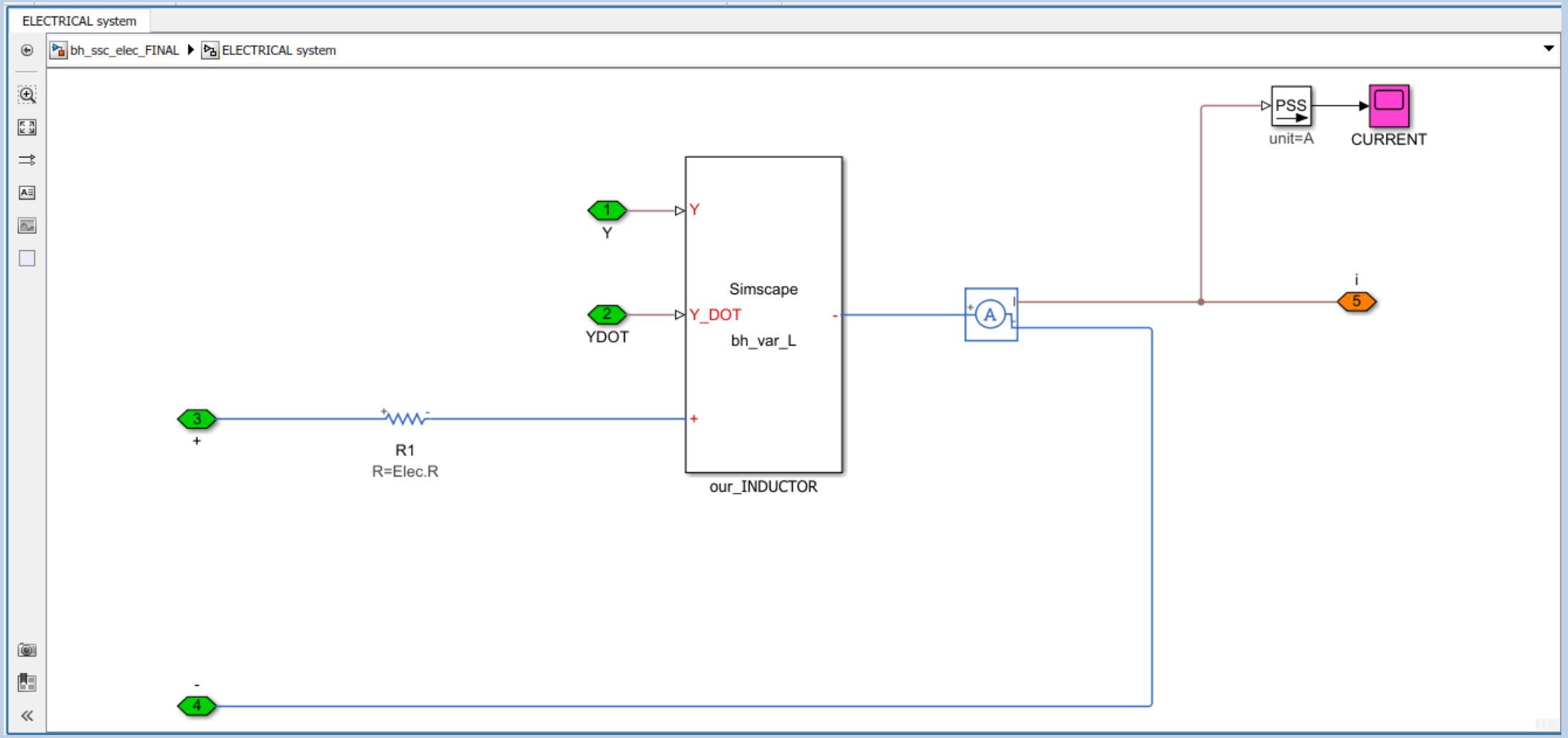
MECHANICAL: **step_03**



If we increase the size of our excitation force …. We see the effect of our mechanical HARDSTOPS

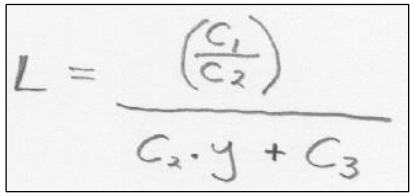


Let's start building the **ELECTRICAL** system



Before we start:

- The inductance has been coded into a simscape file called **bh_var_L.ssc**



$$V_L = L.\frac{di}{dt} + i.\frac{dL}{dy}.\frac{dy}{dt}$$

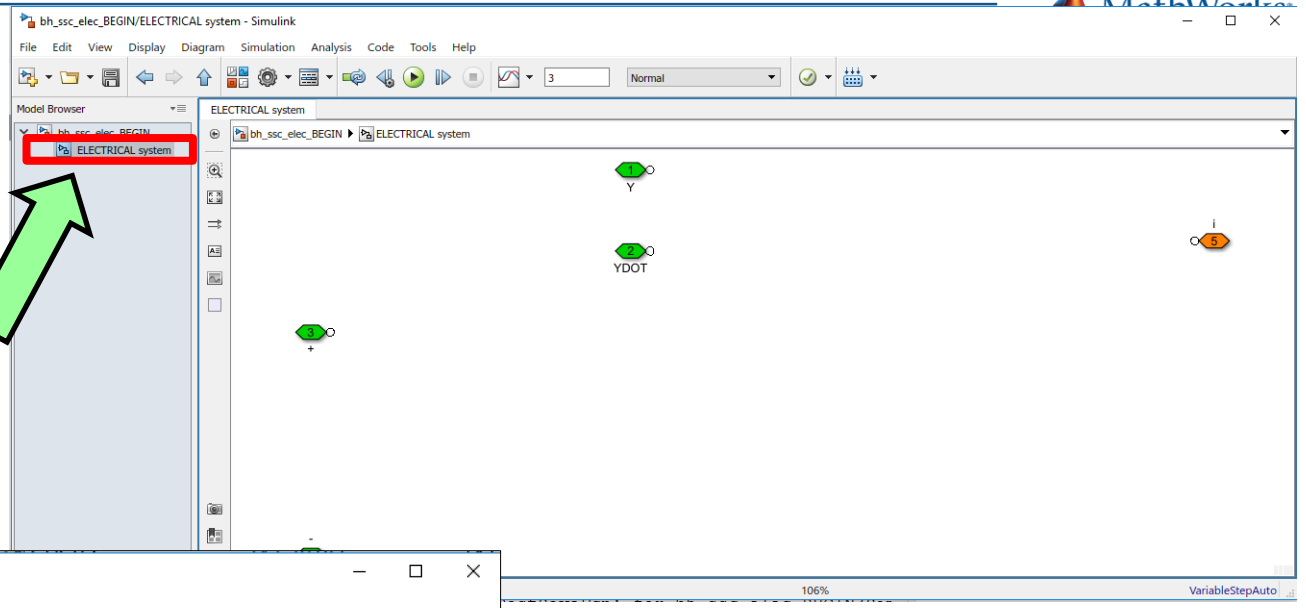
Editor - C:\bh\users_EV\Workshops\CLIENT_SESSIONS\MQ_solenoid\THE_LAB\THE_LIBRARY\bh_var_L.ssc

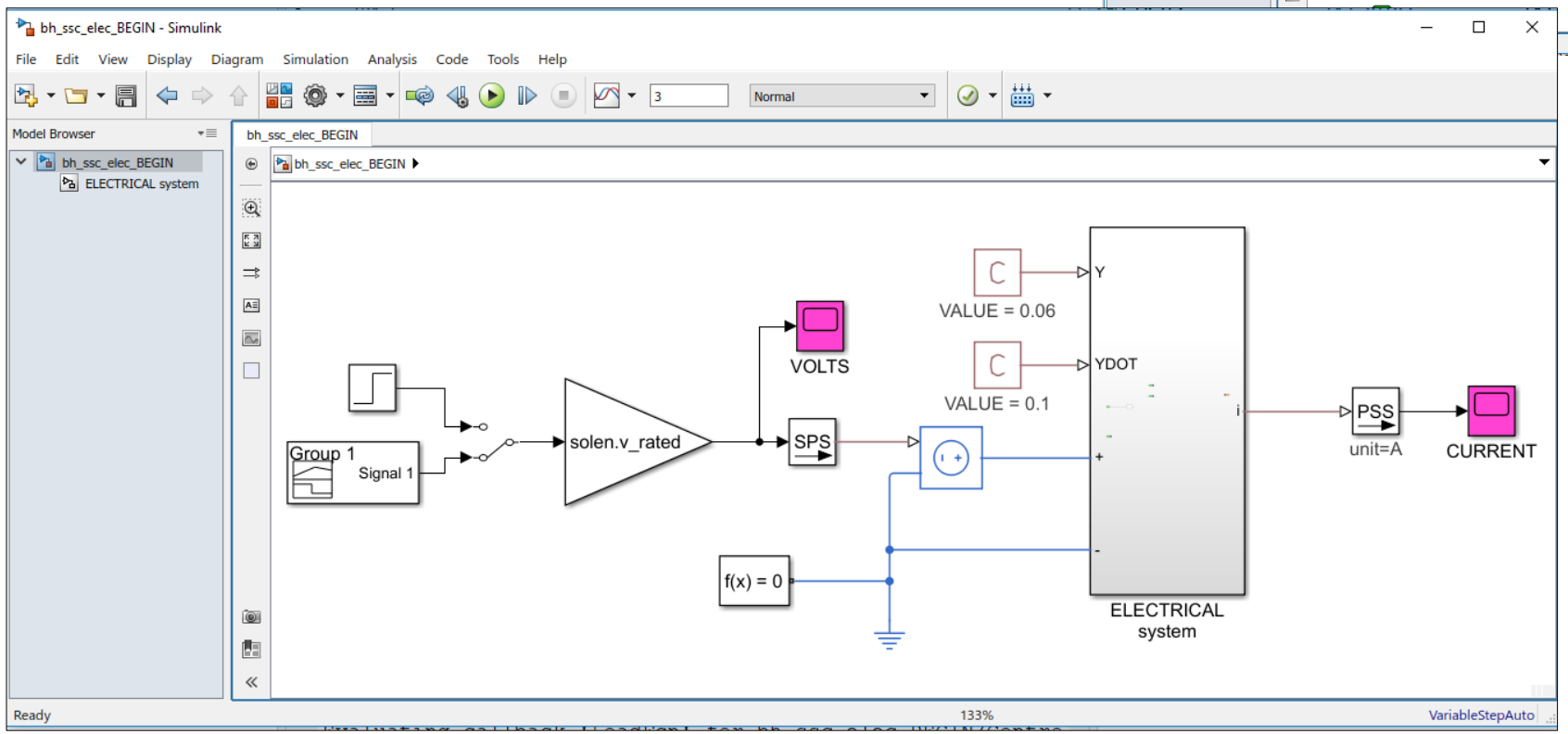
bh_model_params.m | bh_var_L.ssc

```
component bh_var_L

inputs
    y    = { 1.0, 'm' };  % Y:left
    ydot = { 1.0, 'm/s'}; % Y_DOT:left
end

nodes
    p = foundation.electrical.electrical; % +:left
    n = foundation.electrical.electrical; % -:right
end

parameters
    C1 = {1,'H*(m^5)'};
    C2 = {1,'m^2'};
    C3 = {1,'m^3'};
end

variables
    i = { 0, 'A' }; % Current
    v = { 0, 'V' }; % Voltage
end

branches
    i : p.i -> n.i;
end

equations
    let
       L    =           (C1/C2) /   (C2*y + C3)    ;
       dLdy = -1 * C2 * (C1/C2) / ( (C2*y + C3)^2 );
    in
       v == L*i.der  +  i*dLdy*ydot;
       v == p.v - n.v;
    end
end

end
```

Electrical: **BEGIN**

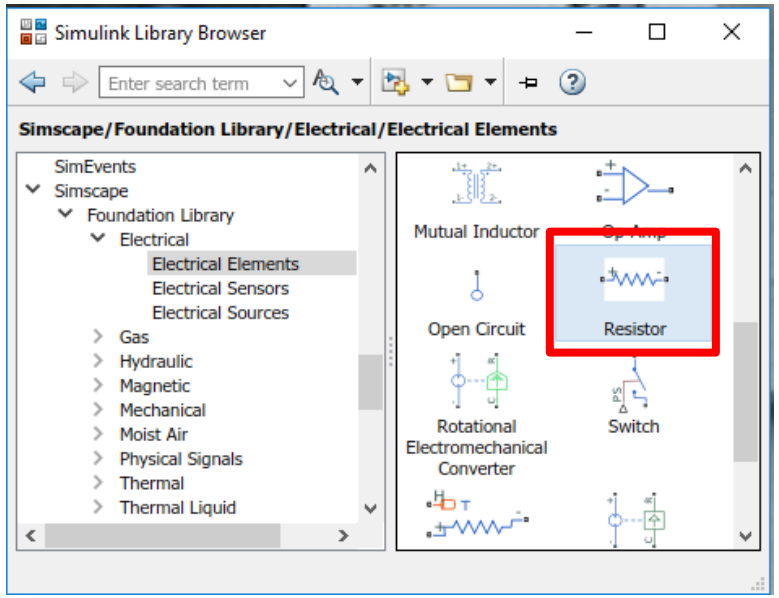
We'll finish building this system

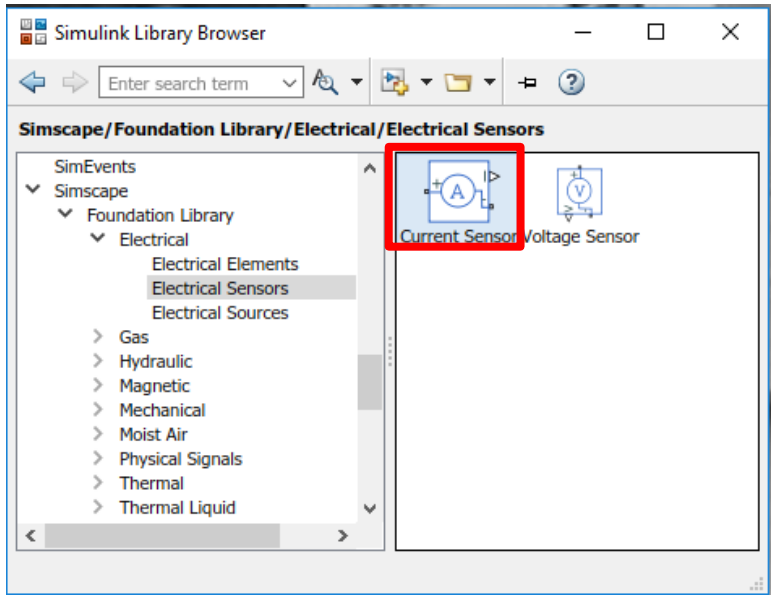


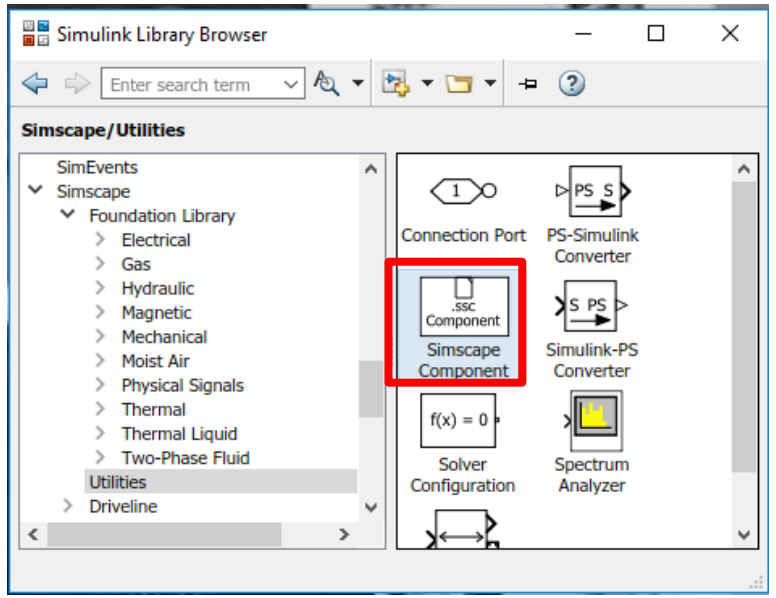


Electrical: **BEGIN**

1 of each

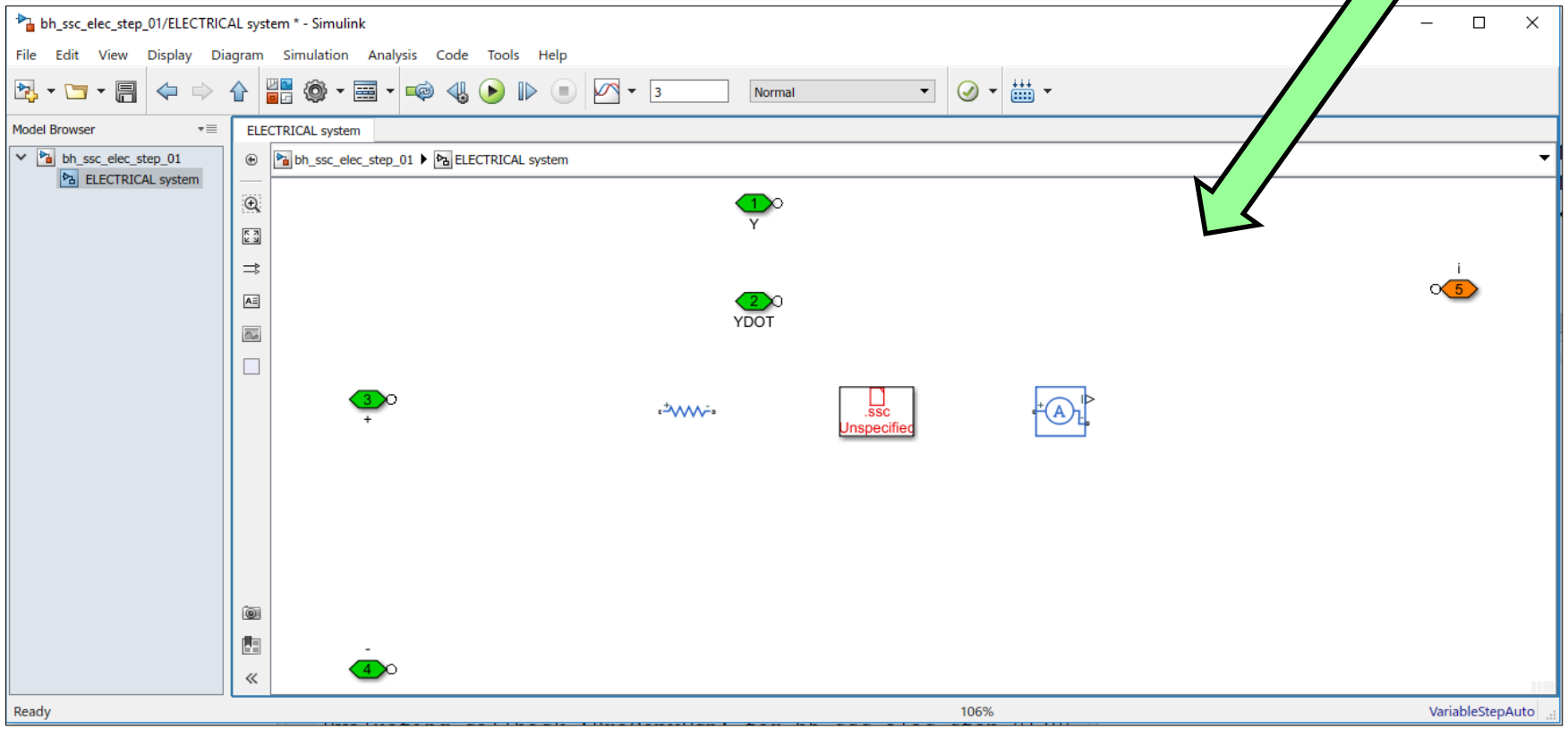




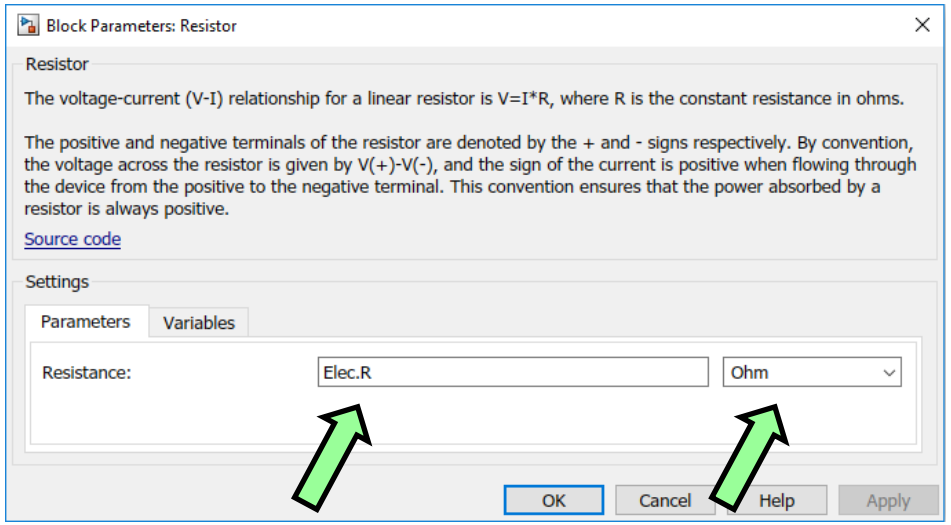


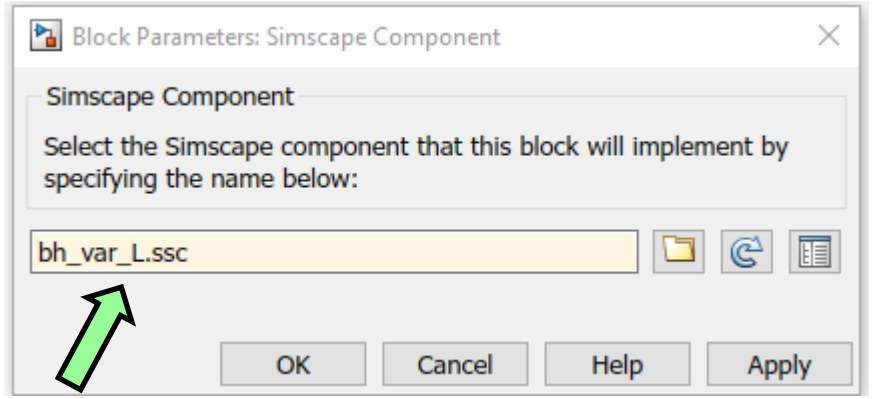
Electrical: **step_01**

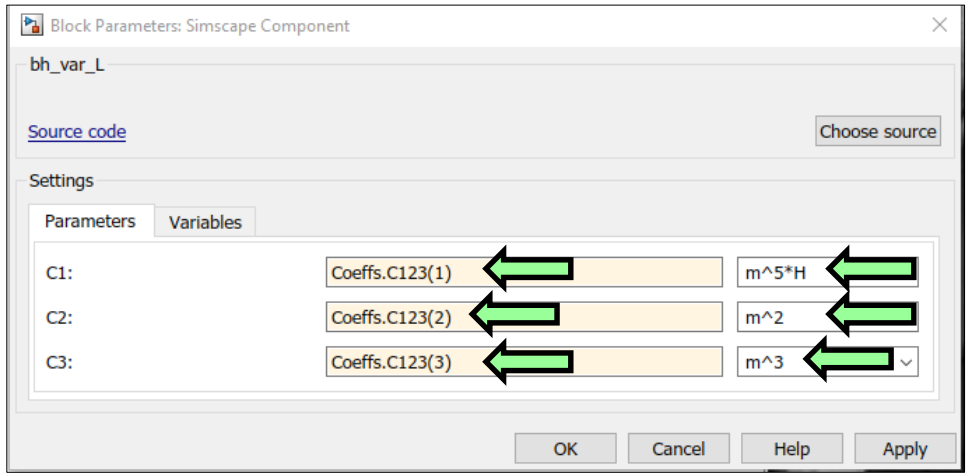
So at the end of **step_01** your model should look like this



Electrical: **BEGIN**

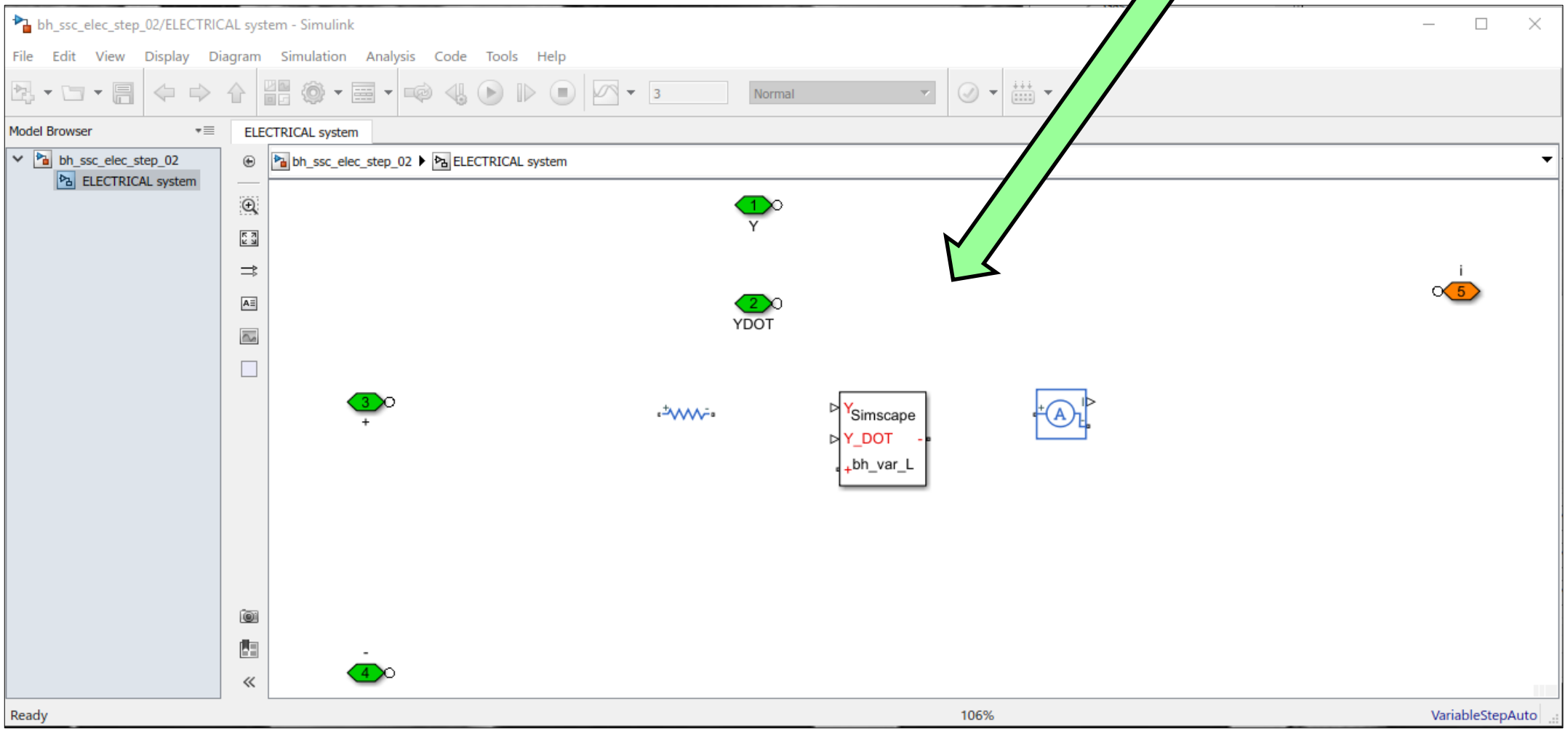






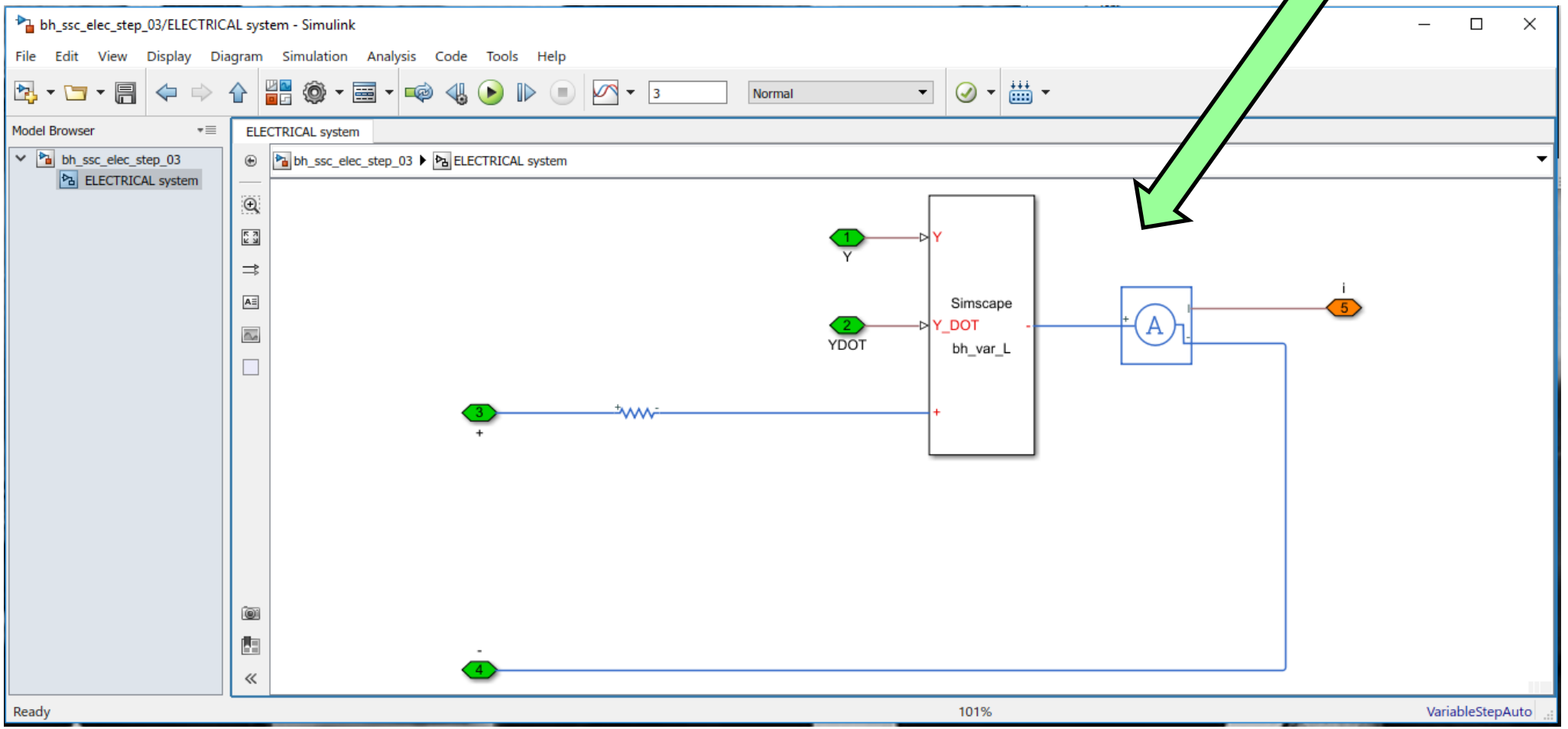
Electrical: **step_02**

So at the end of **step_02** your model should look like this



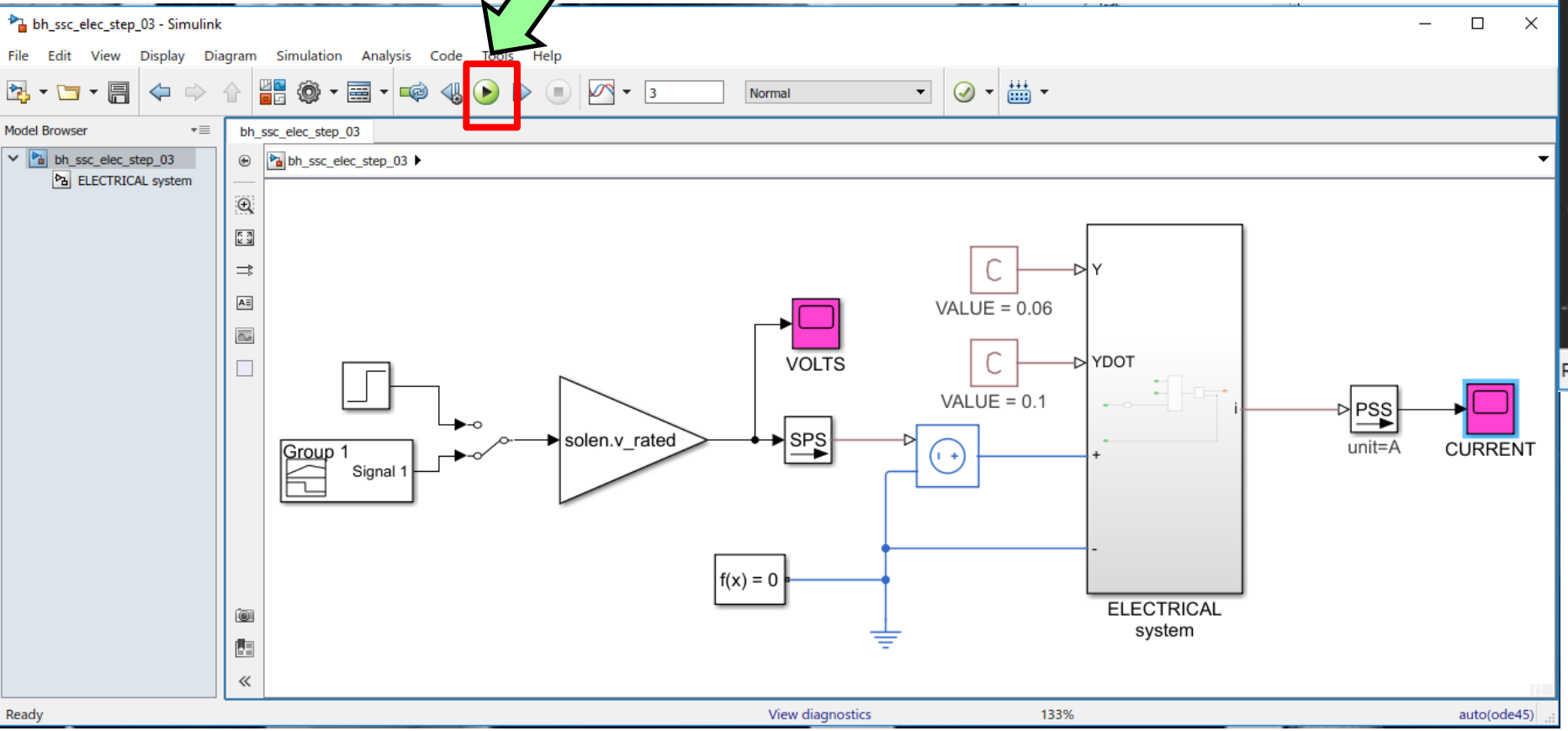
Electrical: **step_03**

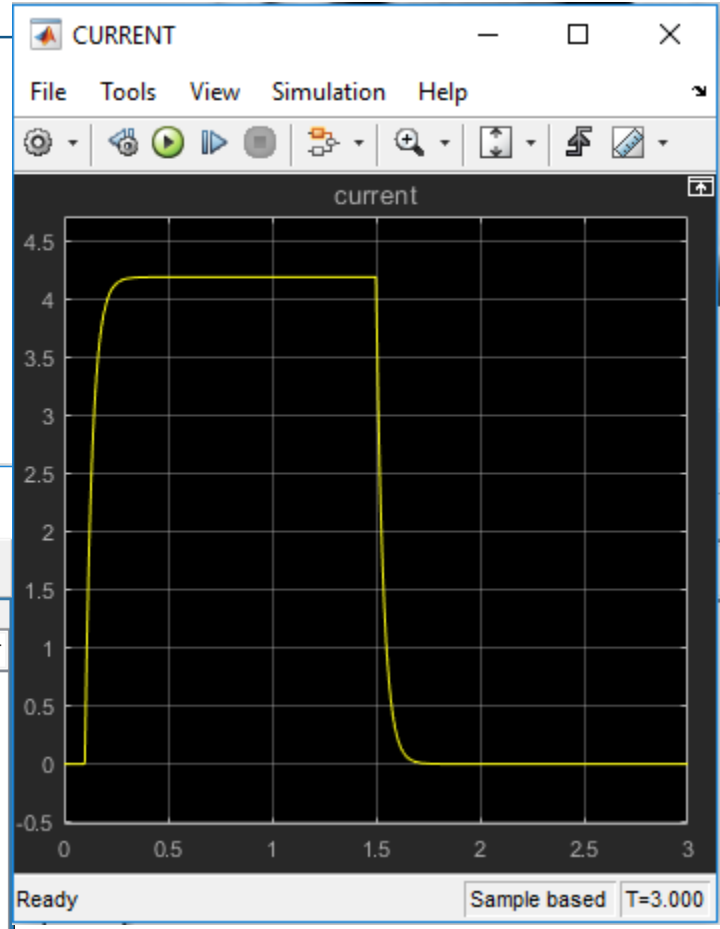
So at the end of **step_03** your model should look like this



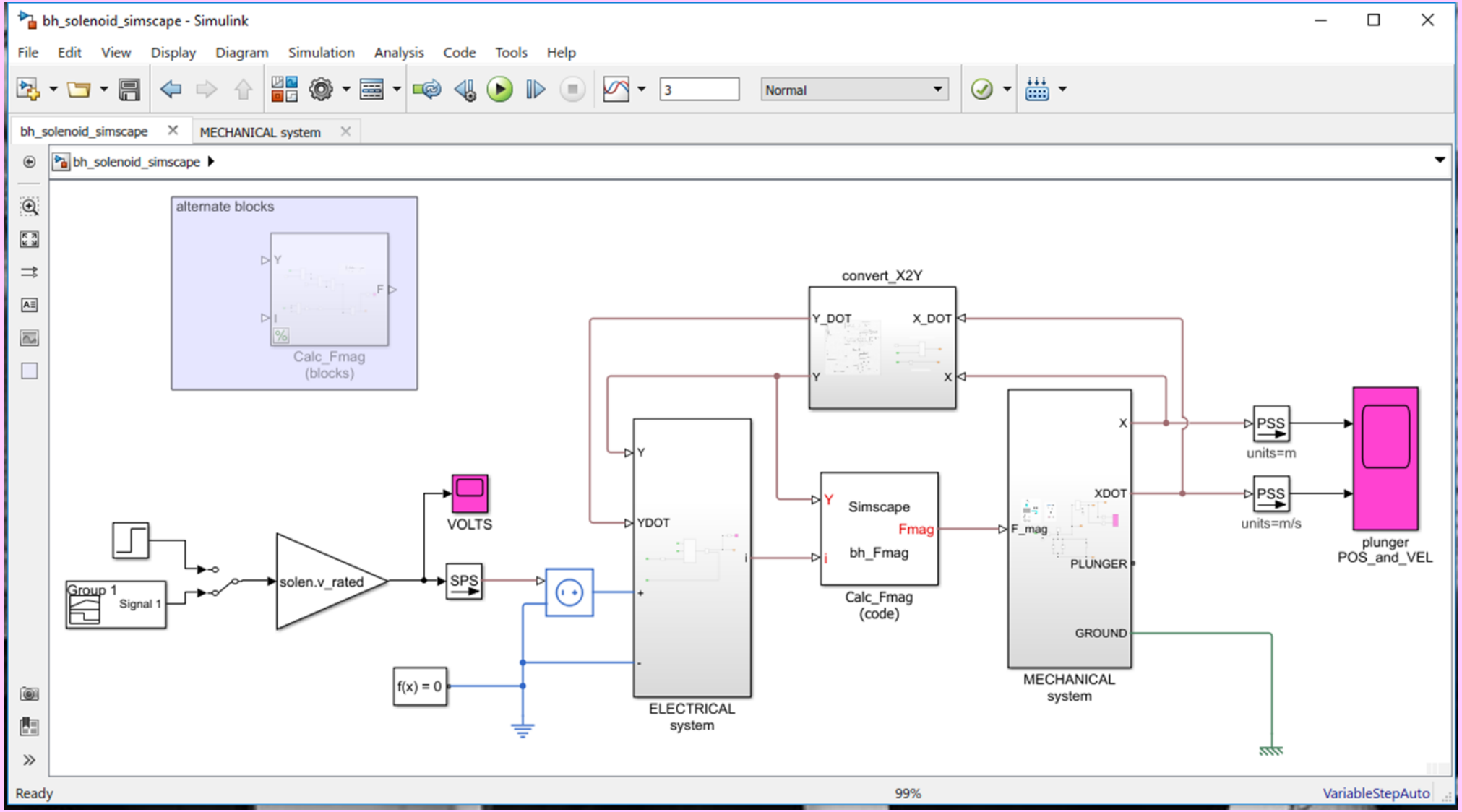
Electrical: **step_03**

Run your simulation

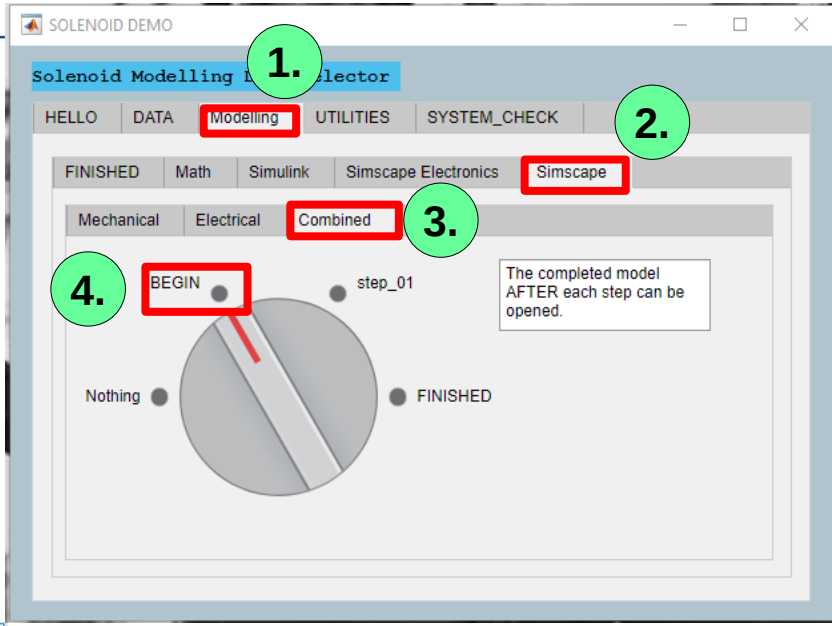


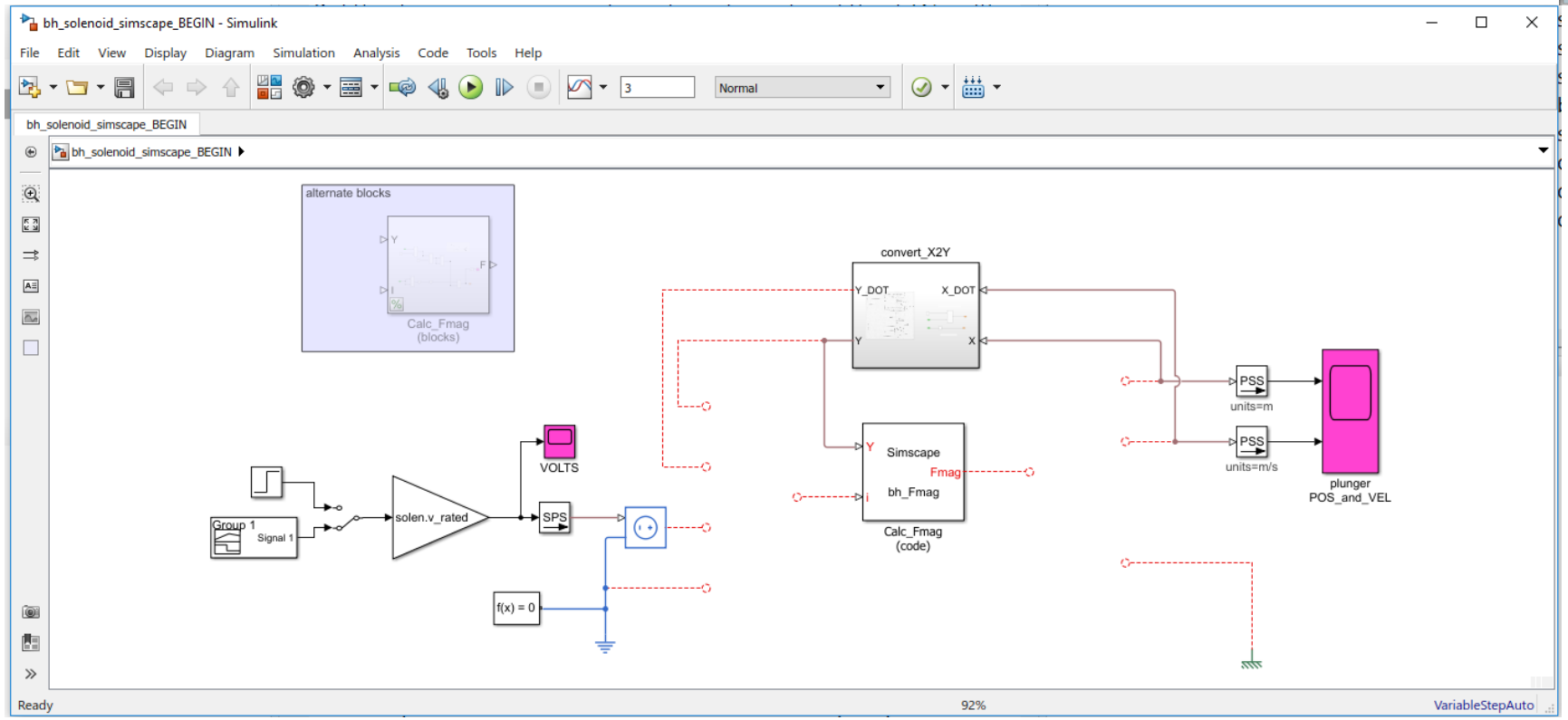


Let's bring together the **MECHANICAL** and **ELECTRICAL** systems

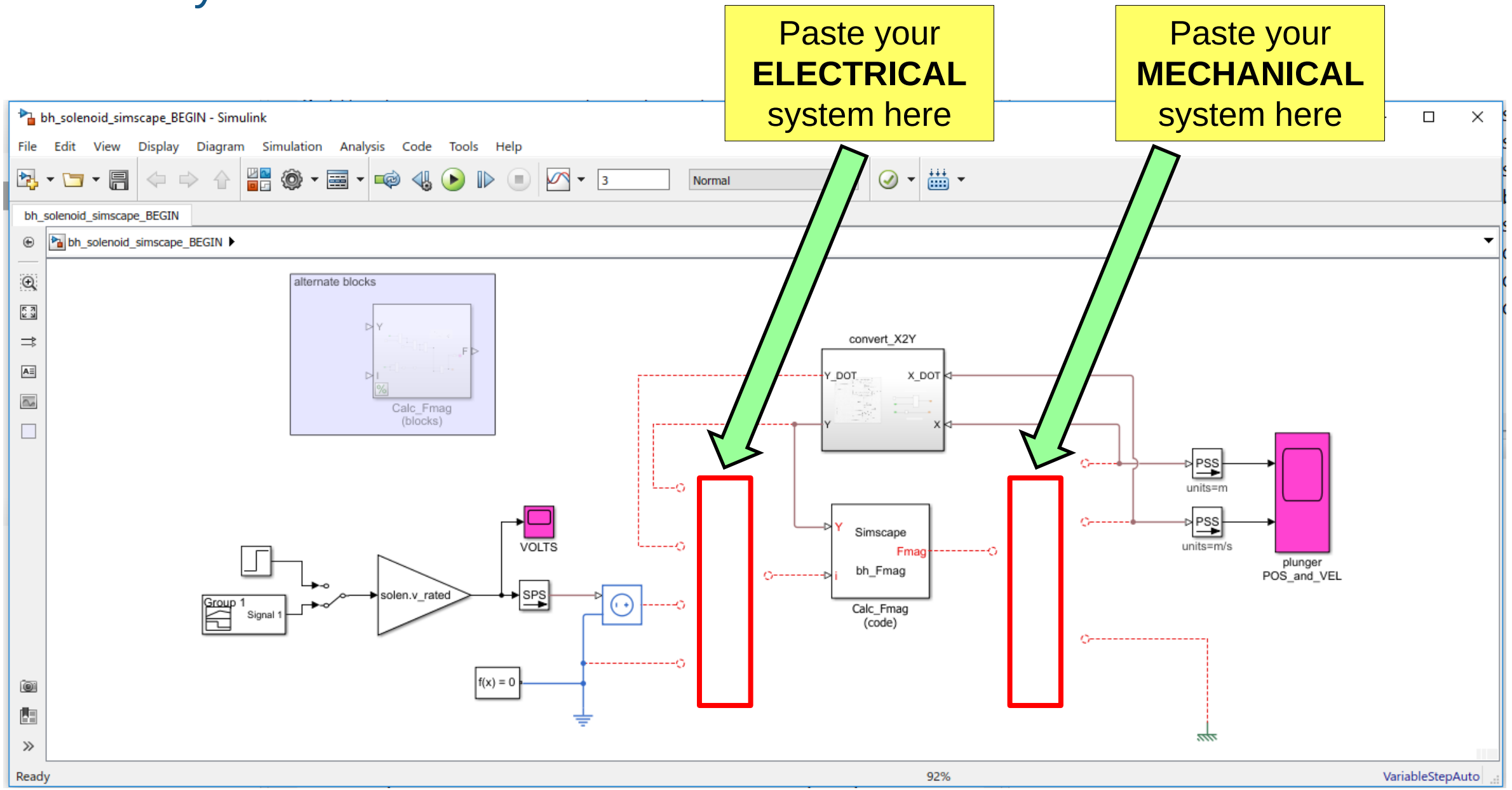


FULL system: **BEGIN**

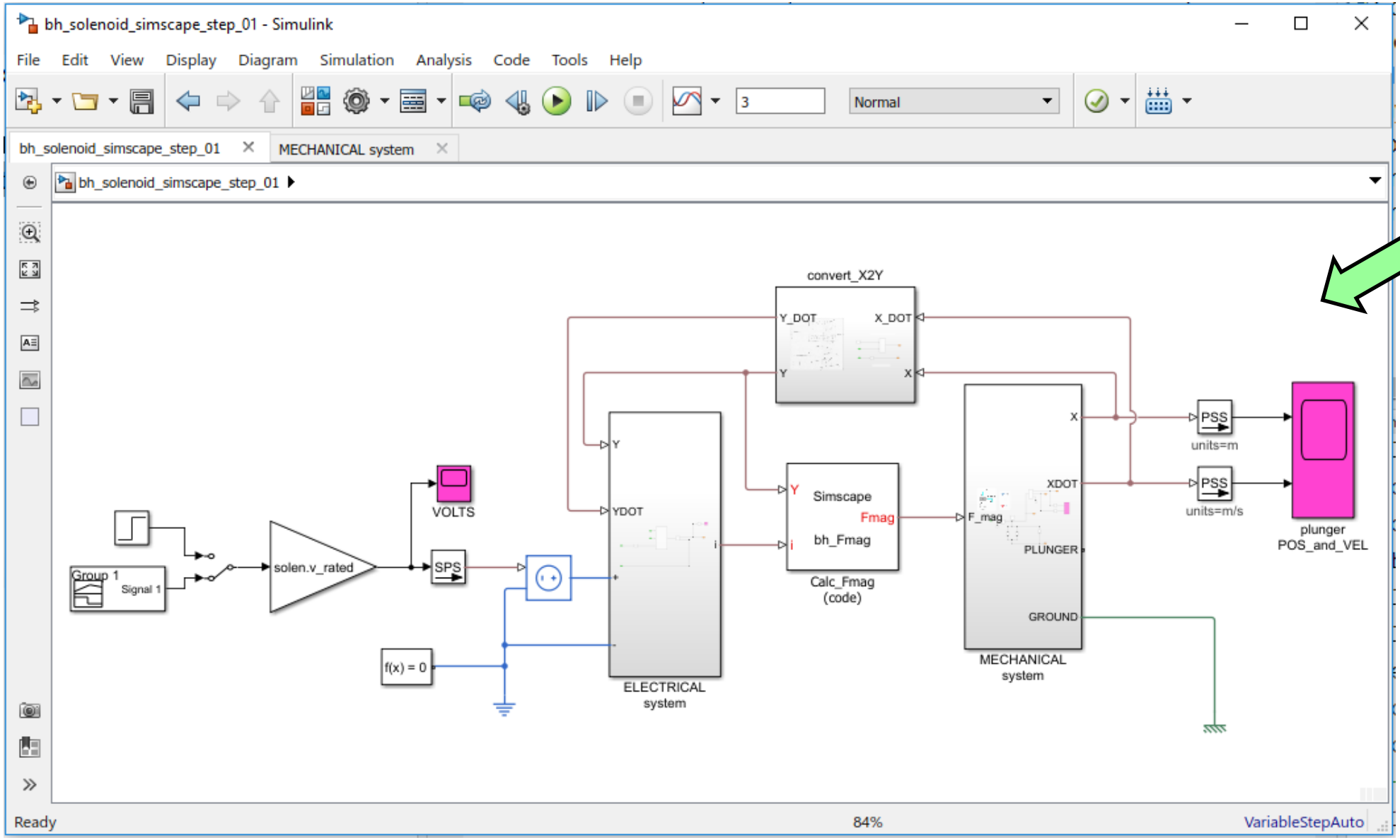




FULL system: **BEGIN**



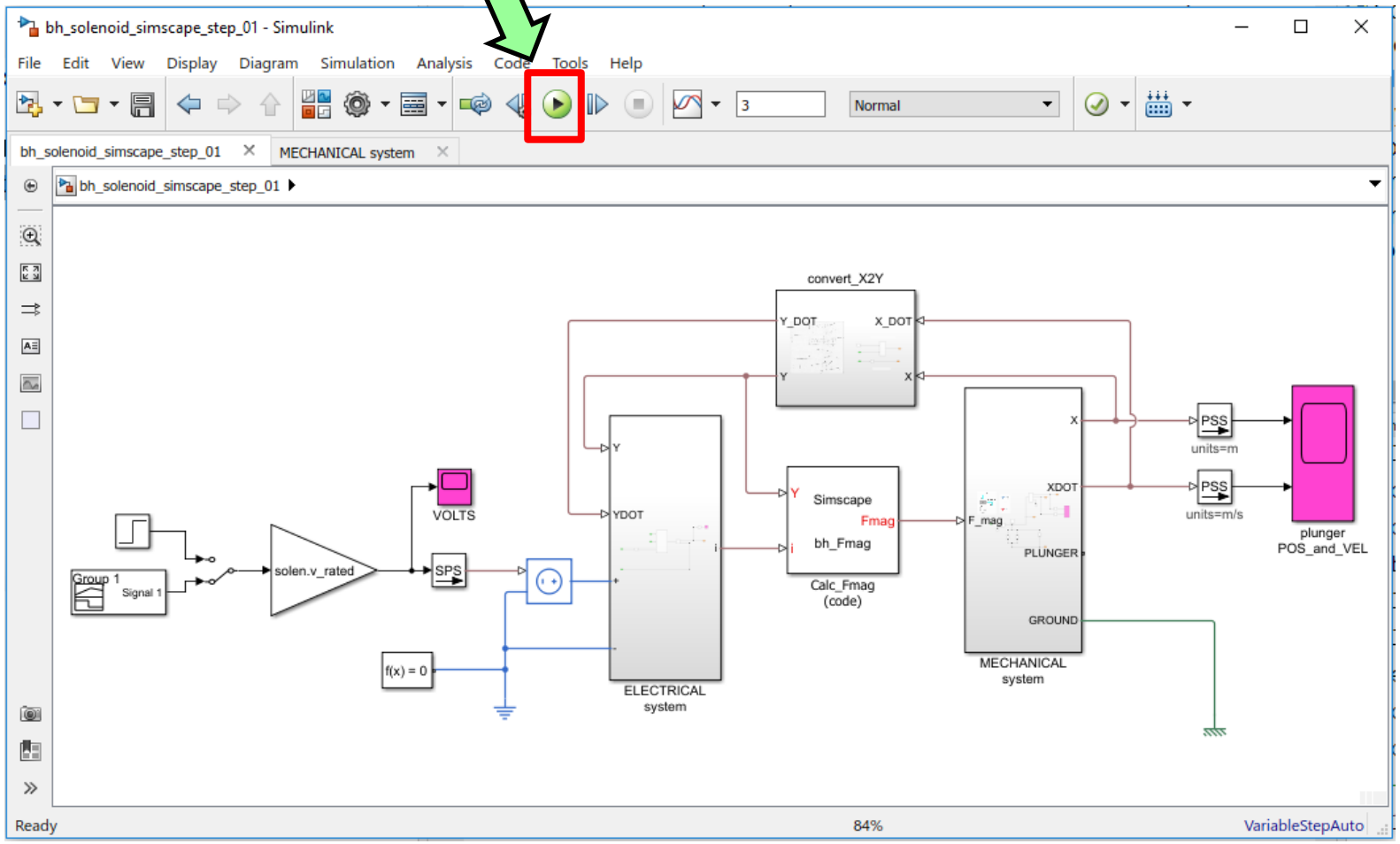
FULL system: **step_01**

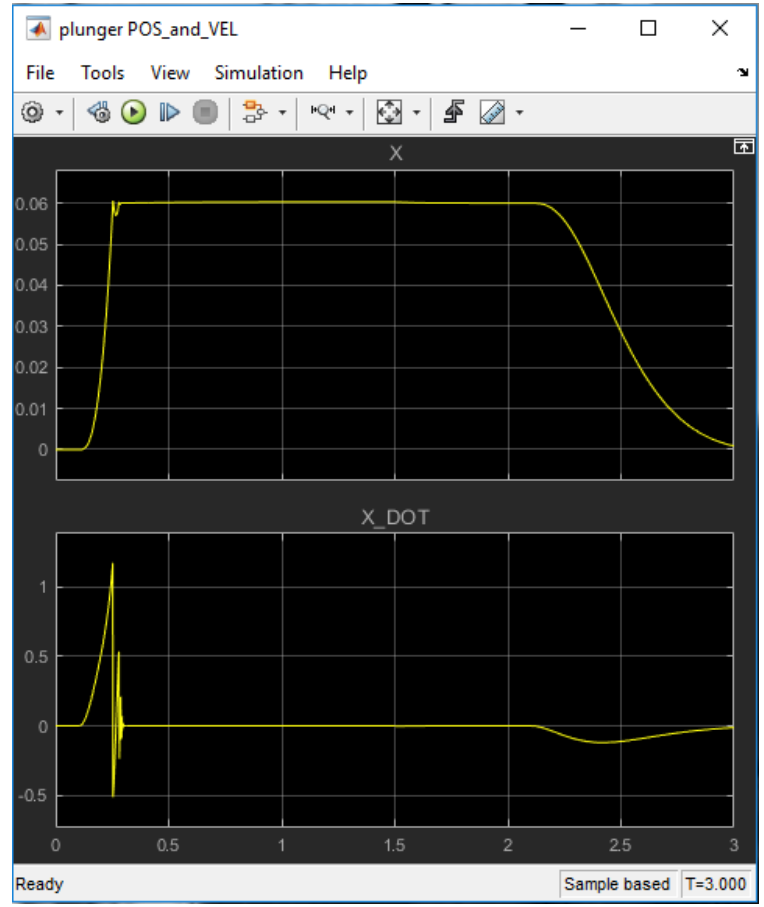


So at the end of **step_01** your model looks like this

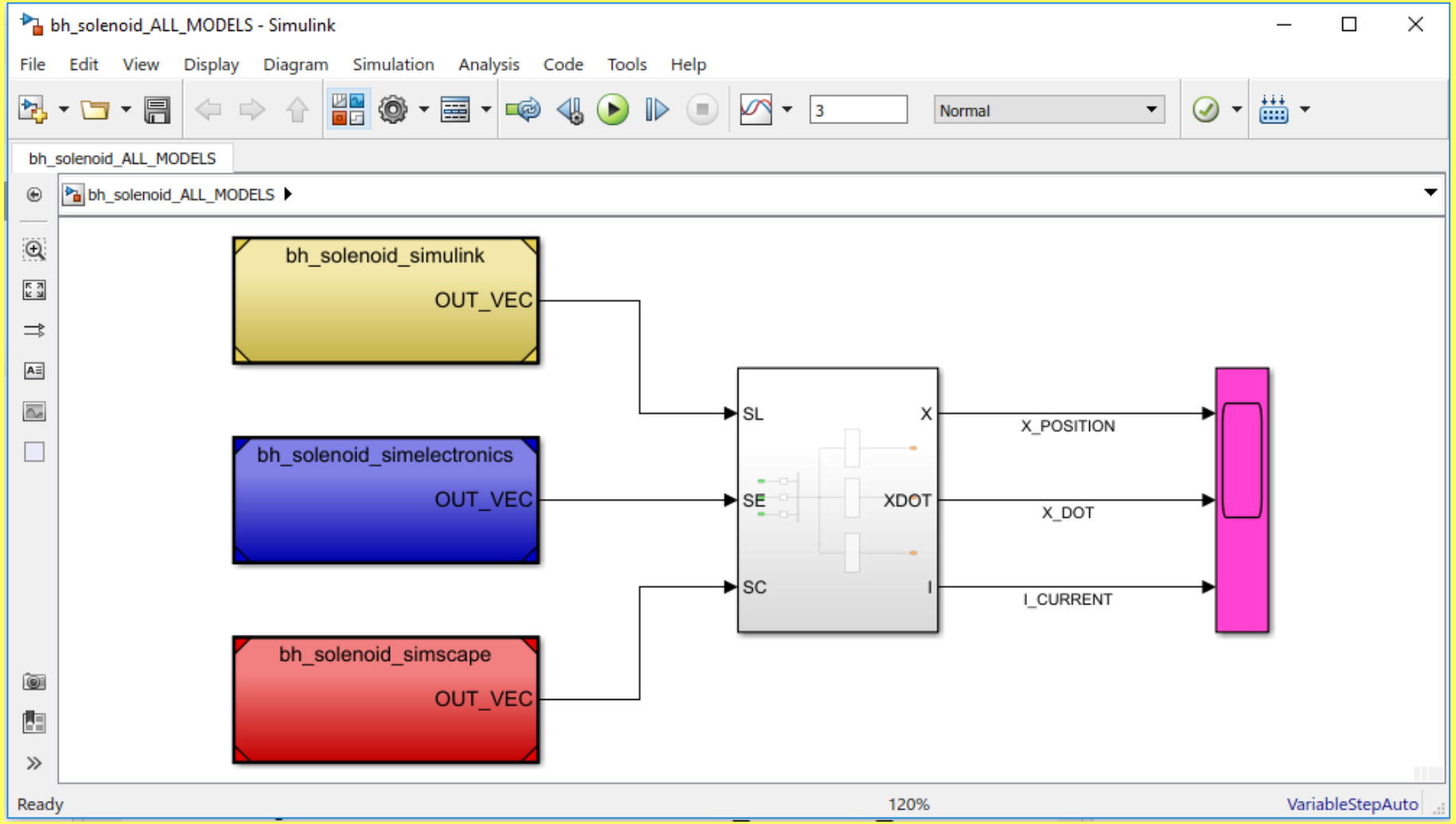
FULL system: **step_01**

Run your simulation

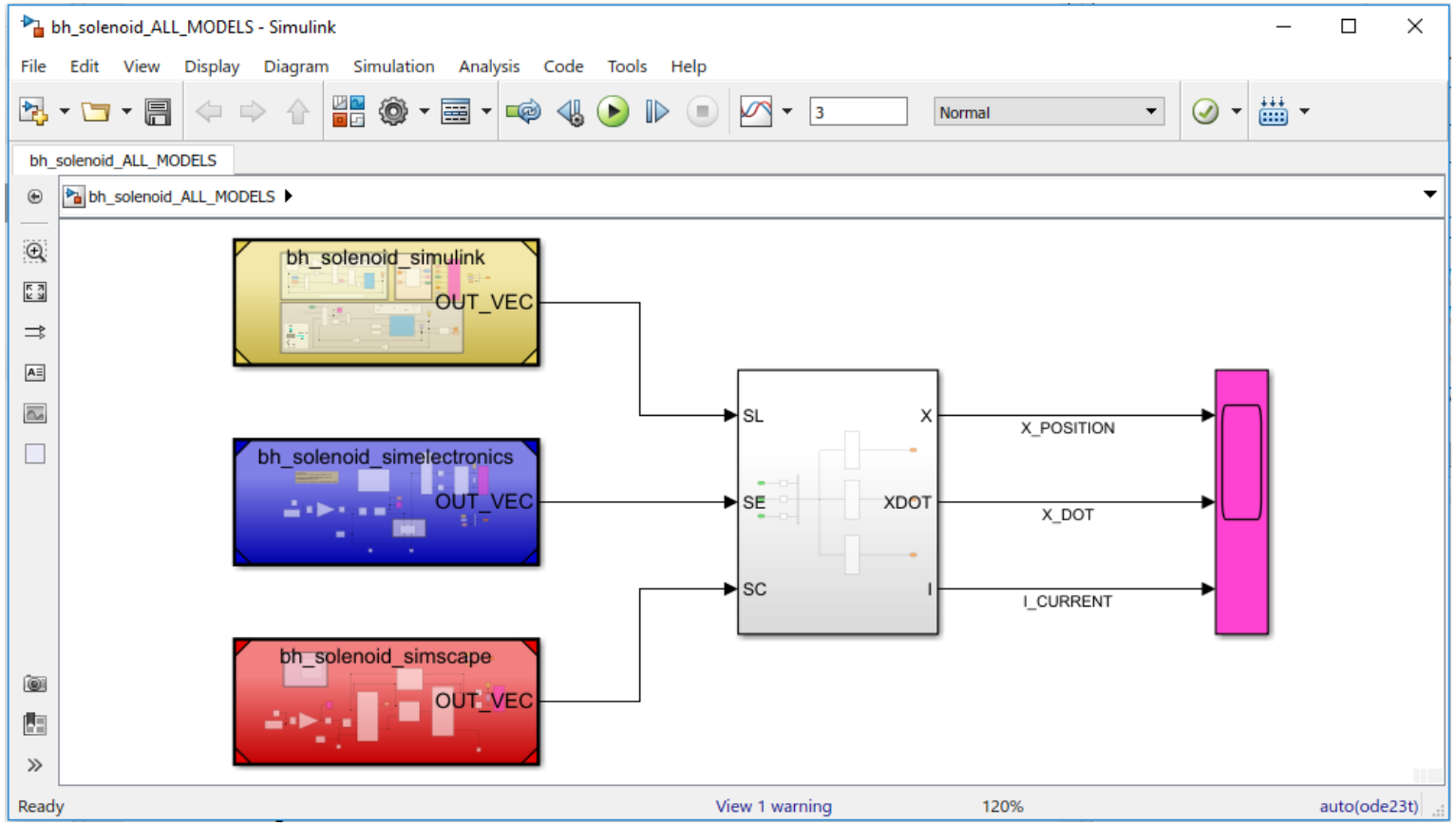


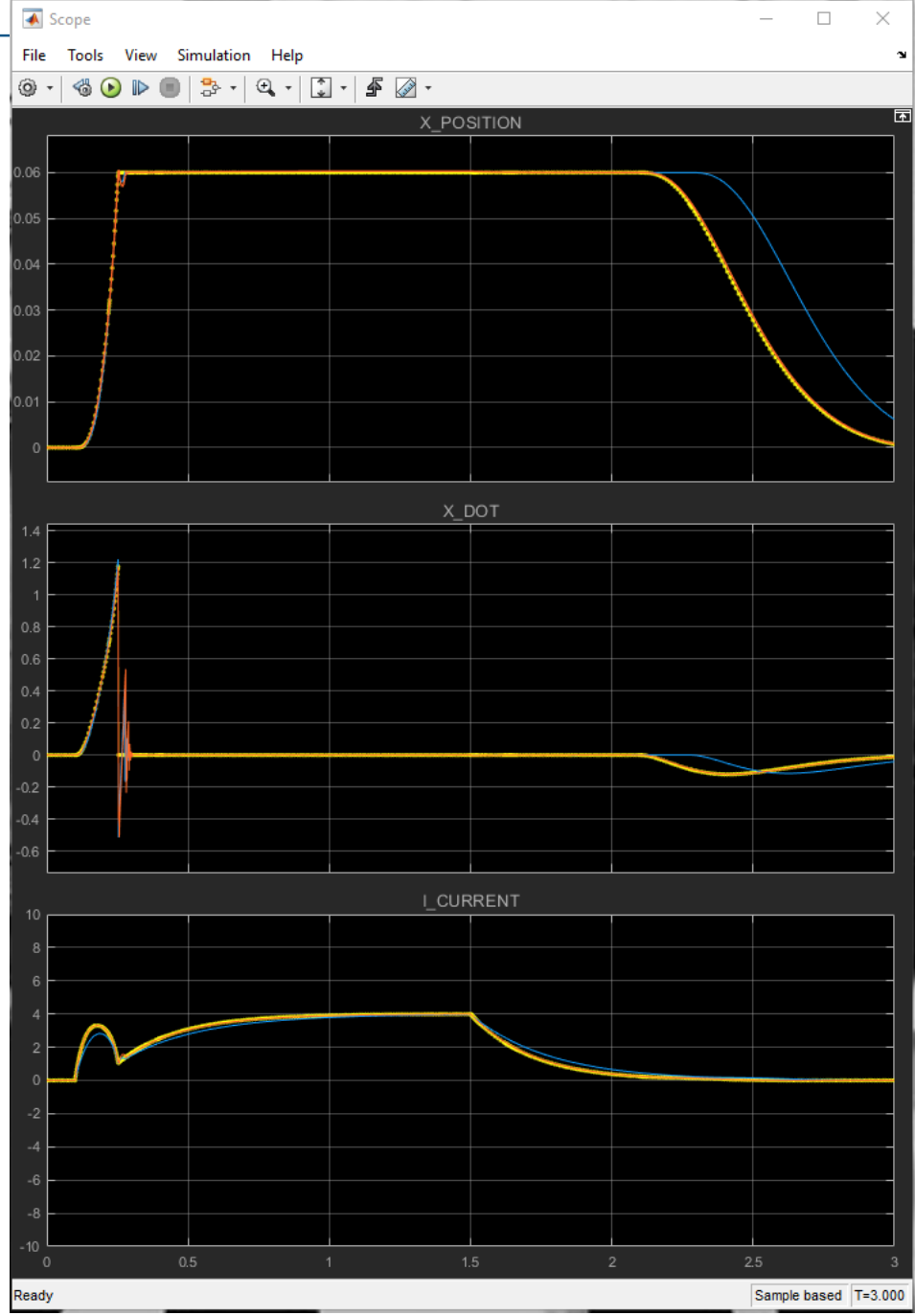


Model: COMPARE



Compare the 3 modelling approaches:





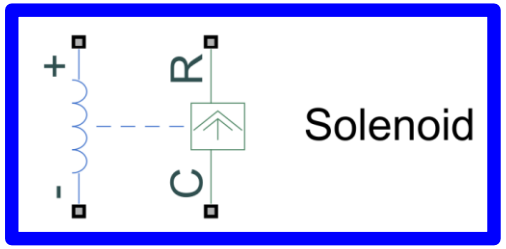
Compare the 3 modelling approaches:

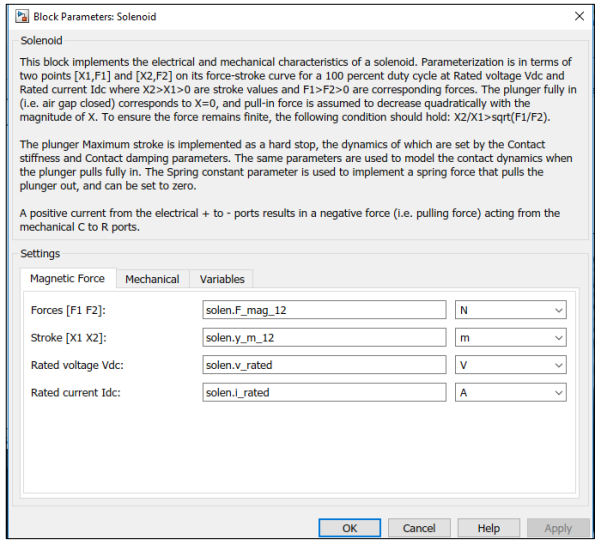
The results:

- The Simulink and Simscape models are very close
- The Simscape Electronics model is a little different … especially during the discharge phase after the power was turned off

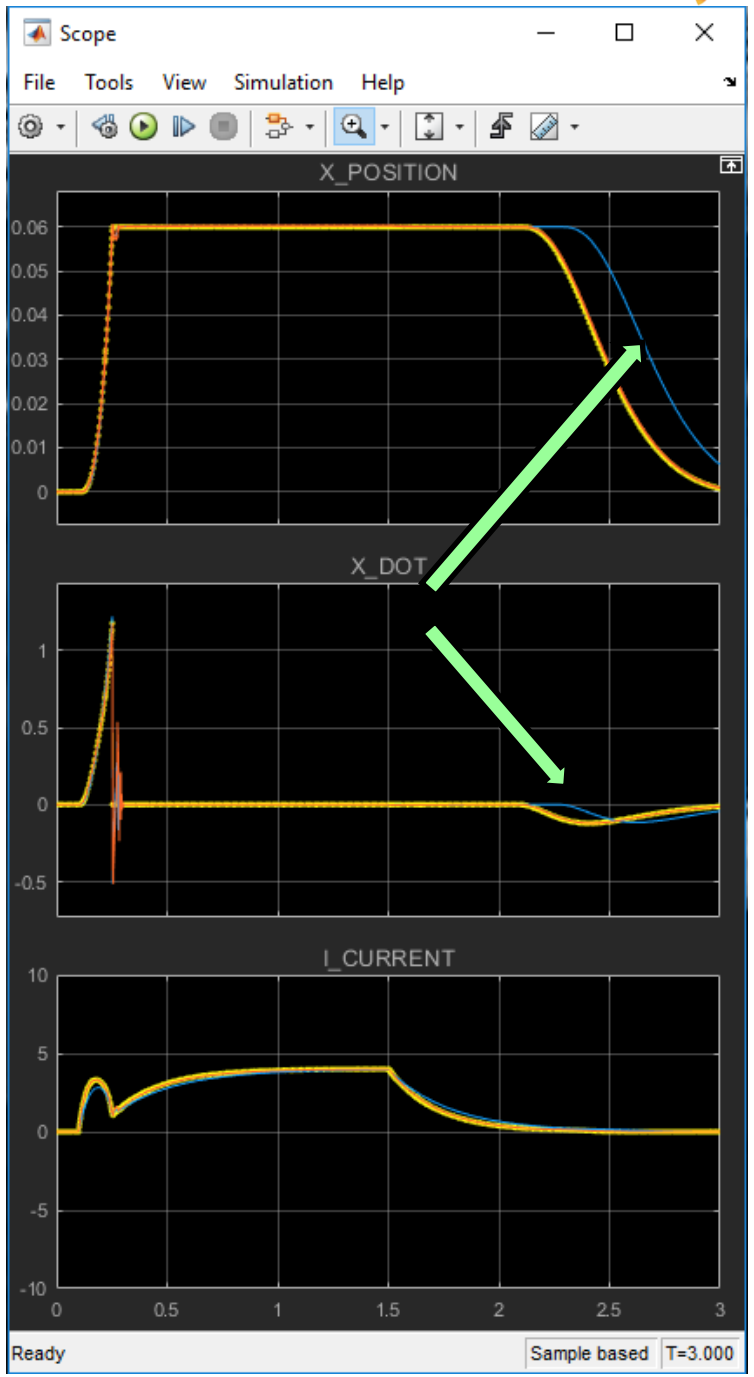
- WHY ?
  - When we parameterised the Simscape Electronics block we were only asked to provide 2 data points on the **Fmag_vs_stroke** curve. AND these 2 data points are used to determine the coefficients of the Fmag and L expressions. So ?
  - So the choice of the 2 data points will effect the curve fit.

See examples on next slide



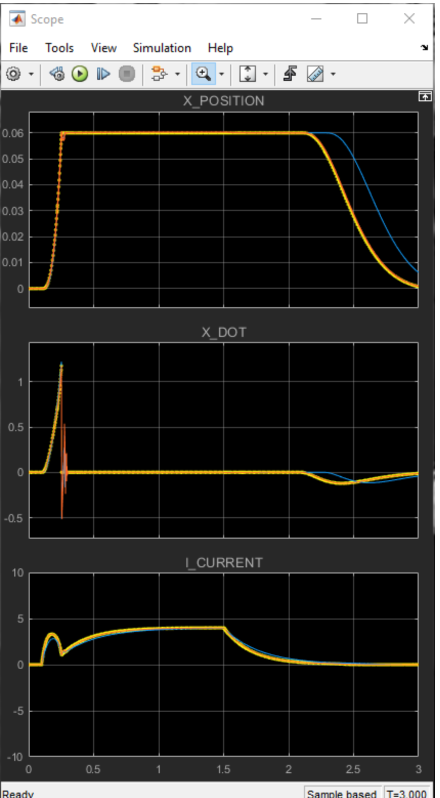


k	y(k)	Fmag(k)
1	0	280.7692
2	0.0050	157.6923
3	0.0100	84.6154
4	0.0150	50.0000
5	0.0200	26.9231
6	0.0250	15.3846
7	0.0300	9.6154
8	0.0350	7.6923
9	0.0400	6.9231
10	0.0450	6.5385
11	0.0500	6.1538
12	0.0550	6.1538
13	0.0600	6.1538

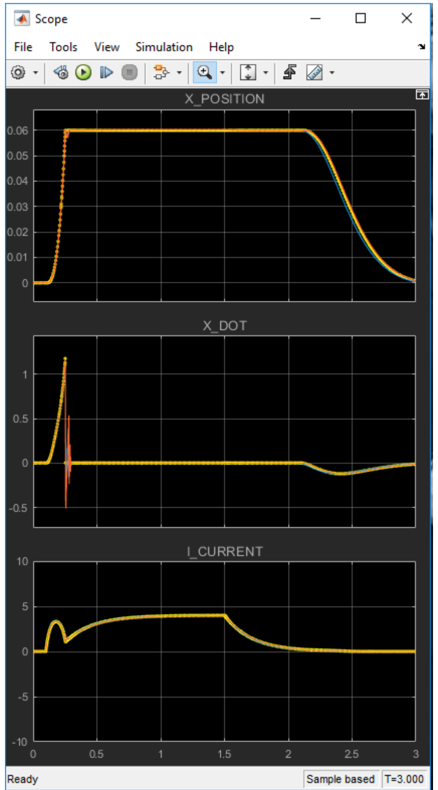


Compare the 3 modelling approaches:

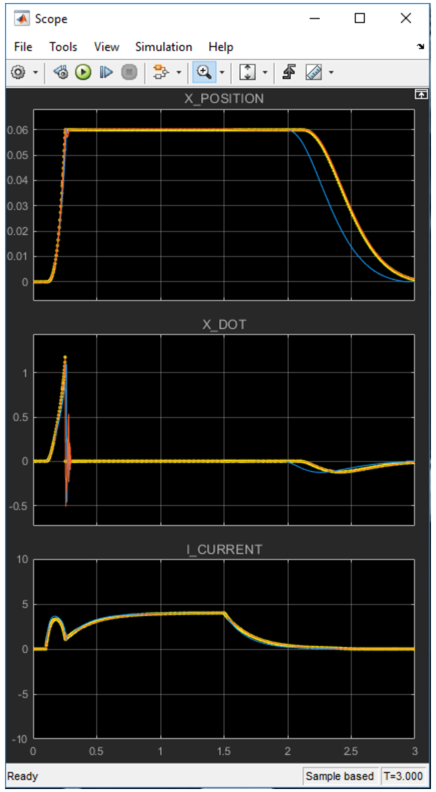
k	y(k)	Fmag(k)
1	0	280.7692
2	0.0050	157.6923
3	0.0100	84.6154
4	0.0150	50.0000
5	0.0200	26.9231
6	0.0250	15.3846
7	0.0300	9.6154
8	0.0350	7.6923
9	0.0400	6.9231
10	0.0450	6.5385
11	0.0500	6.1538
12	0.0550	6.1538
13	0.0600	6.1538



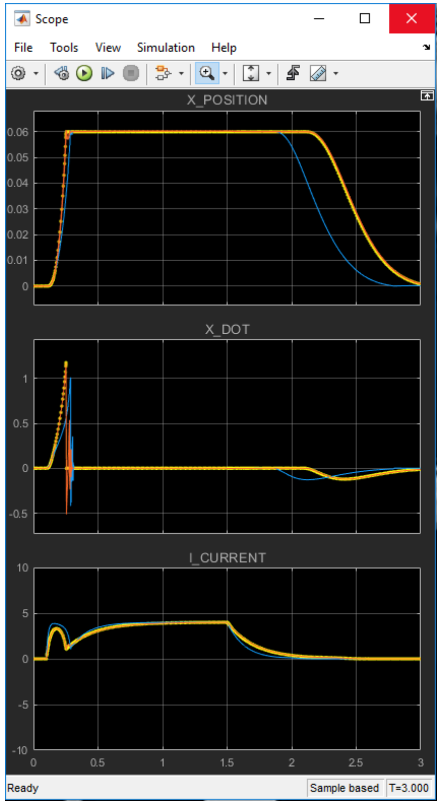
k	y(k)	Fmag(k)
1	0	280.7692
2	0.0050	157.6923
3	0.0100	84.6154
4	0.0150	50.0000
5	0.0200	26.9231
6	0.0250	15.3846
7	0.0300	9.6154
8	0.0350	7.6923
9	0.0400	6.9231
10	0.0450	6.5385
11	0.0500	6.1538
12	0.0550	6.1538
13	0.0600	6.1538



k	y(k)	Fmag(k)
1	0	280.7692
2	0.0050	157.6923
3	0.0100	84.6154
4	0.0150	50.0000
5	0.0200	26.9231
6	0.0250	15.3846
7	0.0300	9.6154
8	0.0350	7.6923
9	0.0400	6.9231
10	0.0450	6.5385
11	0.0500	6.1538
12	0.0550	6.1538
13	0.0600	6.1538

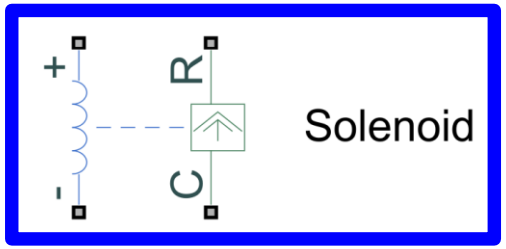


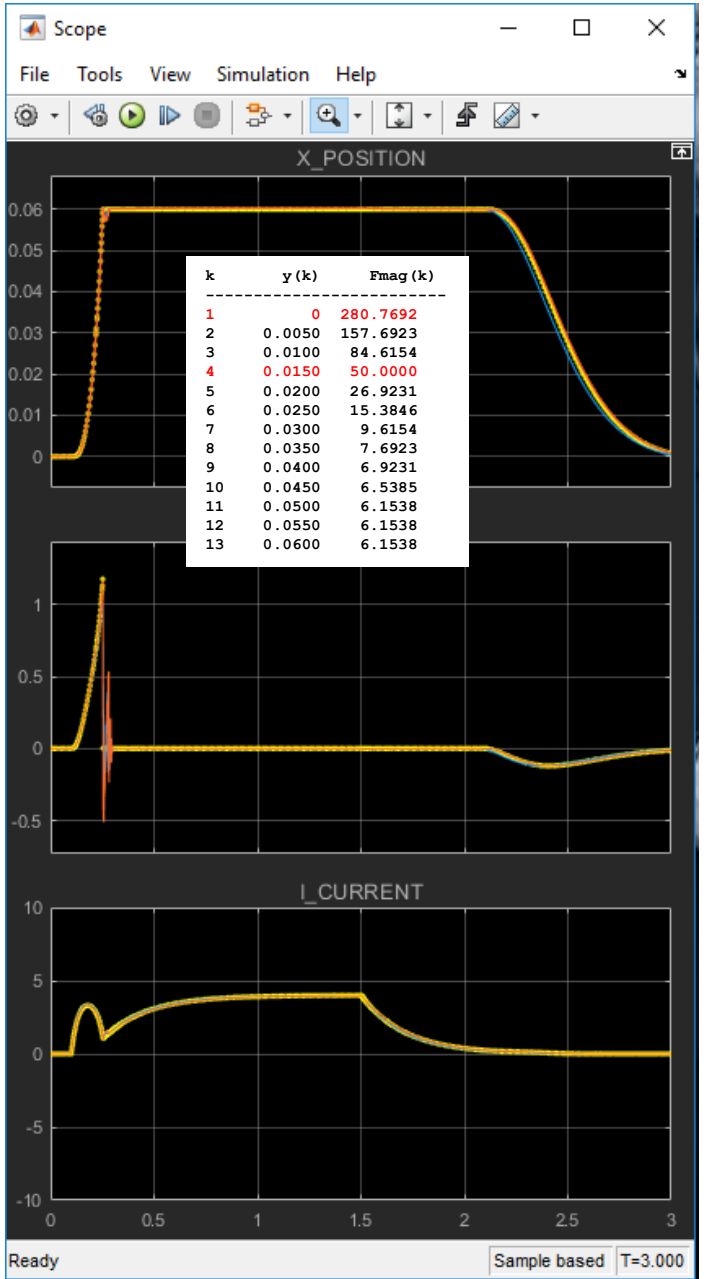
k	y(k)	Fmag(k)
1	0	280.7692
2	0.0050	157.6923
3	0.0100	84.6154
4	0.0150	50.0000
5	0.0200	26.9231
6	0.0250	15.3846
7	0.0300	9.6154
8	0.0350	7.6923
9	0.0400	6.9231
10	0.0450	6.5385
11	0.0500	6.1538
12	0.0550	6.1538
13	0.0600	6.1538

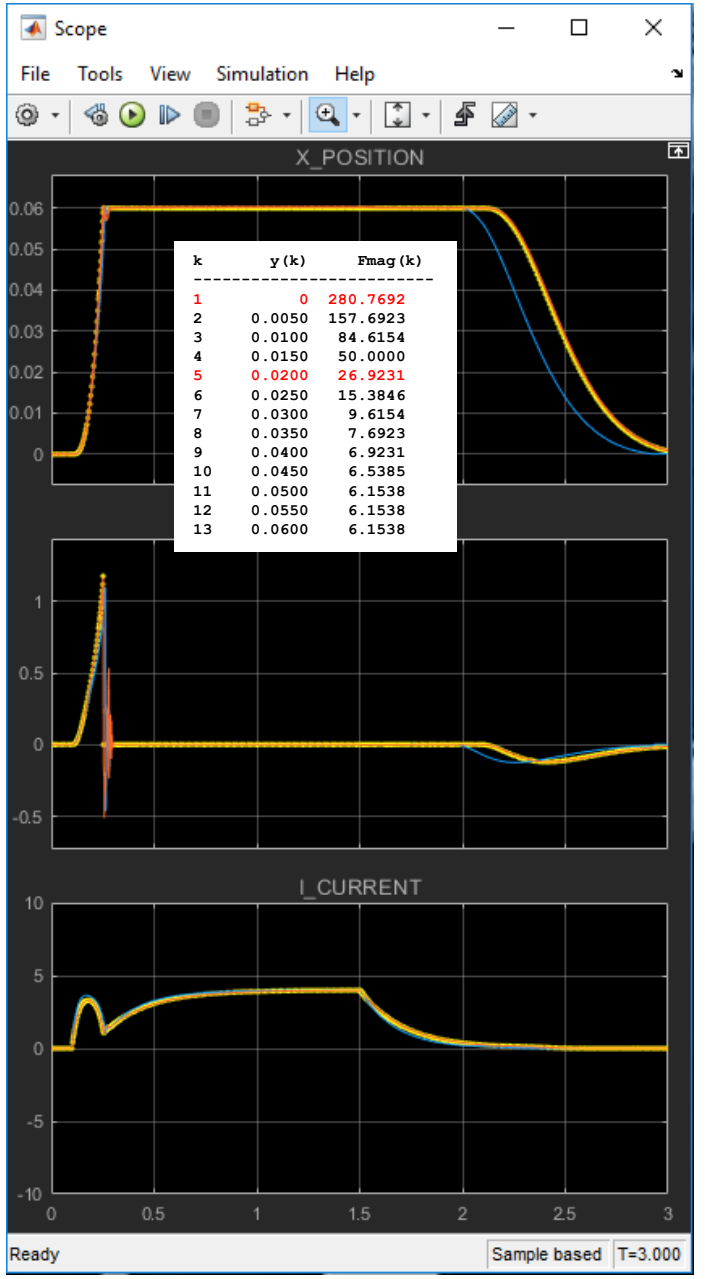


Compare the 3 modelling approaches:

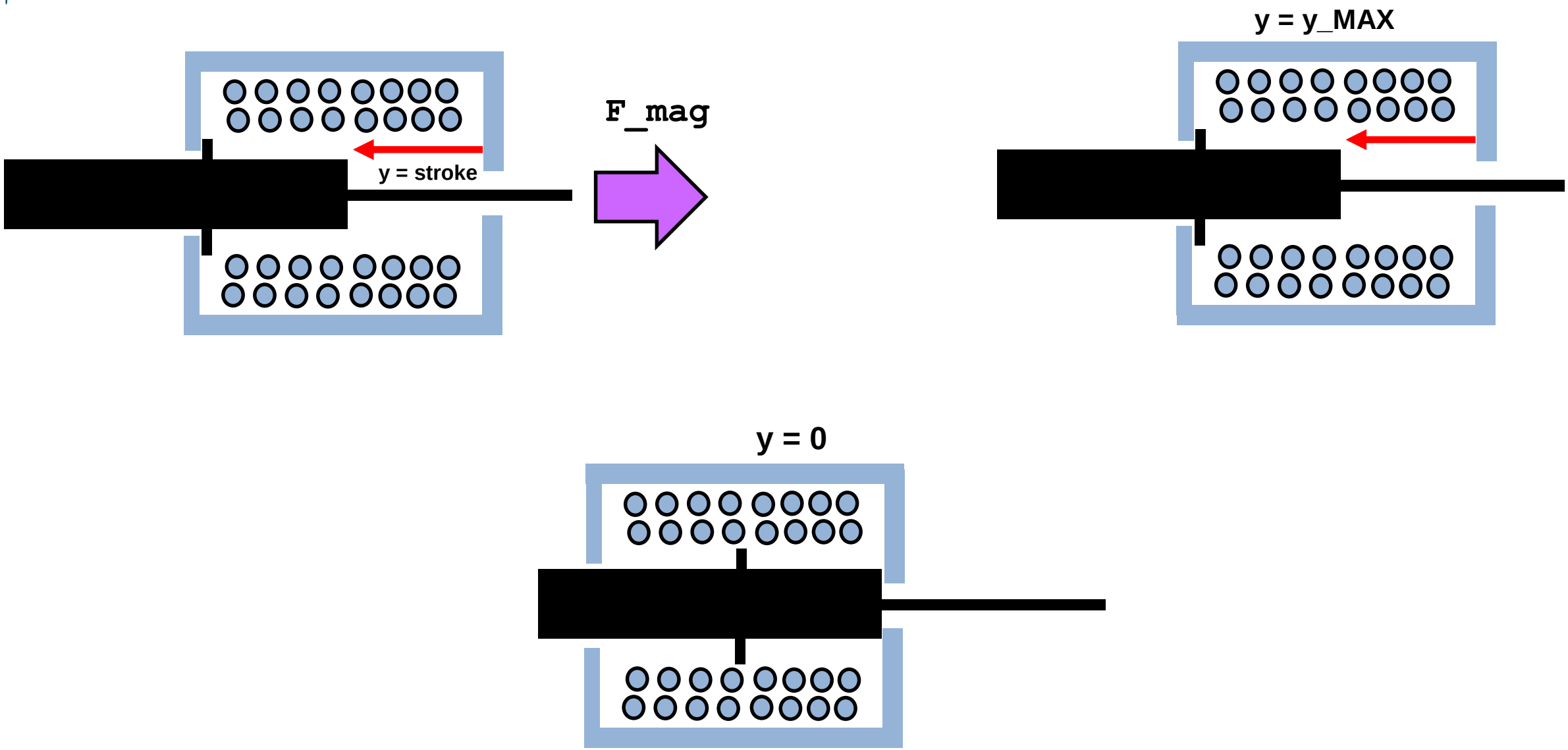
- So ?
  - So choose your 2 data points so that they cover 25-33% of the stroke

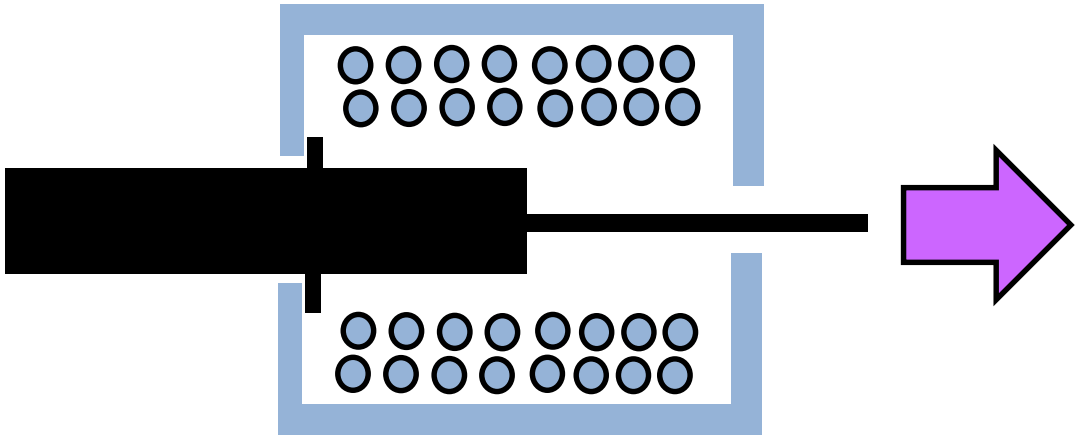


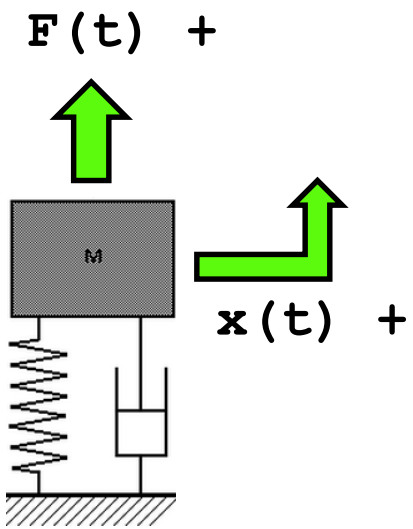


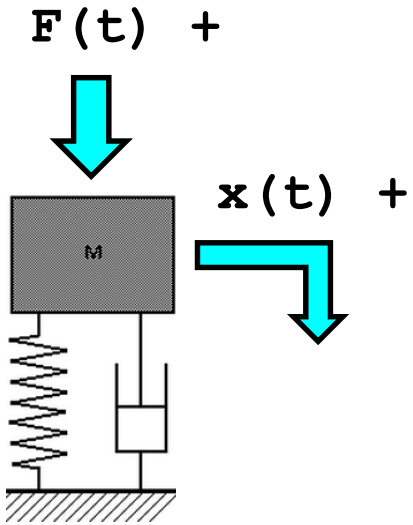


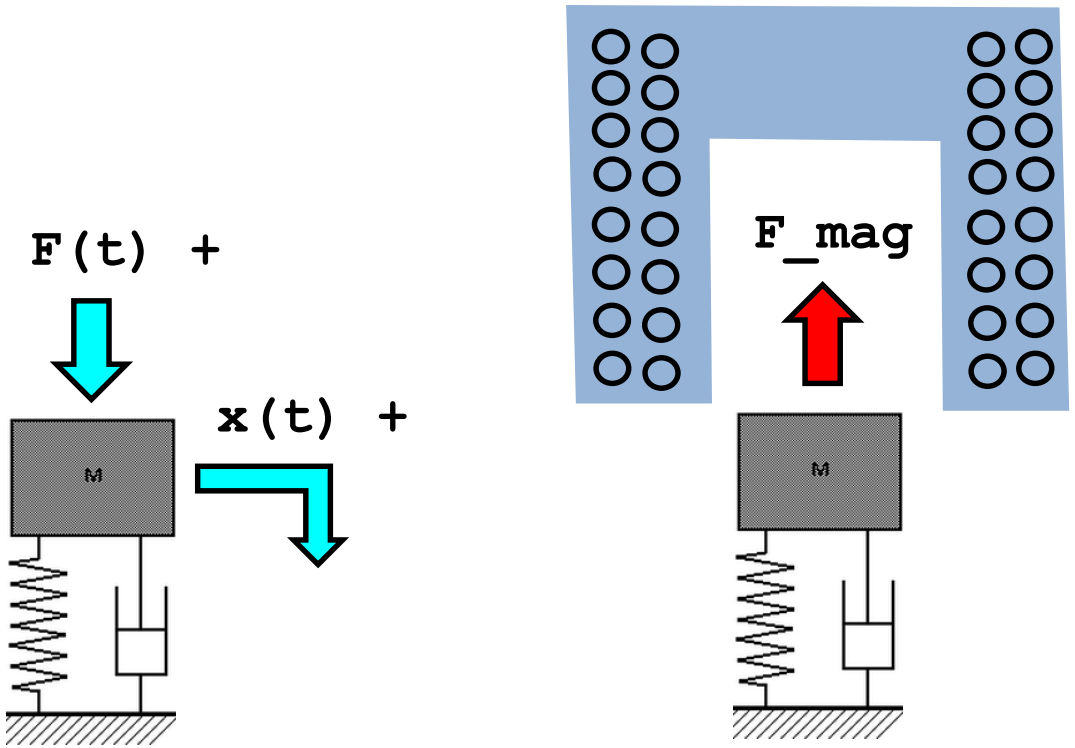
APPENDIX: sketches


y = y_MAX
F_mag
y = stroke
y = 0




F(t) +
M
x(t) +


F(t) +
M
x(t) +


F(t) +
x(t) +
M
F_mag
M