STEPPER MOTOR CONTROL

EEE 316
POWER ELECTRONICS LABORATORY

GROUP - 02

INSTRUCTORS-

Munia Ferdoushi
Satyaki Banik

GROUP MEMBERS-

Elin Ranjan Das- 1706002
Md. Rafiqul Islam- 1706007
Golam Mahmud Samdani-1706029
Ayan Biswas- 1706032

Objective of The Project

This project is based on stepper motor and its' operation under different modes and all. The goals of this project are

- Control unipolar stepper motor with PWM signal.
- Vary inputs e.g., angle, direction and speed
- Observe results for different inputs and stepping mode.
- Analyse functionality of stepper motor.

Working Principle of Stepper Motor

A stepper motor, also known as step motor or stepping motor, is a brushless DC electric motor that divides a full rotation into several equal steps. The motor's position can be commanded to move and hold at one of these steps without any position sensor or anything. That means stepper motor has accurate motor control without any feedback mechanism.

There are two types of stepper motor in large scale-

- **Unipolar Stepper Motor:** The unipolar stepper motor operates with one winding with a centre tap per phase. Each section of the winding is switched on for each direction of the magnetic field. Each winding is made simple with the commutation circuit, this is done since the arrangement has a magnetic pole which can be reversed without switching the direction of the current.
- **Bipolar Stepper Motor:** There is only a single winding per phase. The driving circuit needs to be more complicated to reverse the magnetic pole, this is done to reverse the current in the winding.

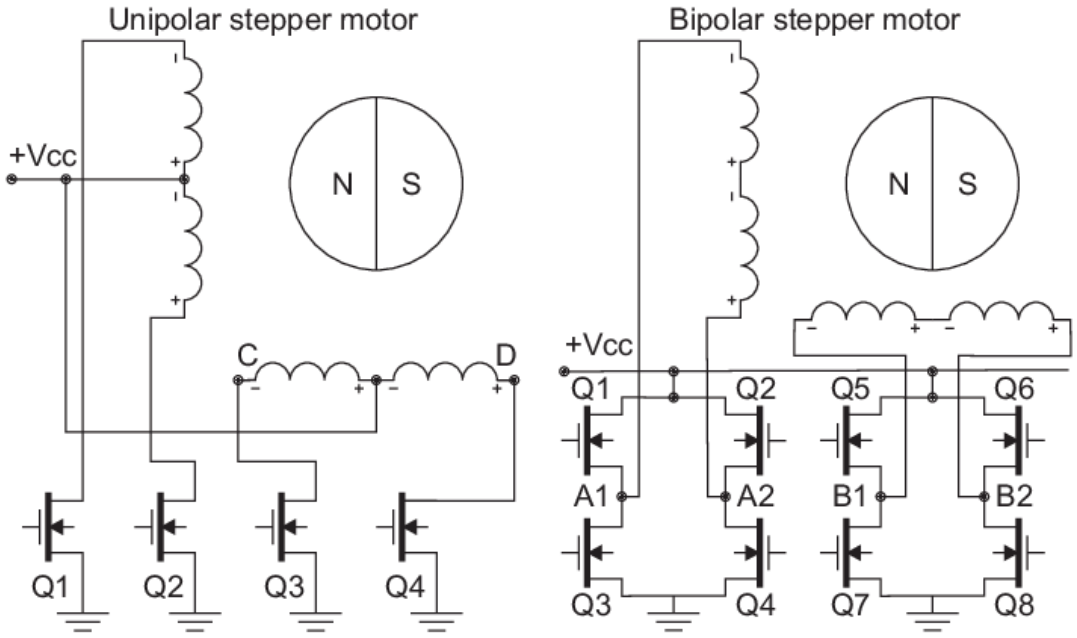


Figure- Internal circuit connection of Unipolar & Bipolar stepper motor

Then again there are three other types of stepper motor-

- **Permanent Magnet Stepper Motor:** Permanent magnet motors use a permanent magnet (PM) in the rotor and operate on the attraction or repulsion between the rotor PM and the stator electromagnets.
- **Variable Reluctance Stepper Motor:** Variable reluctance (VR) motors have a plain iron rotor and operate based on the principle that minimum reluctance occurs with minimum gap, hence the rotor points are attracted toward the stator magnet poles. Here, the rotor's angular position depends on the magnetic circuit's reluctance that can be formed among the teeth of the stator as well as a rotor.
- **Hybrid Synchronous Stepper Motor:** Hybrid stepper motors are named because they use a combination of permanent magnet (PM) and variable reluctance (VR) techniques to achieve maximum power in small package sizes.

A stepper motor can have different number of phases in the design.

For better simulation and execution, we used

- **Unipolar Stepper Motor**
- **4-phase coil**
- **Variable Reluctance Stepper Motor.**

Since it has 4 phases, the construction will include 4 coils and 4 gate pulses to control them.

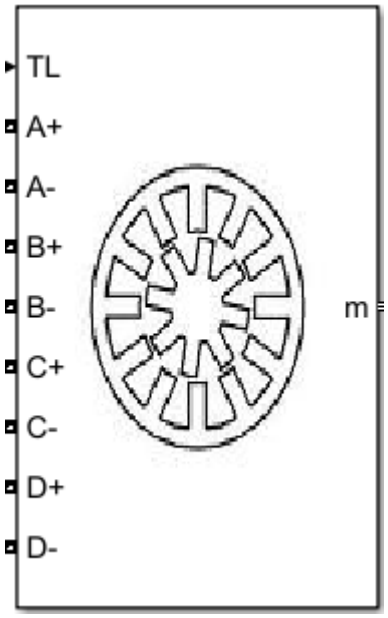


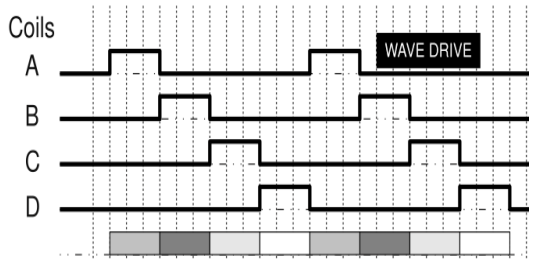
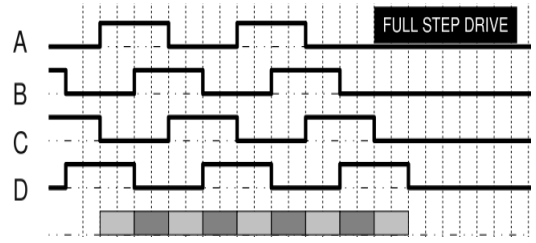
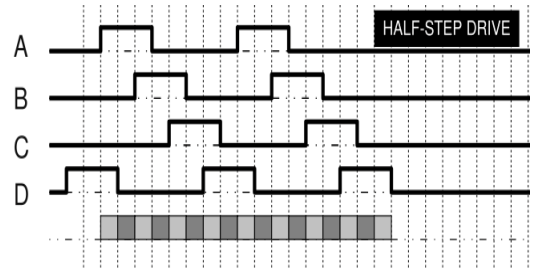
Figure- Simulink block of 4-phase variable reluctance unipolar stepper motor.

Operation Mode of Stepper Motor

There are in total three driving modes for stepper motor-

- **Wave Drive Mode:** This is the basic method of driving a stepper motor. In this technique every phase otherwise stator next to each other will be triggered one by one alternatively with a special circuit.
- **Full Stepping Mode:** In this technique, two stators are activated at a time instead of one in a very less time period. This technique results in high torque & allows the motor to drive the high load.
- **Half Stepping Mode:** This technique is fairly related to the Full step drive because the two stators will be arranged next to each other so that it will be activated first whereas the third one will be activated after that. This kind of cycle for switching two stators first &after that third stator will drive the motor.

So, the comparison table of the features of these modes-

Mode	Features	Graphical View
Wave Drive	• Only a single phase activated at a time. • All four coils energized in sequence. • Less consumed power and less output torque. • Motors not usually driven in this mode	
Full Stepping	• Two motor coils energized together • Double Power consumed than the wave drive mode • High Torque provided • Motors usually driven in this mode	
Half Stepping	• Alternation between 2 phases on & a single phase on • Double step resolution • Precise and accurate motor angle control • Moderate power consumption and output torque	

In our Simulink project, we included each of the stepping modes and observed different operation of stepper motor under different modes.

Parameters of Stepper Motor

The stepper motor parameters mainly include step angle, steps for each revolution and RPM.

 # Step Angle: The step angle is the basis of the movement of a stepping motor. The step angle depends on the total number of magnetic poles of the motor. The step angle is determined by the formula:

Step angle = 360 degrees / (steps/rev)

 # Steps per Each Revolution: The steps for each resolution can be defined as the number of step angles necessary for a total revolution. The formula for this is:

360 degrees/Step Angle.

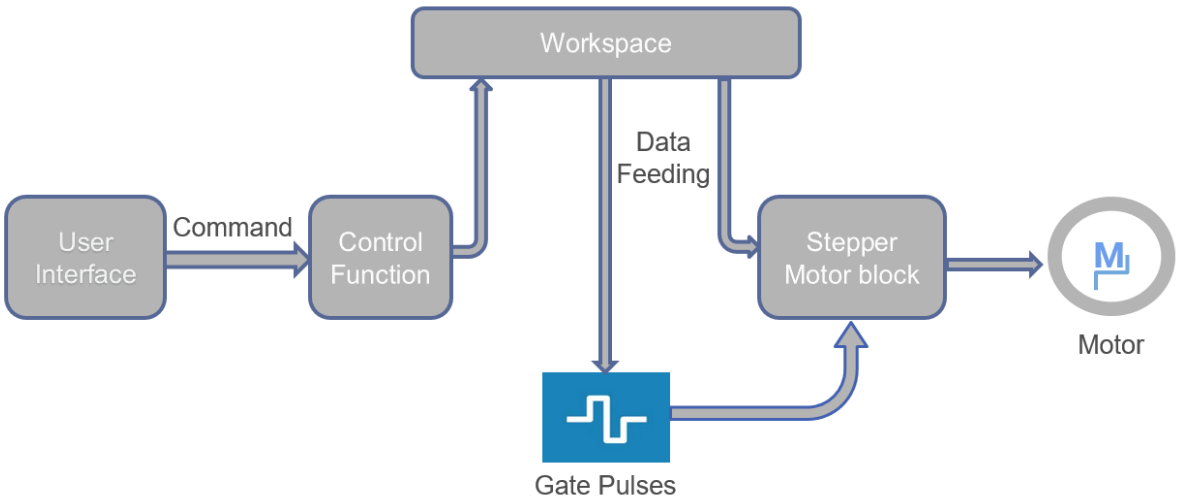
 # Revolution per Minute: It is used to measure the frequency of revolution. So, by using this parameter, we can calculate the number of revolutions in a single minute. This parameter has the following relation with time period of the circulating roto:

Time Period = 60/RPM

As step angle and steps/rev is connected to one another, just one of them being known is enough for the motor parameter value.

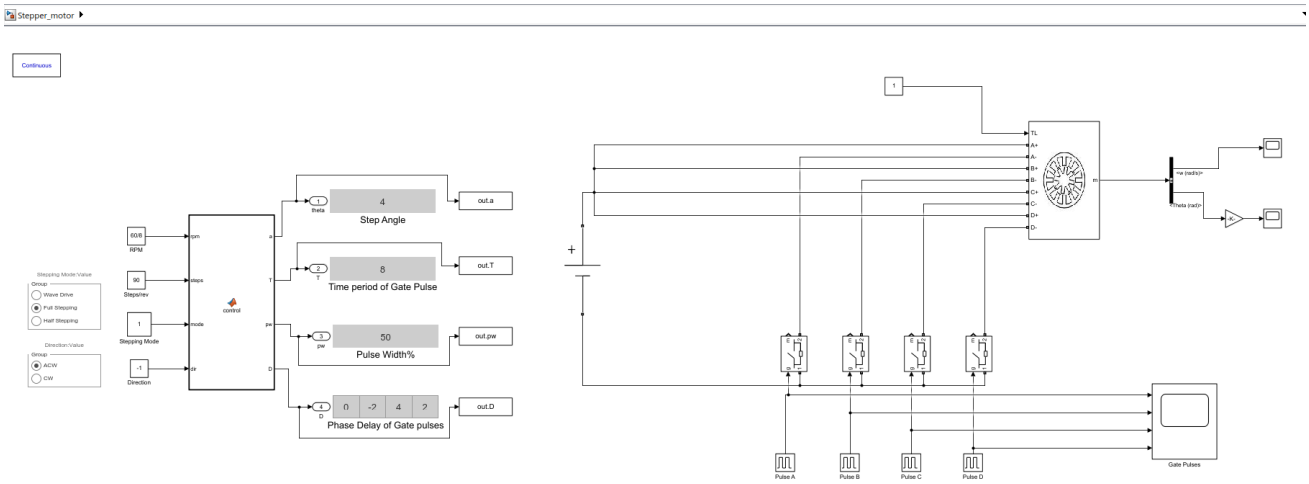
So, in our Simulink project we took two main inputs- RPM & steps/rev.

Workflow



Simulink Model

The Simulink model we executed in this project-



Here, we have two sections-

- **First/Left section** is to take the stepper motor parameter- RPM, steps/rev; stepping mode, direction as inputs and calculate important values like step angle, time period, pulse width and phase delays. This section first calculates the data and then feed them to the second section. This is the **Control** part.
- **Second/Right section** is the main stepper motor part where, with the necessary information fed into, the stepper motor works accordingly. This is the **Application** part.

First/Left Section- Control part

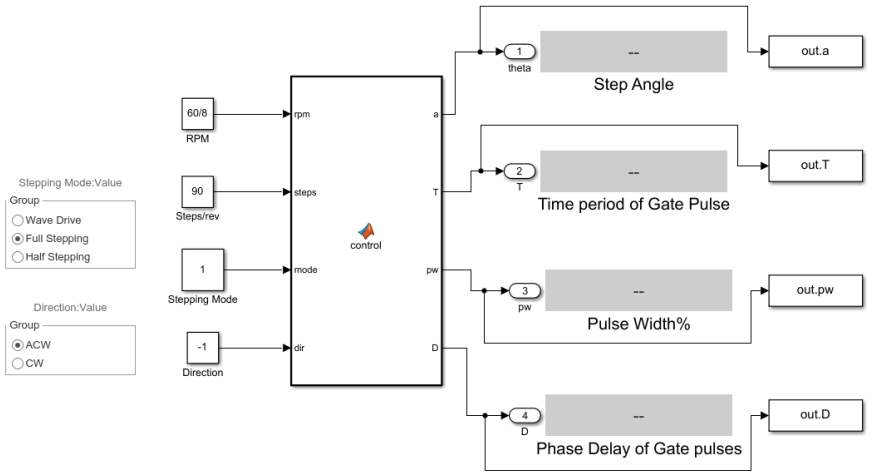


Figure- Simulink model of the control section

The operation of this section:

- Takes inputs
- Calculates stepper motor driving values
- Stores the output values in the workspace

Inputs:

There are 4 inputs in this section.

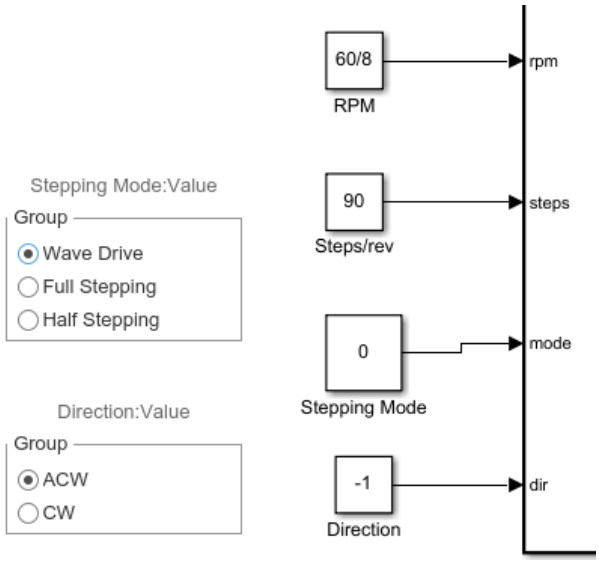


Figure- inputs of first section

1. **RPM:** This is taken as a constant value.
2. **Steps/rev:** This value is also taken as a constant value
3. **Stepping Mode:** This mode options were installed in a Radio button block which was connected to a constant dedicated for this input. There are three options with different states for each of them. The states for the stepping modes were set like-

States:

STATE VALUE	STATE LABEL
0	Wave Drive
1	Full Stepping
2	Half Stepping

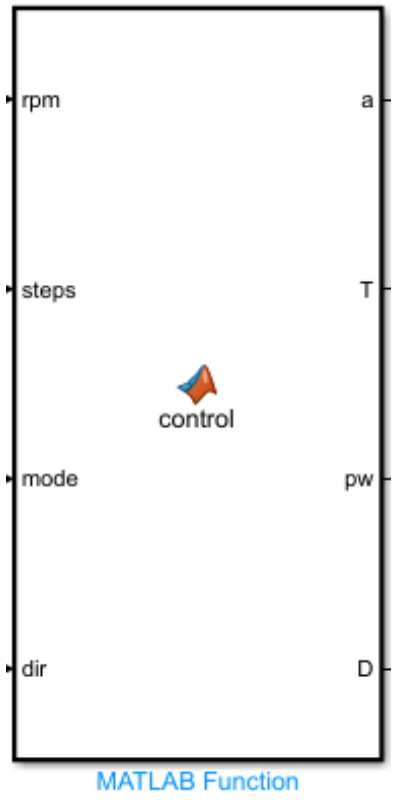
4. **Direction:** This is also installed in a Radio button block with connected & dedicated constant value for the input. The states for two directions were-

States:

STATE VALUE	STATE LABEL
-1	ACW
1	CW

Control part- MATLAB function

This segment will execute all the calculation, decision and give output values.



Control Block

Stepper_motor ▸ MATLAB Function

```
1   function [a,T, pw, D] = control (rpm, steps, mode, dir)
2
3   D = zeros(1,4);
4   w = 0;
5
6   T = 60/rpm;
7
8
9   if mode == 0
10      w = 25;
11  end
12  if mode == 1
13      w = 50;
14  end
15  if mode == 2
16      w = 37.5;
17  end
18
19
20  if dir == 1
21      D(2) = T/4;
22      D(3) = T/2;
23      D(4) = -T/4;
24
25  else
26      D(2) = -T/4;
27      D(3) = T/2;
28      D(4) = T/4;
29  end
30
31  a = 360/steps;
32  T = 60/rpm;
33  pw = w;
34  end
```

Calculation Code

- With a specific RPM value, this block will calculate **Time Period**
- For a specific given steps/rev, this block will find the **Step Angle.**
- According to our given stepping mode, this block will set the **Pulse Width**
- Having a certain direction, this block will set the **Phase Delays** properly.

Outputs & Data Storing in Workspace

There are 4 outputs in this segment-

1. **Step Angle:** For a stepper motor with steps/rev, step angle will be a constant output.
2. **Time period:** For a specific RPM value, time period will be constant.
3. **Pulse Width of the Gate pulses:** For a certain stepping mode, pulse width will be the same for each of the Gate pulses.
4. **Phase Delays:** With a fixed direction, there will be fixed four phase delays for the gate pulses.

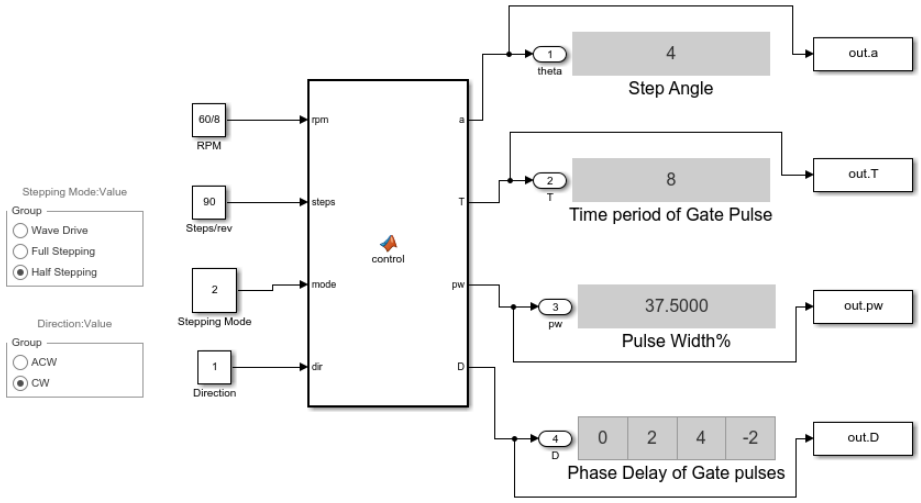


Figure- Fixed output values for certain given inputs.

These outputs are then stored in the workspace via **to workspace** block. The format of these saved data was **structure** mode which stores data in 2-D array with corresponding time samples. As Simulink runs a file repeatedly for a certain end time with certain time gap, the output parameters have values for each of the time samples. So, the outputs a, T, pw, D are not integer/double values, they are array stored with time sample data.

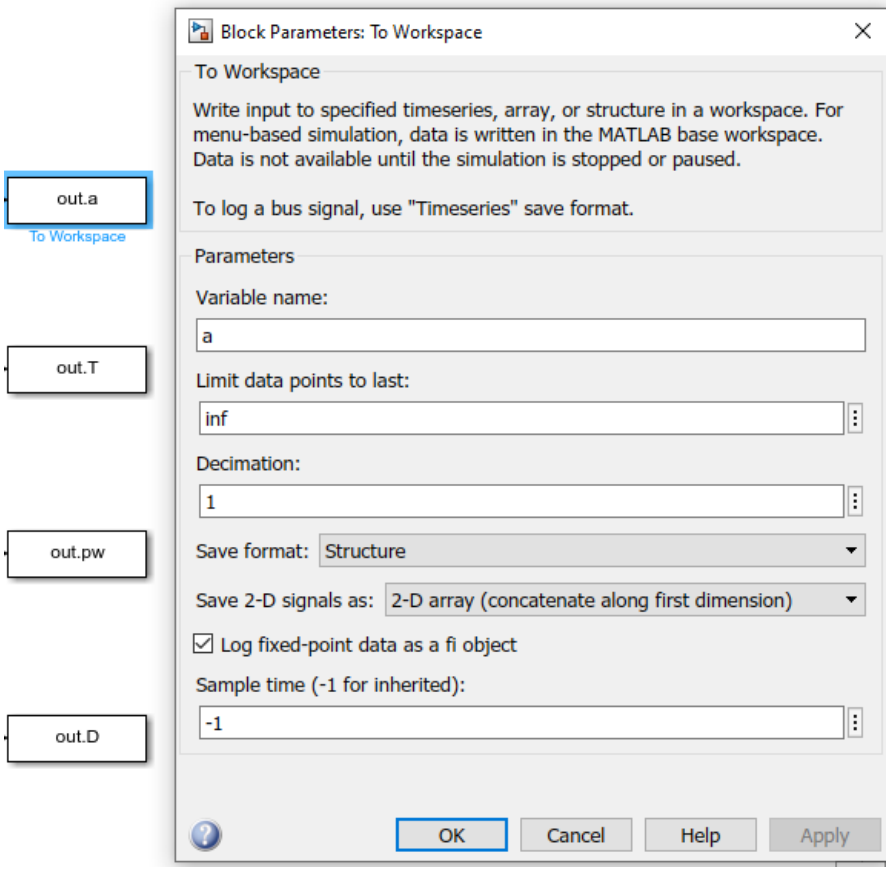
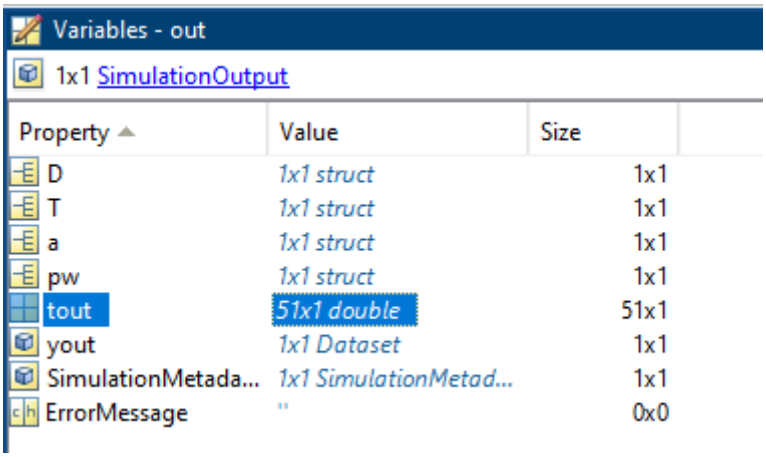


Figure- Data storing via to workspace block and save format.

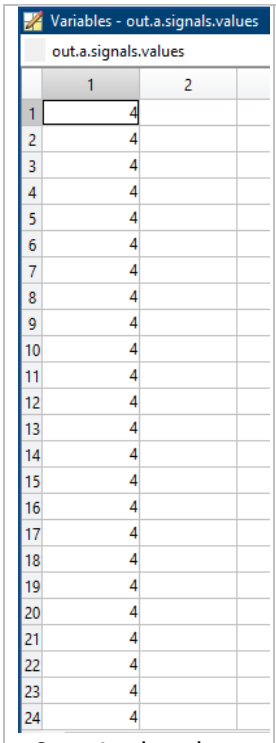
There are in total 51-time samples taken by Simulink file.



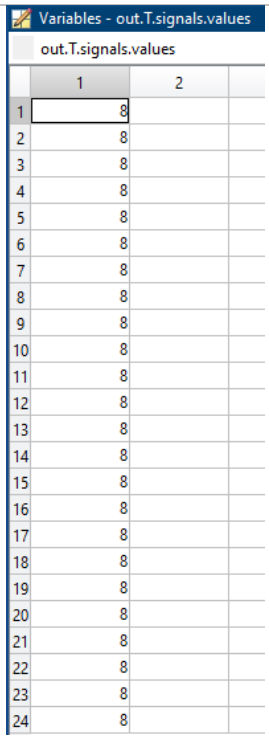
We used 4-phase stepper motor. So, there should be 4 phase delays in the output. The dimension and values of output parameters-

Parameters	Dimension	Values
Step Angle (a)	[1]	51x1 double
Time Period (T)	[1]	51x1 double
Pulse Width (pw)	[1]	51x1 double
Phased Delays (D)	[1,4]	1x4x51 double

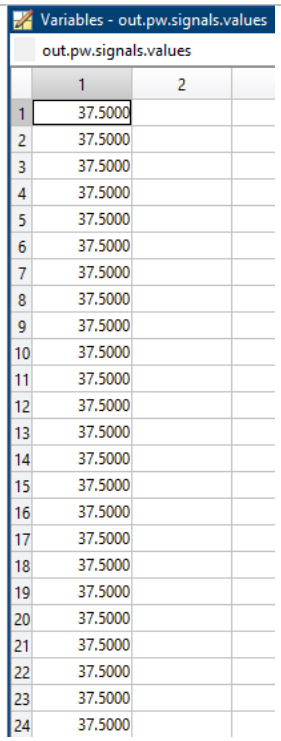
But we didn't change the input with the time sample taken. So, for each of the time sample, the outputs are same. With each of the execution, control block executed the same code with same inputs having same outputs and same values are stored in the array.



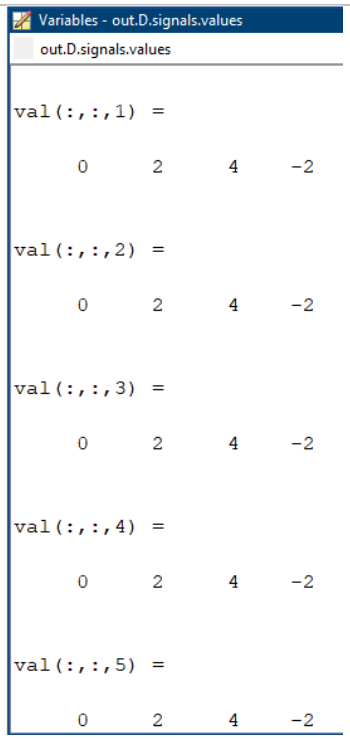
Step Angle values



Time Period values



Pulse Width values



Phase Delay values

The ultimate target of this section was to feed these control values to the second section- stepper motor part. There is no need to feed every data sample from here. Just one single sample is enough to control the stepper motor accordingly. So, transferred data from control section to the motor section are-

1. out.a.signals.values(1,1) - first sample of signal values of a as step angle.
2. out.T.signals.values(1,1) - first sample of signal values of T as time period.
3. out.pw.signals.values(1,1) - first sample of signal values of pw as pulse width.
4. out.D.signals.values(1 , : , 1) - first sample of signal values of D as phase delays.

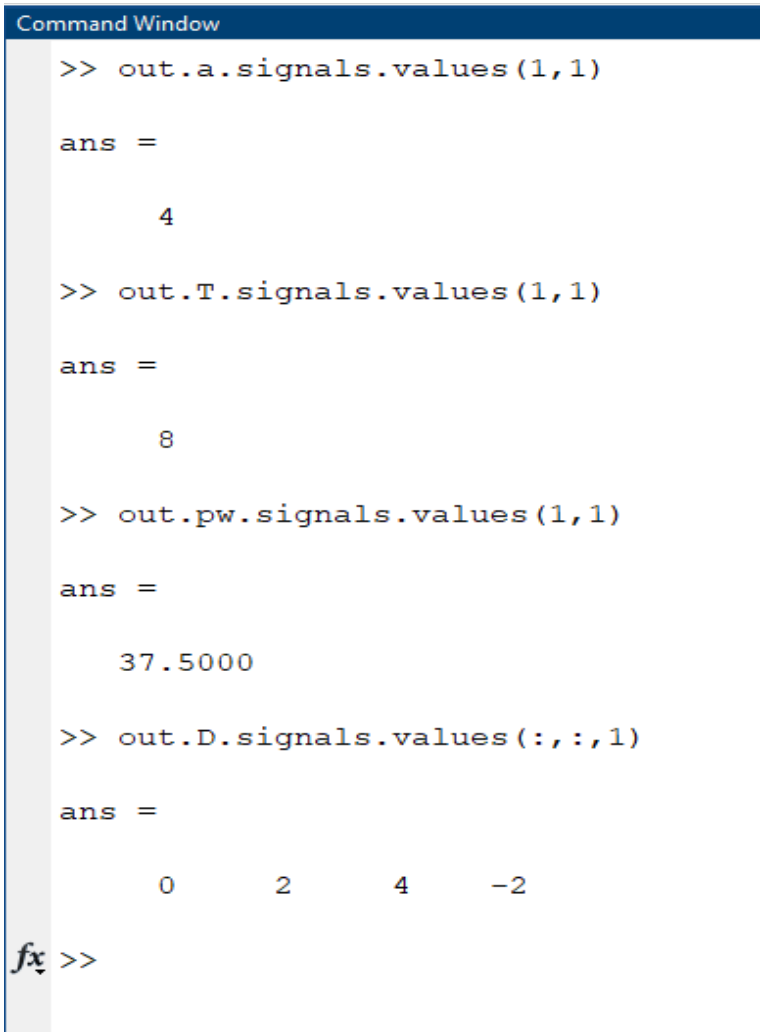


Figure- transferred control parameters for certain input taken in first segment.

One important feature of this project is the second section- stepper motor part cannot run till the data feeding takes place. Since Simulink runs the file as a whole, we must comment out the second section- stepper motor part before simulating the first section. So, we must execute the control section first, store data in the workspace and run the stepper motor part afterwards according to our workflow.

Second/Right section- Stepper Motor Part

This is where the stepper motor works with proper control parameters fed to it.

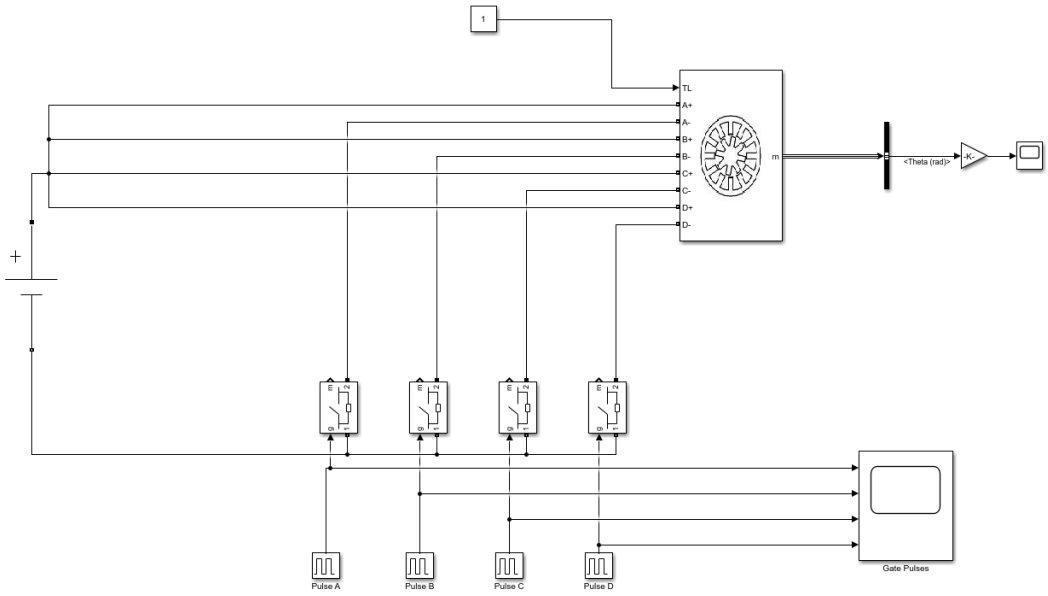


Figure- Simulink model of the stepper motor section

The operation of this section:

- Takes control parameters from workspace as input in pulse generator & stepper motor block
- Executes pulse generator and then stepper motor.
- Plots the gate pulses and rotor position in the scopes.

This section consists of several blocks:

Stepper Motor Block:

We took the built in **Unipolar Stepper Motor** block here in Simulink for this project. The parameters of this stepper motor block were-

- Motor type- **Variable Reluctance**
- Phases- **4 phases**
- Maximum & Minimum winding inductance (H)- default value
- Winding resistance (Ohm)
- Step angle- this data is being fed from the workspace with **out.a.signals.values(1,1)**
- Total inertia (kg.m.m) & friction (N.m.s)- default values
- Initial speed (rad/s) & position (degrees)- default value, which is zero otherwise defined, is taken
- Sample time (-1 for inherited)- default value –1 was taken.

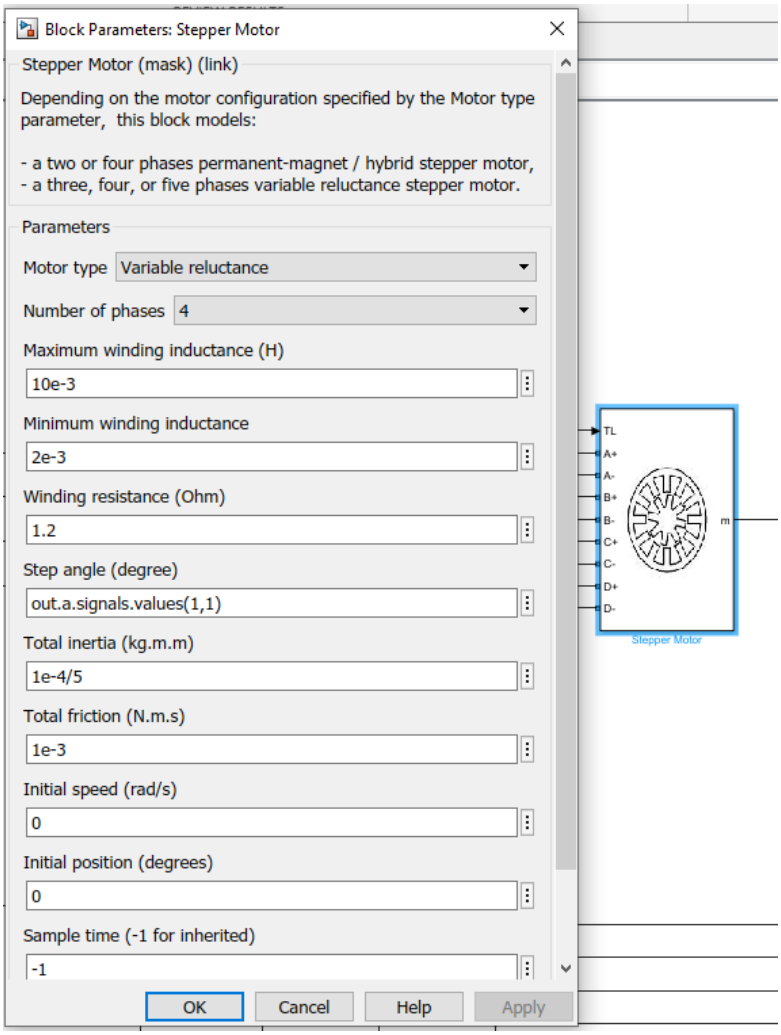


Figure- Stepper motor block parameters.

So, this motor block has two connections (input)

1. **TL**- Mechanical load torque, in N.m. TL is positive in motor operation and negative in generator operation. We set the value at 1 for motor operation in full load condition.
2. **4 phases**- 4 coils with 4 positive and negative ports.

The positive ports of the coils were connected to the positive end of a 25 Volt DC source which will excite the coils.

The negative ports of the coils were connected to the negative end the DC voltage source via switches. These switches are being controlled by the gate pulses.

Ultimately, the coils get excited according to our given/predefined gate pulses with the controlled switching of the coil-source connections. This is how the stepper motor is controlled. We control the switching of the coil- voltage connections through gate pulses so the controlled excitation of the coils results in controlled rotor movements- thus, the ultimate motor control is established.

Gate Pulse Generation

Initially, we execute the first section which loads the control parameter values of the stepper motor in the workspace. For pulse generation, we used built-in pulse generator block where we fed sufficient data from the workspace.

For a certain stepping mode and direction

- Time period of each of the gate pulses are same.
- Pulse width of each of the gate pulses are same.
- Phase delay will be different.

Block parameters of the pulse generators-

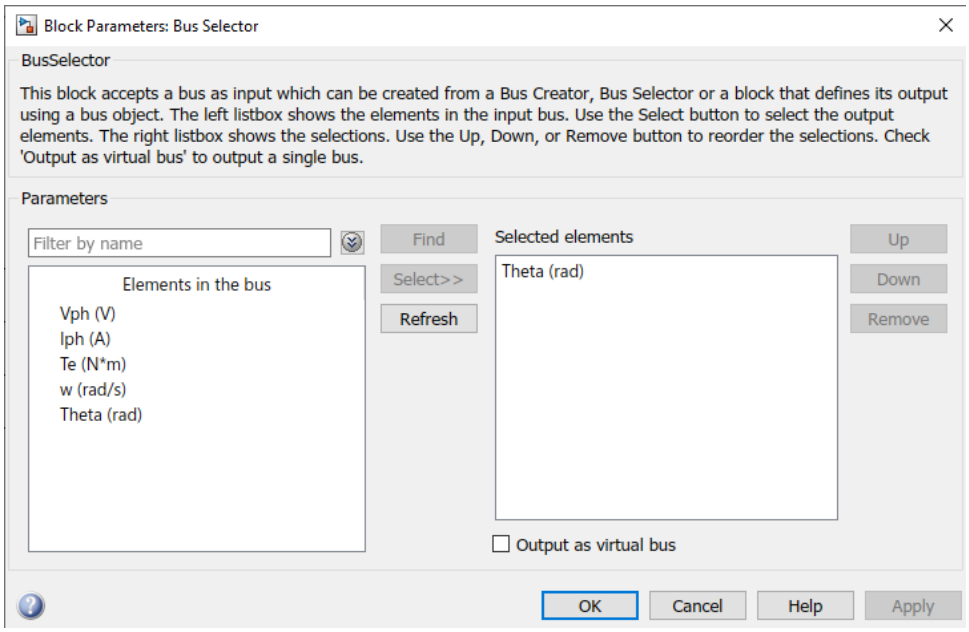
Pulses	Time Period	Pulse Width	Phase Delay
A	out.T.signals.values(1,1)	out.pw.signals.values(1,1)	out.D.signals.values(1,1,1)
B			out.D.signals.values(1,2,1)
C			out.D.signals.values(1,3,1)
D			out.D.signals.values(1,4,1)

Stepper Motor Output

The Simulink output of the block is m which a vector containing five signals-

Signal	Definition	Units	Symbol
1	Phase voltage	V	V_{ph}
2	Phase current	A	I_{ph}
3	Electromagnetic torque	N.m	T_e
4	Rotor speed	rad/s	w
5	Rotor position	rad	Theta

We don't need to observe all the signals. So, we used a bus selector to demultiplex the signals and took only one output to observe- **Rotor position.**



As the rotor position, theta, is supplied in radian unit, we used a gain of 180/pi to convert it into degree unit.

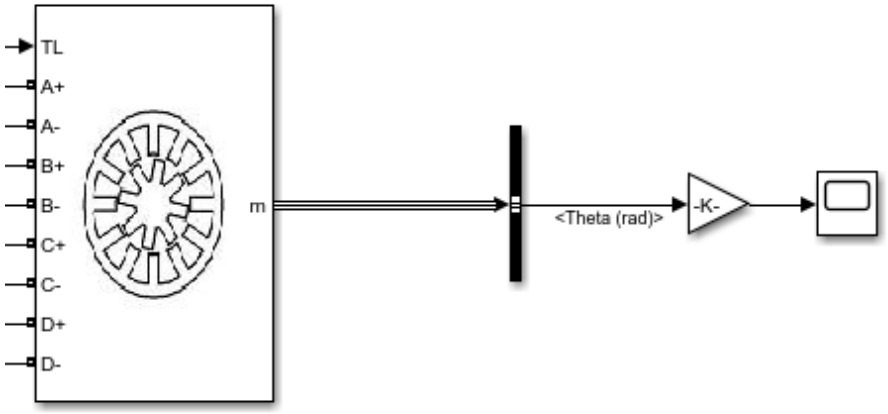


Figure- Stepper motor output

Results & Graphs

For each of the system designed with specific inputs, we will observe two results/graphs

- Generated gate pulses
- Rotor position

There are three stepping mode, two direction

- **Wave Drive mode**
- **Full Stepping mode**
- **Half Stepping mode**

- **ACW- backward direction**
- **CW- forward direction**

These modes can be shown under two initial conditions-

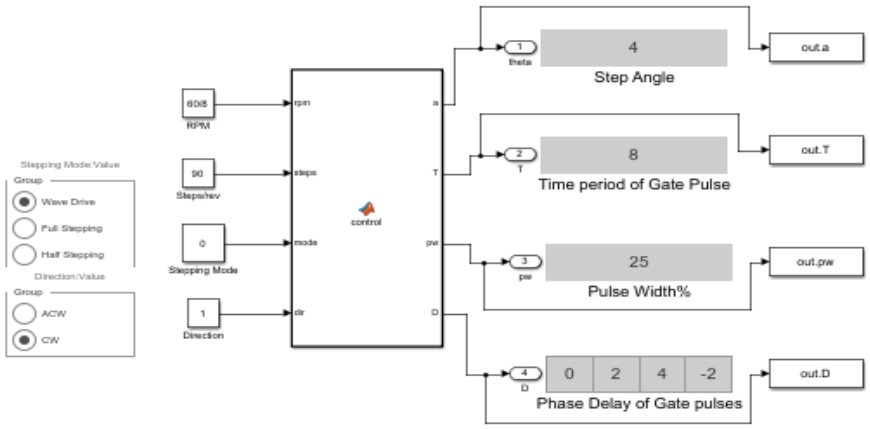
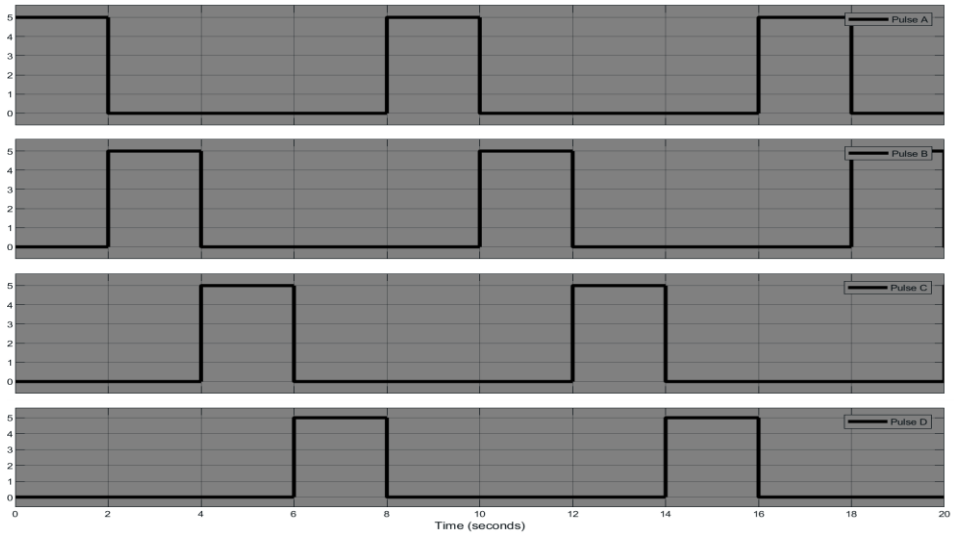
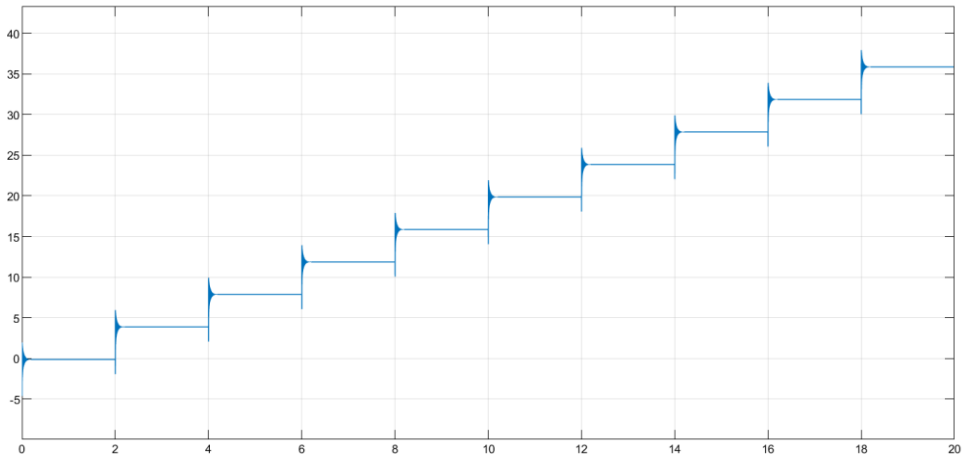
- With zero initial rotor speed and zero initial rotor position
- With non-zero, fixed initial rotor speed & position

So, for these two initial conditions, stepper motor result and output are shown for different stepping modes and random directions.

Zero Initial Position

#Wave Drive mode with CW direction

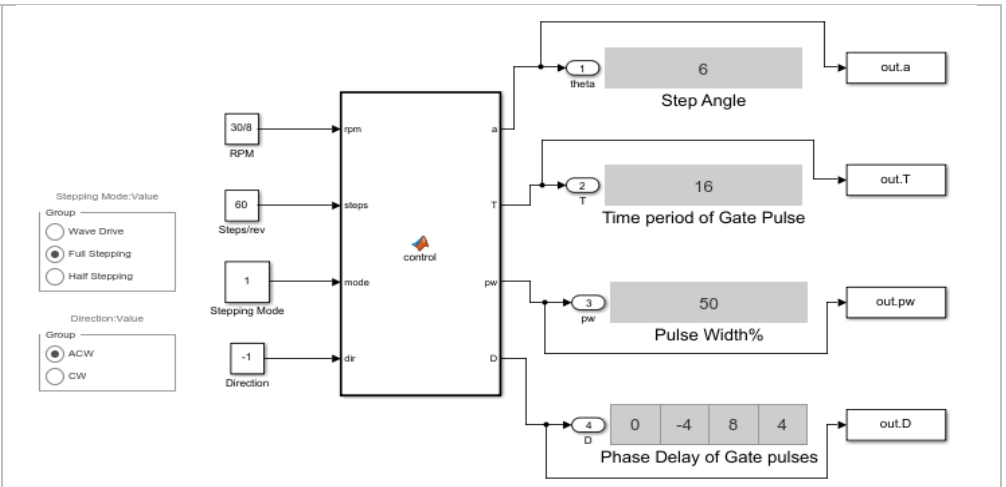
The inputs are taken- **RPM = 60/8 rev/min** & **Steps/rev = 90**

Outputs-	
Gate pulses	
Rotor Position	

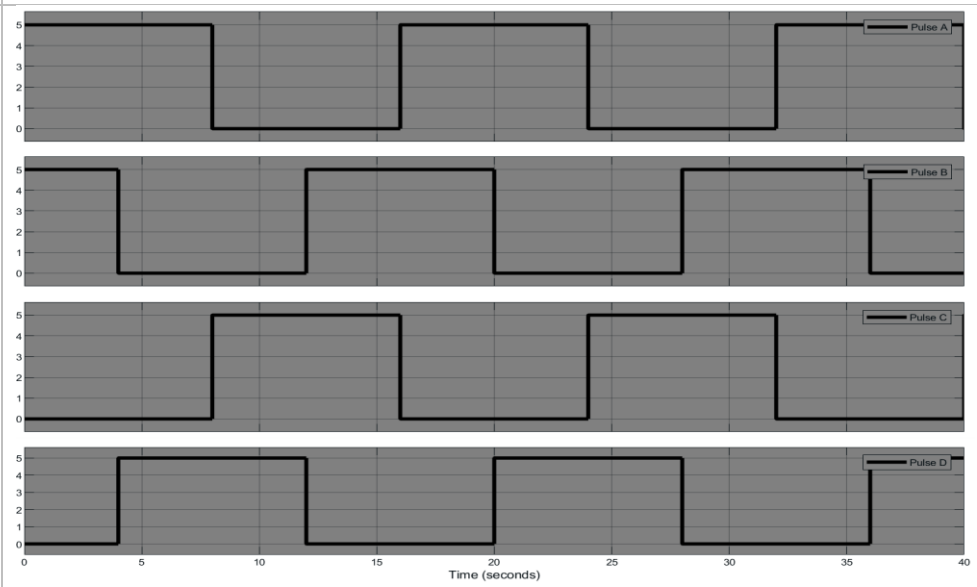
#Full Stepping mode with ACW direction:

The inputs are taken- **RPM = 30/8 rev/min** & **Steps/rev = 60**

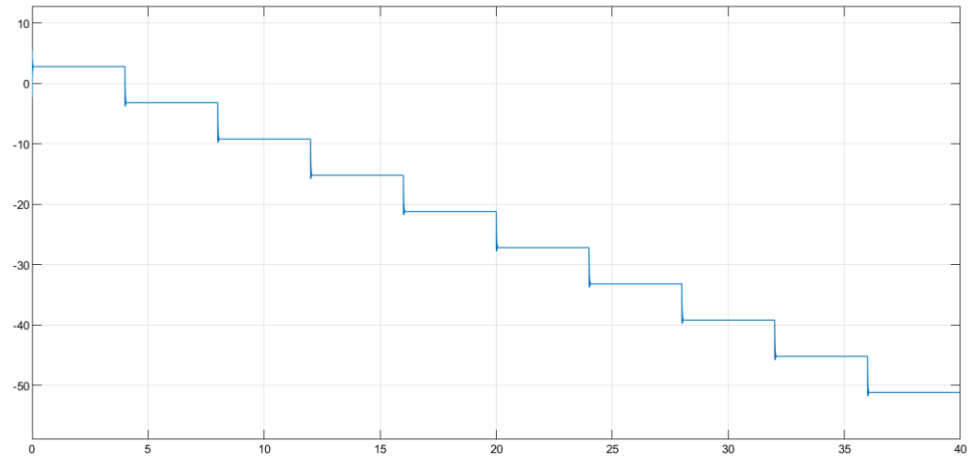
Outputs-



Gate pulses



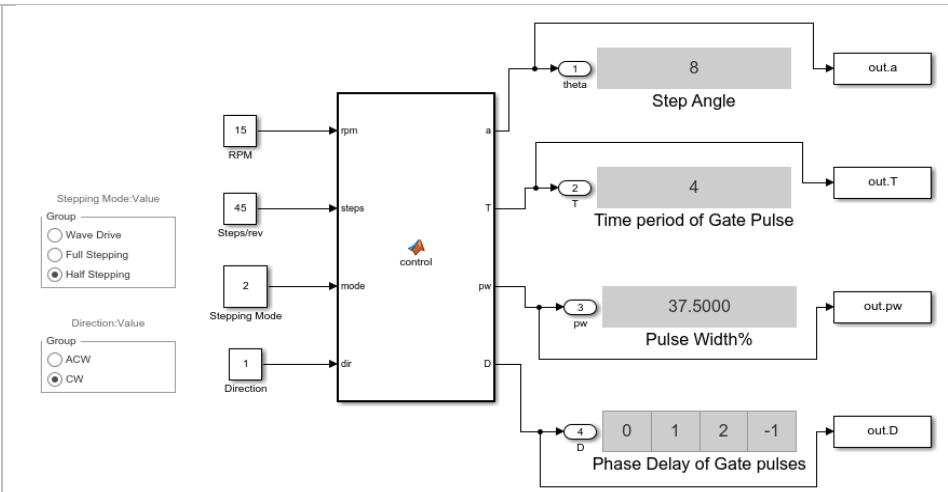
Rotor Position



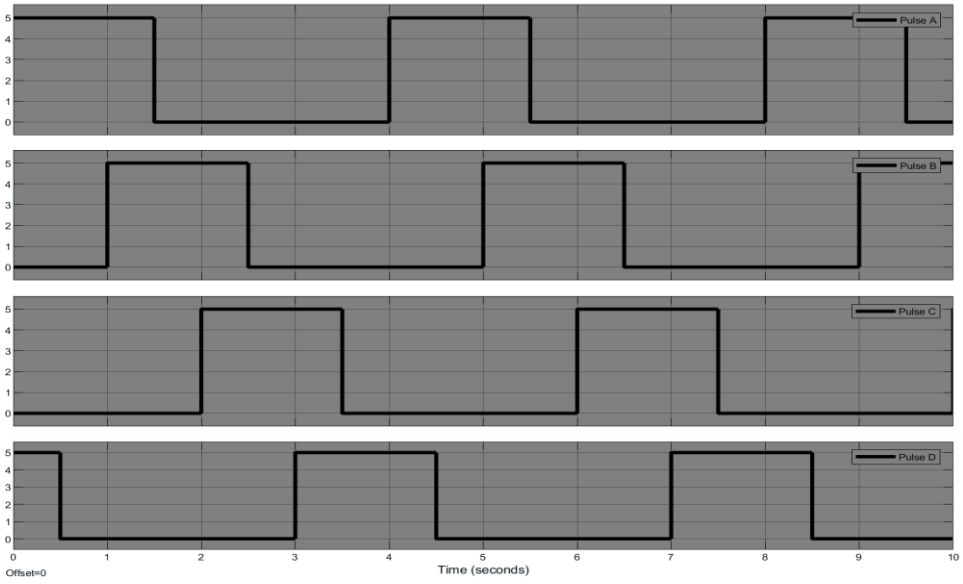
#Half Stepping mode with CW direction:

The inputs taken are- **RPM = 15 rev/min** & **Steps/rev = 45**

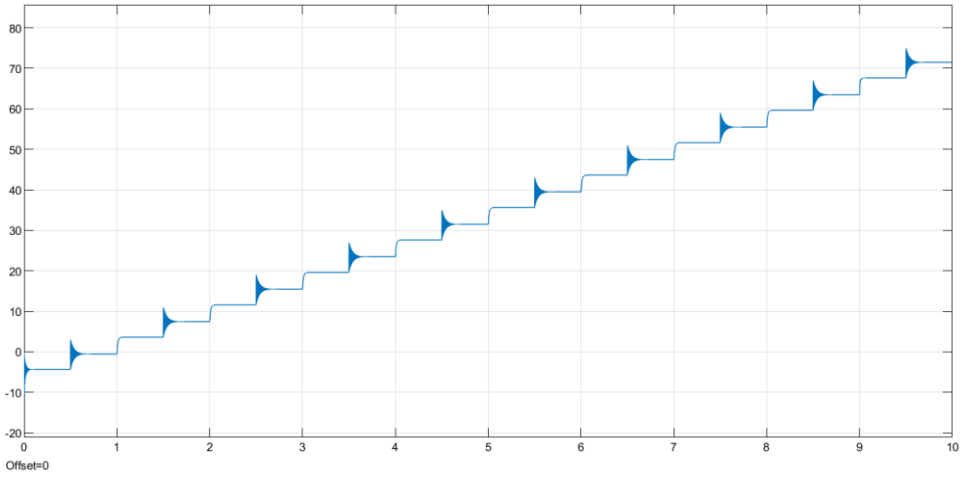
Outputs-



Gate pulses



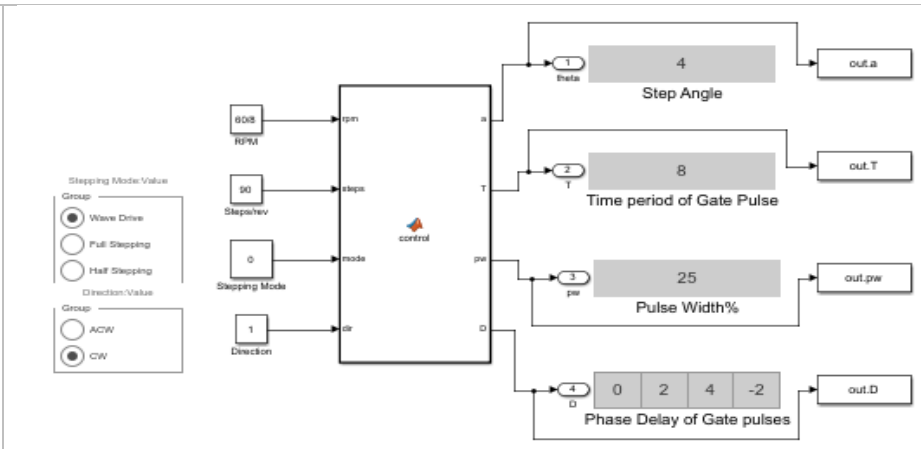
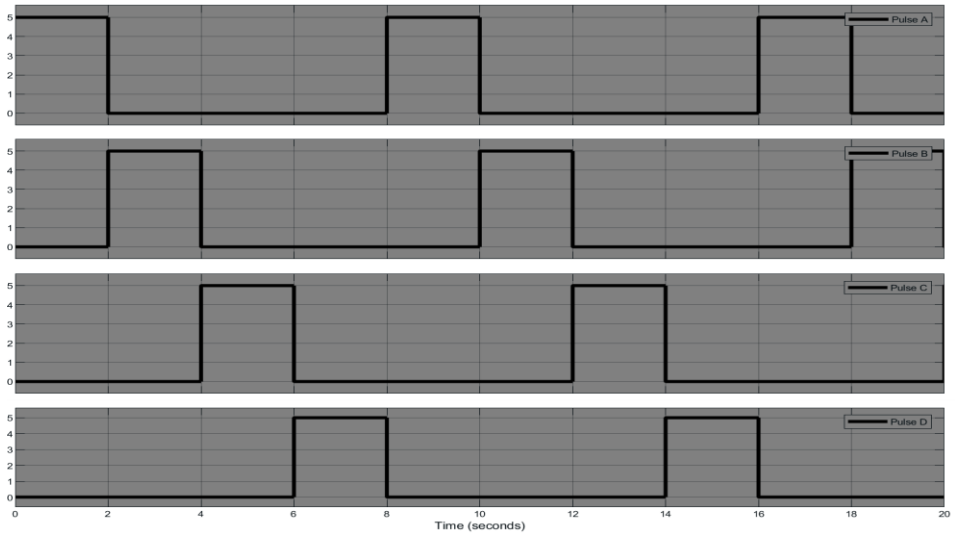
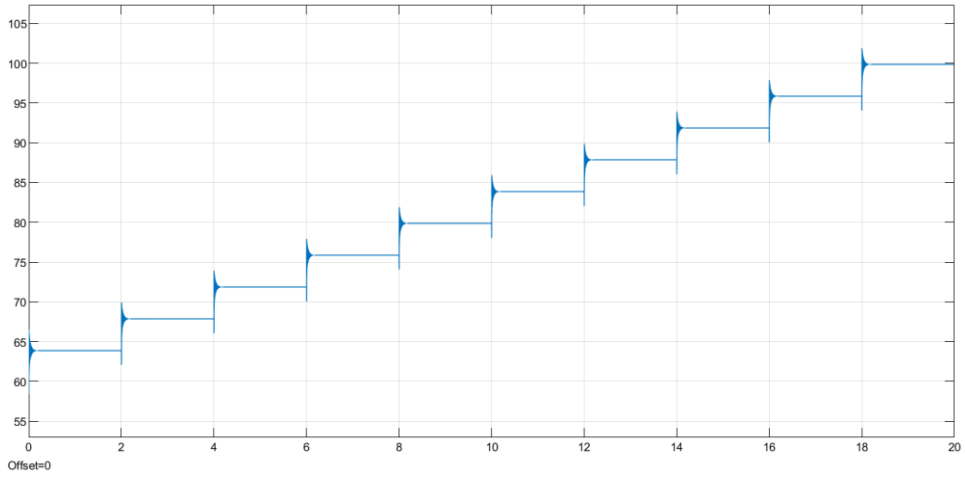
Rotor Position



With Initial Position- 60 degrees

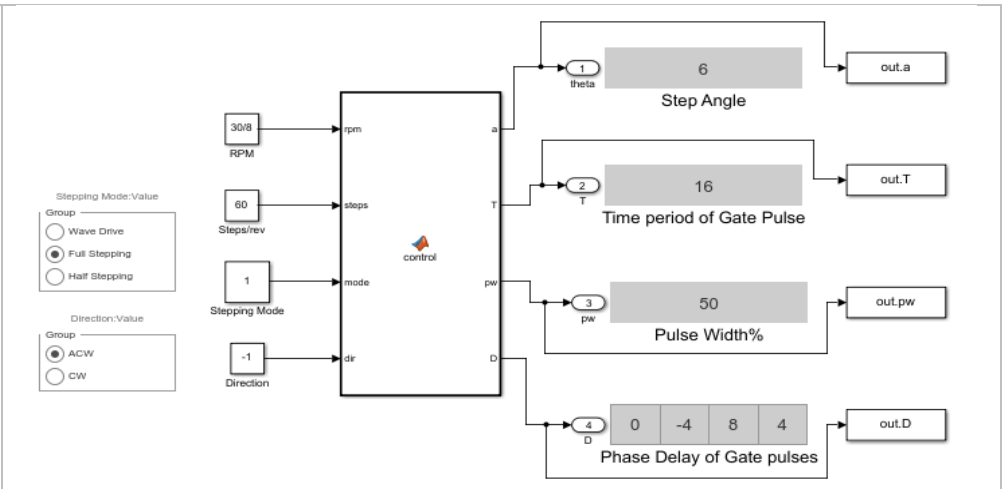
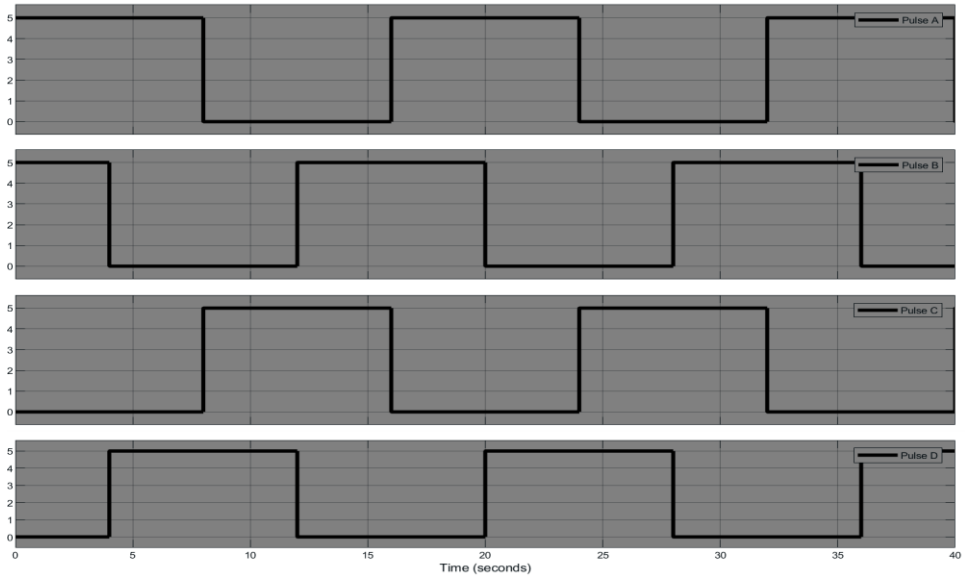
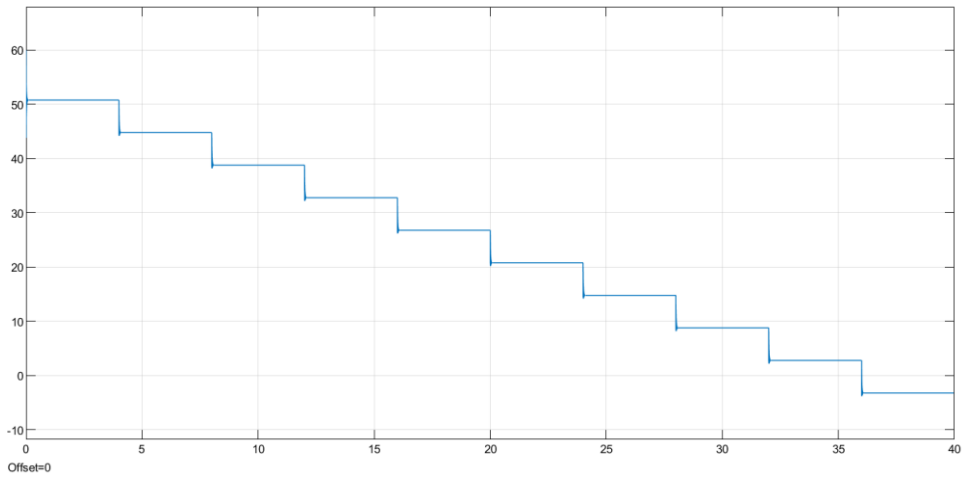
#Wave Drive mode with CW direction:

The inputs are taken- **RPM = 60/8 rev/min** & **Steps/rev = 90**

Outputs-	
Gate pulses	
Rotor Position	

#Full Stepping mode with ACW direction:

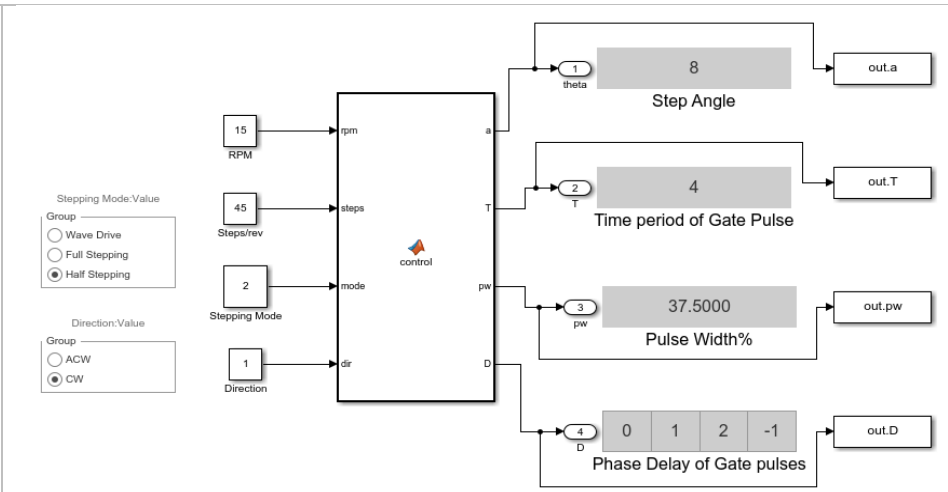
The inputs are taken- **RPM = 30/8 rev/min** & **Steps/rev = 60**

Outputs-	
Gate pulses	
Rotor Position	

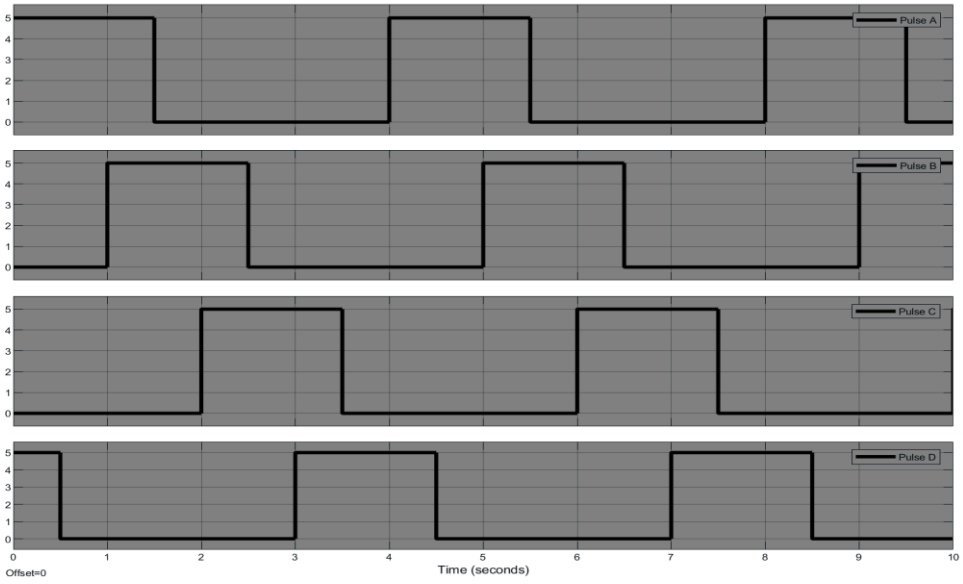
#Half Stepping mode with CW direction:

The inputs taken are- **RPM = 15 rev/min** & **Steps/rev = 45**

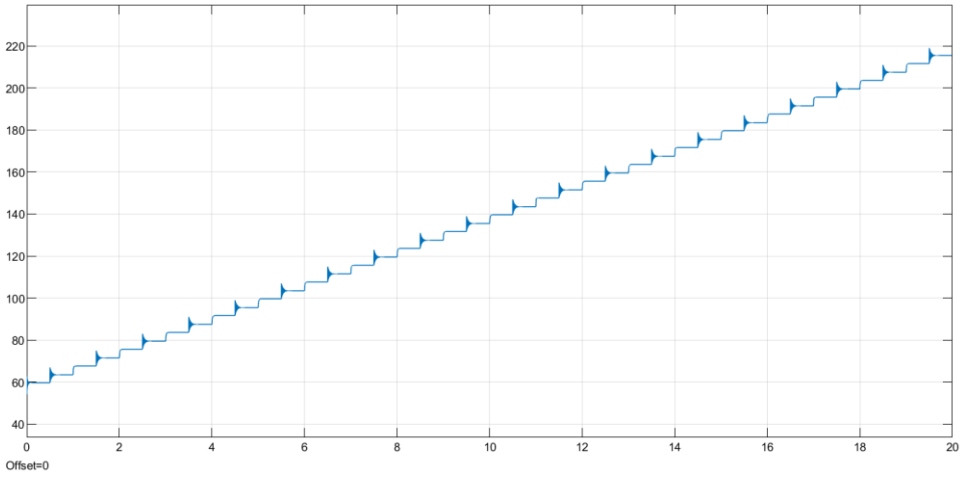
Outputs-



Gate pulses



Rotor Position



Result Analysis

From the graphs of outputs, we can conclude that-

- The basic control output of the control box- MATLAB function operated successfully. Each time, the calculated control parameters were correct.
- There is no error in pulse generation. For each of the cases, gate pulses were generated accurately as they were expected.
- For zero initial rotor position, the rotor position matched perfectly with expectation of theoretical knowledge. They all started from zero position.
- For fixed initial rotor position, the theta outputs were like the expectation with slight deviation offset values.
- There were certain amounts of overshoot in the rotor position responses for each of the cases. This is because the rotor has inertia for each of the steps it took while rotating.

Limitation

However, there were some limitations in this project such as-

- Hardware implementation was not possible.
- Detailed circuit could have been designed instead of the built-in blocks from Simulink Library. MSFET and detailed transformer could have been used to perform this project work.
- As unipolar stepper motor block was directly used from the Simulink library, implementing bipolar stepper motor control in the same file was not possible.
- There was considerable amount of overshoot in the rotor position response which could have been mitigated by designing feedback system with proper combination of PID controller / Lag-Lead compensator.
- Analysis of load inertia, resonance effect, overheating in the stepper motor was not feasible to perform.

Conclusion

The target of this Simulink-based project was to establish the control of stepper motor with inputs like- angle, speed and direction which was performed accurately. With proper opportunities, this project could have been implemented in hardware using microcontroller. But the simulation provided insights about the operation and features of stepper motor in different modes and all.