Faster Than Real-Time (FTRT) Dynamics Simulation

Generic Wind Turbine Model Type 3

Authors

Ming Chen, Xinyi Yu, Sida Liu

Instructor

Dr. Alexander J. Flueck

(Illinois Institute of Technology, Electrical and Computer Engineering Department, Chicago, IL)

8.21.2014

IIT Armour College of Engineering
ILLINOIS INSTITUTE OF TECHNOLOGY

ABSTRACT

As science and technology grow dramatically nowadays, the demand of energy has gone through a significant increment. Fossil energy plays a prominent role in human society, but damage to the ecological system has always been an obstacle of our development. New energies, including wind energy, have been more welcomed by people since they do much less harm to the environment. Therefore, more and more wind turbines are built and the stability of generating electricity becomes extremely important to the upper network. Our final goal is to provide actionable information to operators during power system disturbances and cascading outages. This paper focuses on the dynamic modeling of Type 3 Generic Wind Turbine, with each model being explained in detail. SIEMENS has documents describing the models including Generator, Electrical, Mechanical and Pitch, but not accurate enough. During the research effort put into this paper, we have figured out the detailed parts inside each model, with the help of PSS®E and MATLAB Simulink. It also discusses the remaining problems of this type of wind turbine and describes future work such as translating the model into C code for completing the "faster than real time" simulator.

I. INTRODUCTION

A. Project Description

Develop a high fidelity "faster than real-time" dynamics simulator capable of predicting complex, large-scale power system behavior based on real-time measurements. Leverage recent and proposed mathematics and computational advances (e.g., PETSc linear solvers, nonlinear solvers, time-stepping algorithms, memory management, multi-core, many-core and GPU processing) to improve speed of dynamics simulations. Leverage recent modeling and simulation advances (e.g., new three-phase unbalanced network models, single-phase induction motor models, protection system models) to improve fidelity of dynamics simulations.

The goal of "faster than real-time power system dynamics simulation" is to provide actionable information to operators during power system disturbances and cascading outages. The project is developing high performance computer modeling and simulation software for predicting the effects of disturbances faster than real-time. Based on the predictive capabilities of this research, operators will be able to respond before the full effects of a cascading outage are realized, thereby avoiding wide spread blackouts.

B. Background of Type 3 Generic Wind Turbine Model

The WT3 PSS®E wind turbine stability model was developed to simulate performance of a wind turbine employing a doubly fed induction generator (DFIG) with the active control by a power converter connected to the rotor terminals.

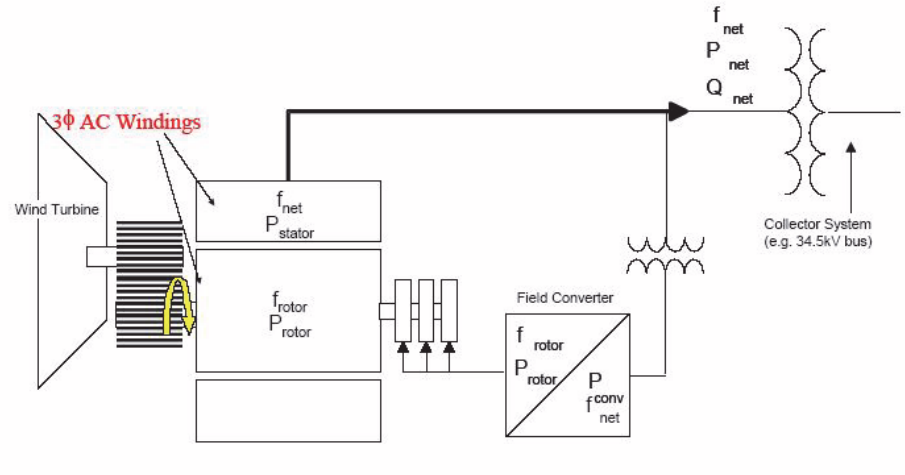


Figure 1-1. Doubly Fed Induction Generator with the Active Control by a Power Converter Connected to the Rotor Terminals

The model is to be used in studies related to the integration of Wind Turbine Generators (WTG) in an Electrical Power System.

There are four components in this model:
- WT3G: generator/converter model
- WT3E: electrical control model
- WT3T: mechanical control (wind turbine) model
- WT3P: pitch control model.

In each component model, there are many control blocks that duplicate integrators, and the controller limits also play an integral role in the dynamic performance of this wind turbine. In addition, the behavior of such models is governed by interactions between the continuous dynamics of state variables, and discrete events associated with limits.

The interaction among generic wind models are shown in Figure 1-2 below:

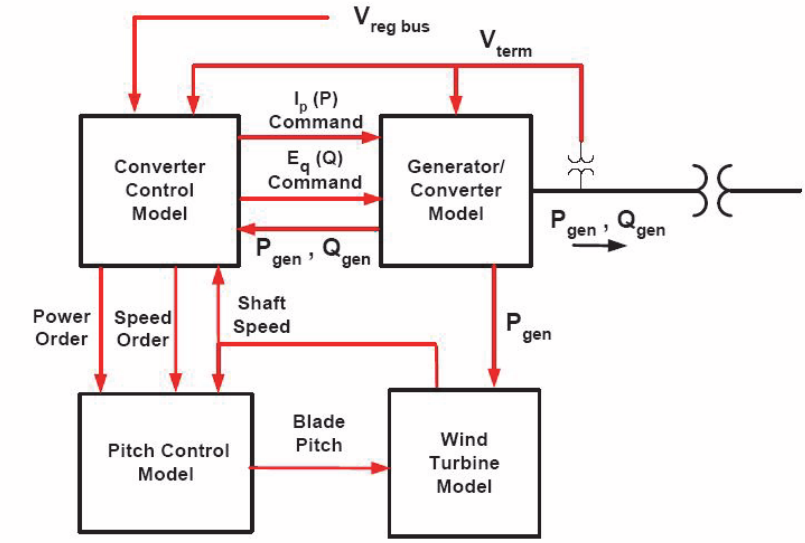


Figure 1-2. Interaction among Generic Wind Models

Since Type 3 Generic Wind Turbine Model are currently the dominant technology for new wind farm developments, they receives considerable attention and it is worthwhile to analyze it.

II. RESEARCH METHOD

The control device model generic wind turbine type 3 responses were benchmarked in a small test system against a commercial software tool for offline power system dynamics simulation. This small test system we used was MATLAB. At the beginning, we implemented WT3 into MATLAB Simulink based on the description in PSS®E documentation. Then it was obvious to us that PSS®E hid a great deal of details in its documentation. Reading online documents helped us a lot in figuring out all of those hidden things. Our team verified our assumption and validity of online resources by comparing the simulation result between PSS®E and our modified MATLAB Simulink models. Varied sets of data for PSS®E power flow setup help to increase the model's and program's reliability and performance.

III. RESULTS

A. Clarification for Hidden Parts of WT3 model in PSS@E

1. Doubly-fed induction generator

There are two different generator/converter models available in PSS®E, namely WT3G1 and WT3G2. The WT3G2 model, which is recommended for new dynamic setups, includes improvements in the original WT3G1. The original WT3G1 model is being retained for reasons of backward compatibility.

We explored both WT3G1 and WT3G2 model and figured out some blocks which lacked of important details by studying relevant documents online as well as building Matlab Simulink models to compare PSSE results with Simulink plots, verifying our assumptions about unknown block details.

a. WT3G1

Both WT3G1 model and WT3G2 model can work with electrical control model WT3E1, but the dynamic data input file should be changed a little bit. When WT3E1 is used with the WT3G1 model, it is recommended that ICON (M+2) be set to 1.

WT3G1 model diagram in PSS®E is shown in Appendix A.1.1 and the final model we established in Simulink is shown in Appendix C.1.1.

The first thing we do is to determine the location of each state in WT3G1 model. Since the diagram fails to show the four state (from K to K+3) in WT3G1 model, we use the descriptions in documentation to locate each one and there order.

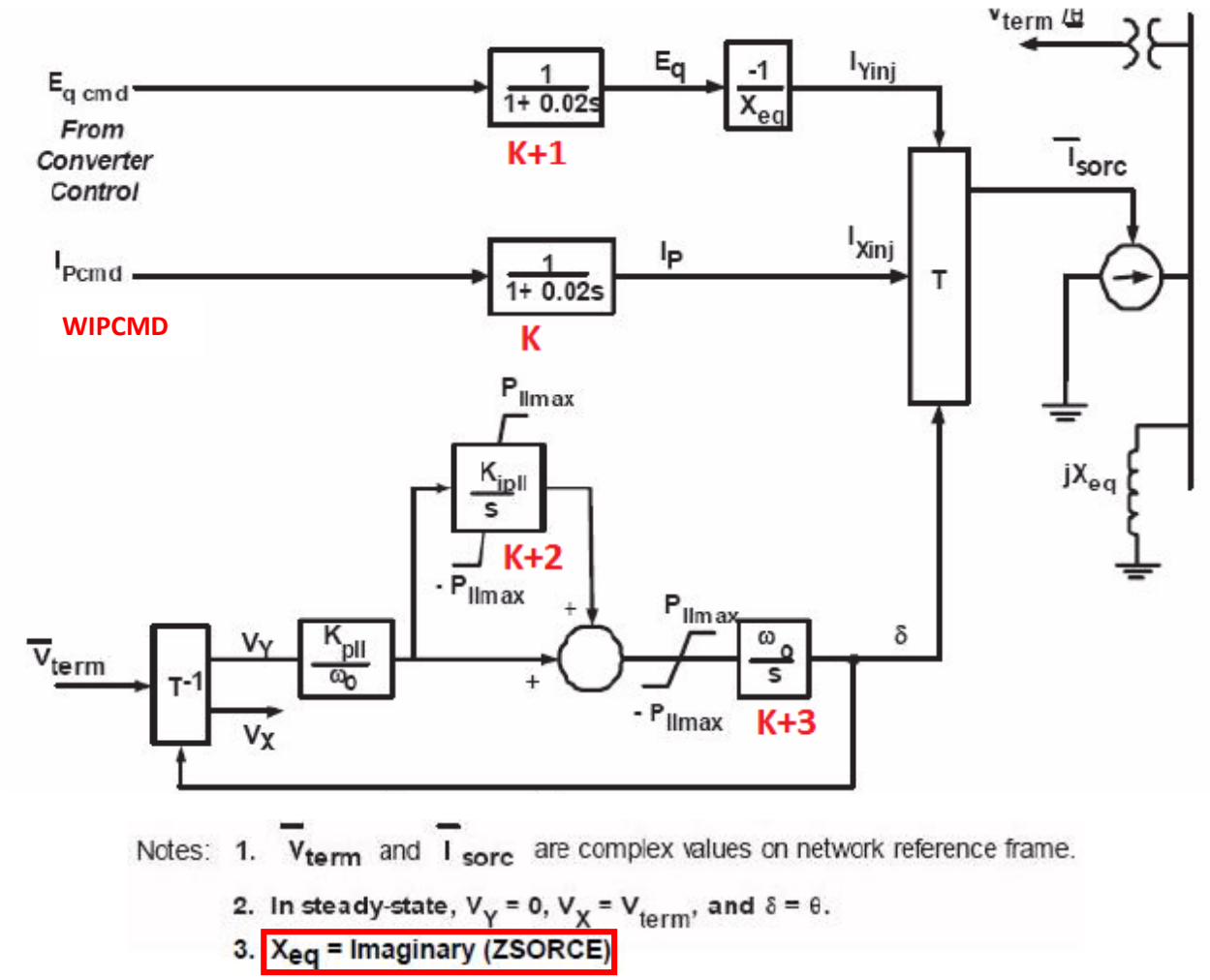


Figure 3-1. PSS®E WT3G1 Model

Then we also modify the Xeq value in input file. Because it shown in documentation that the Xeq equals to the imaginary part of ZSCORE, and in our power flow set up raw file, the imaginary part of ZSCORE is 0.1, so we modify Xeq to 0.1 from 0.8, which is provided from dynamic data input file.

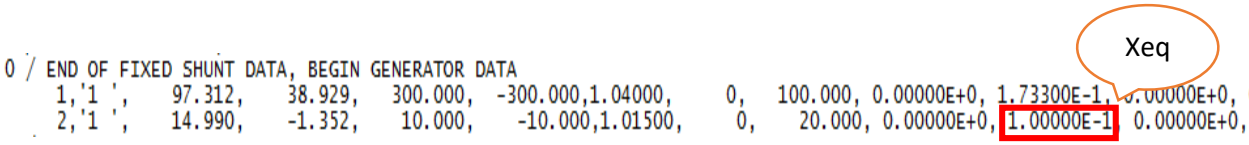
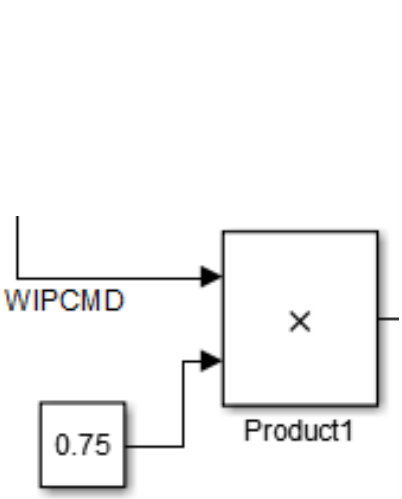


Figure 3-2. Xeq Value in Power Flow Generator Data in Raw file

One thing needed to be clarify is the input in this model. Since the input of WT3G1 and WT3G2 are same, and in WT3G2 it shows that the input Ipcmd is from the output of electrical model, which is WIPCMD, so in WT3G1, Ipcmd means WIPCMD. In addition, since WIPCMD is in system base, so we need to change it to machine base before using it as an input. Therefore, we add a gain of 0.75 after WIPCMD in our Matlab model, shown in Figure 3-3.

The biggest mystery we solved in this model is the T block and T^{-1} block. There is a figure showing the relationship between I_{Yinj}, I_{Xinj} and δ:



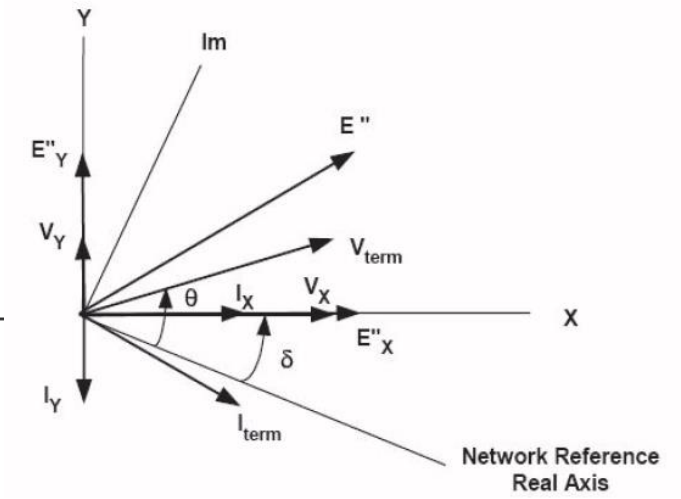


Figure 3-3. WIPCMD Base Change in WT3G1 **Figure 3-4. Relationship of Network Parameters**

According to this diagram, we find out that the T block and T^{-1} block are operation of complex voltage and current. Equations are shown below:

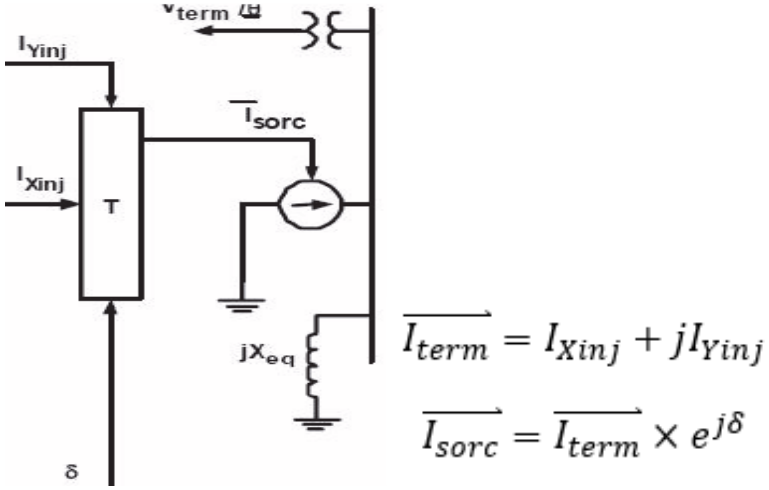
$$\overrightarrow{V_{new}} = (V_{term} \times e^{j\theta})e^{-j\delta}$$

$$\overrightarrow{V_x} = Re\{\overrightarrow{V_{new}}\} \text{ and } \overrightarrow{V_y} = Im\{\overrightarrow{V_{new}}\}$$

$$\overrightarrow{I_{term}} = I_x + jI_y \text{ and } \overrightarrow{I_{sorc}} = \overrightarrow{I_{term}} \times e^{j\delta}$$

$$Pelec + jQelec = (Eterm \times e^{jAngle})I^*_{sorc}$$

For T block:



$$\overrightarrow{I_{term}} = I_{Xinj} + jI_{Yinj}$$

$$\overrightarrow{I_{sorc}} = \overrightarrow{I_{term}} \times e^{j\delta}$$

Figure 3-5. T Block in PSS®E Documentation and the Corresponding Equations

We build MATLAB model as Figure 3-6, in which T block operation is realized.

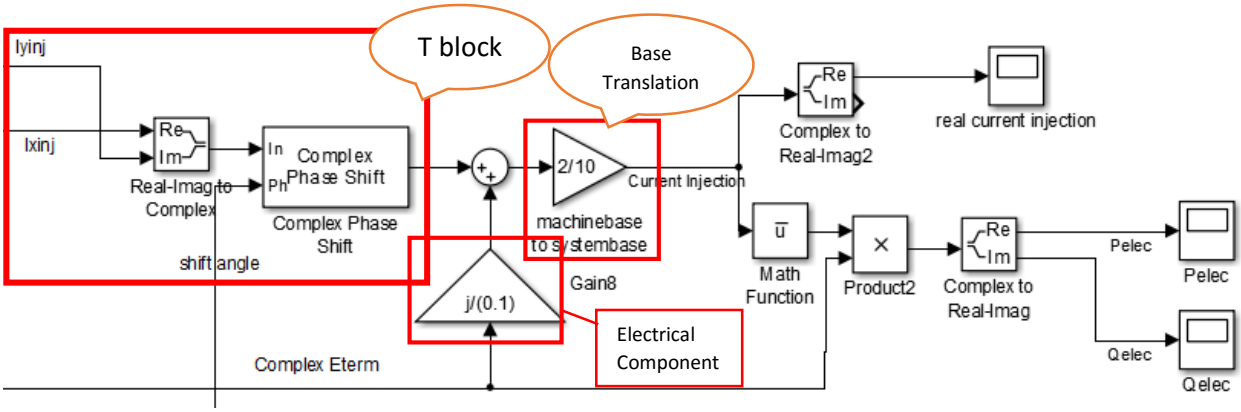


Figure 3-6. MATLAB Implementation of T block and Connecting to Network

The rest part is used to finish the operation of: $\left(Eterm \times e^{jAngle}\right)I^{*}_{sorc} = Pelec + jQelec$

We also modify the unit in this part. We multiply complex Eterm with j/(0.1) because the electrical component in the model (Figure 3-7).

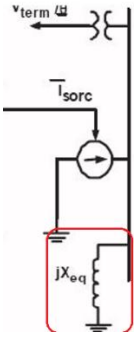


Figure 3-7. jXeq in PSS®E Documentation

After the summation, Gain 8 does the base translation from per unit in machine base multiplied by 20 (MBases value), then divided by 100 back to per unit in system base. The translation between machine base and system base is a tricky thing in PSS®E and researches have to be careful in implementation. Moreover, the angle of Eterm is in radius instead of degree. Therefore we also modify its unit before using it.

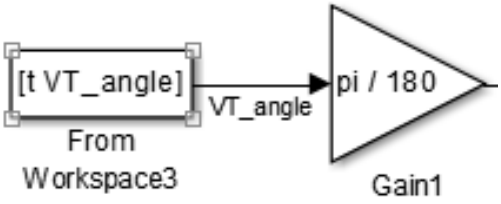
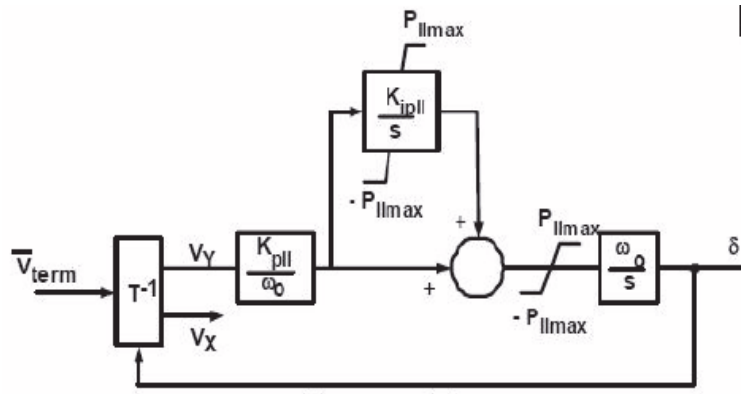


Figure 3-8. Degree to Radius Translation

For T^{-1} block:



$$\overrightarrow{V_{new}} = (V_{term} \times e^{j\theta})e^{-j\delta}$$

$$\overrightarrow{V_x} = Re\{\overrightarrow{V_{new}}\} \text{ and } \overrightarrow{V_y} = Im\{\overrightarrow{V_{new}}\}$$

Figure 3-9. T^{-1} Block in PSS®E Documentation and the Corresponding Equations

We build MATLAB model as Figure 3-10, in which T^{-1} Block block operation is realized.

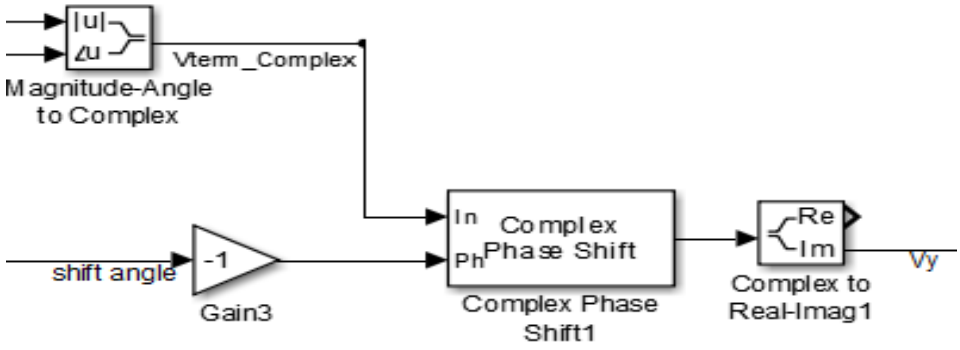


Figure 3-10. MATLAB Implementation of T^{-1} Block

State K+ 2 has a non-windup limit as shown in Figure 3-11. More details of non-windup limit are shown in other sections.

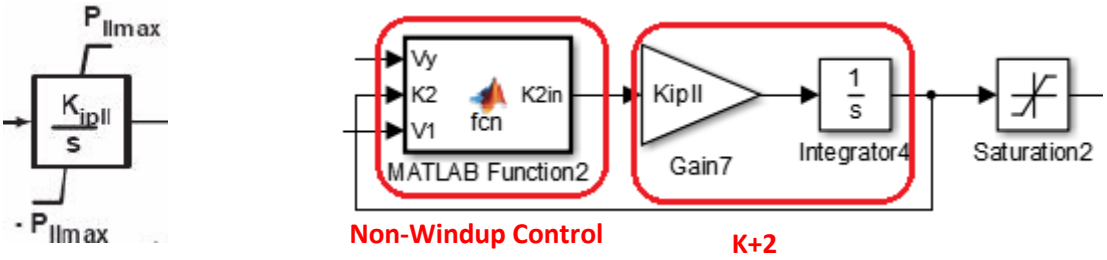


Figure 3-11. Non-Windup Limit in State K+2

b. WT3G2

WT3G2 model is more complicated than WT3G1 model and there are three mysterious blocks needed to be clarified.

WT3G2 model diagram in PSS®E is shown in Appendix A.1.2 and the final model we established in Simulink is shown in Appendix C.1.2.

The first mystery we solved is the V-LVPL function block and the state it controls. We know the constant values of the three variables shown inside it, but the changes of LVPL at the time V less than Vlvpl1 and larger than Vlvpl2 are not clear enough.

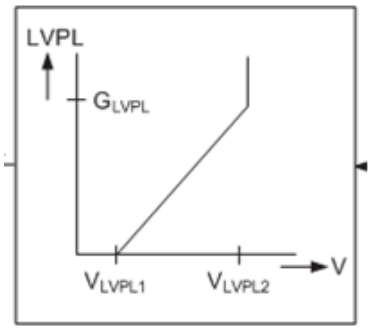
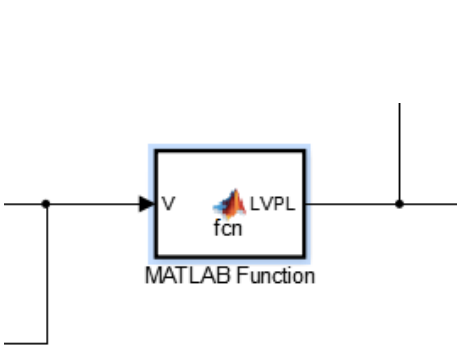


Figure 3-12. V-LVPL Function Block

The output of V-LVPL block (called LVPL) is also the upper limit which restricts the input going in the low voltage active current logic, so we need to link all parts together to test it due to the interaction of them.

MATLAB Simulink block and the corresponding code inside the Function are shown in Figure 3-13.




```
function LVPL=fcn(V)
Vlvpl1=0.50000;
Vlvpl2=0.9000;
Glvpl=1.1100;

if (V<Vlvpl1)
    LVPL=0;
elseif(V>=Vlvpl2)
    LVPL=Inf;
else
    LVPL=(V-Vlvpl1)*(Glvpl/(Vlvpl2-Vlvpl1));

end
end
```

Figure 3-13. MATLAB Code inside V-LVPL Function Block

In addition, there is anti-windup limit on the state K, limiting by LVPL and the input of state K. After testing, the output channel of state K in PSS®E should be behind the limit LVPL instead of before LVPL limit (Figure 3-14).

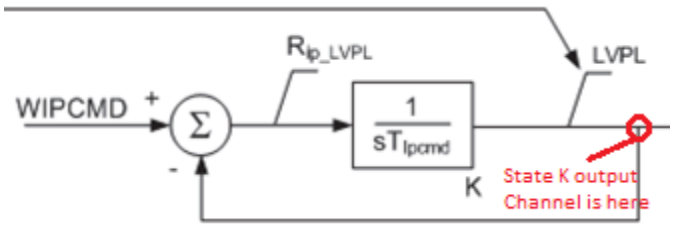


Figure 3-14. State K Output Point in WT3G2

It is implemented in Simulink as shown in Figure 3-15. Function 4 helps to satisfy the condition of non-windup limit on state K. The logic is shown in Figure 3-16, which blocks the state when K hits the limit LVPL and the input of K is larger than 0. Function 1 (Figure 3-15) implements the LVPL limit after the state K.

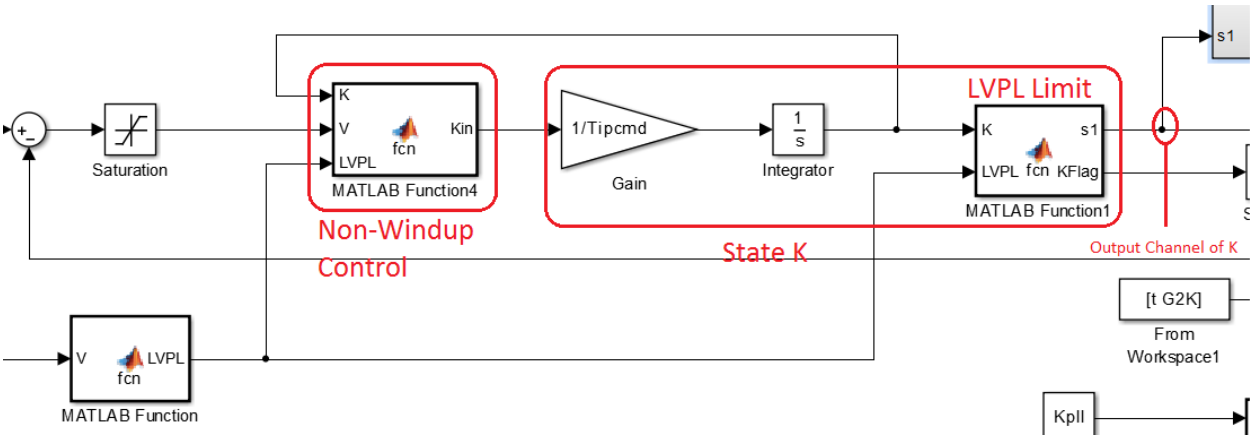


Figure 3-15. MATLAB Implementation of State K

```
function Kin = fcn(K,V,LVPL)

if(K>=LVPL && V>0)
    Kin=0;
else
    Kin=V;

end
end
```

```
function [s1,KFlag]=fcn(K,LVPL)

if (K>=LVPL)
    s1=LVPL;
    KFlag=1;
else
    s1=K;
    KFlag=0;

end
end
```

Figure 3-16. Logic inside MATLAB Function 1 and Function 4

One of the mysteries is two current management logic blocks in WT3G2, which are labeled as "high-voltage reactive current management" and "low-voltage active current management" in PSS®E (shown in Figure 3-17). But there is no clarification or any details in PSS®E documentation. So we did some research online and found the following information.

These blocks represent logic associated with the dynamic model and the ac network solution. The actual implementation of this logic may be software dependent. In the past a simple block diagram was provide in an effort to attempt to explain the logic, this however seemed to have caused more confusion. We present a flow chart, provide by GE, for greater clarification (shown in Appendix B.1 and B.2).

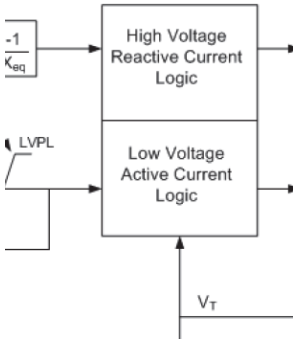


Figure 3-17. Two Current Management Logic in PSS®E

In the data sheet, there are two constant values not be used in the diagram outside the two unknown logic block, which are CURhvrcr (current in HVRC logic) and Vhvrcr (voltage in HVRC logic), so we assume that Vhvrcr represents Vtmax and CURhvrcr represents Qmin, then build a MATLAB Simulink model following this diagram and add anti-windup limit on transfer function as well to verify our assumptions about this High-Voltage Reactive Current Logic.

The diagram inside current management in our Simulink model is shown in Figure 3-18.

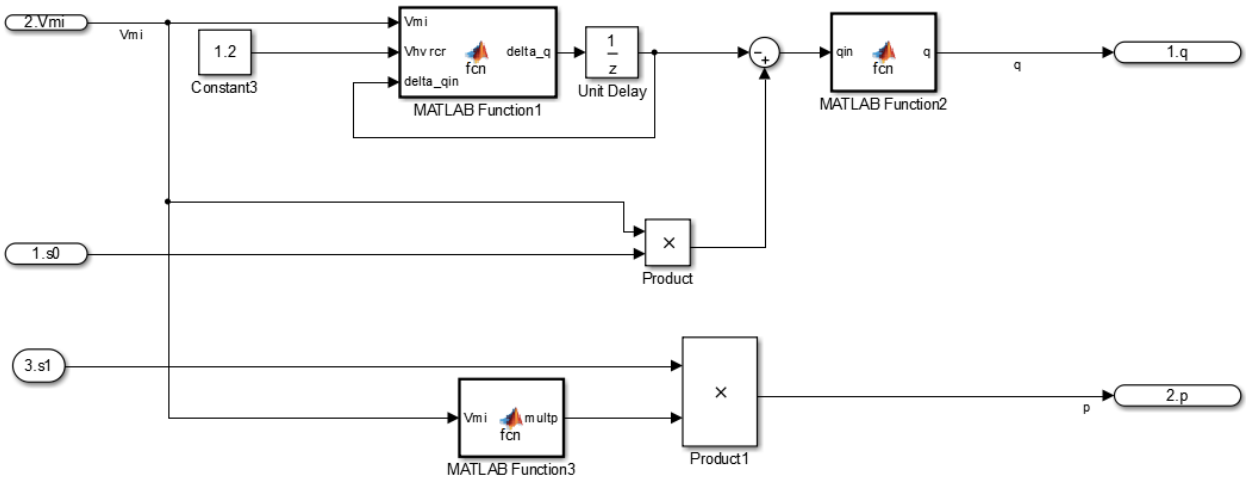


Figure 3-18. MATLAB Implementation of Current Management

The function for "MATLAB Fuction1" is shown in Figure 3-19 corresponding to figure 3-20.

```
function delta_q=fcn(Vmi,Vhvrcr,delta_qin)
accel = 0.8;  % change this value
Vtmax=Vhvrcr;

if (Vmi>Vhvrcr)
    delta_q=delta_qin+accel*(Vmi-Vtmax);
else
    delta_q=delta_qin;

end
end
```

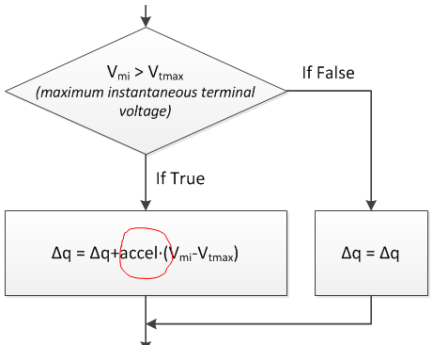


Figure 3-19. Code in MATLAB Function1 **Figure 3-20. PSS®E Logic in High Voltage Logic**

Since there is no material mentions what the Accel value is in this chart, so we set it as a value less than 1, which is 0.8. But since Vmi never reach Vhvrcr when using setted data, so it never go through the branch related to accelerate.

The function for "MATLAB Function2" is shown in Figure 3-21. corresponding to Figure 3-22.

```
function q=fcn(qin)
qmin=-2;

if (qin<qmin)
    q=qmin;
else
    q=qin;
end
end
```

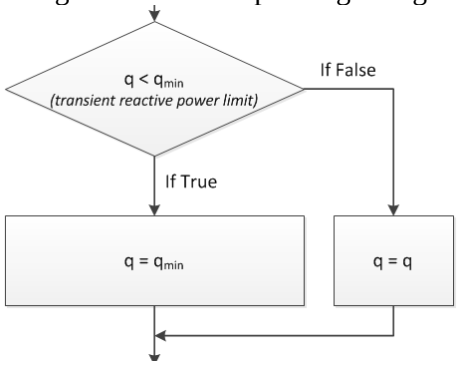


Figure 3-21. Code in MATLAB Function2 **Figure 3-22. PSS®E Logic in Low Voltage Logic**

The code for "MATLAB Function3" is shown in Figure 3-23 corresponding to figure. The value of Lvpnt0 and Lvpnt1 was a problem troubling us, but we installed the PSS®E version 33.4 and found that there were new models for generic wind machine type 3 and type 4. However, there is no sample parameters in Program Application Volume for those new models. Fortunately, [5] is a document having those sample parameters for reference. From [5], Lvpnt0 and Lvpnt1 have the same values with VLVPL1 and VLVPL2 in LVPL (Figure 3-24) respectively. So our assumption is made according to the above investigation.

```
function multp=fcn(Vmi)
Vlvpl1=0.50000;
Vlvpl2=0.9000;

if (Vmi<=Vlvpl1)
    multp=0;
elseif(Vmi>=Vlvpl2)
     multp=1;
else
    multp=(Vmi-Vlvpl1)*(1/(Vlvpl2-Vlvpl1));
end
end
```

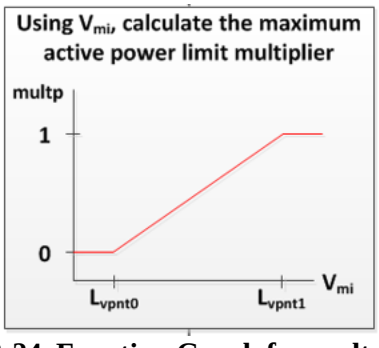


Figure 3-23. Code in MATLAB Function 3 **Figure 3-24. Function Graph for multp Calculation**

Next, in these two current logic, there is two translating confused us a lot (shown in Figure 3-25). But according to the new generator model REGCAU1 in Version 33.4 PSS®E documentation (two logics are shown in Figure 3-26), there is no need to have p=s1*Vmi*multp, which is decribe in Figure 3-26. The correct implementation in PSS®E is simply p=s1*multp, which has been tested in MATLAB and the plots are in section III.B.2.

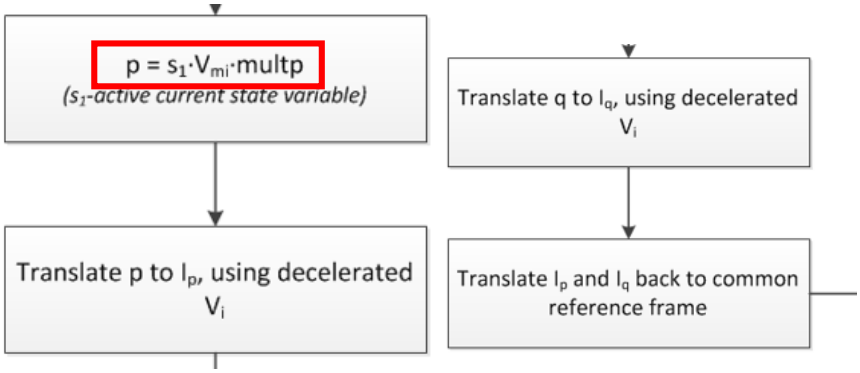


Figure 3-25. Previous Logic Asks for Translation

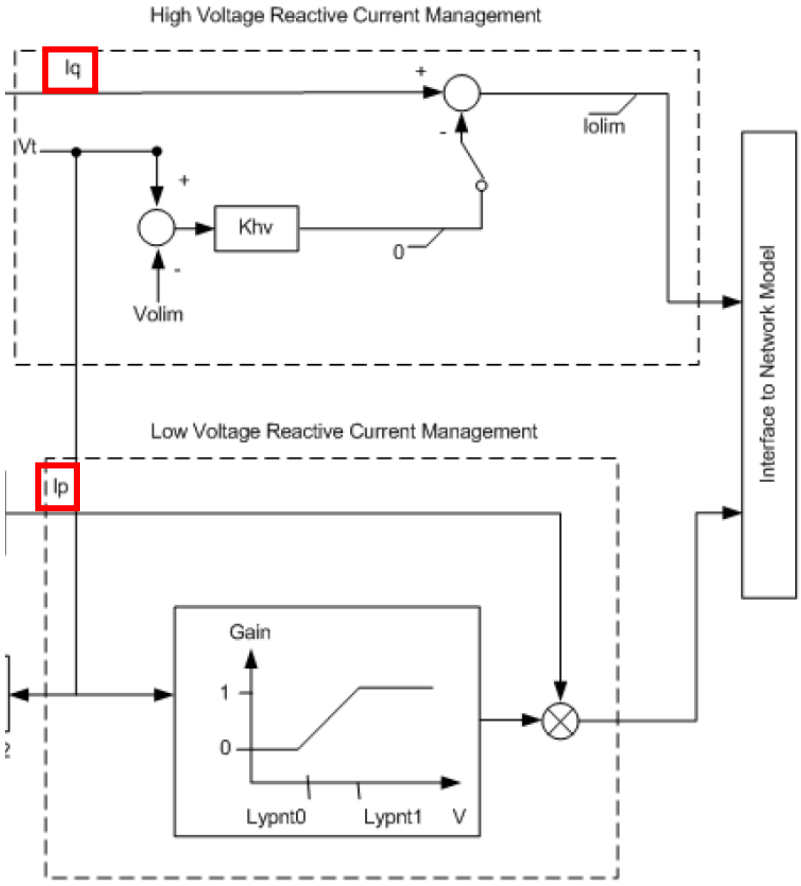


Figure 3-26. Clearer Demonstration in Version 33.4 PSS®E

At last, there is one missing part in PSS®E documentation that has to be empasized. After the current management, the inplementation of interface to network is the same as in G1 (Figure 3-6). Detail

description is in G1 of this section. However, PSS®E G2 doesn't have the electrical component in the network when G1 has one (Figure 3-27).

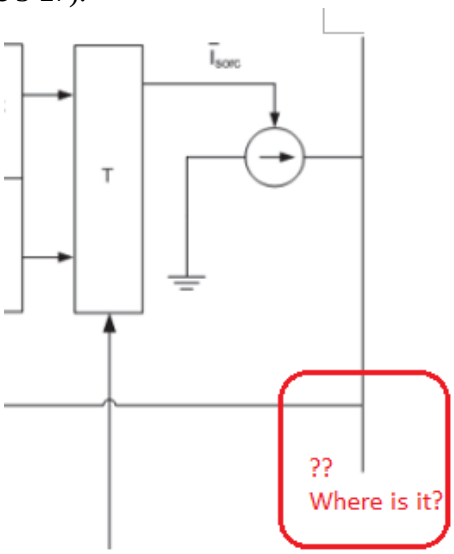


Figure 3-27. Missing Component in PSS®E Documentation

After the testing in MATLAB, we determined to keep the gain j/0.1 as in the G1 Simulink. Sinmulink model is shown in Figure 3-28.

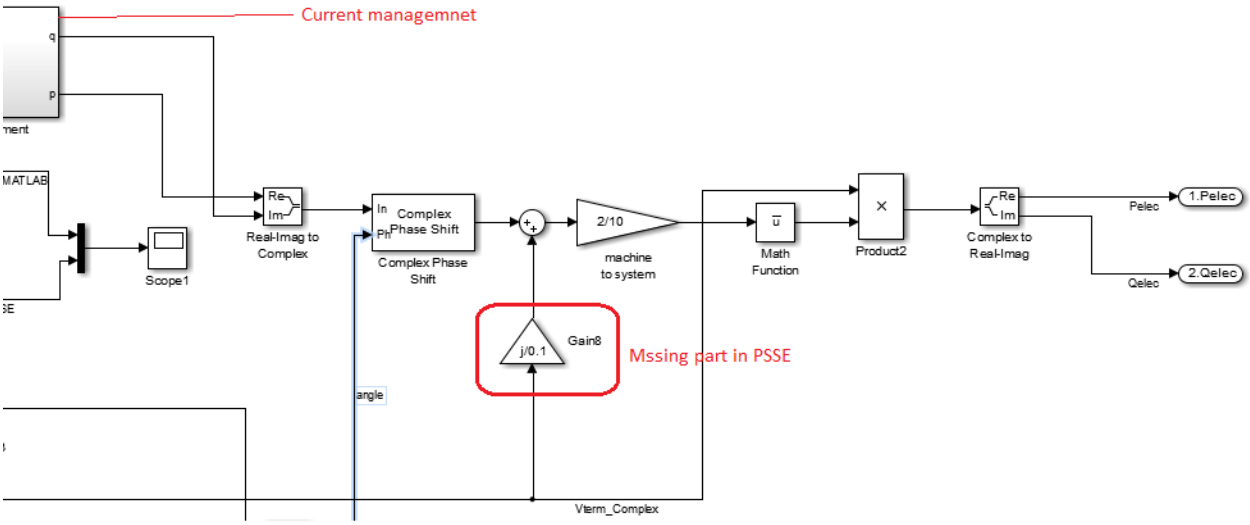


Figure 3-28. MATLAB Implementation of the Missing Part

2. Electrical Control (WT3E1)

The electrical control model is composed of separate active and reactive power control functions. Reactive power control is very fast, due to the power electronic converter. Full details of WT3E1 diagram from PSS®E are provided in Appendix A.2.

a. Upper branch—left part

1) When VARFLG=1, the left part of upper branch is:

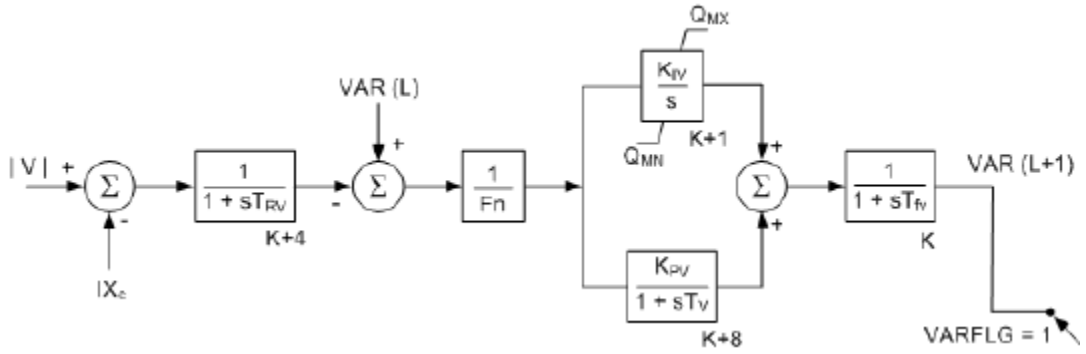


Figure 3-29. PSS®E WT3E1 Upper Left Branch

The MATLAB model of this branch is built as shown in Figure 3-30.

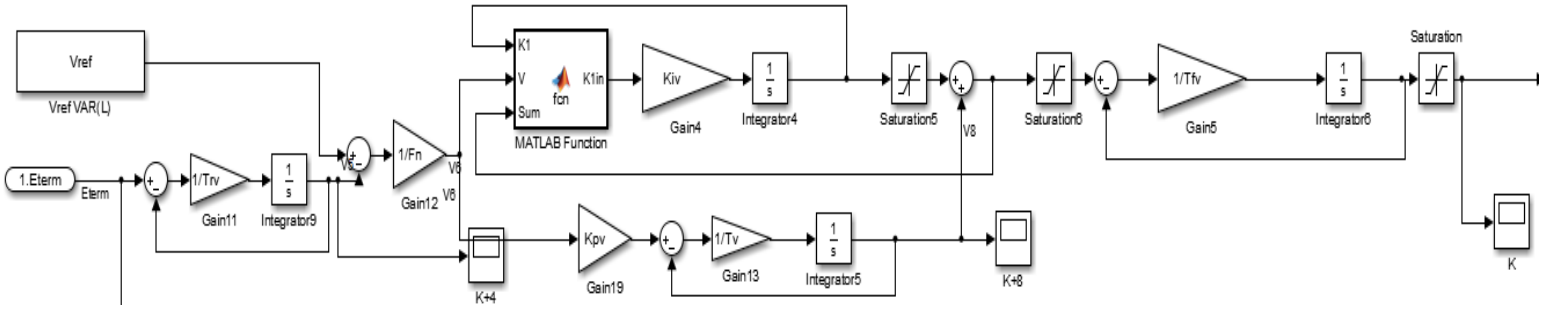


Figure 3-30. MATLAB Implementation of WT3E1 Upper Left Branch

It is important to notice that even if the diagram shows there is only an anti-windup limit on K+1, the system has a hidden limiter at the OUT in Figure 3-31 and block K+1 together with (a).

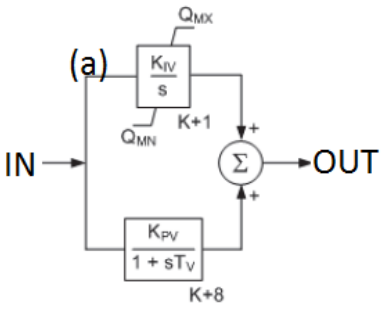


Figure 3-31. State K+1 and K+8 in PSS®E

In order to match the result from PSS®E in this branch, the following Simulink model is built.

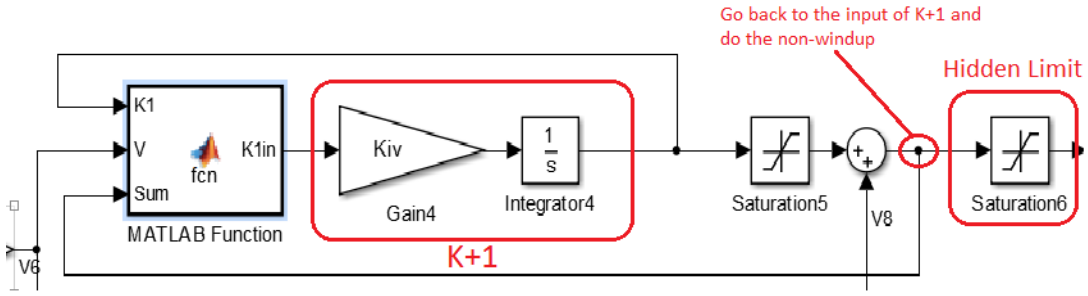


Figure 3-32. Sophisticated State K+1 Non-windup Limit Implementation in MATLAB

The code in MATLAB Function is shown in Figure 3-33.

```
function K1in = fcn(K1,V,Sum)

Qmx=0.4360;
Qmn=-0.4360;

if((V>0 && K1>=Qmx) || Sum>=Qmx || Sum <=Qmn)
    K1in=0;
elseif((V<0 && K1<=Qmn) || Sum>=Qmx || Sum <=Qmn)
    K1in=0;
else
    K1in=V;
end

end
```

Figure 3-33. MATLAB Function Code

2) When VARFLG=-1, the left part of upper branch switches to the model shown in Figure 3-34.

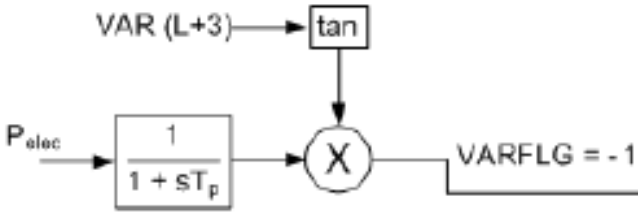


Figure 3-34. PSS®E Model when VARFLG=-1

The MATLAB model we built is:

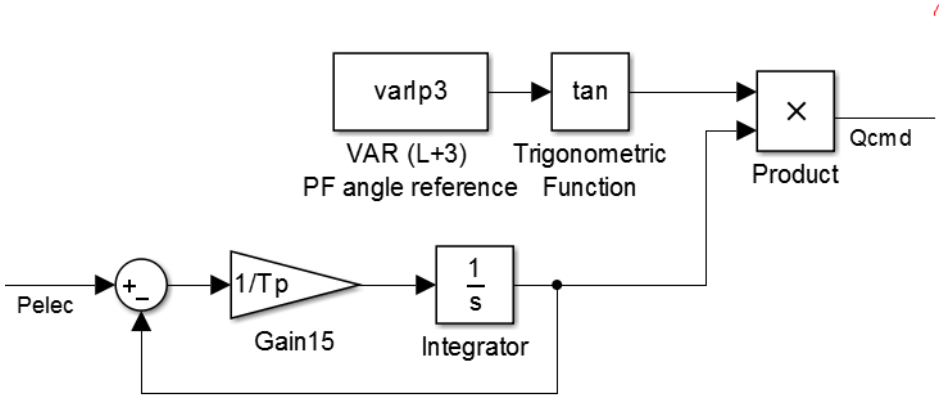


Figure 3-35. MATLAB Implementation of WT3E1 Upper Left Part when VARFLG=-1

3) When VARFLG=0, the left part of upper branch switches to VAR(L+2), which is also called $Q_{reference}$.

b. Upper Branch—Right Part

PSS®E Model is shown in Figure 3-36.

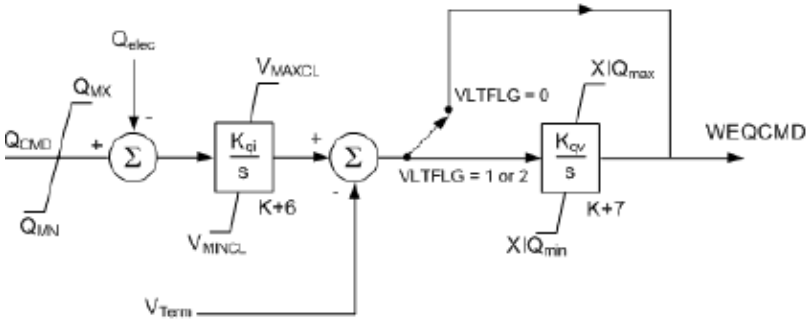


Figure 3-36. WT3E1 Upper Right Part in PSS®E

State K+6 has a non-windup limit. It is implemented in MATLAB as shown below.

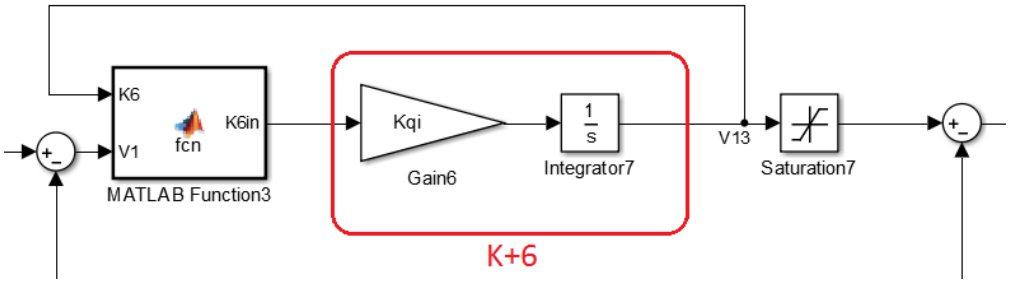


Figure 3-37. MATLAB Implementation of State K+6

Code for MATLAB Function3 is shown as Figure 3-38.

```
function K6in = fcn(K6,V1)

Vmaxcl=1.1;
Vmincl=0.9;

if(V1>0 && K6>=Vmaxcl)
    K6in=0;
elseif(V1<0 && K6<=Vmincl)
    K6in=0;
else
    K6in=V1;
end

end
```

Figure 3-38. Code for MATLAB Function 3

There are three paths after state K+6. This switch is controlled by VLTFLG set by users (Figure). Details are shown in three different cases.

M+2[1]		VLTFLG: 0 Bypass terminal voltage control 1 Eqcmd limits are calculated as VTerm + XIQmin and VTerm + XIQmax, i.e., limits are functions of terminal voltage 2 Eqcmd limits are equal to XIQmin and XIQ max

Figure 3-39. Different Cases when VLTFLG Changes

1) When VLTFLG=1

State K+7 has non-windup limit and the limits (Figure 3-40 Saturation1) are XIQmin and XIQMax, which is a normal limiter. Code for MATLAB Function4 is shown in Figure 3-41.

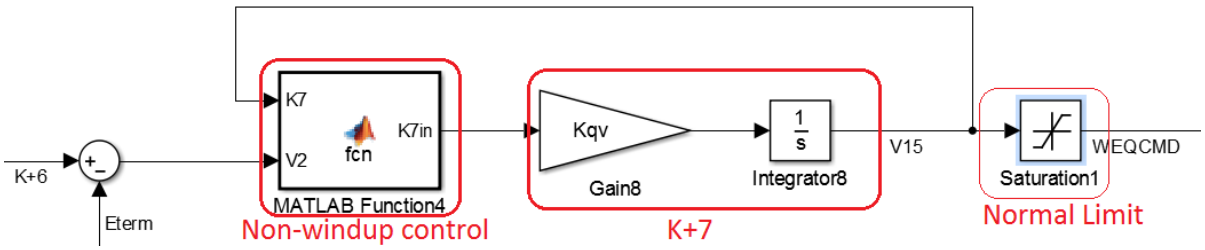


Figure 3-40. MATLAB Implementation of State K+7 when VLTFLG=1

```
function K7in = fcn(K7,V2)

XIQmax=1.45;
XIQmin=0.5;

if(V2>0 && K7>=XIQmax)
    K7in=0;
elseif(V2<0 && K7<=XIQmin)
    K7in=0;
else
    K7in=V2;
end

end
```

Figure 3-41. Code for MATLAB Function 4

2) When VLTFLG=2

The EQCMD Limits of VLTFLG=2 case (New Limit in Figure 3-42) are different from the EQCMD limits of VLTFLG=1 case. Upper limit equals to VTerm + XIQmax while the lower limit equals to VTerm + XIQmin. Code for two Matlab Functions is shown in Figure 3-43 and Figure 3-44.

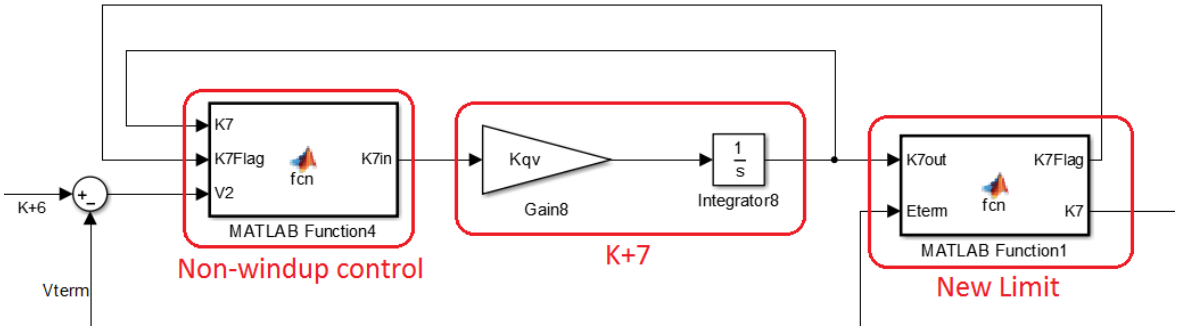


Figure 3-42. MATLAB Implementation of State K+7 when VLTFLG=2

```
function K7in = fcn(K7,K7Flag,V2)

if(V2>0 && K7Flag==1)
    K7in=0;
elseif(V2<0 && K7Flag==2)
    K7in=0;
else
    K7in=V2;
end

end
```

Figure 3-43. MATLAB function4 Code (Non-windup control)

```
function [K7Flag,K7] = fcn(K7out,Eterm)

XIQmax=0.40;
XIQmin=-0.5;
upperLimit=Eterm+XIQmax;
lowerLimit=Eterm+XIQmin;
K7Flag=0;

if(K7out>=upperLimit)
    K7=upperLimit;
    K7Flag=1;

elseif(K7out<=lowerLimit)
    K7=lowerLimit;
    K7Flag=2;
else
    K7=K7out;
end
```

Figure 3-44. Matlab function1 code (New Limit)

3) When VLTFLG=0

K+7 is bypassed and there is no limit for WEQCMD (Figure 3-45).

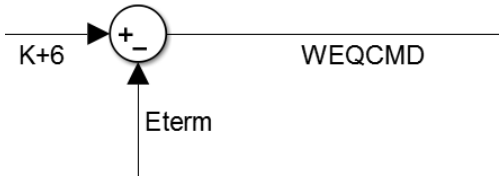


Figure 3-45. State K+7 is Bypassed When VLTFLG=0

c. Bottom Branch:

1) Speed- Pelec function:

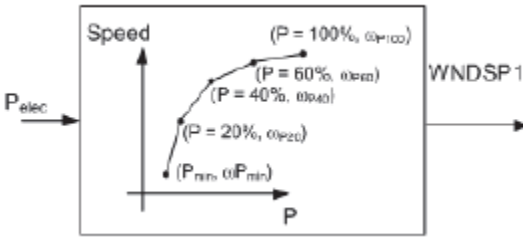


Figure 3-46. Speed - Pelec Function in WT3E1 Lower Branch

Note: P=20% means Pelec equals 20% of Prated.

Since this diagram fall to indicate the speed value when Pelec lower than Pmn*Prated or larger than Prated. Therefore, we figure out this function by adjust Pgen value in raw file and collect speed value by adding one to state K+5 value. An excel table is made to list all the data and a line chart is formed using this data, as shown in Figure 3-47.

Pgen	K+5	Pelec	speed
0.5	-0.62619	0.060	0.37381
0.6	-0.62601	0.060	0.37399
0.7	-0.61289	0.070	0.38711
0.8	-0.59973	0.080	0.40027
0.9	-0.58657	0.090	0.41343
1	-0.57345	0.100	0.42655
2	-0.44219	0.200	0.55781
3	-0.31024	0.300	0.68976
4	-0.28013	0.400	0.71987
5	-0.25017	0.499	0.74983
6	-0.22013	0.600	0.77987
7	-0.15479	0.698	0.84521
8	-8.91E-02	0.797	0.910945
9	-2.07E-02	0.899	0.979276
10	1.56E-02	0.997	1.015619
11	5.20E-02	1.096	1.05195
11.1	5.55E-02	1.106	1.05552
11.11	3.30E-02	1.044	1.032959
11.15	3.47E-02	1.049	1.034675
11.2	3.68E-02	1.055	1.03683
11.3	4.12E-02	1.067	1.041241
11.33	4.24E-02	1.070	1.042444
11.335	4.27E-02	1.071	1.042747
11.336	4.27E-02	1.071	1.042706
11.337	0.19994	1.132	1.19994
11.34	0.19999	1.132	1.19999
11.35	0.19999	1.133	1.19999
11.4	0.19999	1.137	1.19999
11.5	0.19999	1.144	1.19999

Figure 3-47. Multiple Data Collected to Reveal This Mysterious Function

Then we find out that, speed keeps constant at wPmin when Pelec is less than Pmn*Prated, and also keeps constant at wP100 when Pelec is larger than Prated. Meanwhile, speed will first reach its maximum value at the point Pelec=Pmin*Prated, and keeps stable afterwards. Hence, if Pmin=1, then the relationship will follow the diagram shown above, which is from PSS®E documentation. However, if Pmin is less than one, then the function should be drawn as below:

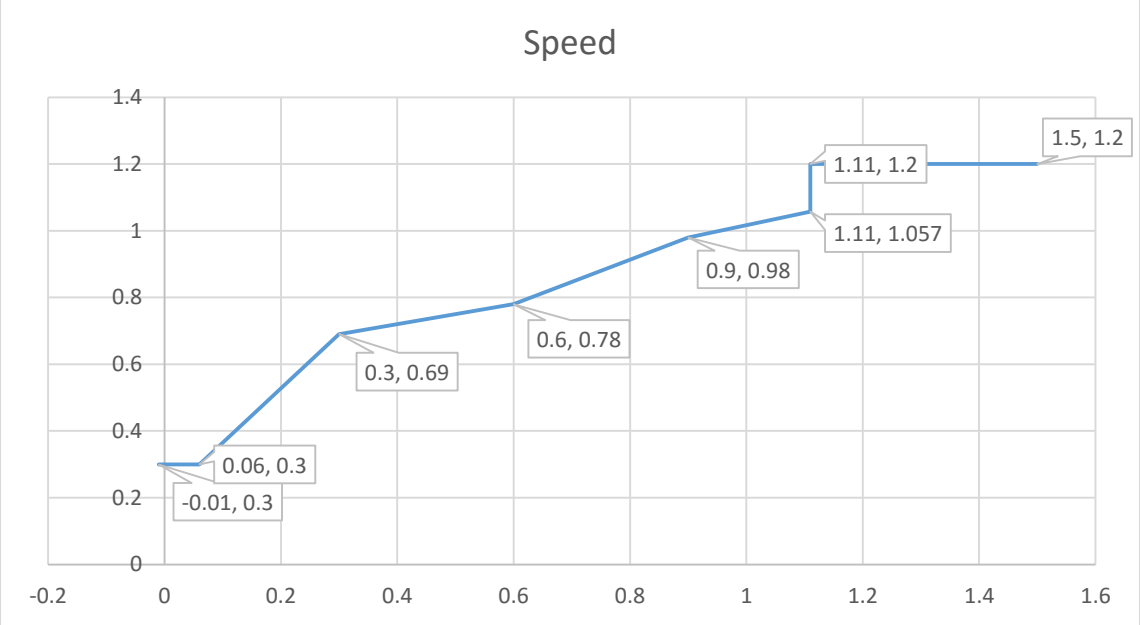


Figure 3-48. New Line Chart Generated by Excel

This is an ideal function when WT3G2 constant values are set as following:

CONs	#	Value	Description
J+24			Fn, A portion of online wind turbines
J+25		0.3	ωP_{min}, Shaft speed at P_{min} (pu)
J+26		0.69	ωP_{20}, Shaft speed at 20% rated power (pu)
J+27		0.78	ωP_{40}, Shaft speed at 40% rated power (pu)
J+28		0.98	ωP_{60}, Shaft speed at 60% rated power (pu)
J+29		0.74	P_{min}, Minimum power for operating at ωP_{100} speed (pu)
J+30		1.2	ωP_{100}, Shaft speed at 100% rated power (pu)
J+8		0.04	P_{MN}, Min limit in torque regulator (pu)

Figure 3-49. Parameters of This Line Function

So we modify the math function in MATLAB model to implement this part:

```
function y = fcn(u)
 wPmin=0.3;
 wP20=0.69;
 wP40=0.78000;
 wP60=0.98;
 wP100=1.2000;
 Pmin=0.74000;
 Pmn=0.04;
 Prated=1.5;

if(u>Pmin*Prated)
    y=1.2;
elseif(u>0.6*Prated)
    y=(wP100-wP60)/(0.4*Prated)*(u-0.6*Prated)+wP60;
elseif(u>0.4*Prated)
    y=(wP60-wP40)/(0.2*Prated)*(u-0.4*Prated)+wP40;
elseif(u>0.2*Prated)
    y=(wP40-wP20)/(0.2*Prated)*(u-0.2*Prated)+wP20;
elseif(u>=Pmn*Prated)
    y=(wP20-wPmin)/ (0.2*Prated-Pmn*Prated)*(u-Pmn*Prated)+wPmin;
else
    y=wPmin;
end
end
```

Figure 3-50. Code for MATLAB Function to Implement This Line Function

2) Non-windup limit on State K+2

Simulink model and code for MATLAB Function2 are shown below.

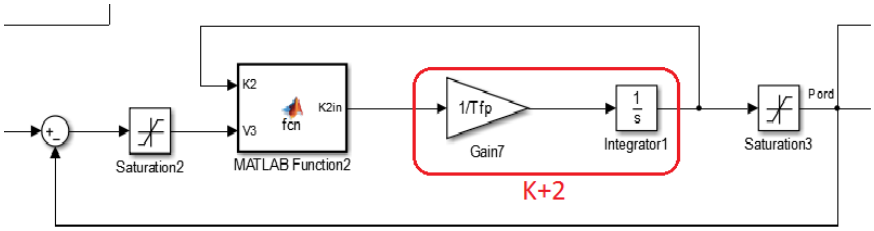


Figure 3-51. Simulink Model of State K+2

```
function K2in = fcn(K2,V3)

Pmxe=1.12;
Pmn=0.04;

if(V3>0 && K2>=Pmxe)
    K2in=0;
elseif(V3<0 && K2<=Pmn)
    K2in=0;
else
    K2in=V3;
end

end
```

Figure 3-52. Code for MATLAB Function 2

3. Pitch control (WT3P1)

The pitch control model WT3P from PSS®E is shown in Appendix A.4. This model looks like the simplest one within these four models, but the idea of implementation of the non-windup limiter on the pitch angle θ is quite sophisticated. Simulink model of WT3P1 is shown in Appendix C.3.

According to Ian's report (Figure shows the WT3P1 in his report),

"The Pitch Control and Pitch Compensation integrators are non-windup integrators as a function of the pitch, i.e., the inputs of these integrators are set to zero when the pitch is in limits (PImax or PImin) and the integrator input tends to force the pitch command further against its limit."

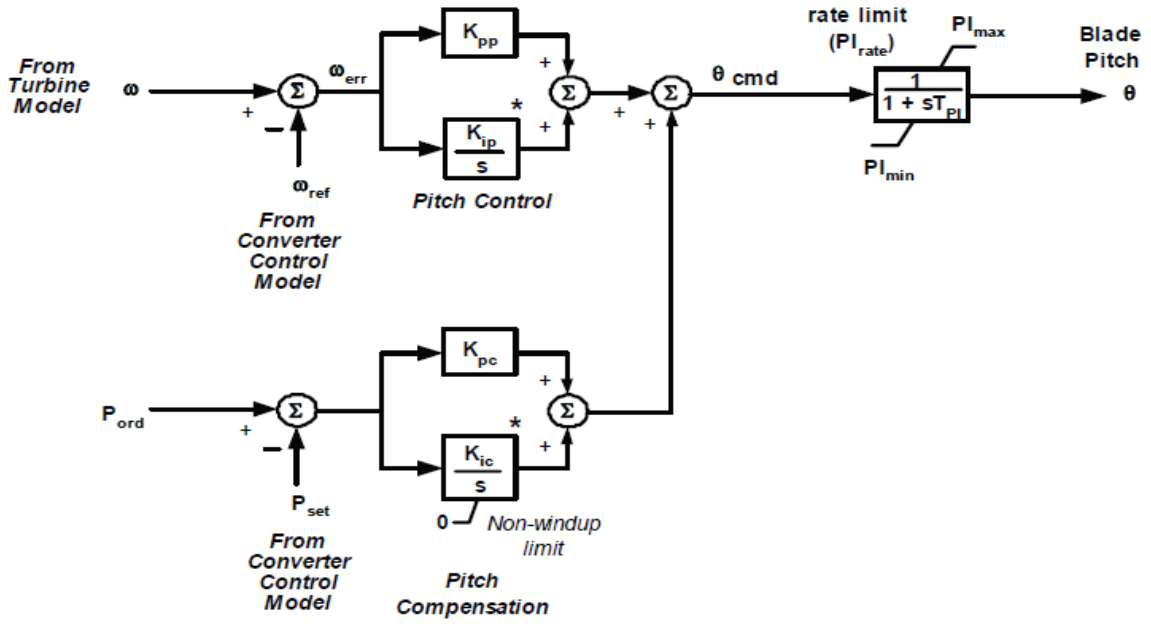


Figure 3-53. WT3P1 Model Diagram

To illustrate, consider the case where θ is on its lower limit PImin. A negative input to the pitch-control integrator would cause the corresponding pitch control state (K+1 in PSS®E instead) to reduce, which in turn would force θ further against its PImin limit. To prevent that wind-up effect, the integrator is blocked under such conditions. Similarly, the pitch-compensation integrator (K+2 in PSS®E) is blocked when its input is negative. When θ is on its upper limit PImax, blocking of the up-stream integrators occurs when their respective inputs are positive. [1]

From the above description, we can summarize that state K+1 and state K+2 has non-windup limit which is under controlled by state K (Appendix A.4). Also, state K has a non-windup limit controlled by its own limits and its input. In Figure 3-54, K1Flag is a signal to show whether state K is blocked. Then K1Flag helps to control the state K+1 and K+2 (Figure 3-55 and Figure3-56).

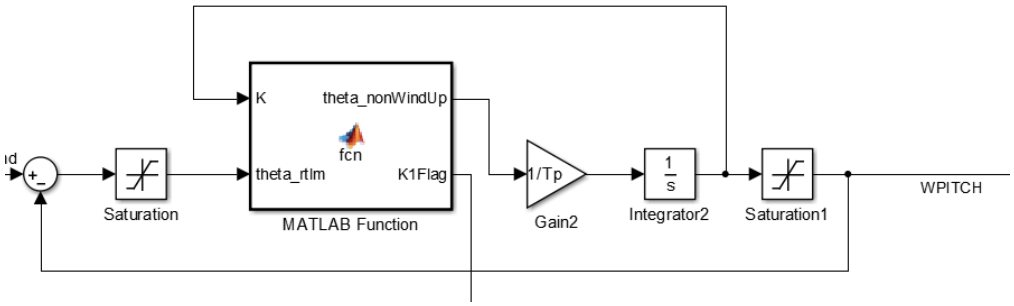


Figure 3-54. K

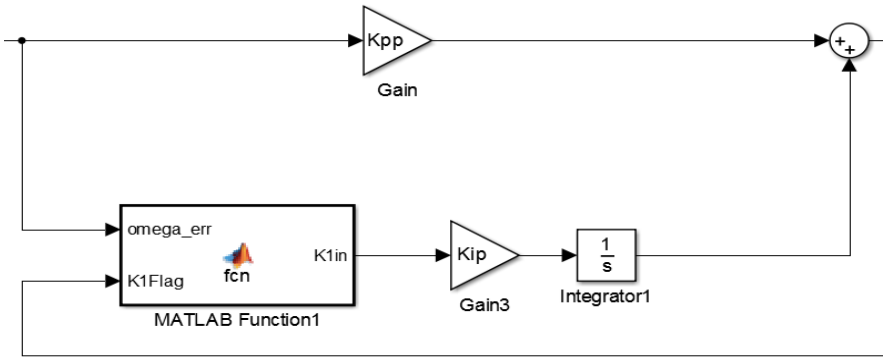


Figure 3-55. K+1

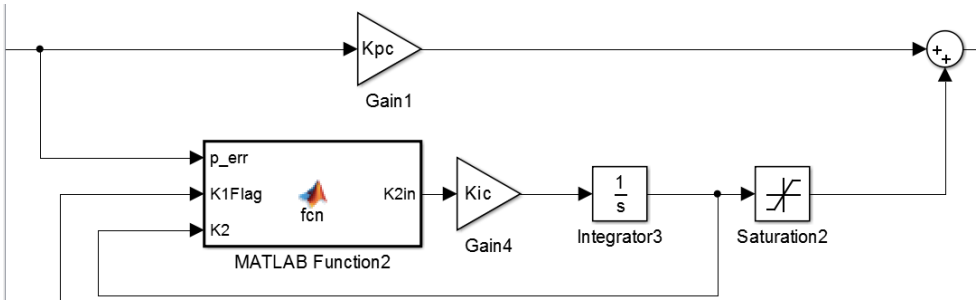


Figure 3-56. K+2

Corresponding sets of code for MATLAB Function, MATLAB Function 1, and MATLAB Function 2 are shown below as Figure 3-57, Figure 3-58, Figure 3-59 respectively.

```
function [theta_nonWindUp,K1Flag]= fcn(K,theta_rtlm)

TetaMax=27;
TetaMin=0;

if(theta_rtlm>0 && K>=TetaMax)
    theta_nonWindUp=0;
    K1Flag=1;

elseif(theta_rtlm<0 && K<=TetaMin)
    theta_nonWindUp=0;
    K1Flag=2;
else
    theta_nonWindUp = theta_rtlm;
    K1Flag=3;
end

end
```

Figure 3-57. Code for MATLAB Function

```
function K1in = fcn(omega_err,K1Flag)

if(K1Flag==1 || K1Flag==2)
    K1in=0;
else
    K1in=omega_err;
end

end
```

Figure 3-58. Code for MATLAB Function 1

```
function K2in = fcn(p_err,K1Flag,K2)

if(K1Flag==1 || K1Flag==2 || (p_err<0 && K2<=0))
    K2in=0;
else
    K2in=p_err;
end

end
```

Figure 3-59. Code for MATLAB Function 2

WPCMND should be distinguished from WIPCMD. After our testing and verification in other documentation, WPCMND is Pord from WT3E1.

At last, after a lot of testing, one of the inputs for this model WNDSP1 is considered to be state K+5 in WT3E1. Figure 3-60 illustrates our finding.

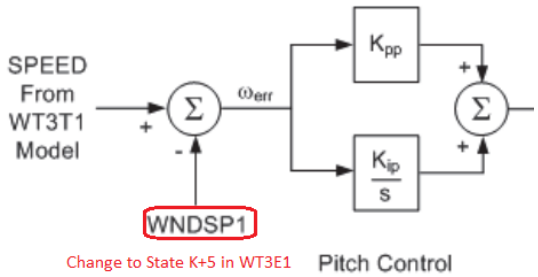


Figure 3-60. WNDSP1 being STATE K+5 from Electrical Model

4. Mechanical control (WT3T1)

The mechanical control model used in the wind machine system is WT3T1. In this model, the parameter Htfrac has significant impact on the model diagram and can result in notable differences.

When Htfrac is not 0, the diagram is given by the WT3T1 model in Appendix (A3).

When Htfrac is 0, the diagram is different as some branches are disconnected or connected differently. It can be shown that WATROT directly goes to Tmech, and WTRBSP is equal to SPEED. According to PSS®E simulation, STATE K+1 is 0, while WTRBSP is not, which behaves different than the diagram showing WTRBSP is "the same" as STATE K+1. On this occasion, STATE K is also 0. However, STATE K+2 does have value and is equal to SPEED, and thus equal to WTRBSP.

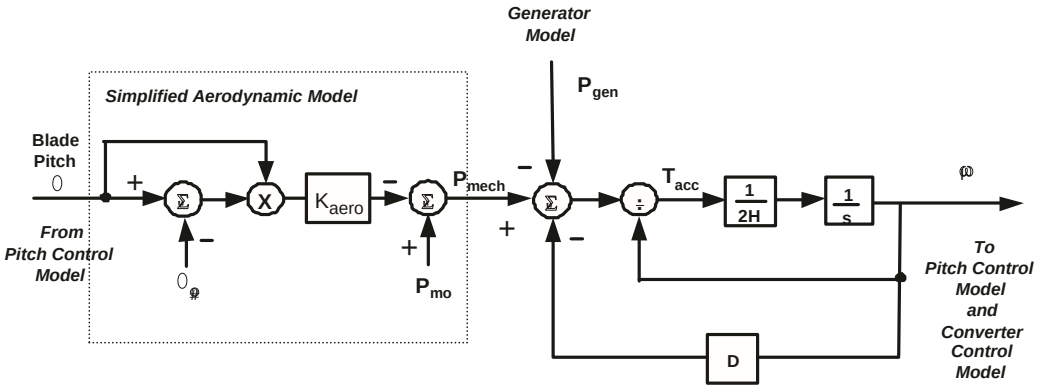


Figure 3-61. Mechanical model interconnects

In general, the diagram can be simplified as following, if Htfrac is 0:

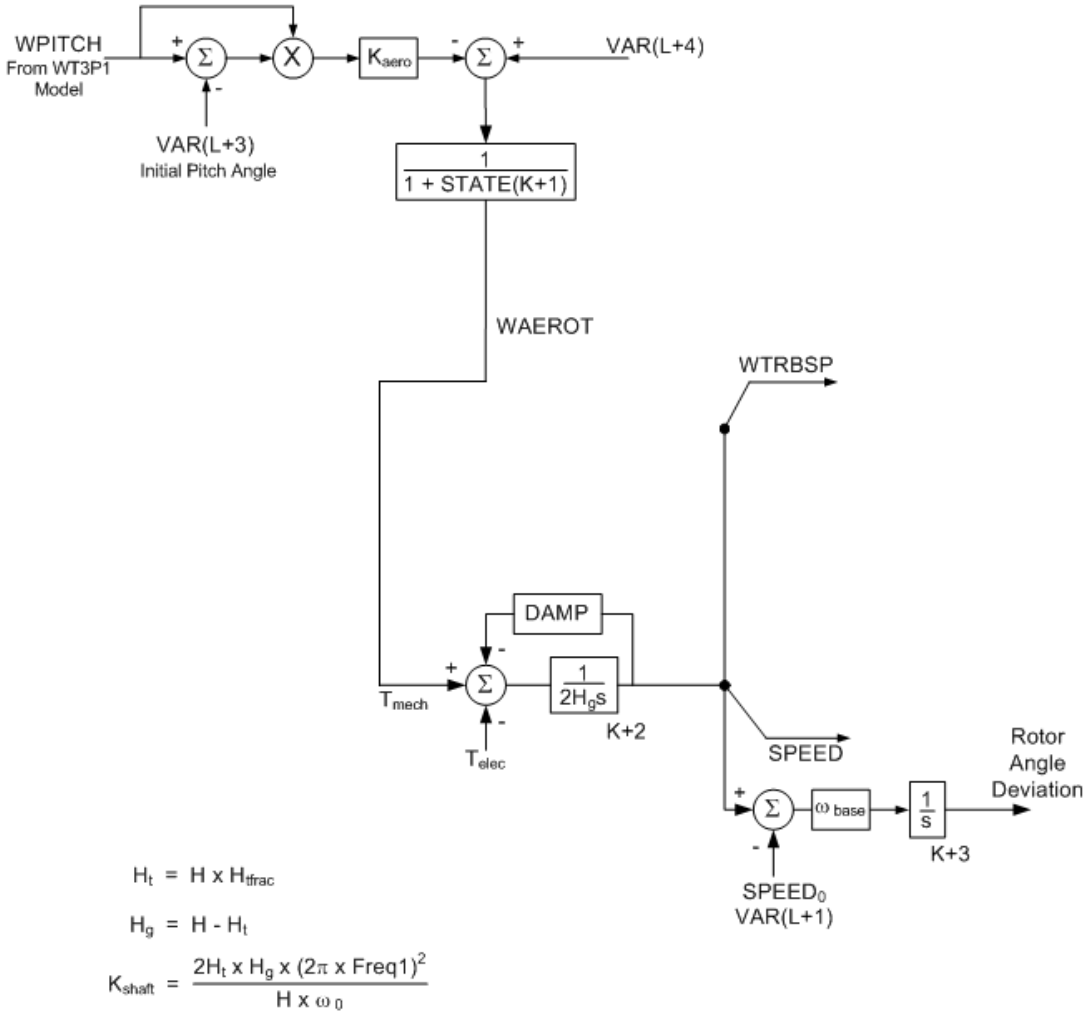


Figure 3-62. Simplified Mechanical model when Htfrac is 0

Actually, Since STATE K+1 is 0, the block showing $\frac{1}{1+\text{STATE(K+1)}}$ is actually 1, just like there is no such divider implemented. Due to the way of connection shown above, MATLAB Simulink models have to be built separately for Htfrac equals and not equals to zero.

When Htfrac is not 0, STATE K will have value. Simulation shows STATE K+1 equal to WTRBSP and STATE K+2 equal to SPEED. The MATLAB model is shown in Appendix C.4.

Since the initial value of state K+2 is 0.2, and VAR (L+4) is 0.7495, the initial value of WAEROT is equals to $\frac{0.7495}{1+0.2} = 0.624583$ $\left(i.e.\frac{\text{VAR(L+4)}}{1+\text{STATE(K+1)}}\right)$.

At the very beginning, the input of state (K+1) should be zero, and the initial value of Tmech is from the initialization of state K. So, we modify it to 0.624583 in order to make the subtraction of WAEROT and Tmech equals to zero.

Since in PSS®E the state K begins at 0.7495, we modify the model by making the integration value multiply 1+STATE (K+1) to reach almost 0.75.

We did had a hard time to match MATLAB and PSS®E output of WAEROT. If we do it directly as shown above, there will be discrepancy, plotted as Figure 3-63.

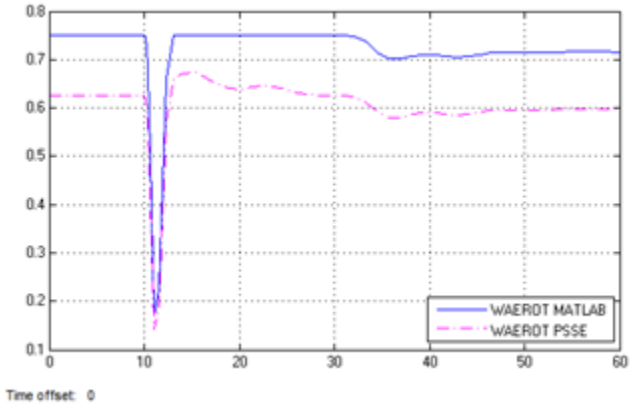


Figure 3-63. WAEROT discrepancy between MATLAB and PSS®E

This proves that WAEROT output in PSS®E is not Tmech, but rather some kind of translated into something else. Initially we attempted to let WAEROT be divided by $1 + \mathrm{STATE}(K + 2)$, just like Figure 3-64, and make it Tmech.

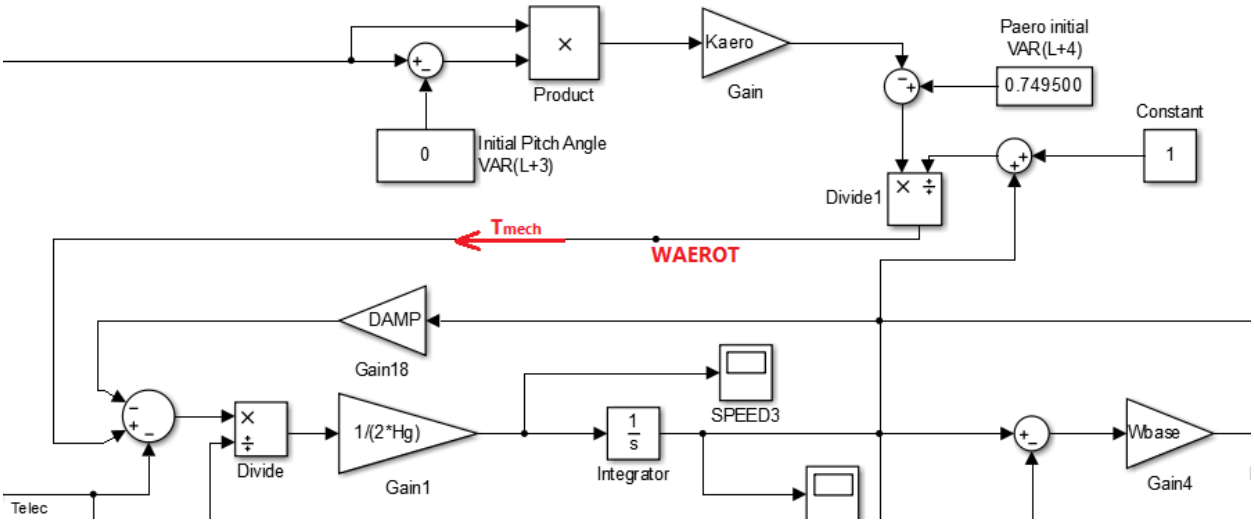


Figure 3-64. Attempting To Let Divided WAEROT Be Utputted as Tmech

We ended up none of the outputs match expected results. The SPD and Rotor Angle Deviation did not make any sense. This is reasonable as we are able to make these outputs match PSS®E if we do not divide WAEROT by such.

We imagined that PSS®E might divide WAEROT by $1 + \mathrm{STATE}(K + 2)$ when outputting it, but the actual connection to Tmech is indeed not divided. This imagination was proven correct, so the MATLAB model was revised as Figure 3-65.

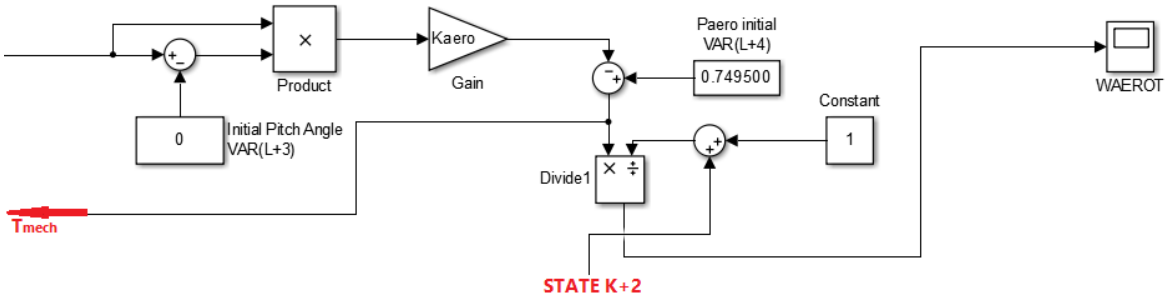


Figure 3-65. Revised MATLAB Model Regarding Tmech and WAEROT in Mechanical Model

We realize this in MATLAB as following:

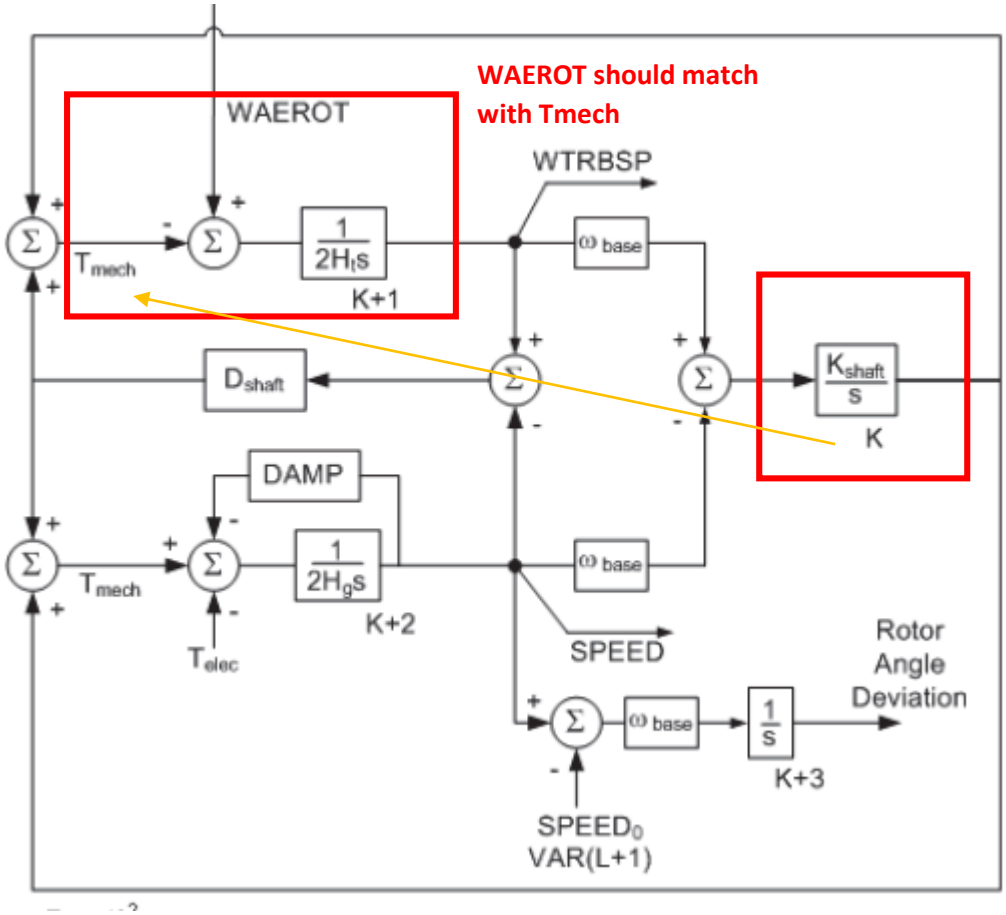


Figure 3-66. System diagram

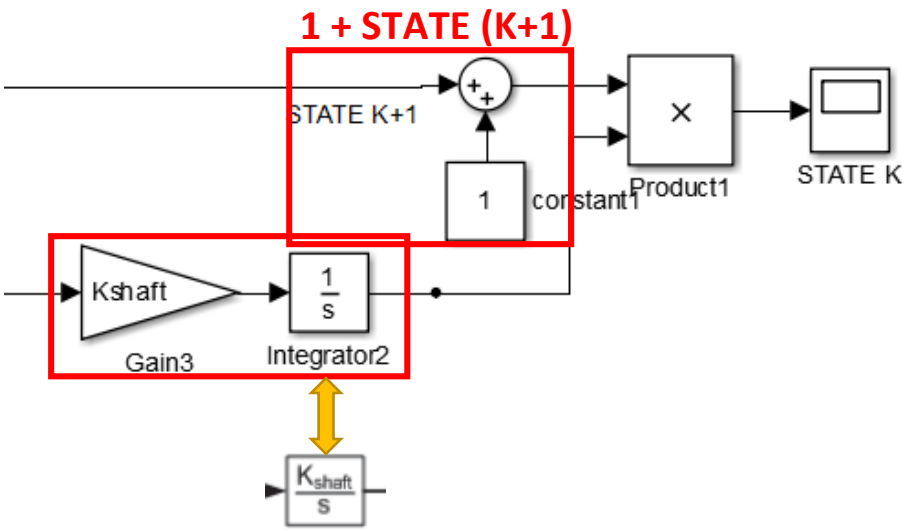


Figure 3-67. MATLAB realization

When Htfrac is 0, the model is shown as Figure 3-68.

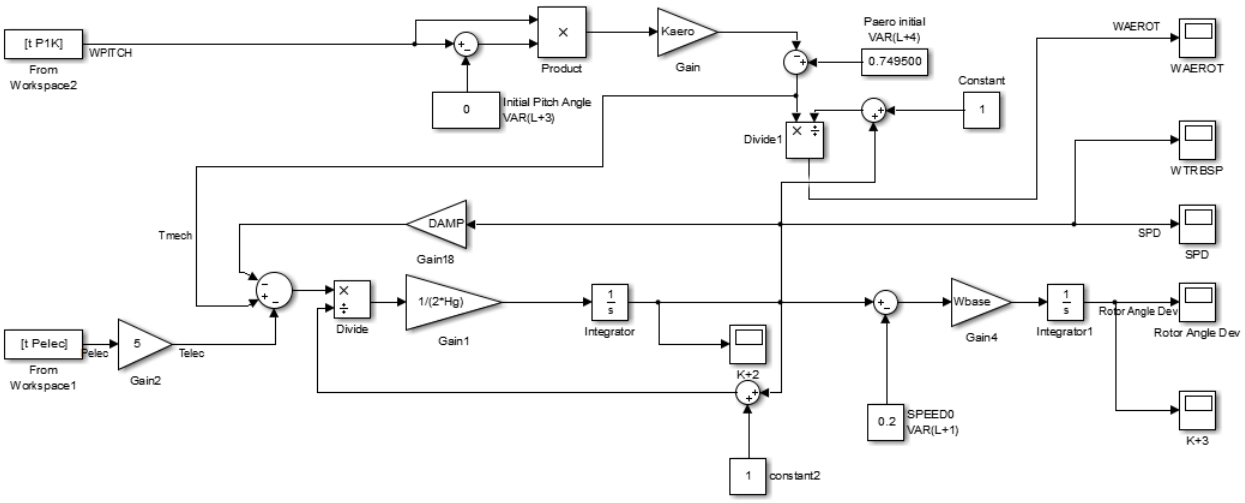


Figure 3-68. MATLAB Model for Mechanical Model when Htfrac is 0

B. PSS®E and Matlab Simulation

In type 3 Generic Wind Machine, there are two generators. The results in PSS®E will have huge differences if the users choose WT3G1 or WT3G2 as the generator in this whole model. Here we separate these two cases for comparing Matlab result and PSS®E result.

The first part in each case is the power flow setup in PSS®E, providing the detail data and command we entered in PSS®E. Next, in each case, some parts of network data after running the idv file in PSS®E are provided. Then, plots of important bus and machine quantity plots from PSS®E are shown. At last, in order to show our Matlab model matches PSS®E model, the comparison of vital states and output between PSS®E and Matlab simulation are shown. Most of these two simulations described in this section match each other well, but there are still some mysteries remain further investigation described in section IV DISSCUSSION.

1. Using WT3G1 as the Generator

Power Flow Setup

Raw file and dyr file, provided in Figure 3-B1 and Figure 3-B2 should be well prepared before creating command lines in idv file. Here we set the wind machine control mode to 2 and the power factor to 0.9 in bus 2 generator data. Ten machines are connected to the network.

```
0,   100.00, 33, 0, 1, 60.00     / PSS(R)E-33.3    MON, JUL 14 2014  15:09


    1,'BUS1       ', 345.0000,3,   1,   1,   1,1.04000,   0.0000,1.10000,0.90000,1.10000,0.90000
    2,'BUS2       ', 345.0000,2,   1,   1,   1,1.01500,  -3.0087,1.10000,0.90000,1.10000,0.90000
0 / END OF BUS DATA, BEGIN LOAD DATA
  2,'11',1,   1,   1,     0.000,     0.000,     0.000,     0.000,   107.990,   -38.840,   1,1,0
0 / END OF LOAD DATA, BEGIN FIXED SHUNT DATA
0 / END OF FIXED SHUNT DATA, BEGIN GENERATOR DATA
    1,'1 ',    97.312,    38.929,   300.000,  -300.000,1.04000,     0,   100.000, 0.00000E+0, 1.73300E-1, 0.00000E+0, 0.00000E+0,1.00000,1,  100.0,   999.000,    10.000,   1,1.0000
    2,'1 ',    14.990,    -1.352,    10.000,    -10.000,1.01500,     0,    20.000, 0.00000E+0, 1.00000E-1, 0.00000E+0, 0.00000E+0,1.00000,1,  100.0,    15.000,     0.000,   1,1.0000,
                     ,        ,        ,        ,        ,      ,2, 0.9000
0 / END OF GENERATOR DATA, BEGIN BRANCH DATA
  1,   2,'1 ', 1.00000E-4, 5.76000E-2,   0.00010,  250.00,  250.00,  250.00,  0.00000,  0.00000,  0.00000,  0.00000,1,1,   0.00,   1,1.0000
0 / END OF BRANCH DATA, BEGIN TRANSFORMER DATA
0 / END OF TRANSFORMER DATA, BEGIN AREA DATA
  1,    1,    0.000,   10.000,'            '
0 / END OF AREA DATA, BEGIN TWO-TERMINAL DC DATA
0 / END OF TWO-TERMINAL DC DATA, BEGIN VSC DC LINE DATA
0 / END OF VSC DC LINE DATA, BEGIN IMPEDANCE CORRECTION DATA
0 / END OF IMPEDANCE CORRECTION DATA, BEGIN MULTI-TERMINAL DC DATA
0 / END OF MULTI-TERMINAL DC DATA, BEGIN MULTI-SECTION LINE DATA
0 / END OF MULTI-SECTION LINE DATA, BEGIN ZONE DATA
0 / END OF ZONE DATA, BEGIN INTER-AREA TRANSFER DATA
0 / END OF INTER-AREA TRANSFER DATA, BEGIN OWNER DATA
0 / END OF OWNER DATA, BEGIN FACTS DEVICE DATA
0 / END OF FACTS DEVICE DATA, BEGIN SWITCHED SHUNT DATA
0 / END OF SWITCHED SHUNT DATA, BEGIN GNE DATA
0 / END OF GNE DATA, BEGIN INDUCTION MACHINE DATA
0 / END OF INDUCTION MACHINE DATA
Q
```

Figure 3-B1. RAW file

```
1,'GENROU',1, 6.1000, 0.0500, 1.0000, 0.1500,  3.3800, 0.0000, 1.5750, 1.5120, 0.2910, 0.3900, 0.1733, 0.0787, 0, 0/

1,'ESDC1A',1, 0.0, 25.0, 0.2, 0.0, 0.0, 100.0, -100.0, -0.05, 0.57, 0.091, 0.35, 0.0, 2.846, 0, 3.795, 0/

2 'WT3G1' 1    10
  0.10000 30.0000 0.00000 0.10000 1.5000 /

2 'WT3E1' 1    0 1 1 0 0 '1 '
  0.15000 18.000 5.0000 0.0000 0.50000E-01 3.0000 0.60000 1.1200 0.10000 0.29600 -0.43600 1.1000 0.50000E-01 0.45000 -0.45000 5.0000
  0.50000E-01 0.90000 1.2000 40.000 -0.50000 0.40000 0.50000E-01 0.50000E-01 1.0000 0.69000 0.78000 0.98000 1.1200 0.74000 1.2000 /

2 'WT3T1' 1
  1.2500 4.9500 0.0000 0.70000E-02 21.980 0.0000 1.8000 1.5000 /

2 'WT3P1' 1
  0.30000 150.00 25.000 3.0000 30.000 0.0000 27.000 10.000 1.0000 /


//ajf 1 'IEESGO',1, 0.1,  0.0,0.12, 0.25,6.0, 0.0,18.94,   0.7,  0.0, 100,  -100/

//ajf 1 'PSS2A', 1, 1, 0, 3, 0, 5, 1, 2.0, 2.0, 0.0, 2.0, 0.0000000, 2.0, 0.18, 1.0, 0.5, 0.1, 15.0, 0.15, 0.03,
//    0.15, 0.03, 1000, -1000/
```

Figure 3-B2. DYR file

There are three different idv files shown below. We ran the simulation in 60 seconds, within which no fault condition, 6 cycles fault condition in 10 second (long fault) and 3 cycles fault condition in 10 second (short fault) are created. Details are shown in Figure 3-B3, Figure 3-B4 and Figure 3-B5 respectively.

```
@! File:"C:\Users\ECE497\Documents\ex2bus_10nodist_30fault_uirecord.idv", generated on FRI, MAY 23 2014  14:03, release 33.03.00
BAT_PROGRESS_OUTPUT,2,'ex2bus_nodist10_fault30_Qwind.prg',0,0
BAT_REPORT_OUTPUT,2,'ex2bus_nodist10_fault30_Qwind.prg',0,0
BAT_READ,0,'ex2bus_wt_Qconst_sol_v33_15.0_nodist.raw'
BAT_FNSL,0,0,0,1,2,0,0,0
BAT_CONG,0
BAT_CONL,0,1,1,0,0,0.0, 100.0,0.0, 100.0
BAT_CONL,0,1,2,0,0,0.0, 100.0,0.0, 100.0
BAT_CONL,0,1,3,0,0,0.0, 100.0,0.0, 100.0
BAT_ORDR,0
BAT_FACT
BAT_TYSL,0
BAT_DYRE_NEW,1,1,1,1,'ex2bus_wt_Qconst_G1.dyr',,,,;
BAT_DYNAMICS_SOLUTION_PARAM_2,50,,,,,,,, 0.8,, 0.005,,,,,,;
BAT_DOCU,0,1,0,3,1
@! add
BAT_MACHINE_ARRAY_CHANNEL,1,1,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,2,2,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,3,3,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,4,4,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,5,5,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,6,6,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,7,7,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,8,8,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,9,9,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,10,10,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,11,11,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,12,12,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,13,13,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,14,14,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,15,15,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,16,16,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,17,17,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,18,18,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,19,19,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,20,20,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,21,21,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,22,22,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,23,23,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,24,24,2,'1'," "
BAT_STATE_CHANNEL,25,12,'G1 STATE K'
BAT_STATE_CHANNEL,26,13,'G1 STATE K+1'
BAT_STATE_CHANNEL,27,14,'G1 STATE K+2'
BAT_STATE_CHANNEL,28,15,'G1 STATE K+3'
BAT_STATE_CHANNEL,29,16,'E1 STATE K'
BAT_STATE_CHANNEL,30,17,'E1 STATE K+1'
BAT_STATE_CHANNEL,31,18,'E1 STATE K+2'
BAT_STATE_CHANNEL,32,19,'E1 STATE K+3'
BAT_STATE_CHANNEL,33,20,'E1 STATE K+4'
BAT_STATE_CHANNEL,34,21,'E1 STATE K+5'
BAT_STATE_CHANNEL,35,22,'E1 STATE K+6'
BAT_STATE_CHANNEL,36,23,'E1 STATE K+7'
BAT_STATE_CHANNEL,37,24,'E1 STATE K+8'
BAT_STATE_CHANNEL,38,25,'E1 STATE K+9'
BAT_STATE_CHANNEL,39,26,'T1 STATE K'
BAT_STATE_CHANNEL,40,27,'T1 STATE K+1'
BAT_STATE_CHANNEL,41,28,'T1 STATE K+2'
BAT_STATE_CHANNEL,42,29,'T1 STATE K+3'
BAT_STATE_CHANNEL,43,30,'P1 STATE K'
BAT_STATE_CHANNEL,44,31,'P1 STATE K+1'
BAT_STATE_CHANNEL,45,32,'P1 STATE K+2'
BAT_VAR_CHANNEL,46,2,'G1 VAR L'
BAT_VAR_CHANNEL,47,3,'G1 VAR L+1'
BAT_VAR_CHANNEL,48,4,'G1 VAR L+2'
BAT_VAR_CHANNEL,49,5,'G1 VAR L+3'
BAT_VAR_CHANNEL,50,6,'E1 VAR L'
BAT_VAR_CHANNEL,51,7,'E1 VAR L+1'
BAT_VAR_CHANNEL,52,8,'E1 VAR L+2'
BAT_VAR_CHANNEL,53,9,'E1 VAR L+3'
BAT_VAR_CHANNEL,54,10,'E1 VAR L+4'
BAT_VAR_CHANNEL,55,11,'E1 VAR L+5'
BAT_VAR_CHANNEL,56,12,'E1 VAR L+6'
BAT_VAR_CHANNEL,57,13,'T1 VAR L'
BAT_VAR_CHANNEL,58,14,'T1 VAR L+1'
BAT_VAR_CHANNEL,59,15,'T1 VAR L+2'
BAT_VAR_CHANNEL,60,16,'T1 VAR L+3'
BAT_VAR_CHANNEL,61,17,'T1 VAR L+4'
BAT_VOLTAGE_AND_ANGLE_CHANNEL,62,-1,-1,1,'',''
BAT_VOLTAGE_AND_ANGLE_CHANNEL,64,-1,-1,2,'',''
BAT_BRANCH_P_AND_Q_CHANNEL,66,-1,-1,1,2,'1','',''
@! add
BAT_STRT,1,'ex2bus_wt_Qconst_sol_v33_15.0_nodist.out'
BAT_RUN,1, 40.0,10,1,0
BAT_REPORT_OUTPUT,1,'',0,0
BAT_PROGRESS_OUTPUT,1,'',0,0
```

Figure 3-B3. No fault

```
BAT_STRT,1,'ex2bus_nodist10_fault30_Qwind.out'
BAT_RUN,1, 10.0,10,1,0
BAT_DIST_BUS_FAULT,2,1, 345.0,0.0,-0.2E+10
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_nodist10_fault30_Qwind.out'
BAT_RUN,1, 10.1,10,1,0
BAT_DIST_CLEAR_FAULT,1
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_nodist10_fault30_Qwind.out'
BAT_RUN,1, 60.0,10,1,0
```

Figure 3-B4. 6 cycles fault (long): change the last part of Idv file

```
BAT_STRT,1,'ex2bus_nodist10_fault30_Qwind.out'
BAT_RUN,1, 10.0,10,1,0
BAT_DIST_BUS_FAULT,2,1, 345.0,0.0,-0.2E+10
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_nodist10_fault30_Qwind.out'
BAT_RUN,1, 10.05,10,1,0
BAT_DIST_CLEAR_FAULT,1
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_nodist10_fault30_Qwind.out'
BAT_RUN,1, 60.0,10,1,0
```

Figure 3-B5. 3 cycles fault (short): change the last part of Idv file

Network data after running idv file

After running the idv file, bus data, plant data, and machine data in PSS®E network data are collected and shown below.

	Bus Number	Bus Name	Base kV	Area Num	Area Name	Zone Num	Zone Name	Owner	Owner Name
	1	BUS1	345.0	1		1		1	
	2	BUS2	345.0	1		1		1	
✳									

Code	Voltage (pu)	Angle (deg)	Normal Vmax (pu)	Normal Vmin (pu)	Emergency Vmax (pu)	Emergency Vmin (pu)
2	1.0400	-131.94	1.1000	0.9000	1.1000	0.9000
2	1.0150	-134.95	1.1000	0.9000	1.1000	0.9000

Figure 3-B6. Bus data

	Bus Number	Bus Name	Id	Area Num	Area Name	Zone Num	Zone Name	Code	VSched (pu)	Remote Bus
	1	BUS1 345.00	1	1		1		2	1.0400	0
	2	BUS2 345.00	1	1		1		2	1.0150	0
✳										

In Service	PGen (MW)	PMax (MW)	PMin (MW)	QGen (Mvar)	QMax (Mvar)	QMin (Mvar)	Mbase (MVA)
☑	96.2746	999.0000	10.0000	47.4925	300.0000	-300.0000	100.00
☑	14.9900	15.0000	0.0000	-1.3516	7.2600	-7.2600	20.00
☑							

R Source (pu)	X Source (pu)	RTran (pu)	XTran (pu)	Gentap (pu)	Owner	Fraction 1	Owner
0.000000	0.173300	0.00000	0.00000	1.00000	1	1.000	0
0.000000	0.100000	0.00000	0.00000	1.00000	1	1.000	0

Fraction 2	Owner	Fraction 3	Owner	Fraction 4	Positive R (pu)	Subtransient X (pu)	Transient X (pu)
1.000	0	1.000	0	1.000			
1.000	0	1.000	0	1.000			

Figure 3-B7. Machine data

Synchronous X (pu)	Negative R (pu)	Negative X (pu)	Zero R (pu)	Zero X (pu)	Grounding Z units	Grounding R
					P.U. (Per Unit)	
					P.U. (Per Unit)	
					P.U. (Per Unit)	

Grounding X	Wind machine Control Mode	Wind Machine Power factor
	Not a wind machine	1.000
	+, - Q limits based on WP	0.900
	Not a wind machine	

Figure 3-B8. Machine data-continued

	Bus Number	Bus Name		Area Num	Area Name	Code	PGen (MW)	QGen (Mvar)
	1	BUS1	345.00	1		2	96.3	47.5
	2	BUS2	345.00	1		2	15.0	-1.4
*								

QMax (Mvar)	QMin (Mvar)	VSched (pu)	Remote Bus	Remote Bus Name	Voltage (pu)	RMPCT
300.0	-300.0	1.0400	0		1.0400	100.00
7.3	-7.3	1.0150	0		1.0150	100.00

Figure 3-B9. Plant data

Bus and Machine Quantity

With these three cases of fault, some key bus and machine quantity plots are shown below.

As shown in the general trend of no fault condition, it is under expectation that all quantities are constants, which is because they are in steady states from the beginning to the end. As can be seen in Figure, active power, reactive power and Vterm magnitude have reasonable values, which are close to the value we set in the Raw file.

Initial values are also reasonable in both 6 cycles fault and 3 cycles fault condition. But for 6 cycles fault, final values keep oscillating and are unable to settle to a stable value. However, for 3 cycles fault case, values can eventually settle to a stable value though some quantities decrease or increase a little bit comparing to the initial values. The comparison plots between three cycles fault and six cycles fault case are shown in Figure.

This might be because the Type 3 wind model need improvement when longer fault occurs in the network.

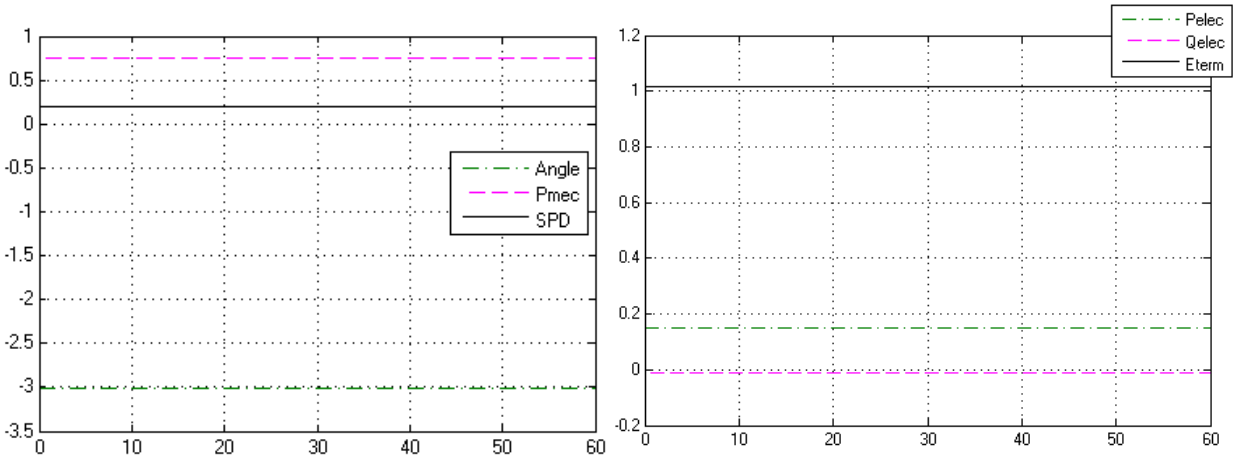
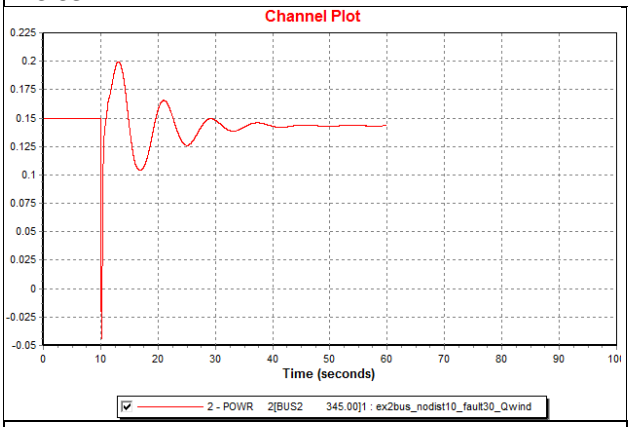
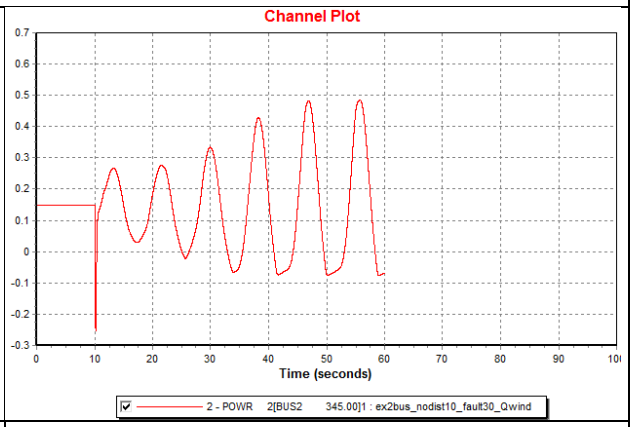
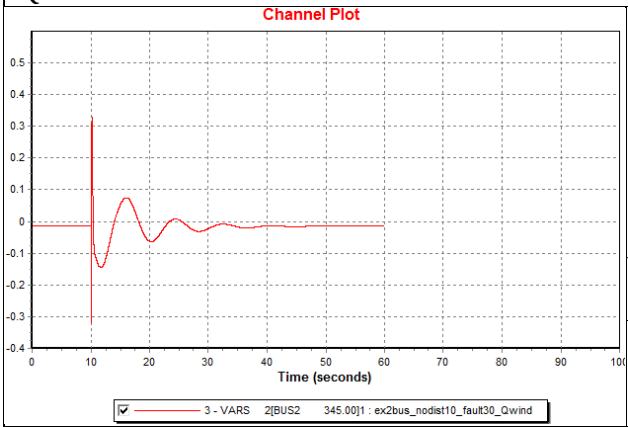
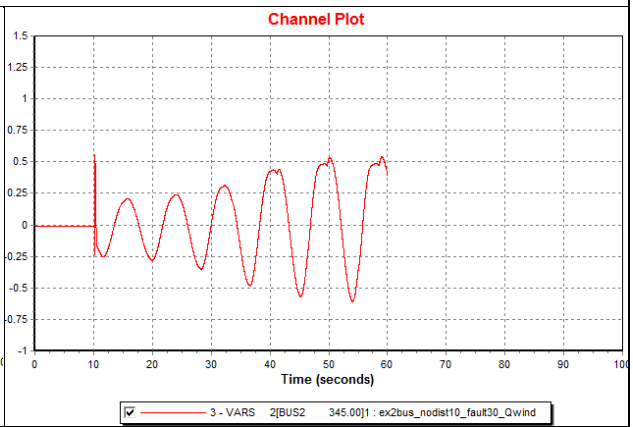
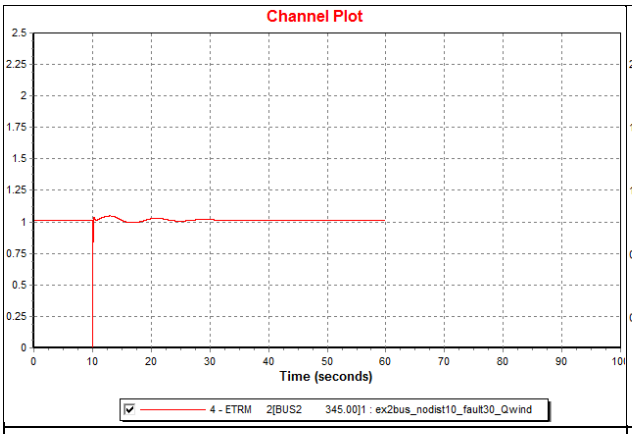
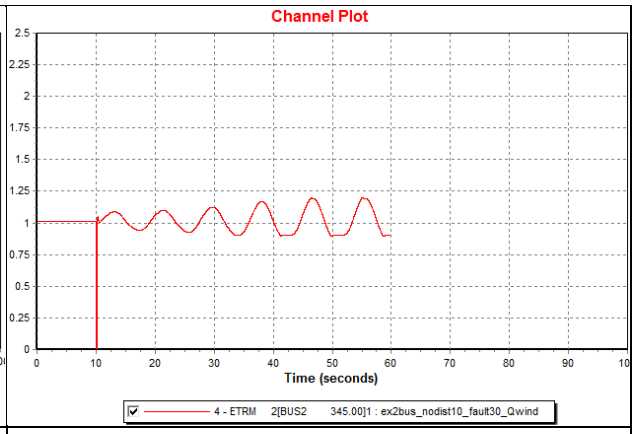


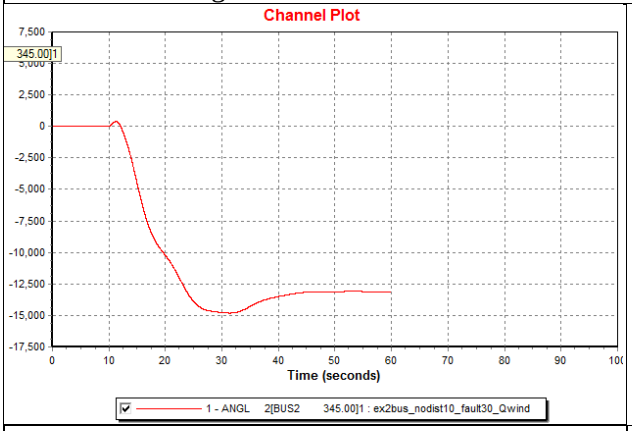
Figure 3-B10. No fault

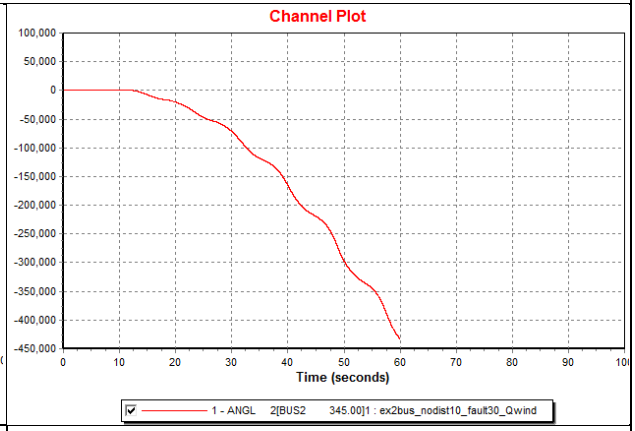
Three cycles fault	Six cycles fault
Pelec	
	
Qelec	
	
Eterm	



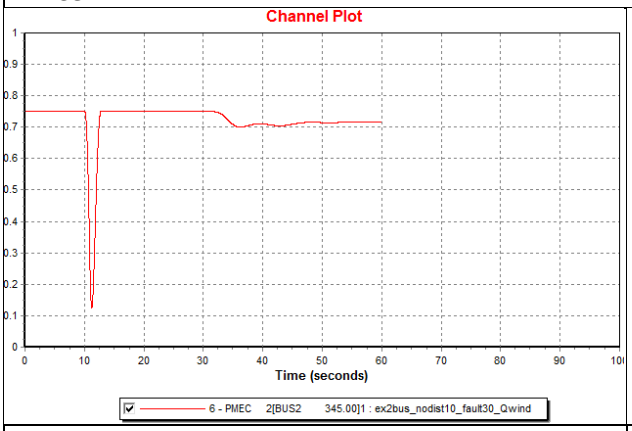


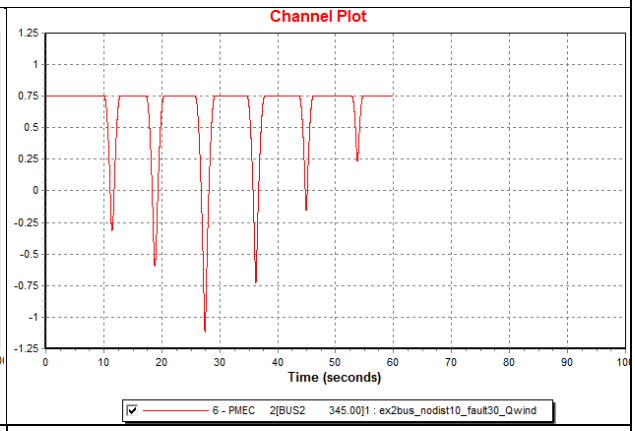
Bus 2 Vterm Angle



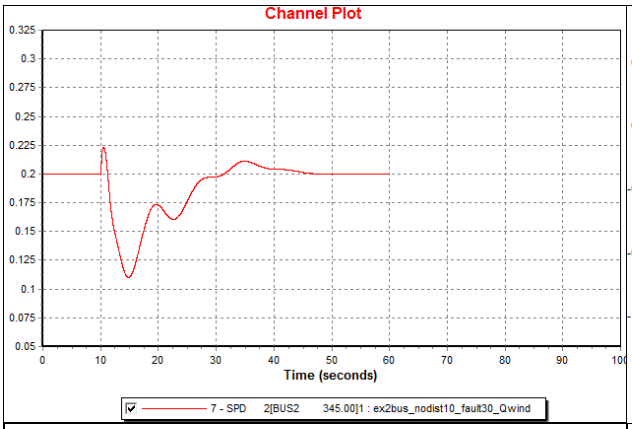


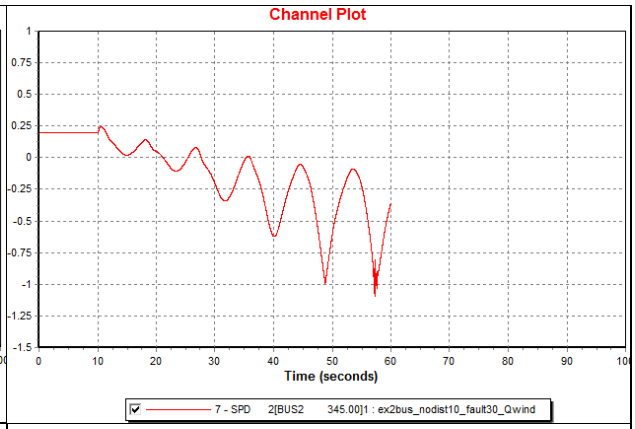
Pmec



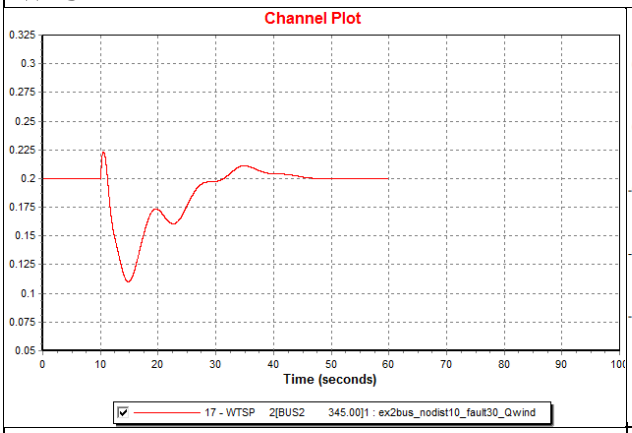


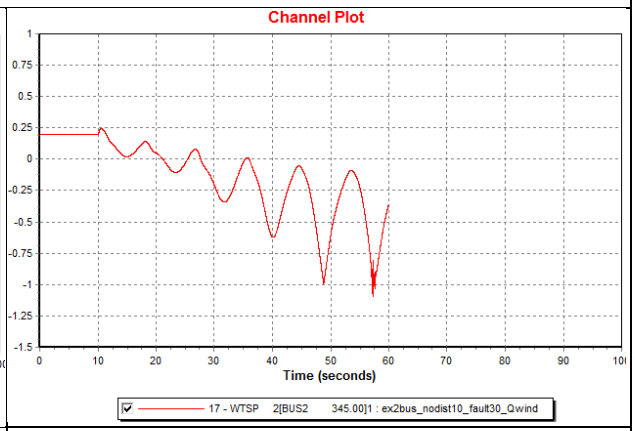
SPD



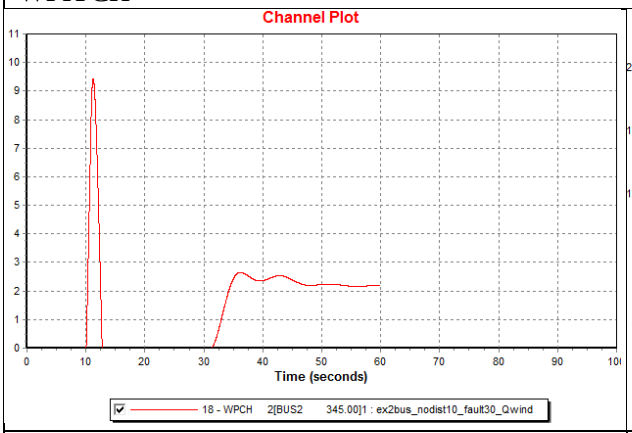


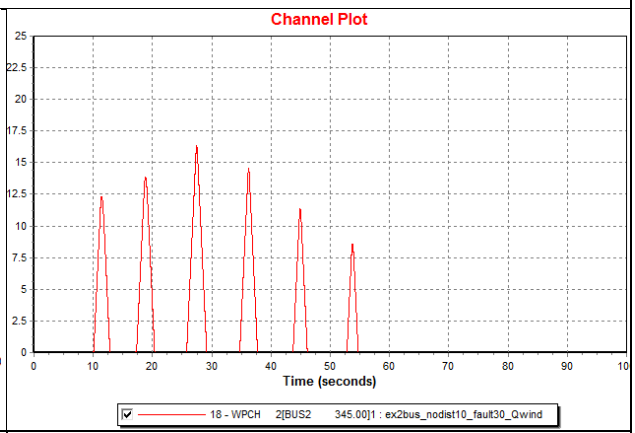
WTSP



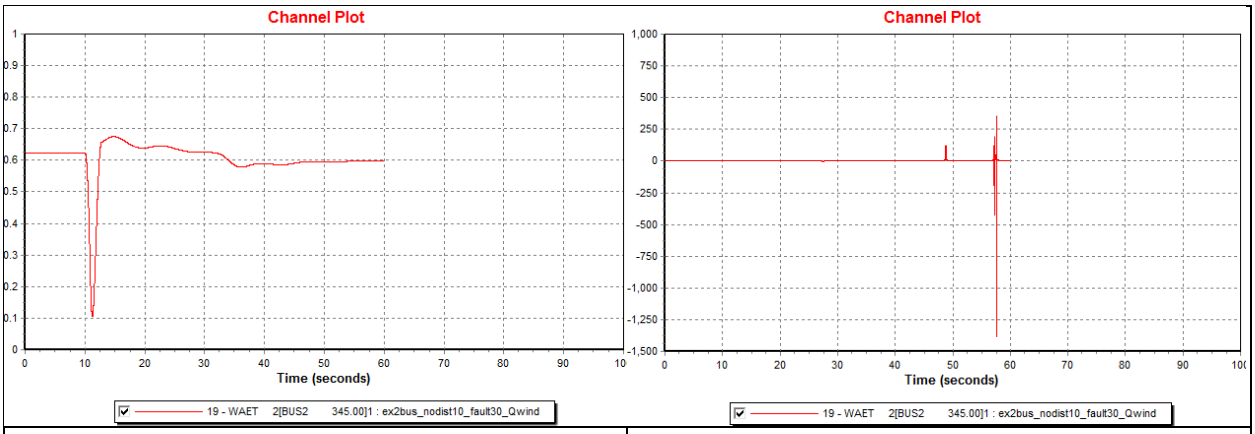


WPITCH





WAEROT



WPCM

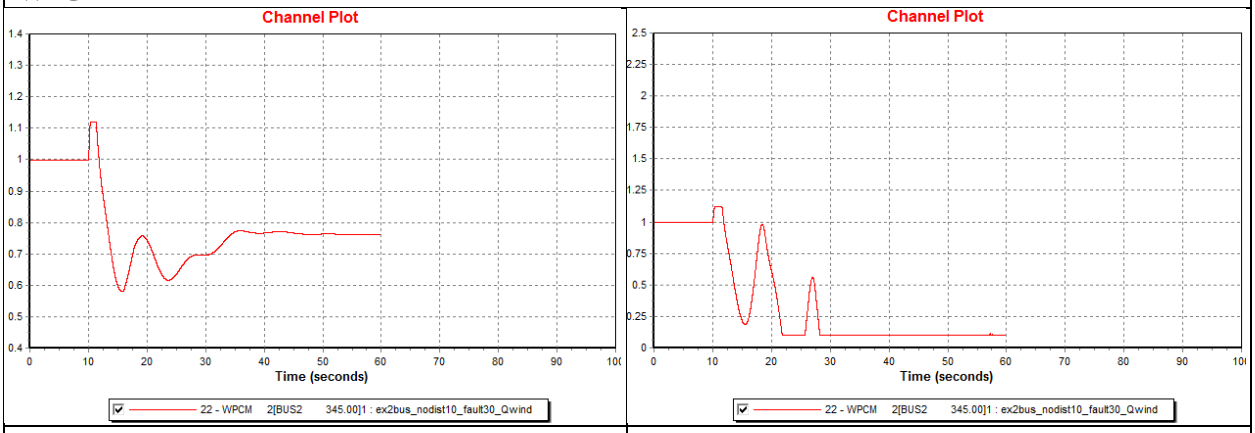


Figure 3-B11. three cycles fault and six cycles fault

Internal States and Model Output Comparison

a. WT3G1

Taking the input WEQCMD (State K+7 in E1 model), WIPCMD (Figure), Vt magnitude and Vt angle from PSS®E, we tested the output Pelec and Qelec of WT3G1 in Simulink.

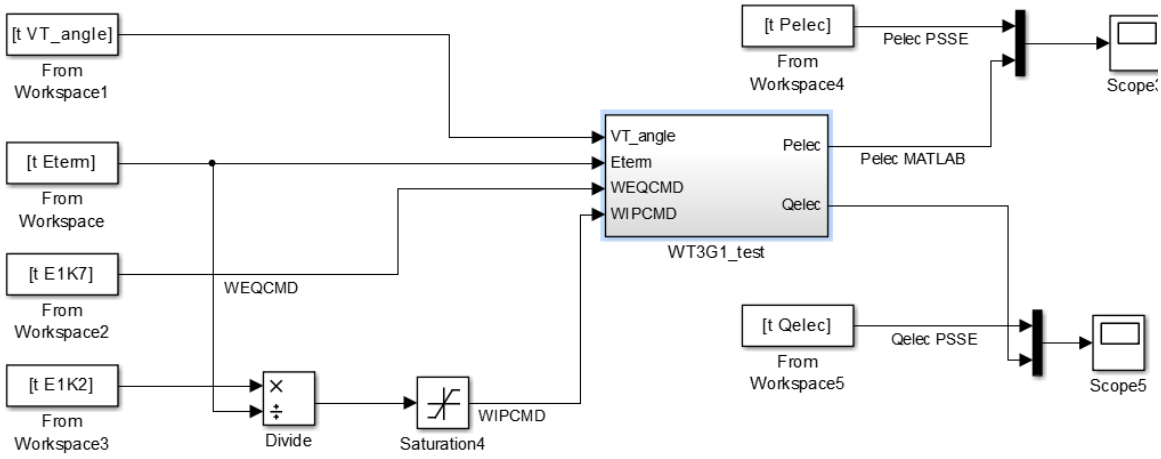


Figure 3-B12. Testing WT3G1 in MATLAB Simulink

As shown in the comparison plots below, all the internal states of WT3G1 (K, K+1, K+2, K+3) in our Simulink model matches very well with PSS®E. The implementation of translating Isourc to Pelec and Qelec are shown in section III.A.a. The plots of Pelec and Qelec in Simulink are close to PSS®E.

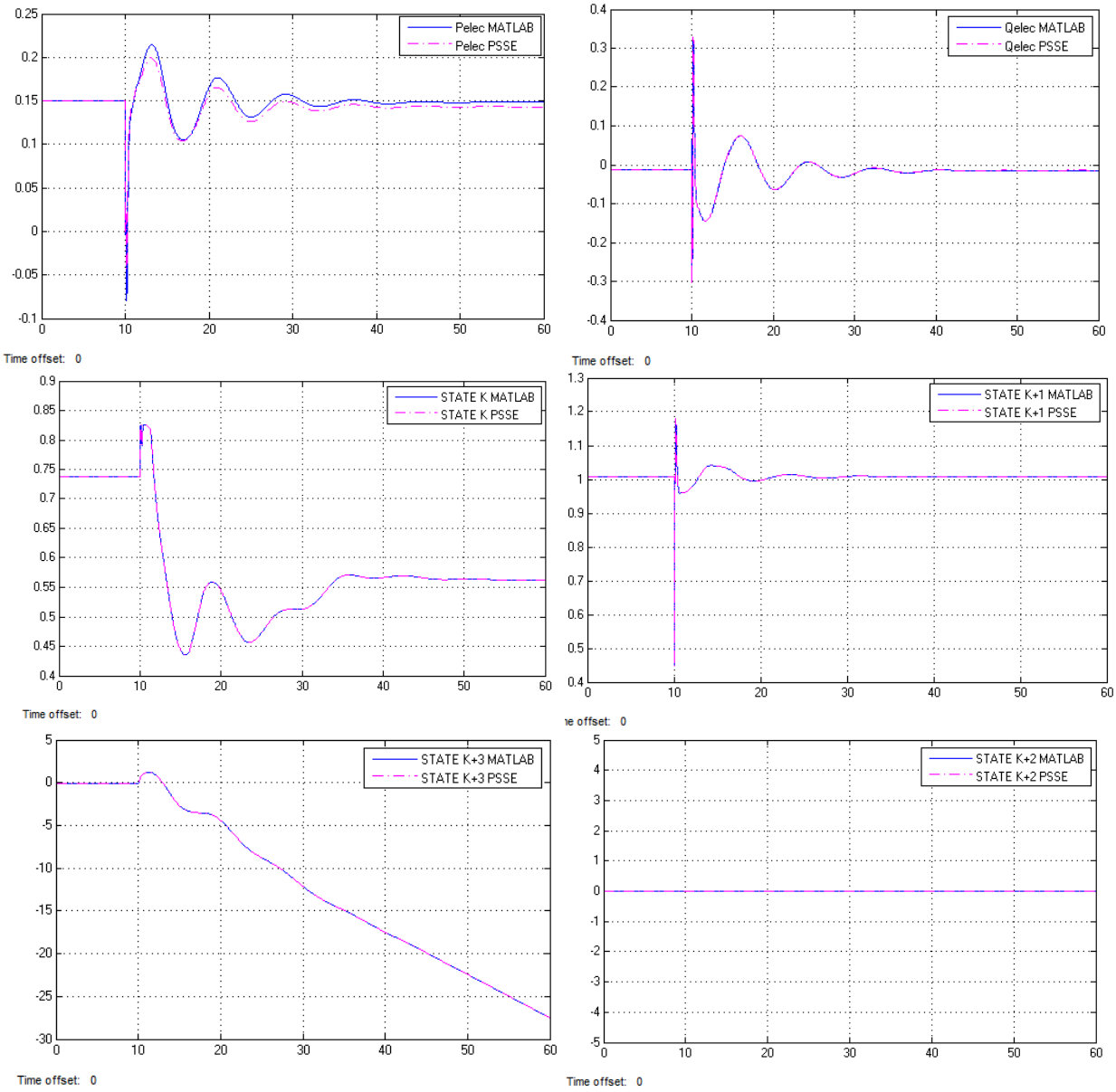


Figure 3-B13. Comparison between MATLAB and PSS®E for WT3G1

b. WT3E1

Taking the input Eterm, SPD, Qelec and Pelec from PSS®E, we tested the output WEQCMD (state K+7), Pord (K+2) and state K+5 of WT3G2 in Simulink. (Figure 3-B14)

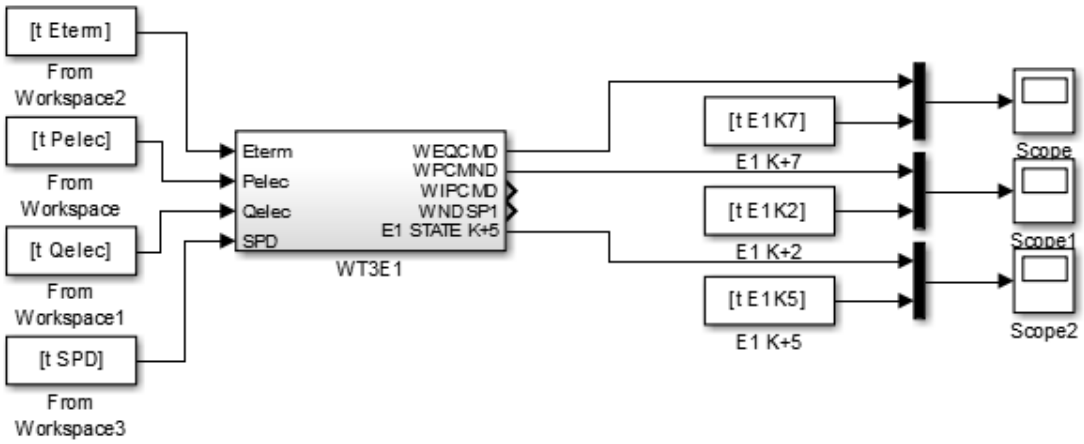


Figure 3-B14. Testing WT3E1 in MATLAB Simulink

1) Upper Branch of WT3E1

As shown in the comparison plots below, all the internal states of WT3E1 (K, K+1, K+4, K+6, K+8) except for K+7 (WEQCMD) in our Simulink model matches well with PSS®E.

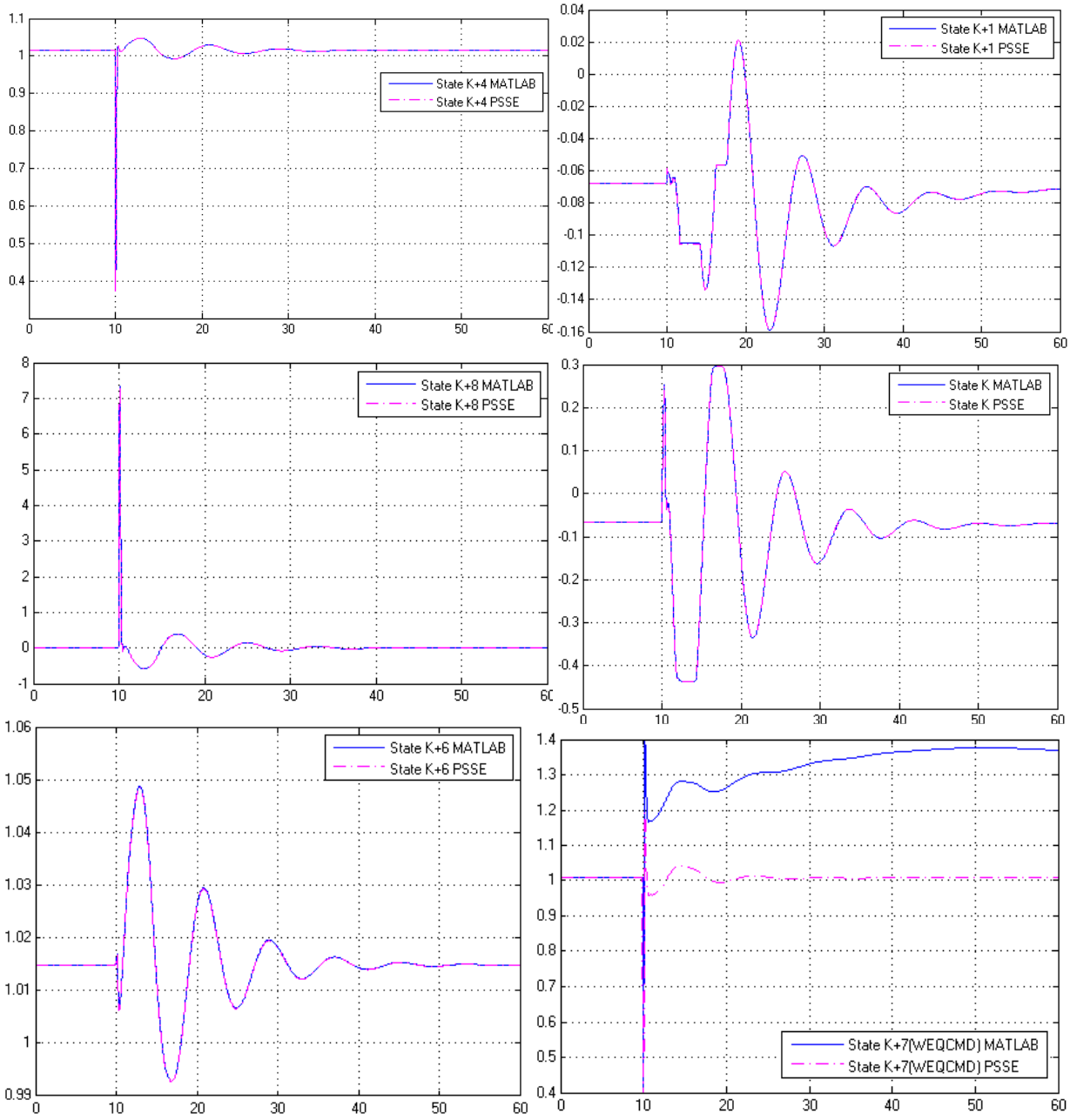


Figure 3-B15. Plotted states for WT3E1, upper branch

But actually for all the states here, when we zoomed in and found that there was still a small difference between these two simulations (Figure 3-B16 shows K+4). Further, we took a difference at state K+4 between PSS®E and MATLAB and zoomed in (Figure 3-B17). These slight differences might due to different simulation method inside MATLAB and PSS®E, which will lead to unstable final value such as State K+7 (WEQCMD). Further discussion about State K+7 will be put in section IV.

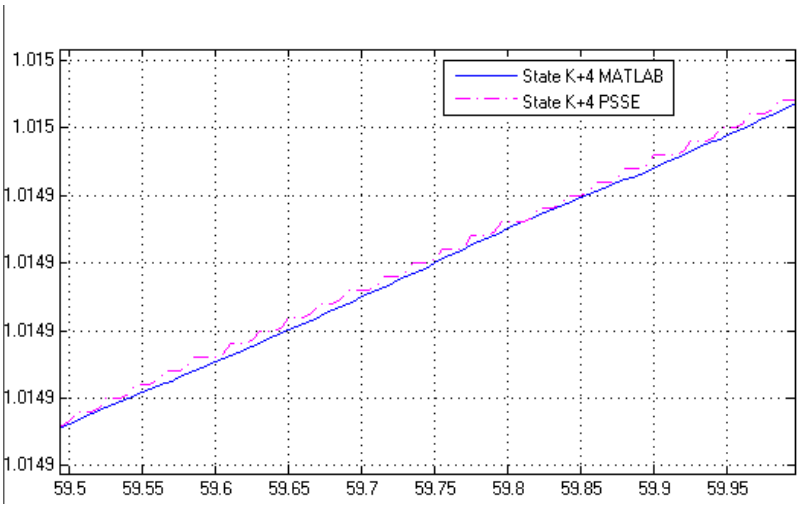


Figure 3-B16. STATE K+4 MATLAB and PSS®E

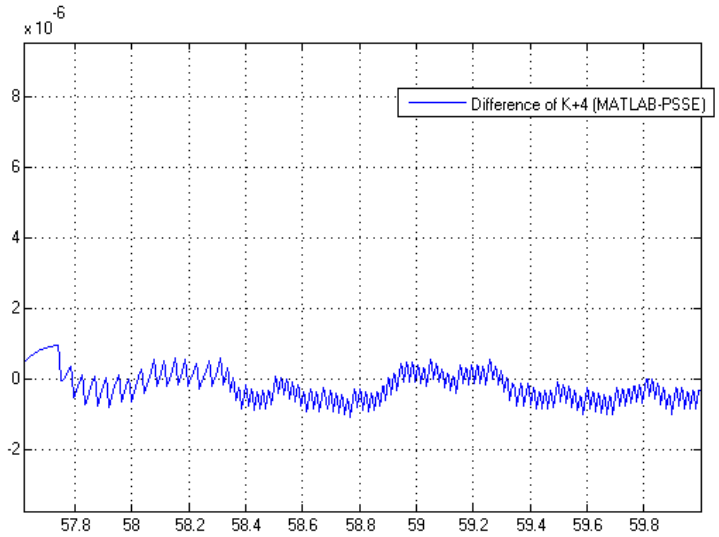


Figure 3-B17. Plotted difference between MATLAB and PSS®E

2) Lower Branch of WT3E1

As shown in the comparison plots below, all the internal states of WT3E1 lower branch (K+2, K+3, K+5) in our Simulink model matches well with PSS®E. State K+5 is Pord, which is used as an input in generator model.

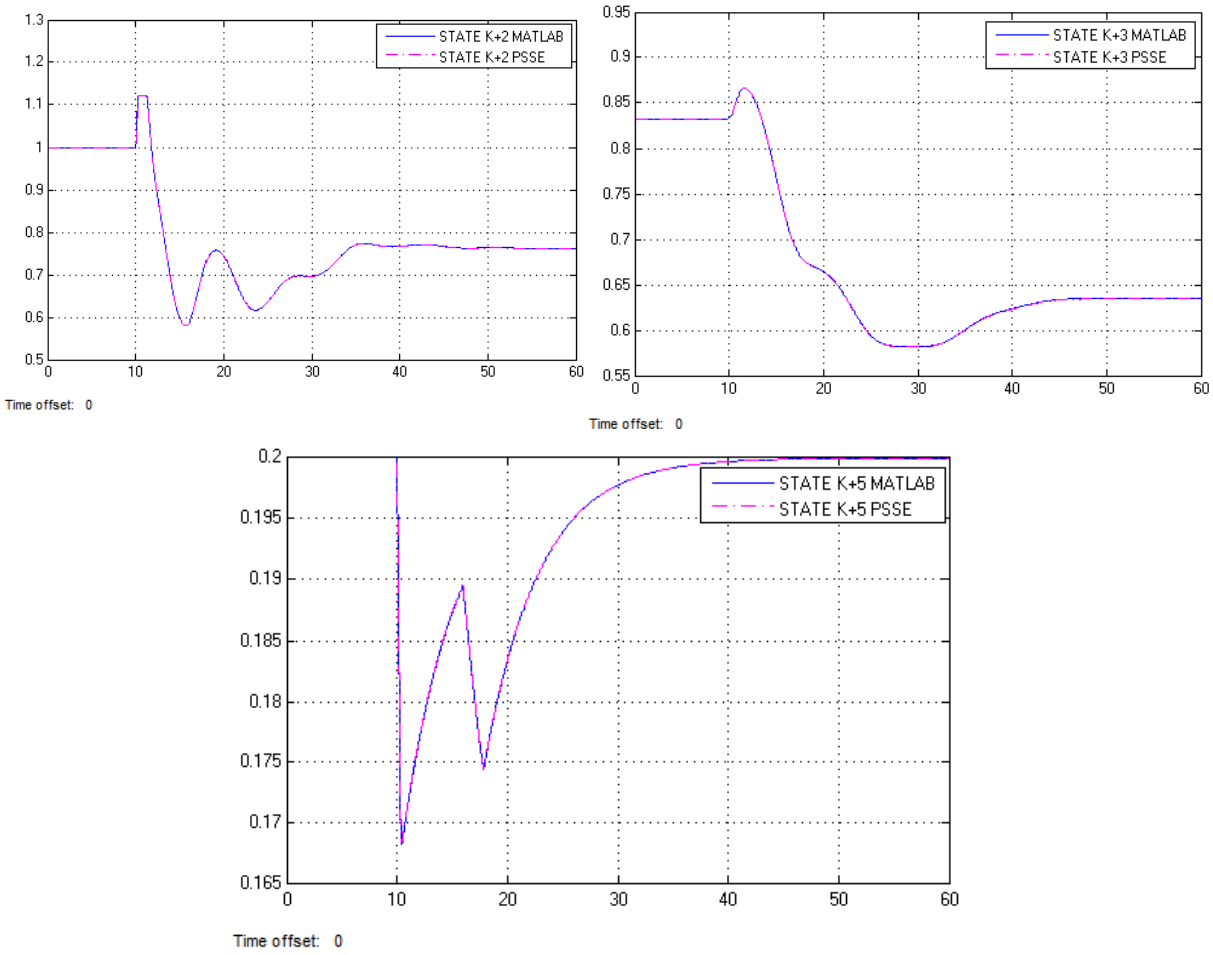


Figure 3-B18. Plotted states for WT3E1, lower branch

c. WT3P1

Taking the input Pord (state E1 K+2), SPD (state T1 K+2), state E1 K+5 from PSS®E, we tested the output WPITCH (state P1 K). (Figure 3-B19)

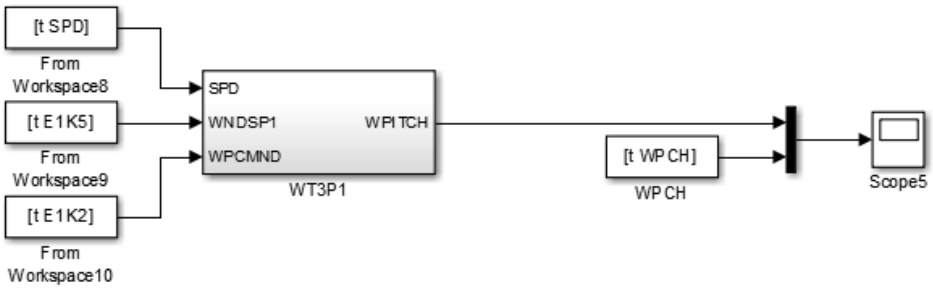


Figure 3-B19. Testing WT3P1 in MATLAB Simulink

As shown in the comparison plots below, internal states K+1, K+2 and K (WPTICH) in our Simulink model matches well with PSS®E, despite a slight difference.

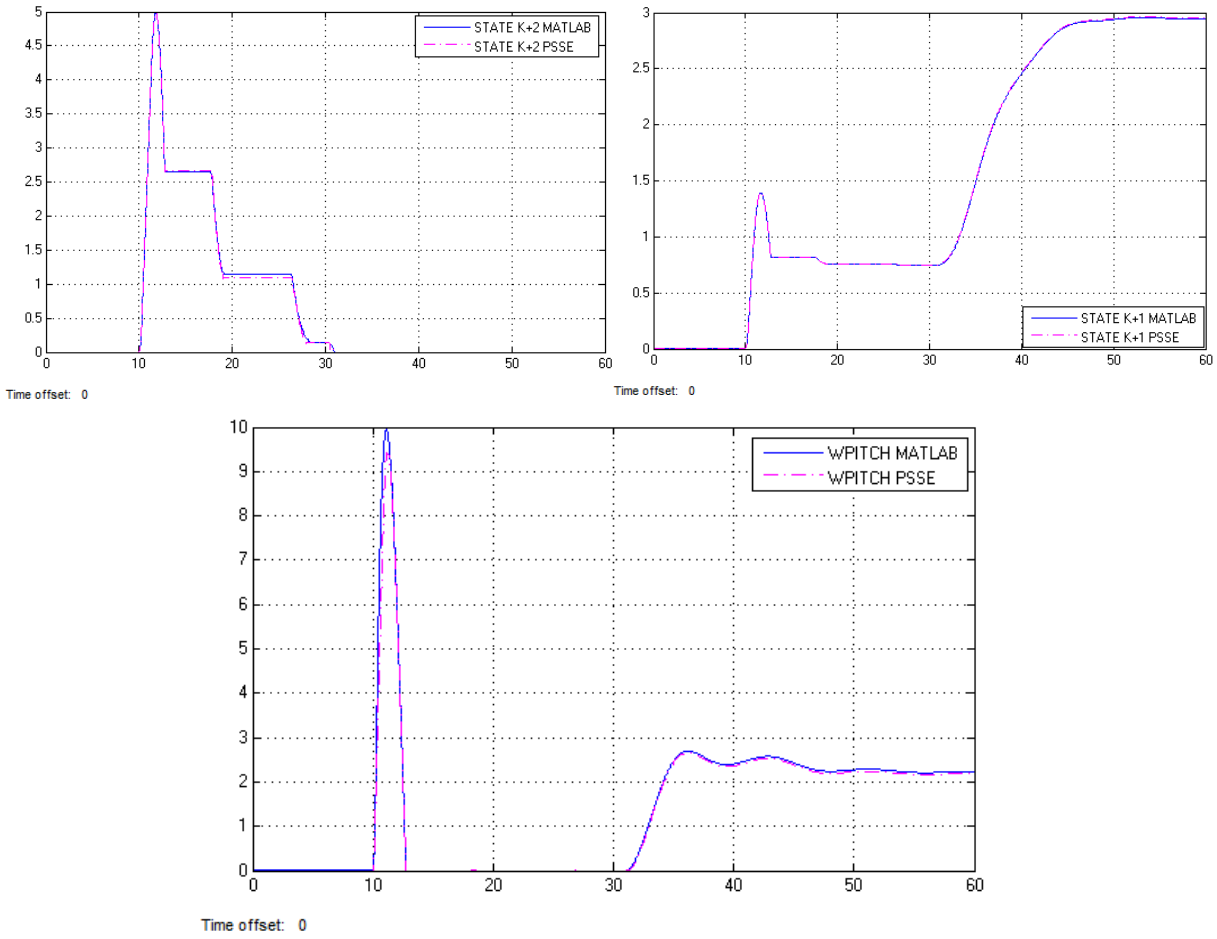


Figure 3-B20. Plotted comparisons between MATLAB and PSS®E for WT3P1

d. WT3T1

Taking the input WPITCH (state P1 K) and Pelec from PSS®E, we tested the output SPEED (state T1 K+2), WTRBSP, Rotor Angle Deviation (state T1 K+3). (Figure 3-B21)

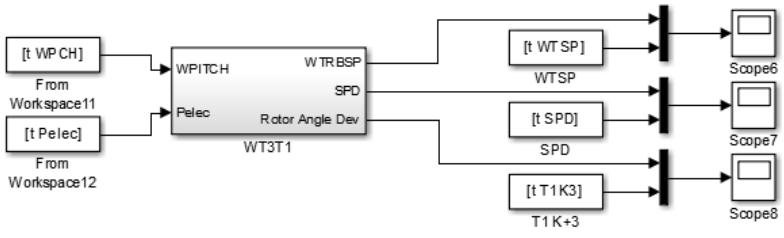


Figure 3-B21. Testing WT3T1 in MATLAB Simulink

1) Htfrac=0

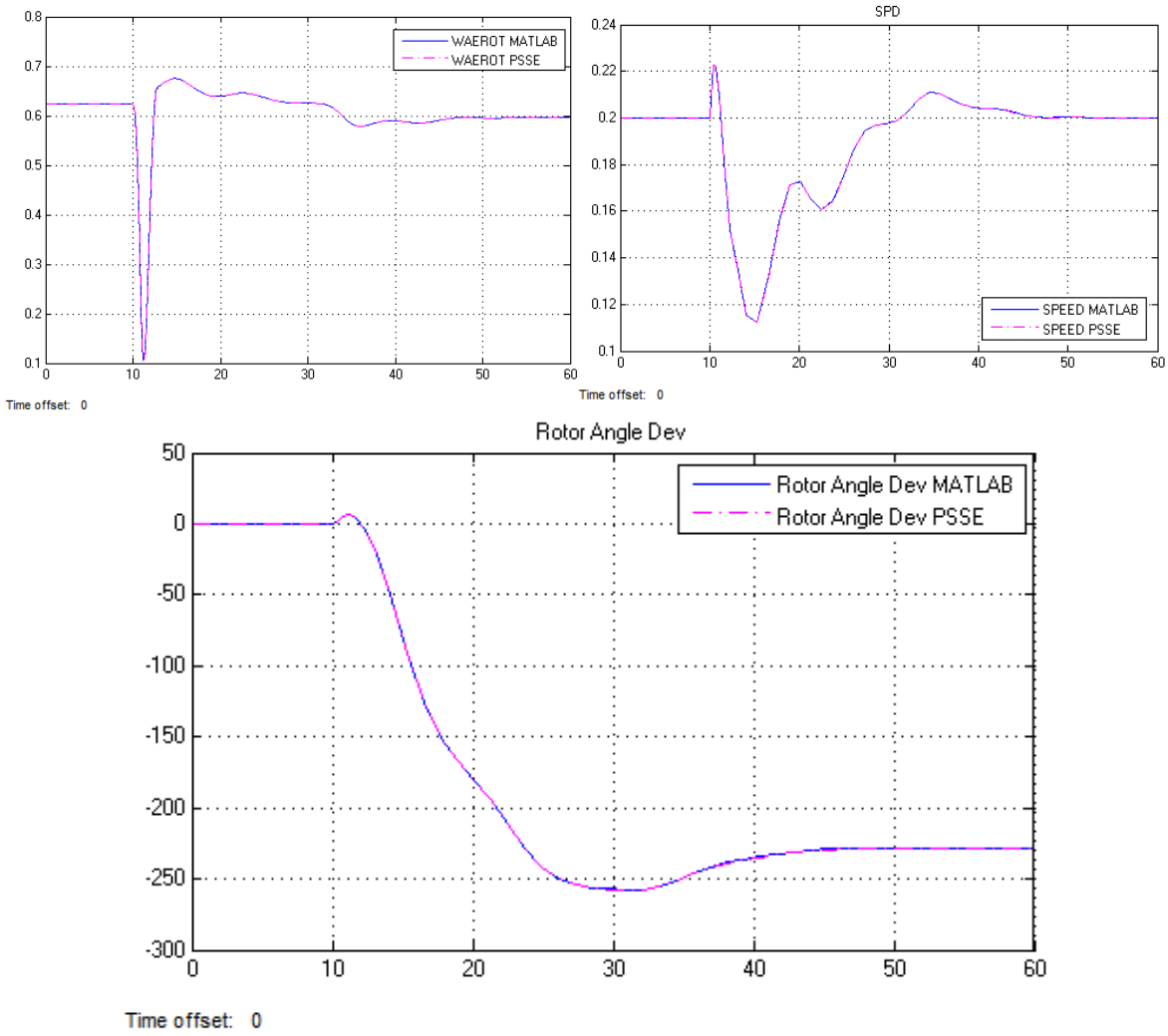


Figure 3-B22. Plotted comparisons for WT3T1, Htfrac=0

2) Htfrac=0.5

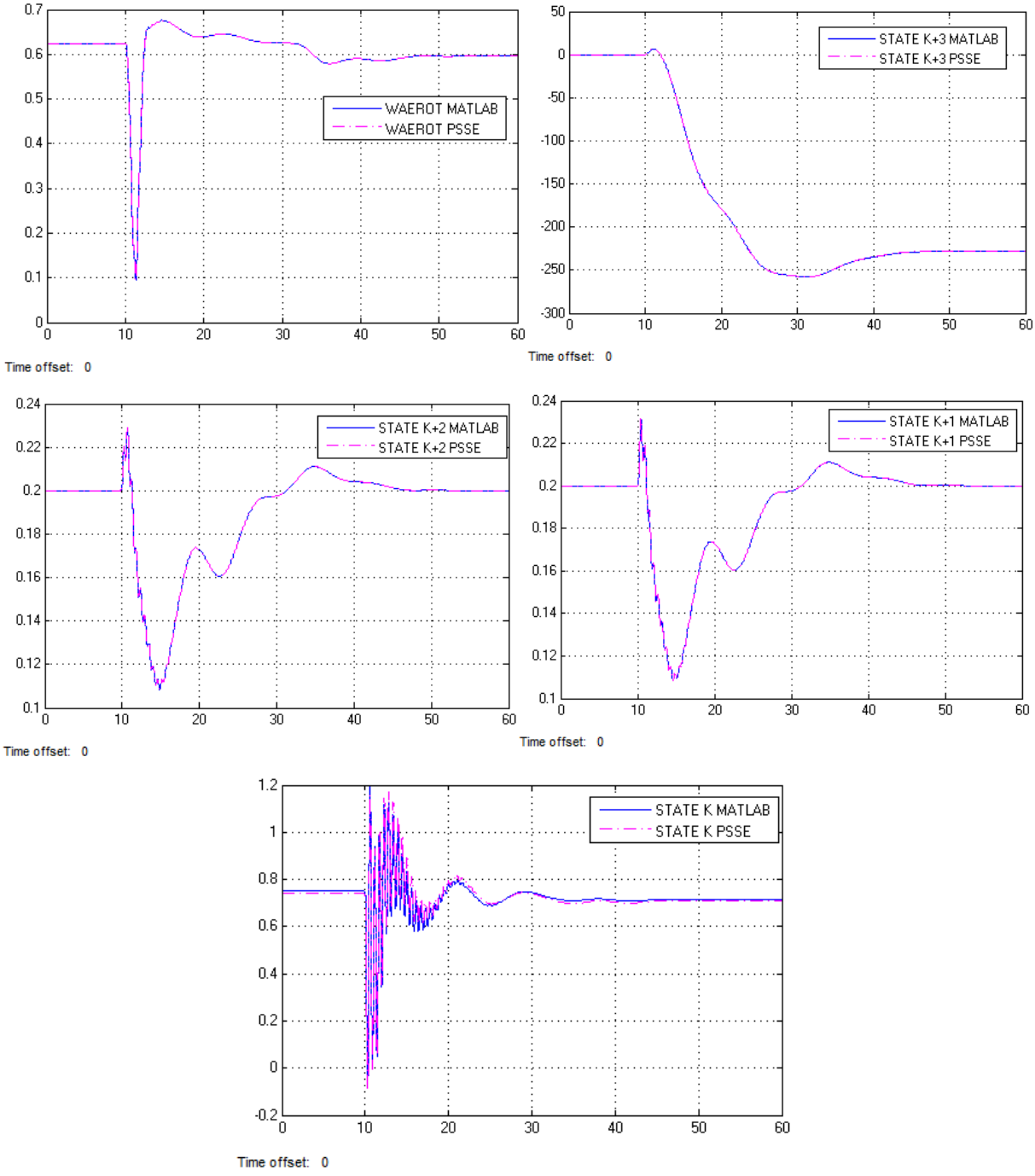


Figure 3-B23. Plotted comparisons for WT3T1, Htfrac=0.5

2. Using WT3G2 as the Generator

Power Flow Set Up

Raw file and dyr file, provided in Figure and Figure should be well repaired before creating command lines in idv file. Here we set the wind machine control mode to 2 and the power factor to 0.9 in bus 2 generator data. Ten machines are connected to the network.

```
0,   100.00, 33, 0, 1, 60.00     / PSS(R)E-33.3    THU, JUN 05 2014  18:18


    1,'BUS1       ', 345.0000,3,   1,   1,   1,1.04000,   0.0000,1.10000,0.90000,1.10000,0.90000
    2,'BUS2       ', 345.0000,2,   1,   1,   1,1.02000,  -3.3867,1.10000,0.90000,1.10000,0.90000
0 / END OF BUS DATA, BEGIN LOAD DATA
    2,'11',1,   1,   1,     0.000,     0.000,     0.000,     0.000,   107.990,   -38.840,   1,1,0
0 / END OF LOAD DATA, BEGIN FIXED SHUNT DATA
0 / END OF FIXED SHUNT DATA, BEGIN GENERATOR DATA
    1,'1 ',   108.863,    39.133,   300.000,  -300.000,1.04000,     0,    100.000, 0.00000E+0, 1.73300E-1, 0.00000E+0, 0.00000E+0,1.00000,1,  100.0,   999.000,    10.000,   1,1.0000
    2,'1 ',    15.000,     8.393,    10.000,   -10.000,1.02000,     0,     20.000, 0.00000E+0, 1.00000E-1, 0.00000E+0, 0.00000E+0,1.00000,1,  100.0,    15.000,     0.000,   1,1.0000,   1,0.0000,   1,0.0000,   1,0.0000,2, 0.9000
0 / END OF GENERATOR DATA, BEGIN BRANCH DATA
   1,    2,'1 ', 1.00000E-4, 5.76000E-2,   0.00010,  250.00,  250.00,  250.00,  0.00000,  0.00000,  0.00000,  0.00000,1,1,   0.00,   1,1.0000
0 / END OF BRANCH DATA, BEGIN TRANSFORMER DATA
0 / END OF TRANSFORMER DATA, BEGIN AREA DATA
  1,    1,     0.000,    10.000,'
0 / END OF AREA DATA, BEGIN TWO-TERMINAL DC DATA
0 / END OF TWO-TERMINAL DC DATA, BEGIN VSC DC LINE DATA
0 / END OF VSC DC LINE DATA, BEGIN IMPEDANCE CORRECTION DATA
0 / END OF IMPEDANCE CORRECTION DATA, BEGIN MULTI-TERMINAL DC DATA
0 / END OF MULTI-TERMINAL DC DATA, BEGIN MULTI-SECTION LINE DATA
0 / END OF MULTI-SECTION LINE DATA, BEGIN ZONE DATA
0 / END OF ZONE DATA, BEGIN INTER-AREA TRANSFER DATA
0 / END OF INTER-AREA TRANSFER DATA, BEGIN OWNER DATA
0 / END OF OWNER DATA, BEGIN FACTS DEVICE DATA
0 / END OF FACTS DEVICE DATA, BEGIN SWITCHED SHUNT DATA
0 / END OF SWITCHED SHUNT DATA, BEGIN GNE DATA
0 / END OF GNE DATA, BEGIN INDUCTION MACHINE DATA
0 / END OF INDUCTION MACHINE DATA
```

Figure 3-B24. RAW file for G2 model

```
1,'GENROU',1, 6.1000, 0.0500, 1.0000, 0.1500,  3.3800, 0.0000, 1.5750, 1.5120, 0.2910, 0.3900, 0.1733, 0.0787, 0, 0/

1,'ESDC1A',1, 0.0, 25.0, 0.2, 0.0, 0.0, 100.0, -100.0, -0.05, 0.57, 0.091, 0.35, 0.0, 2.846, 0, 3.795, 0/

2 'WT3G2' 1    10
  0.02000 0.02000 0.0000 0.0000 0.10000 1.5000 0.50000 0.9000 1.1100 1.2000 2.0000 5.0000 0.20000E-01 /

2 'WT3E1' 1    0 0 2 0 0 '1 '
  0.15000 18.000 5.0000 0.0000 0.50000E-01 3.0000 0.60000 1.1200 0.40000E-01 0.43600 -0.43600 1.1000 0.20000E-01 0.45000 -0.45000
  5.000 0.1000 0.9000 1.1000 40.000 0.5000 1.45000 0.50000E-01 0.50000E-01 1.0000 0.30000 0.69000 0.78000 0.98000 0.74000 1.200 /

2 'WT3T1' 1
  1.2500 4.9500 0.0000 0.70000E-02 21.980 0.0000 1.8000 1.5000 /

2 'WT3P1' 1
  0.30000 150.00 25.000 3.0000 30.000 0.0000 27.000 10.000 1.0000 /

//ajf 1 'IEESGO',1, 0.1,  0.0,0.12, 0.25,6.0, 0.0,18.94,   0.7,  0.0, 100,  -100/

//ajf 1 'PSS2A', 1, 1, 0, 3, 0, 5, 1, 2.0, 2.0, 0.0, 2.0, 0.0000000, 2.0, 0.18, 1.0, 0.5, 0.1, 15.0, 0.15, 0.03,
//    0.15, 0.03, 1000, -1000/
```

Figure 3-B25. DYR file for G2 model

There are three different idv files shown below. We ran the simulation in 40 seconds, within which 6 cycles fault condition in 10 second (long fault) and 3 cycles fault condition in 10 second (short fault) are created. Details are shown in Figure 3-B26, Figure 3-B27 and Figure 3-B28 respectively.

```
@! File:"C:\Users\ECE497\Documents\ex2bus_10nodist_30fault_uirecord.idv", generated on FRI, MAY 23 2014  14:03, release 33.03.00
BAT_PROGRESS_OUTPUT,2,'ex2bus_wt_Qconst_sol_v33_15.0.prg',0,0
BAT_REPORT_OUTPUT,2,'ex2bus_wt_Qconst_sol_v33_15.0.prg',0,0
BAT_READ,0,'ex2bus_wt_Qconst_sol_v33_15.0.raw'
BAT_FNSL,0,0,0,1,2,0,0,0
BAT_CONG,0
BAT_CONL,0,1,1,0,0,0.0, 100.0,0.0, 100.0
BAT_CONL,0,1,2,0,0,0.0, 100.0,0.0, 100.0
BAT_CONL,0,1,3,0,0,0.0, 100.0,0.0, 100.0
BAT_ORDR,0
BAT_FACT
BAT_TYSL,0
BAT_DYRE_NEW,1,1,1,1,'ex2bus_wt_Qconst_G2.dyr',,,,;
BAT_DYNAMICS_SOLUTION_PARAM_2,50,,,,,,,, 0.8,, 0.005,,,,,,;
@!BAT_BSYS,1,0,0.0,0.0,0,2,1,2,0,0
@!BAT_DOCU,1,0,0,3,1
BAT_DOCU,0,1,0,3,1
@! add
BAT_LOAD_ARRAY_CHANNEL,1,1,2,'11'," "
BAT_LOAD_ARRAY_CHANNEL,2,2,2,'11'," "
BAT_MACHINE_ARRAY_CHANNEL,3,1,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,4,2,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,5,3,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,6,4,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,7,5,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,8,6,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,9,7,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,10,8,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,11,9,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,12,10,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,13,11,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,14,12,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,15,13,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,16,14,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,17,15,1,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,18,1,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,19,2,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,20,3,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,21,4,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,22,5,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,23,6,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,24,7,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,25,16,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,26,17,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,27,18,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,28,19,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,29,20,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,30,21,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,31,22,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,32,23,2,'1'," "
BAT_MACHINE_ARRAY_CHANNEL,33,24,2,'1'," "
BAT_STATE_CHANNEL,34,12,'G2 STATE K'
BAT_STATE_CHANNEL,35,13,'G2 STATE K+1'
BAT_STATE_CHANNEL,36,14,'G2 STATE K+2'
BAT_STATE_CHANNEL,37,15,'G2 STATE K+3'
BAT_STATE_CHANNEL,38,16,'G2 STATE K+4'
BAT_STATE_CHANNEL,39,17,'E1 STATE K'
BAT_STATE_CHANNEL,40,18,'E1 STATE K+1'
BAT_STATE_CHANNEL,41,19,'E1 STATE K+2'
BAT_STATE_CHANNEL,42,20,'E1 STATE K+3'
BAT_STATE_CHANNEL,43,21,'E1 STATE K+4'
BAT_STATE_CHANNEL,44,22,'E1 STATE K+5'
BAT_STATE_CHANNEL,45,23,'E1 STATE K+6'
BAT_STATE_CHANNEL,46,24,'E1 STATE K+7'
BAT_STATE_CHANNEL,47,25,'E1 STATE K+8'
BAT_STATE_CHANNEL,48,26,'E1 STATE K+9'
BAT_STATE_CHANNEL,49,27,'T1 STATE K'
BAT_STATE_CHANNEL,50,28,'T1 STATE K+1'
BAT_STATE_CHANNEL,51,29,'T1 STATE K+2'
BAT_STATE_CHANNEL,52,30,'T1 STATE K+3'
BAT_STATE_CHANNEL,53,31,'P1 STATE K'
BAT_STATE_CHANNEL,54,32,'P1 STATE K+1'
BAT_STATE_CHANNEL,55,33,'P1 STATE K+2'
BAT_VOLTAGE_AND_ANGLE_CHANNEL,56,-1,-1,1,'',''
BAT_VOLTAGE_AND_ANGLE_CHANNEL,58,-1,-1,2,'',''
```

Figure 3-B26. Public parts

```
BAT_BRANCH_P_AND_Q_CHANNEL,60,-1,-1,1,2,'1','',''
@! add
BAT_STRT,1,'ex2bus_wt_Qconst_sol_v33_15.0.out'
BAT_RUN,1, 10.0,10,1,0
BAT_DIST_BUS_FAULT,2,1, 345.0,0.0,-0.2E+10
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_wt_Qconst_sol_v33_15.0.out'
BAT_RUN,1, 10.05,10,1,0
BAT_DIST_CLEAR_FAULT,1
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_wt_Qconst_sol_v33_15.0.out'
BAT_RUN,1, 40.0,10,1,0
BAT_REPORT_OUTPUT,1,'',0,0
BAT_PROGRESS_OUTPUT,1,'',0,0
```

Figure 3-B27. 3 cycles fault (short): change the last part of Idv file

```
BAT_BRANCH_P_AND_Q_CHANNEL,60,-1,-1,1,2,'1','',''
@! add
BAT_STRT,1,'ex2bus_wt_Qconst_sol_v33_15.0.out'
BAT_RUN,1, 10.0,10,1,0
BAT_DIST_BUS_FAULT,2,1, 345.0,0.0,-0.2E+10
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_wt_Qconst_sol_v33_15.0.out'
BAT_RUN,1, 10.10,10,1,0
BAT_DIST_CLEAR_FAULT,1
BAT_CHANGE_CHANNEL_OUT_FILE,'ex2bus_wt_Qconst_sol_v33_15.0.out'
BAT_RUN,1, 40.0,10,1,0
BAT_REPORT_OUTPUT,1,'',0,0
BAT_PROGRESS_OUTPUT,1,'',0,0
```

Figure 3-B28. 6 cycles fault (long): change the last part of Idv file

Network data after running idv file

After running the idv file, bus data, plant data, and machine data in PSS®E network data are collected and shown below.

	Bus Number	Bus Name	Base kV	Area Num	Area Name	Zone Num	Zone Name	Owner	Owner Name
	1	BUS1	345.0	1		1		1	
	2	BUS2	345.0	1		1		1	
✳									

Owner	Owner Name	Code	Voltage (pu)	Angle (deg)	Normal Vmax (pu)	Normal Vmin (pu)	Emergency Vmax (pu)	Emergency Vmin (pu)
1		2	1.0440	-163.14	1.1000	0.9000	1.1000	0.9000
1		2	0.9377	-166.11	1.1000	0.9000	1.1000	0.9000

Figure 3-B29. Bus data

	Bus Number	Bus Name	Id	Area Num	Area Name	Zone Num	Zone Name	Code	VSched (pu)	Remote Bus
	1	BUS1 345.00	1	1		1		2	1.0400	0
	2	BUS2 345.00	1	1		1		2	1.0200	0
✳										

In Service	PGen (MW)	PMax (MW)	PMin (MW)	QGen (Mvar)	QMax (Mvar)	QMin (Mvar)	Mbase (MVA)
☑	88.3346	999.0000	10.0000	194.6636	300.0000	-300.0000	100.00
☑	14.0596	15.0000	0.0000	-138.3570	7.2648	-7.2648	20.00
☑							

R Source (pu)	X Source (pu)	RTran (pu)	XTran (pu)	Gentap (pu)	Owner	Fraction 1	Owner
0.000000	0.173300	0.00000	0.00000	1.00000	1	1.000	0
0.000000	0.100000	0.00000	0.00000	1.00000	1	1.000	0

Fraction 2	Owner	Fraction 3	Owner	Fraction 4	Positive R (pu)	Subtransient X (pu)	Transient X (pu)
1.000	0	1.000	0	1.000			
1.000	0	1.000	0	1.000			

Synchronous X (pu)	Negative R (pu)	Negative X (pu)	Zero R (pu)	Zero X (pu)	Grounding Z units	Grounding R
					P.U. (Per Unit)	
					P.U. (Per Unit)	
					P.U. (Per Unit)	

Grounding X	Wind machine Control Mode	Wind Machine Power factor
	Not a wind machine	1.000
	+, - Q limits based on WP	0.900
	Not a wind machine	

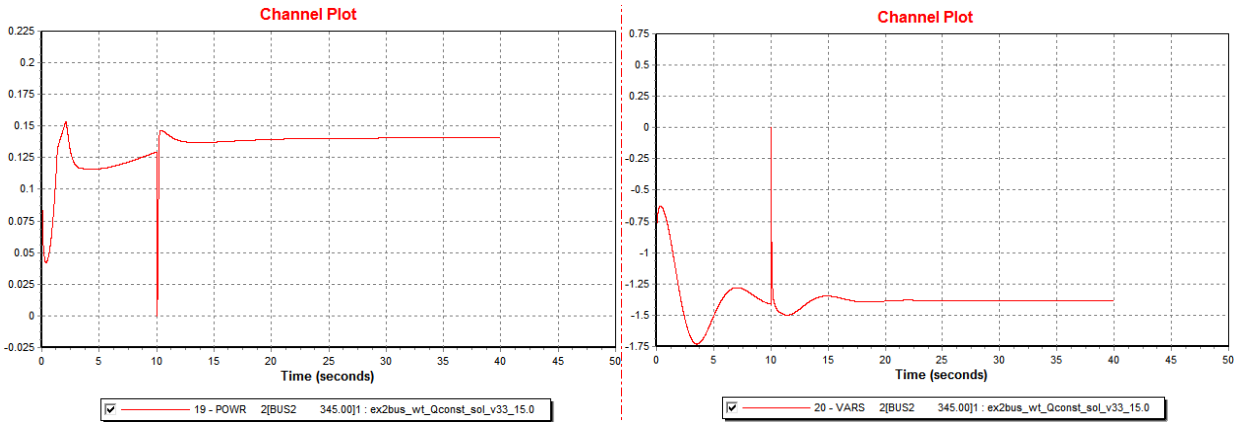
Figure 3-B30. Machine data

	Bus Number	Bus Name	Area Num	Area Name	Code	PGen (MW)	QGen (Mvar)
	1	BUS1 345.00	1		2	88.3	194.7
	2	BUS2 345.00	1		2	14.1	-138.4
*							

QMax (Mvar)	QMin (Mvar)	VSched (pu)	Remote Bus	Remote Bus Name	Voltage (pu)	RMPCT
300.0	-300.0	1.0400	0		1.0440	100.00
7.3	-7.3	1.0200	0		0.9377	100.00

Figure 3-B31. Plant Data

Bus and Machine Quantity



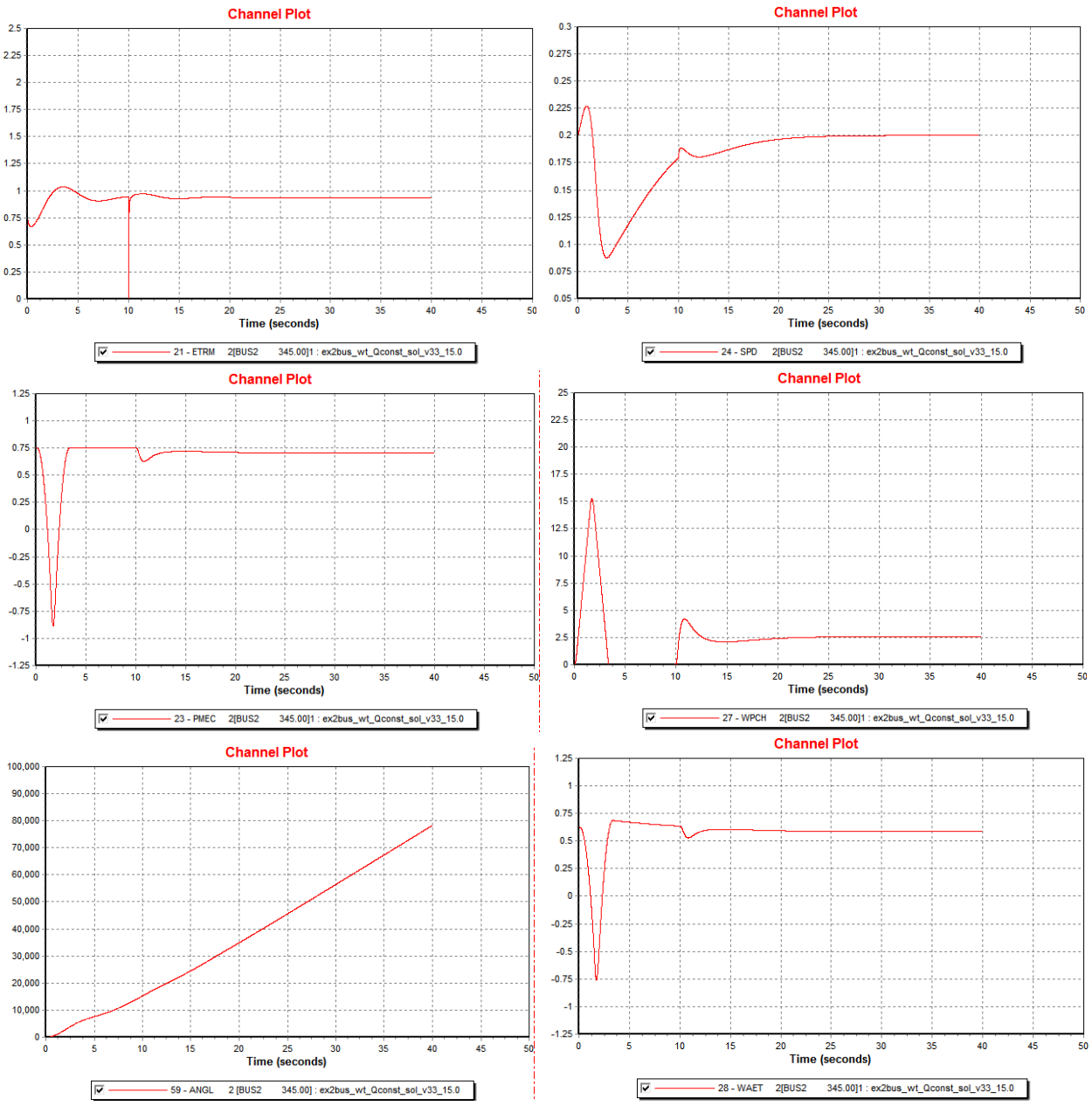


Figure 3-B32. Bus and Machine Quantity

a. WT3G2

Taking the input WEQCMD (State K+7 in E1 model), WIPCMD (Figure), Vt magnitude and Vt angle from PSS®E, we tested the output Pelec and Qelec of WT3G2 in Simulink.

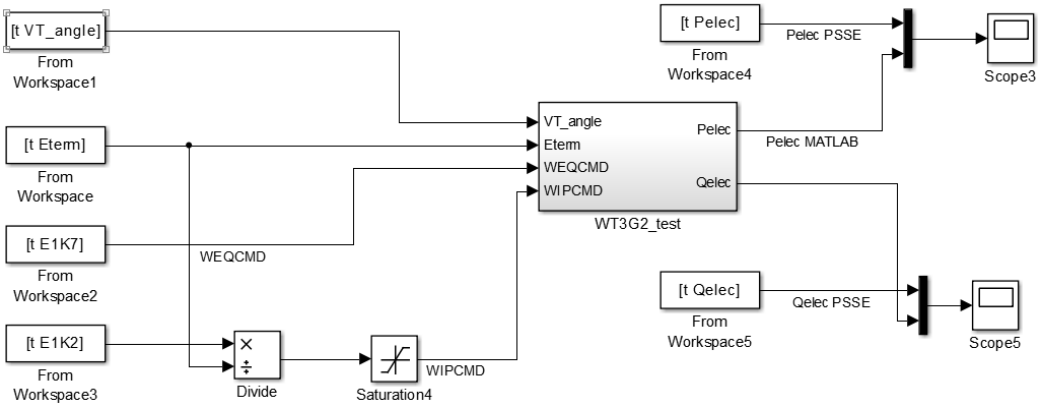


Figure 3-B33. Testing WT3G2 in MATLAB

Comparison of State K, K+1 and output Pelec, Qelec is shown below. The results match PSS®E simulation.

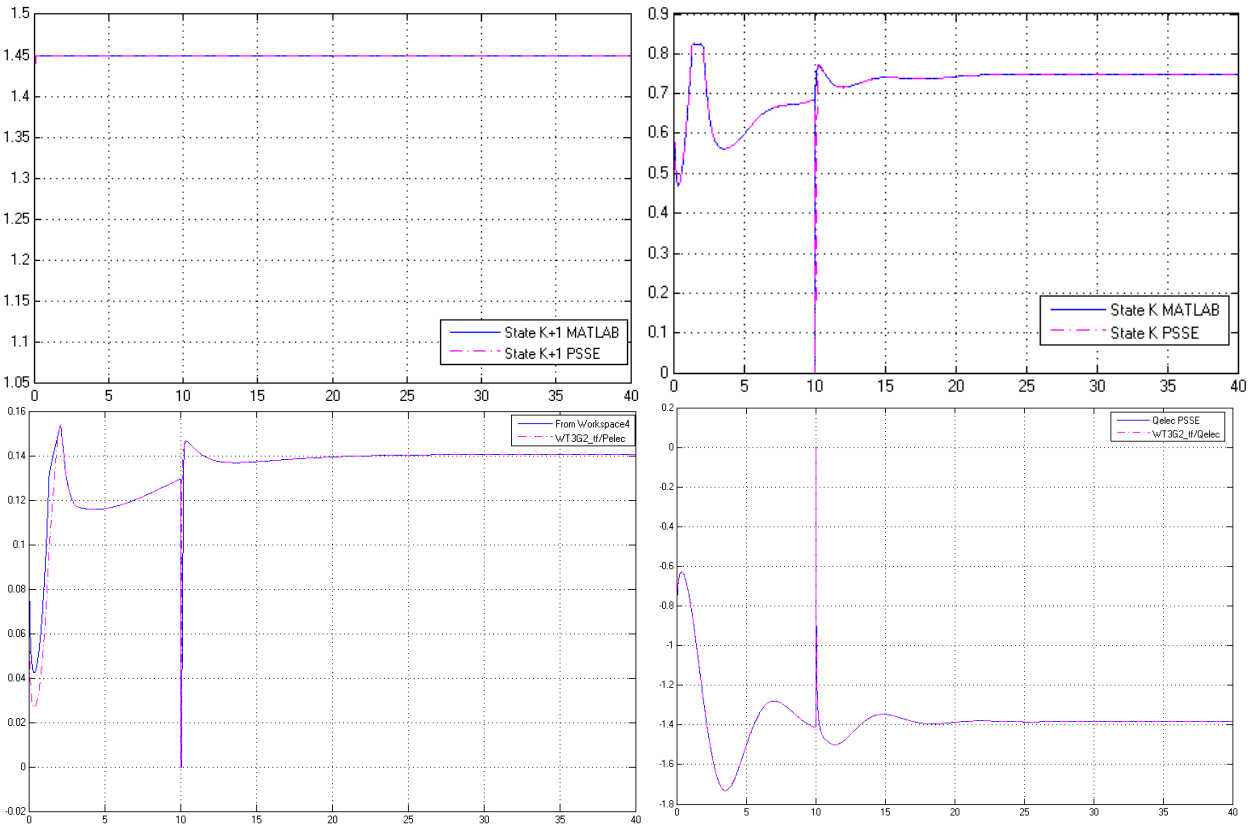


Figure 3-B34. Comparisons between MATLAb and PSS®E

b. WT3E1

Taking the input Eterm, SPD, Qelec and Pelec from PSS®E, we tested the output WEQCMD (state K+7), Pord (K+2) and state K+5 of WT3G2 in Simulink. (Figure 3-B35)

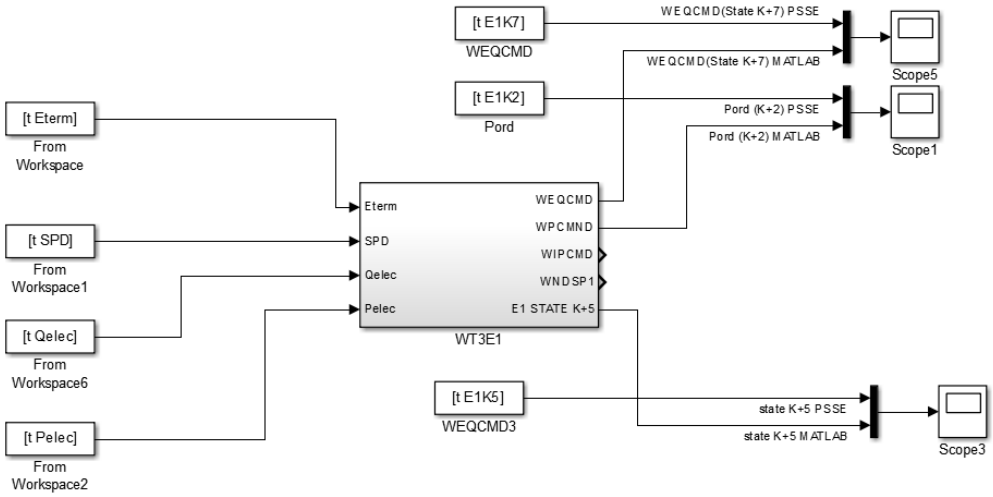
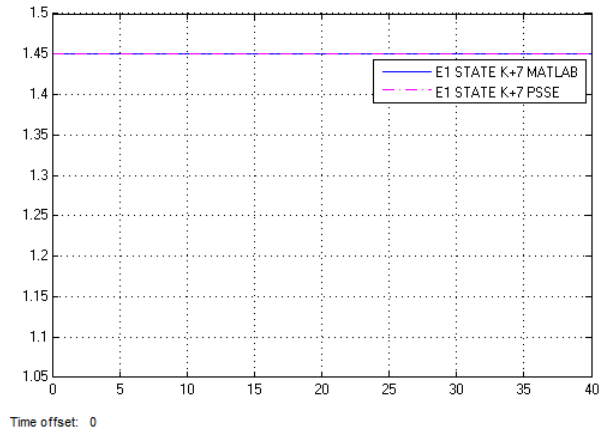
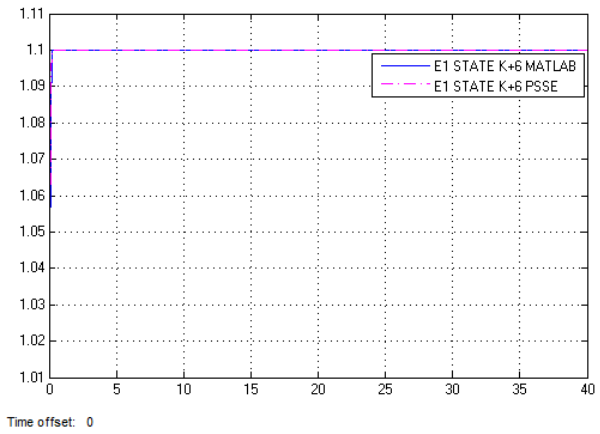
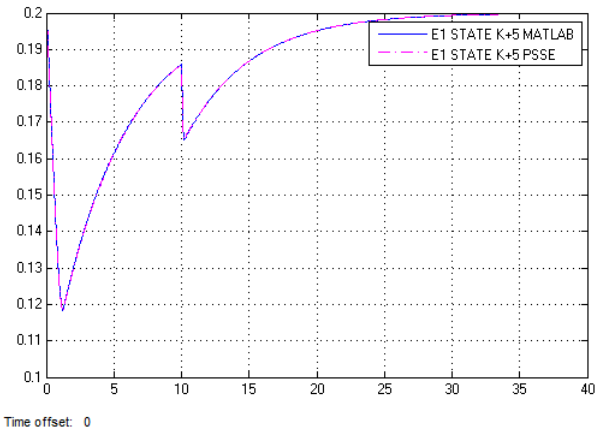


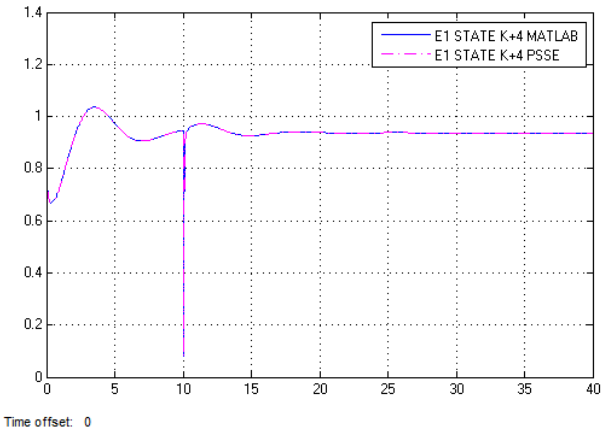
Figure 3-B35. Testing WT3E1 in MATLAB

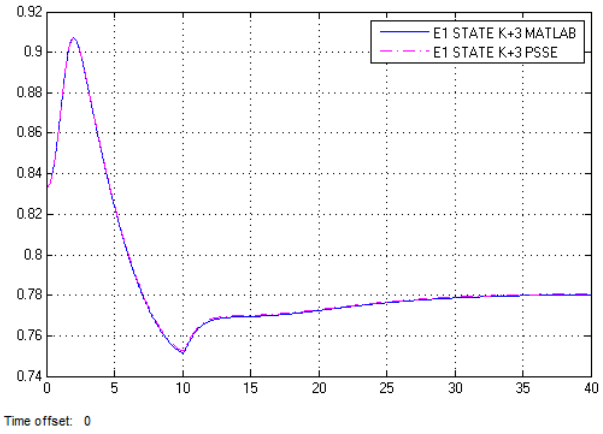
Comparison of the output WEQCMD (state K+7), Pord (K+2),state K+5 and other internal states is shown below. The results match PSS®E simulation.

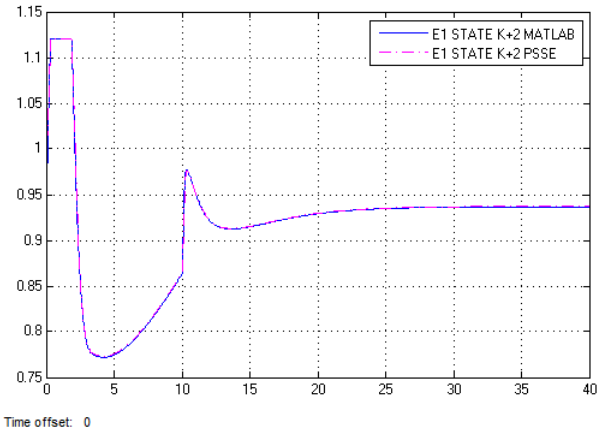












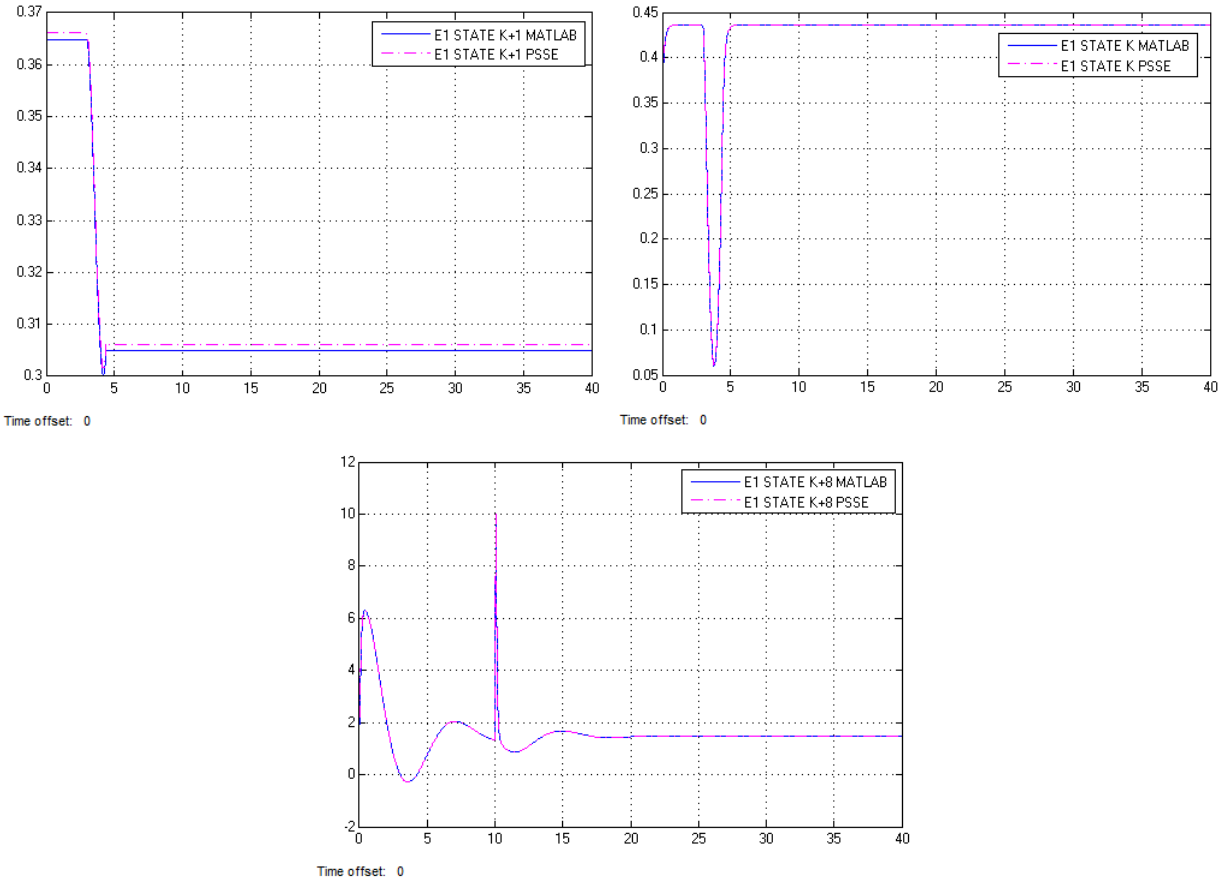


Figure 3-B36. Comparisons between MATLAB and PSS®E

c. WT3P1

Taking the input Pord (state E1 K+2), SPD (state T1 K+2), state E1 K+5 from PSS®E, we tested the output WPITCH (state P1 K). (Figure 3-B37)

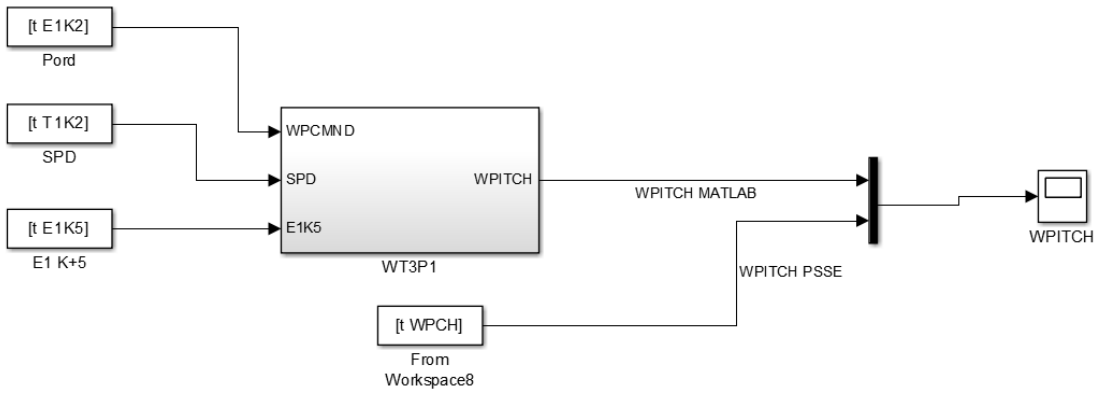


Figure 3-B37. Testing WPITCH in MATLAB

Comparison of the output WPITCH and internal states K+1 and K+2 is shown below. The results do not match PSS®E simulation very well and more investigation is still required.

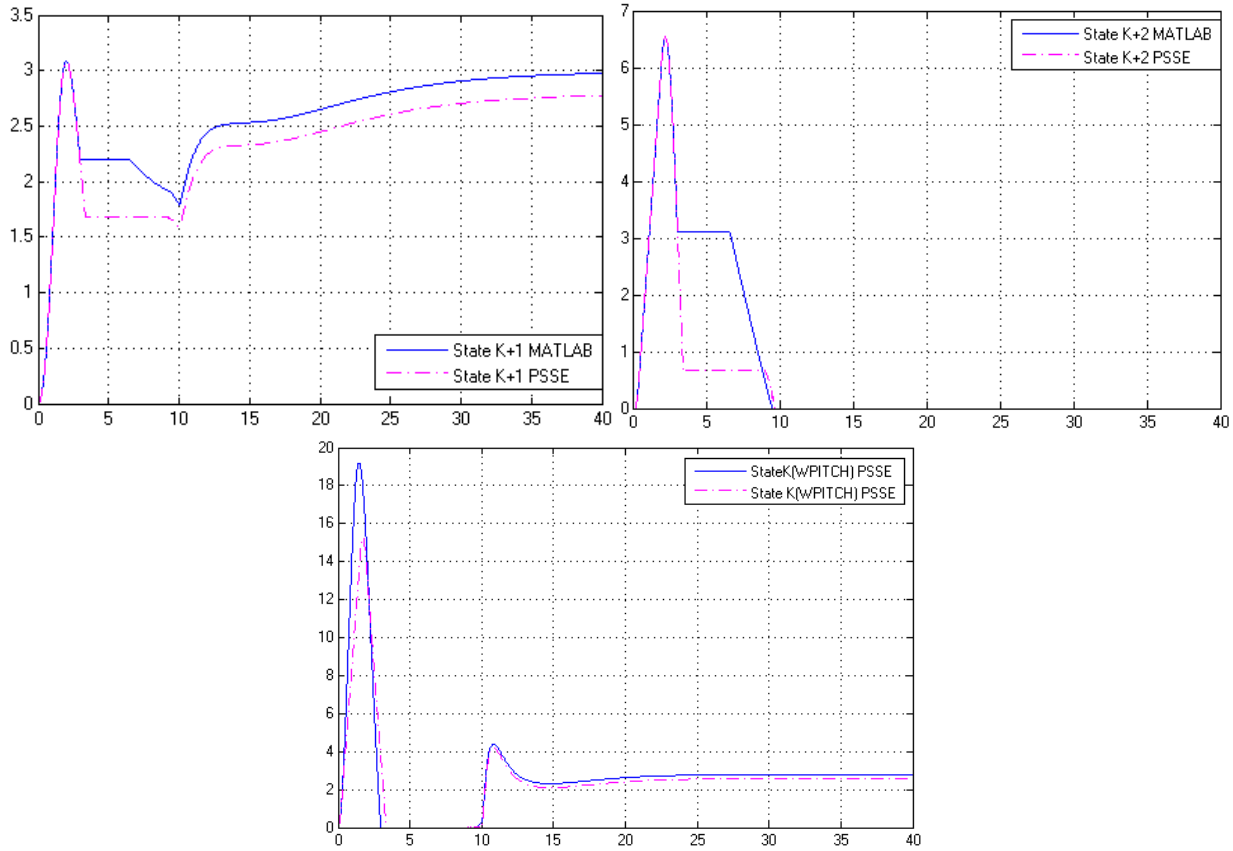


Figure 3-B38. Comparisons between MATLAB and PSS®E for Pitch Model

d. WT3T1

Taking the input WPITCH (state P1 K) and Pelec from PSS®E, we tested the output SPEED (state T1 K+2), WTRBSP, Rotor Angle Deviation (state T1 K+3). (Figure 3-B39)

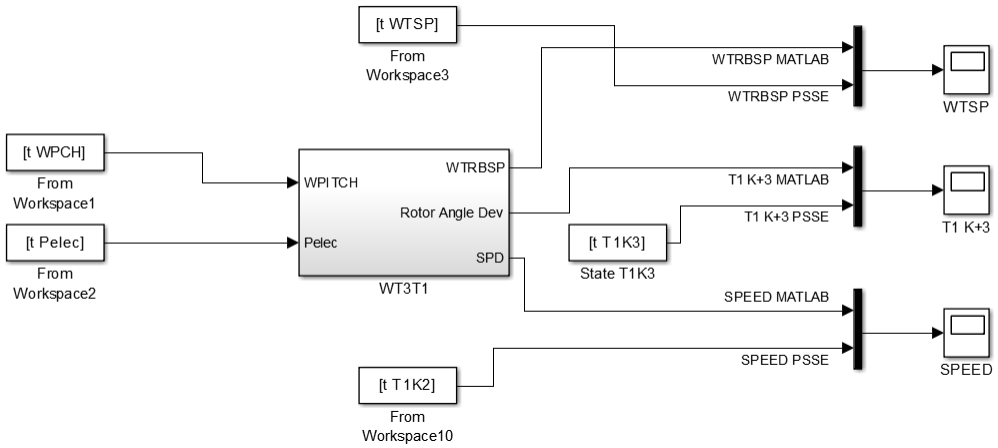


Figure 3-B39. Testing WT3T1 in MATLAB

The results match PSS®E simulation. This comparison is based on Htfrac=0.

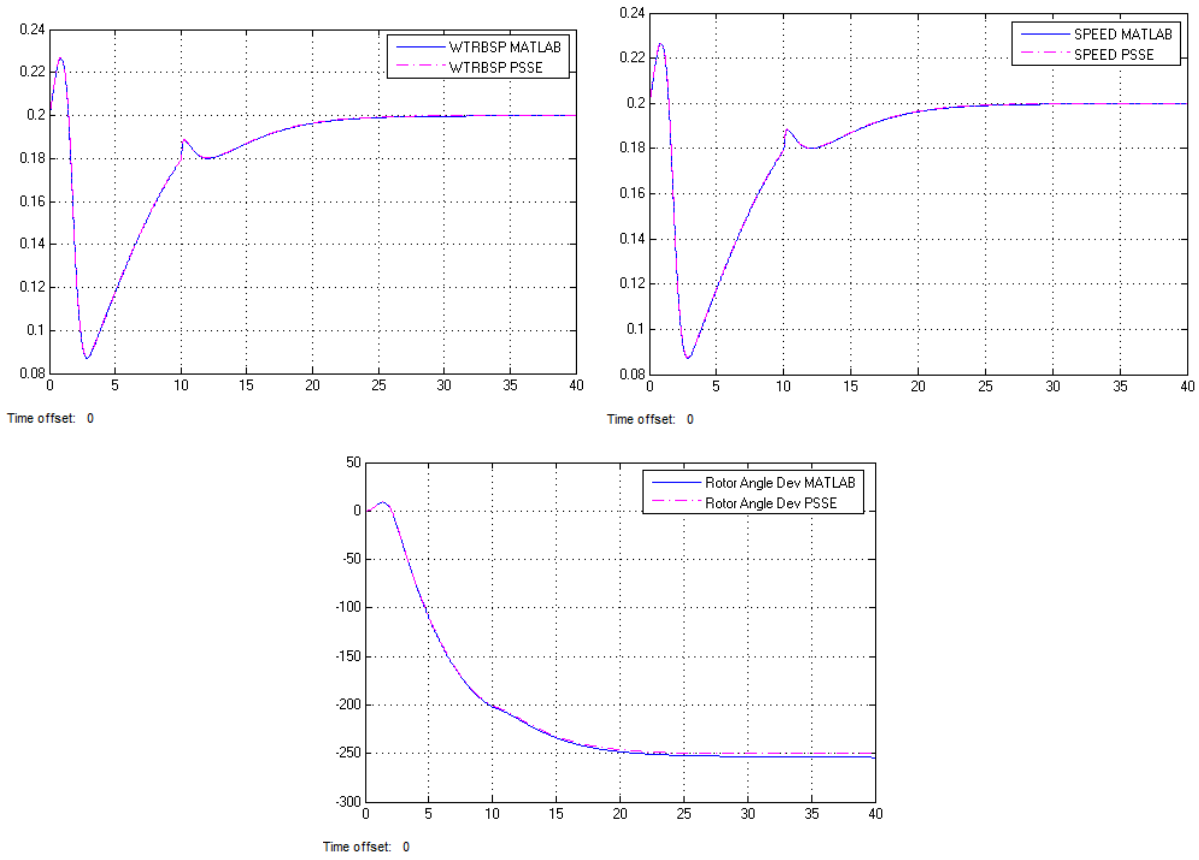


Figure 3-B40. Comparisons between MATLAB and PSS®E for Mechanical Model

IV. DISCUSSION

A. Things Needed Further Investigation

1. Generator model

WT3G1 model works well in PSS®E. When we add a disturbance at 10 seconds, and clear the disturbance at 10.05 seconds, 10.1 seconds, respectively, all values keep constant at their initial set up value from 0 to 10 seconds. But after the disturbance, they go stable at three cycles and unstable at six cycles.

However, WT3G2 model behaves so strange in PSS®E. The running results cannot match with power flow set up, so this is the biggest problem we encounter when exploring WT3G2 model. In addition, all values have same kind of fluctuation before 10 seconds, even though the disturbance is added afterwards, exactly at 10 seconds. Therefore, due to these two undesirable behaviors, PSS®E maybe have some programming error in WT3G2 model which needed to be figure out later.

Besides, parameter "accel" need to be determined in WT3G2 High Voltage Reactive Current Logic, as discussed before.

Furthermore, the lower branch WT3G2, as shown in Figure, gives the phase shift to the T block. Because the sample parameters of Kpll and Kipll equal to zero, angle of Vt is connected to the model but the other branch are not in use. When we tried to set Kpll and Kipll to a different value, state K+2 and K+3 are behaving very weird.

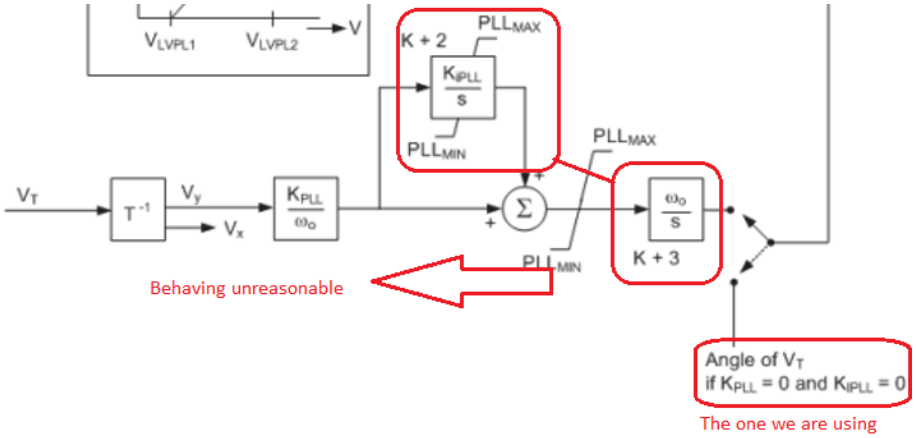


Figure 4-1. WT3G2 Lower Branch

2. Electrical model

The only problem remain mysterious in the most complicated model within Generic WT3 is state K+7. We have already known that there is an non-windup limit on state K+7 and our Simunlink model works very well for WT3G2 input because the control flag mode is set to 2. But for WT3G1 input data, this mode is set to 1 and the limits change is dependent with Vterm magnitude. The result goes too high and cannot go stable by the time going.

It is possible that WEQCMD will drop down when we connected the whole model to the network. The large value of WEQCMD will also increase Vterm and decrease WEQCMD eventually, which is

counteractive and represents the original design idea of generic wind machine-the machine recovers automatically after a disturbance.

3. Mechanical model

The only problem in mechanical model is the initialization of WAEROT. We will discuss this issue in the Initial Value section.

4. Pitch Model

The implementation has been given in Appendix C3. SIEMENS has provided what seems to be the easiest model across all the four, but this turns out with the most complicated anti-windup controls, one of which even has impact on the other two. After figuring out the relations between these controlling models, outputs in MATLAB start to match PSS®E, though not very perfectly.

SIEMENS might also has provided inaccurate diagram. Now we decide WNDSP1 should be STATE K+5 in the WT3E1 model. It is with the in-deep research on WT3E1 model that we can get accurate WNDSP1 and E1 STATE K+5 data from MATLAB, so we can use these data in the pitch model.

By far we have collected data from PSS®E by using both WT3G1 and WT3G2 models. The MATLAB data with WT3G1 almost match PSS®E for the pitch model, but not with WT3G2. With WT3G1 data there is only very minor discrepancy, but with WT3G2 data it is significant. We may still have something missing in this model, but this may be figured out in future works.

5. Interaction of 4 models

When we connected these four models together, MATLAB gave an error for both type 3 wind machine using WT3G1 and WT3G2.

For each individual model like WT3T1, WT3G1 and so on, we are able to match MATLAB outputs with PSS®E. However, when connected together one model may have impact on another, so that may be the reason for unexpected behaviors.

For WT3E1, we have everything work except for WEQCMD. Actually WEQCMD has weird behaviors and we believe this is one factor that render the connected system to behave unexpectedly. Because WEQCMD is fed into WT3G1, we cannot have correct input for it as we did individually for each model. As this is a system that connects four individual models into a loop, if one chain inside the loop fails to work, every chain is going to have problem.

The output results when connected together shows some trend in our simulation results in PSS®E, though MATLAB simulation usually end up with jitters. It is a correct decision that we break the system apart so we can do research on each model individually and carefully investigate the most accurate way of connection inside each model. It is likely that we still need to do the same thing so we can have every small part to be correct, and then we might be able to connect the four models together and get what is expected.

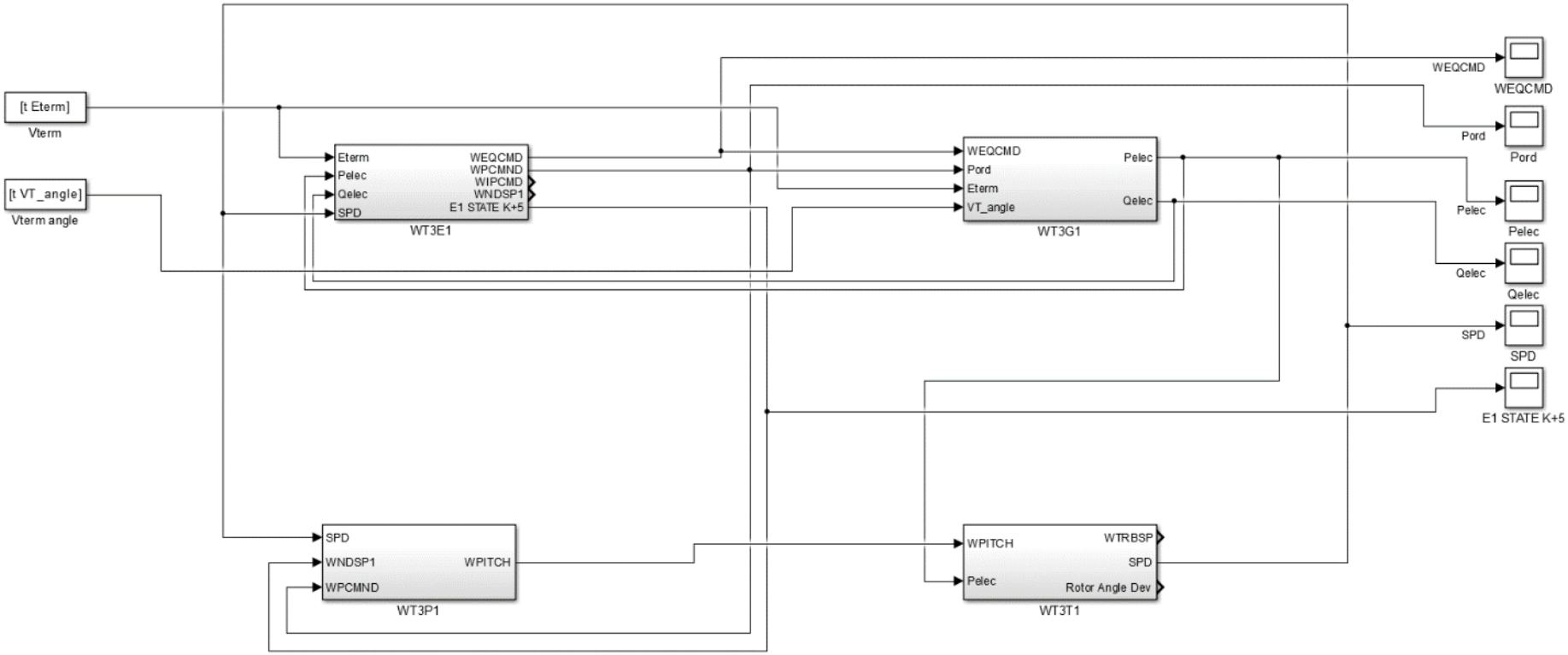


Figure 4-2. Four Models Connected in MATLA

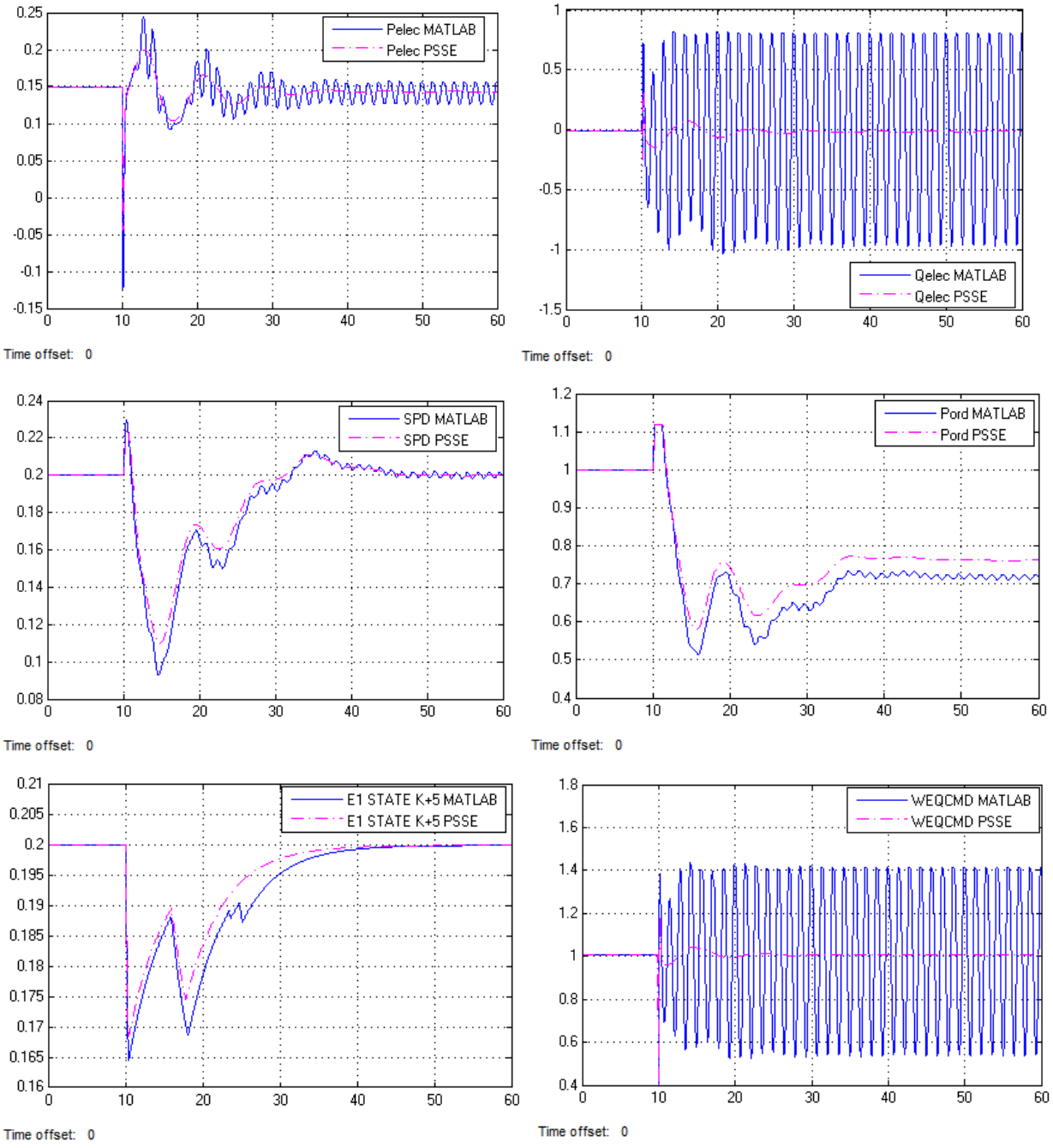


Figure 4-3. Comparisons between MATLAB and PSS®E when Connected

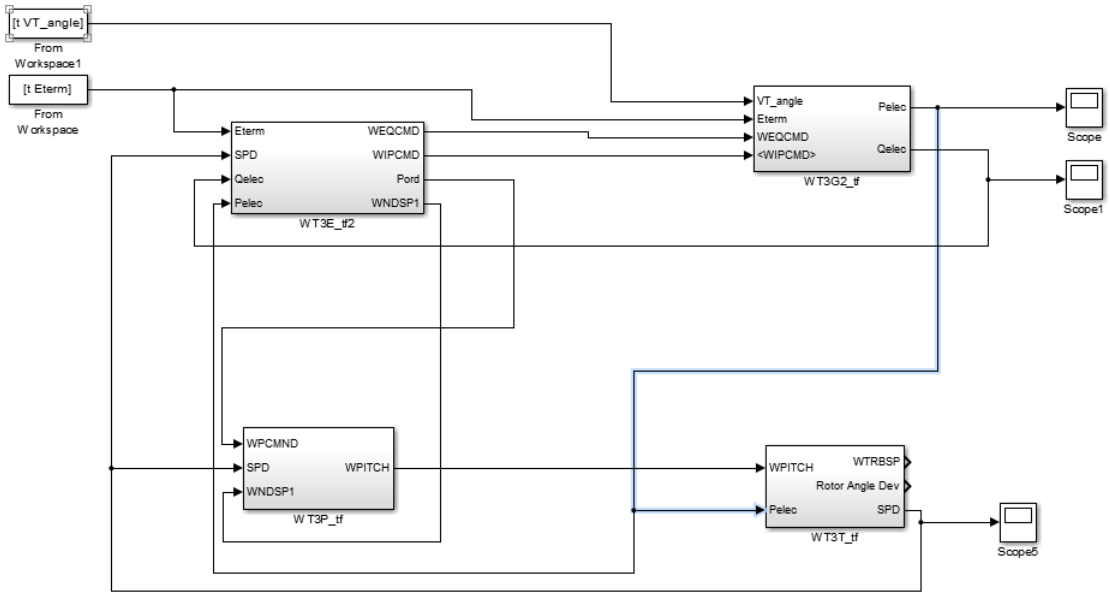


Figure 4-4. Connected Models, Outputting Power and SPD

Derivative input 1 of 'WT3_Combine_0/WT3T_tf/Integrator' at time 10.0075 is Inf or NaN. Stopping simulation. There may be a singularity in the solution. If not, try reducing the step size (either by reducing the fixed step size or by tightening the error tolerances)

Figure 4-5. MATLAB Simulink Runs with Error

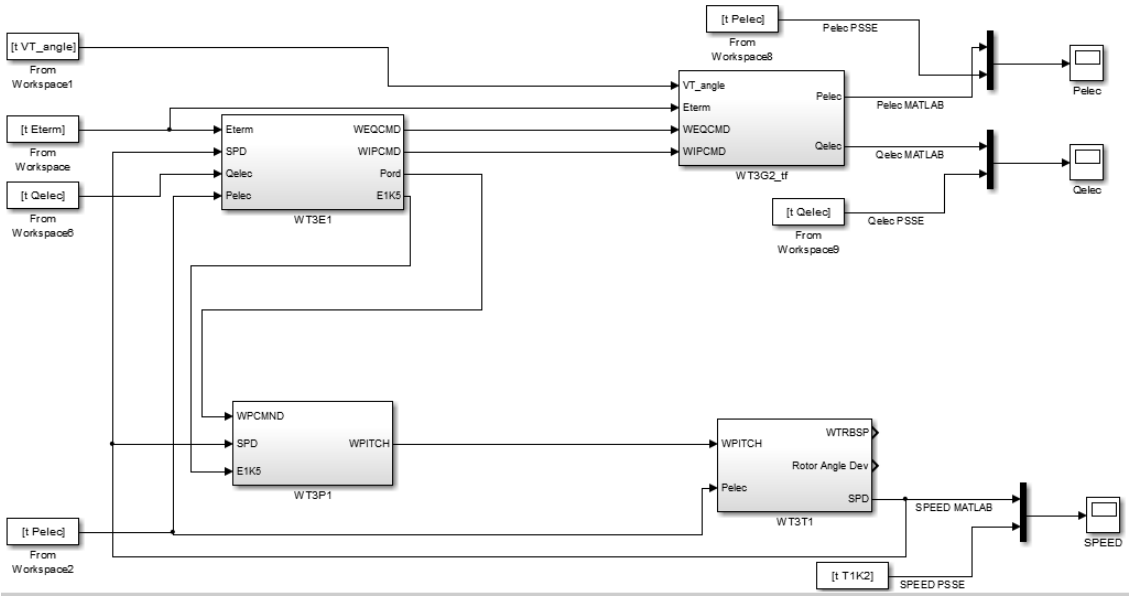


Figure 4-6. Comparing Simulink and PSS®E Data

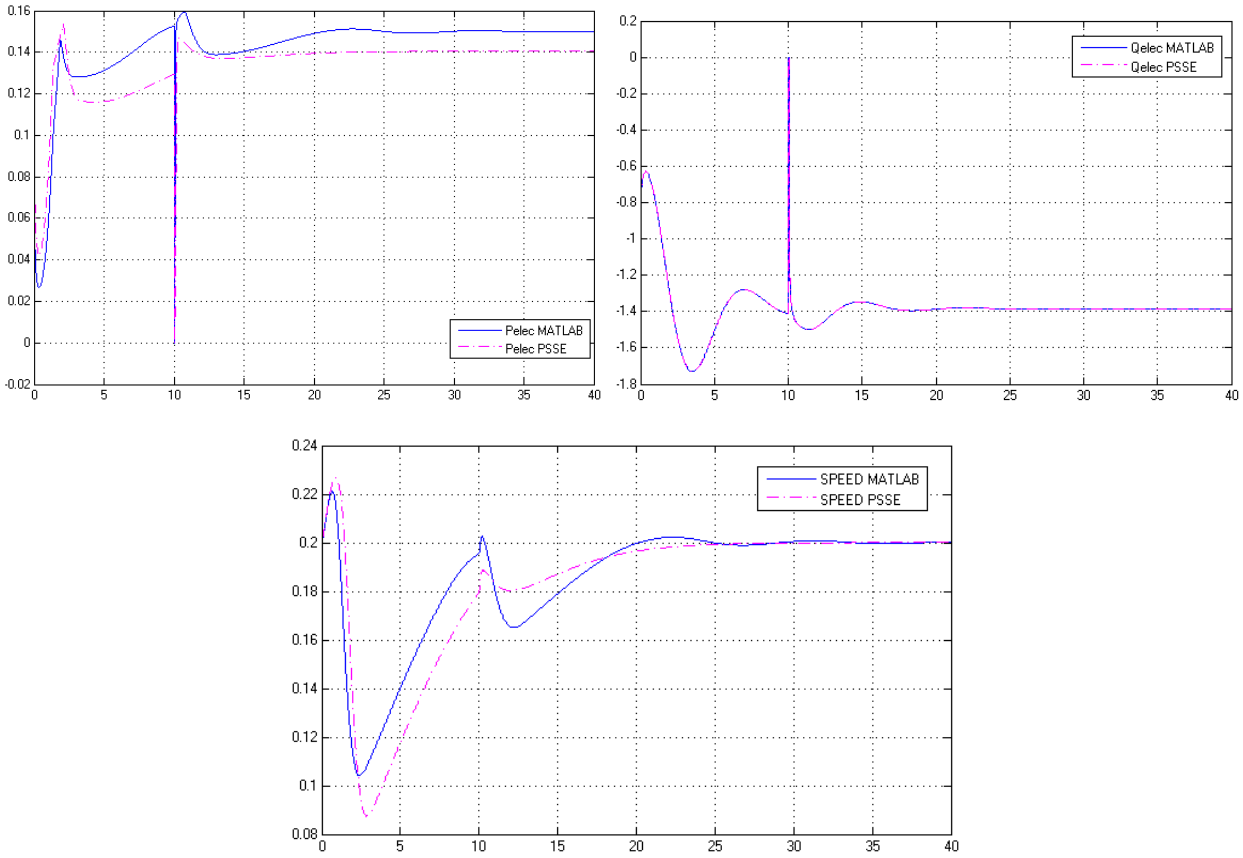


Figure 4-7. MATLAB and PSS®E Comparison for Power and SPEED

6. Determine Initial Values

The way PSS®E does the initialization for all the states and VARs is a mystery, and this is tremendous important if we want to translate our model into C code and do further contribution to the "Faster than Real-Time" Simulator project with Argonne National Laboratory.

We make assumption for initial value determination based on the condition that input equals to output of a lag controller in steady state. The only thing simulator knows about before calculating initial value is Power (Pelec or Pgen), VARS (Qelec or Qgen), Vterm magnitude and Vterm angle. The following discussion covers our assumption on how to make initialization.

First we start at WT3G1. As shown in Figure 4-8, the lower branch has all the initial values set to 0 because the note-"In steady- state, Vy=0, Vx=Vterm, δ=0". The direction of calculation is drawn in the figure to specify the initialization. Complex Vterm, Pelec and Qelec are known, so we can calculation complex Isorc easily. With δ=0, Iyinj and Ixinj can be calculated and we can further get initial values of State K+1 and K. WT3G2 has a similar process instead of δ=Vterm angle.

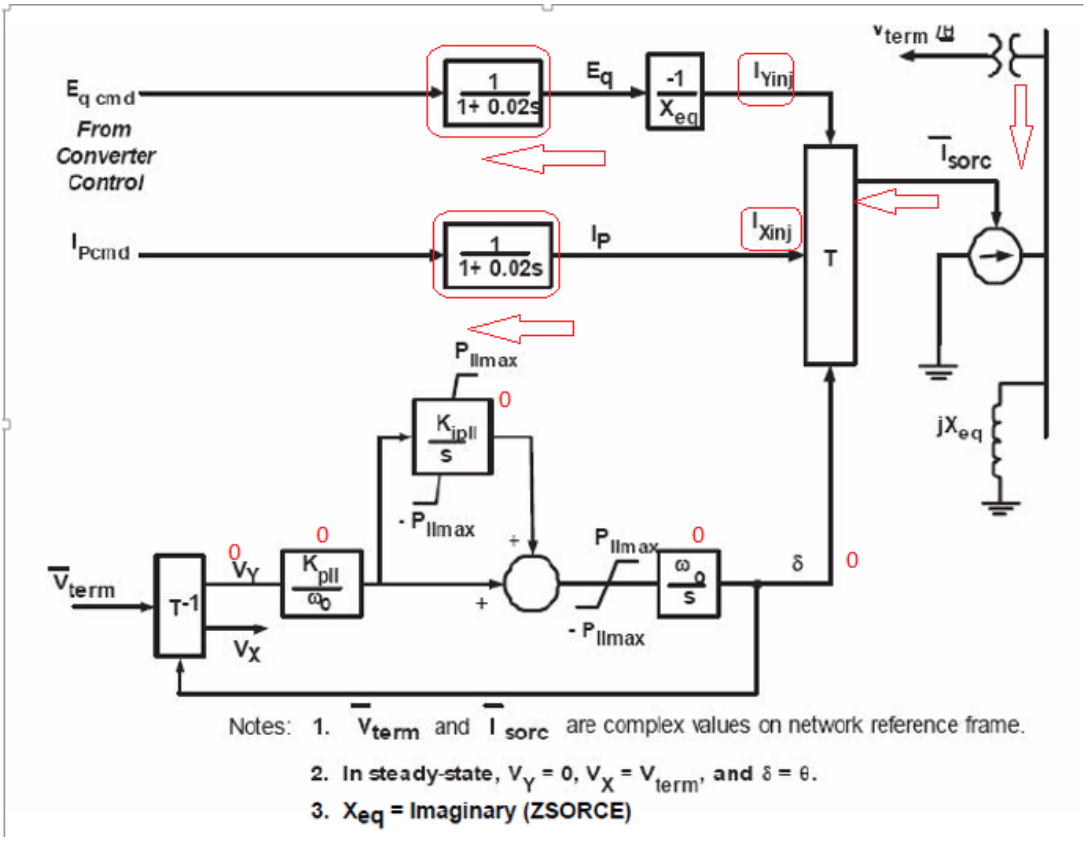


Figure 4-8. WT3G1 Model

Next, calculating WT3E1 will be the best option. In the lower branch of WT3E1, Pelec is known and WNDSP1 can be determined after the function. State K+5 is in steady state, so initial value of K+5 equals to WNDSP1. SPEED should compensate K+5 to give a zero value after different, so SPEED can be known. Then we should jump to the end of this branch and calculate backward. We got WIPCMD in G1 or G2, and Vterm is known, so K+2 is calculated. With a few steps of calculation, we can got K+3 according to the above conditions.

VAR(L) is voltage reference, having the same value of initial Eterm.

WT3T1 initialization is the most difficult one due to the unknown initialization of VAR(L+3) (Initial Pitch Angle). If VAR(L+3) equals to zero, we can easily get the value of WPITCH and solve all calculations in WT3P1. But if VAR(L+3) is not zero, we need to find a way to calculate this value before all the initial values can be determined.

V. CONCLUSION

Through this research project, we have figured out many mysterious parts of type 3 generic wind turbine model which are not revealed in SIEMENS documents by investigating generator model.

In the beginning we built the models with MATLAB exactly following SIEMENS documents. During this process we tried hard to fix compiler errors but were still in vain as the outputs did not match PSS®E at all. Then we realized there are initial values to be set, and confirmed the ways of connection as well. More importantly though, in some of the diagrams SIEMENS indeed has hidden some minor elements like anti-windups, limiters and others. It is really lucky of us that we eventually figured almost everything by looking into each individual block separately and providing input data for each block. Almost everything started to work out, if not all.

There are still some mysteries though, like how the STATE K+7 in WT3E1 should be built up, what key things should be done in WT3G2, how we should connect the models together to make the whole system work and etc. We suspect either there are some programming mistakes in PSS®E, or we are not able to find out something that might be diamond in the rough.

Overall speaking, this project has been a great experience, we were curious, hopeful, desperate, and then hopeful again. We did spend a lot of time figuring some single stuff out, but our efforts eventually paid off, and that’s why we are able to bring up this report.

VI. ACKNOWLEDGEMENTS

This paper is made possible thanks to the support of everybody not only the whole research team of us, but also our family, friends and professors. Please allow us to specifically express our gratitude towards the following individuals.

First, we would like to thank Dr. Alex Flueck for his detailed guidance. As the instructor of our team, he outlined our goals and guided us about the methods of research. Dr. Flueck also contributed huge amount of reference materials so that we can compare one with each other and decide how to most accurately build our models. He is knowledgeable and creative so we are able to get through problems at which we stuck.

Additionally, Yuan Zhi, a graduate student of Dr. Flueck, helped us with using the tools we need in the research. It is because of him that we became familiar with PSS®E, the software we majorly use in this research project for simulation and analysis.

VII. Reference

[1] I. A. Hiskens, *Dynamics of Type-3 Wind Turbine Generator Models*, 2012.

[2] EPRI, *Converter Model Grid Interface*.

[3] EPRI, *WECC Type 3 Wind Turbine Generator Model – Phase II*, Knoxville, Tennessee, 2014.

[4] EPRI, *Proposed Changes to the WECC WT4 Generic Model for Type 4 Wind Turbine Generators*, Knoxville, Tennessee, 2011.

[5] *Typical Machine Data for Power System Simulation Modeling*, 2014.

[6] National Renewable Energy Laboratory, *WECC WIND GENERATOR DEVELOPMENT*, 2010.

[7] Q. Xu and Z. Xu, *A Survey on General Models of Wind Turbine Generators in PSS/E*, Hangzhou, Zhejiang, 2010.

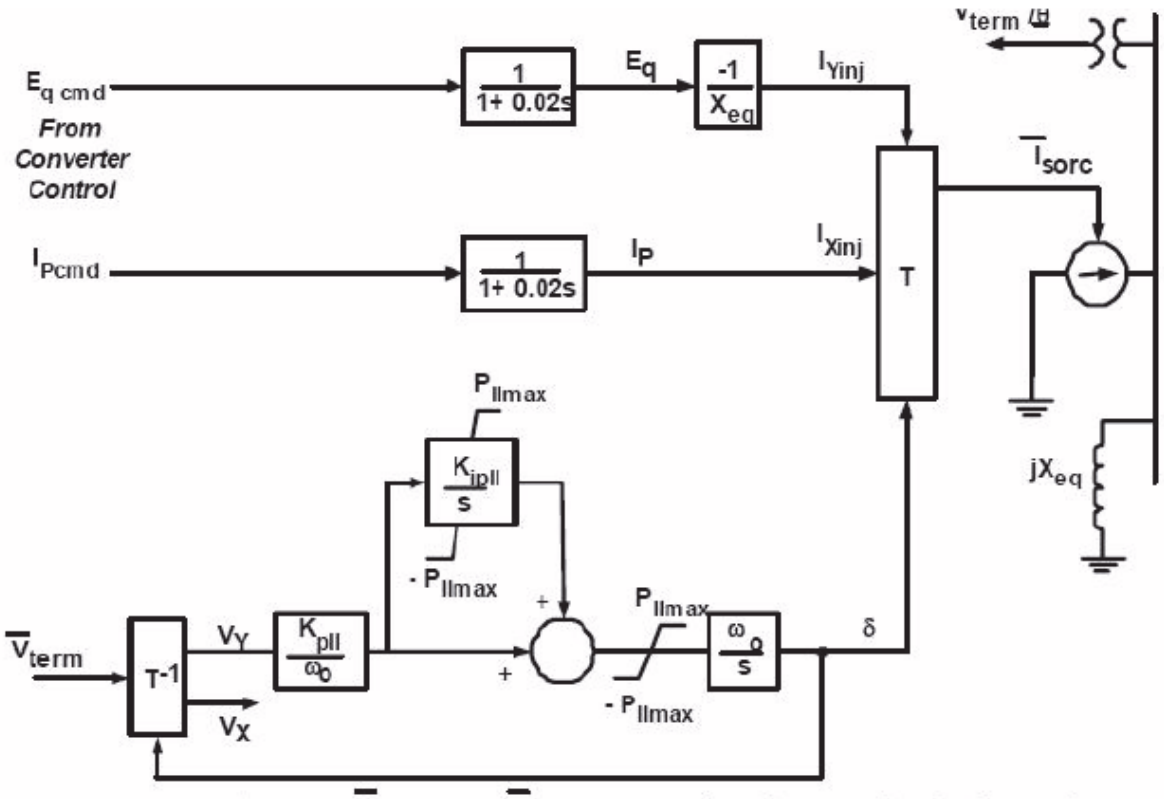
[8] Siemens Energy, Inc., *Memorandum*, 2011.

Appendix

Appendix A Generic Wind Model (Type 3) in PSS(R)

A.1 Doubly-fed induction generator (Type 3)

A.1.1 WT3G1

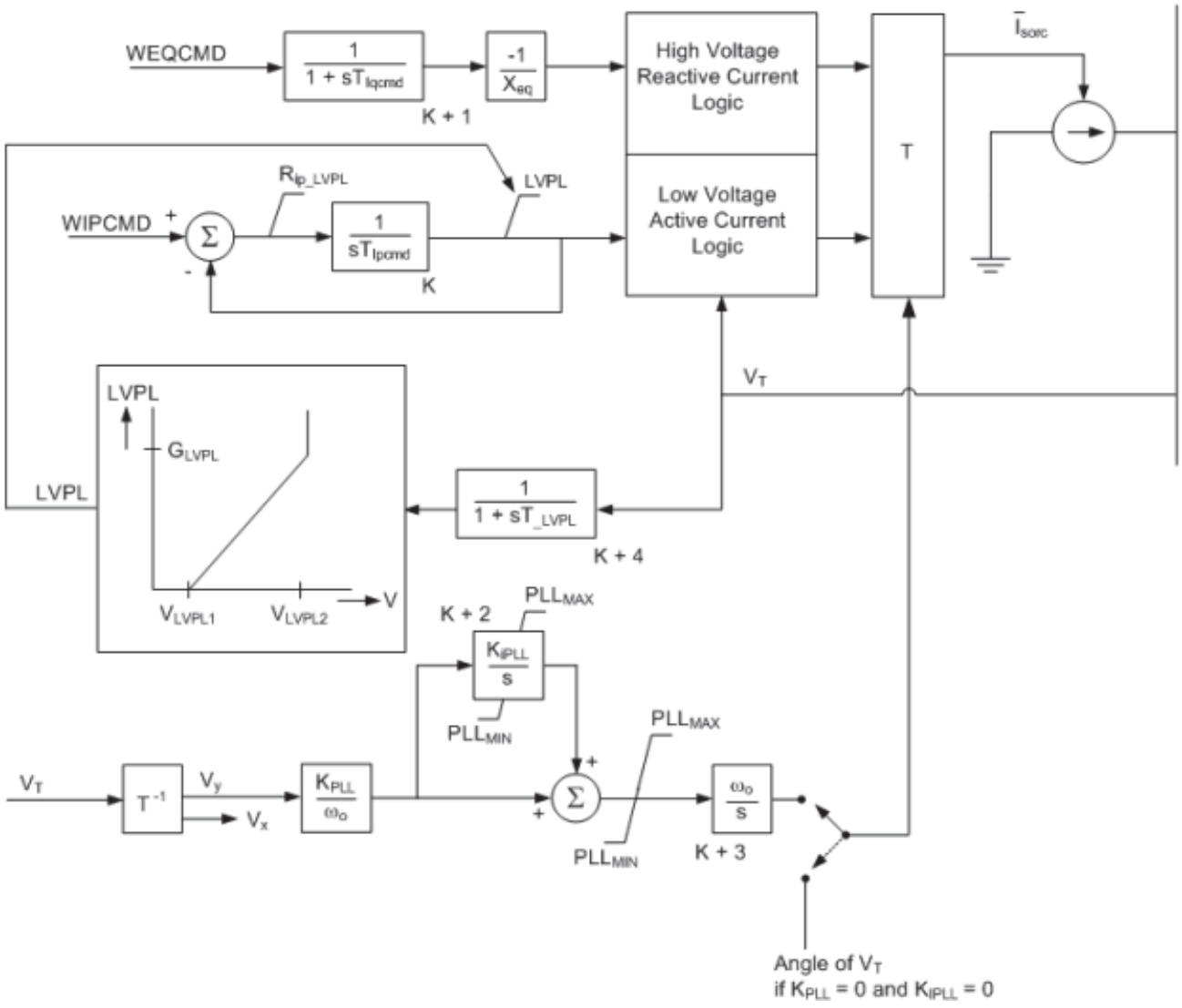


Notes: 1. $\overline{V}_{term}$ and $\overline{I}_{sorc}$ are complex values on network reference frame.

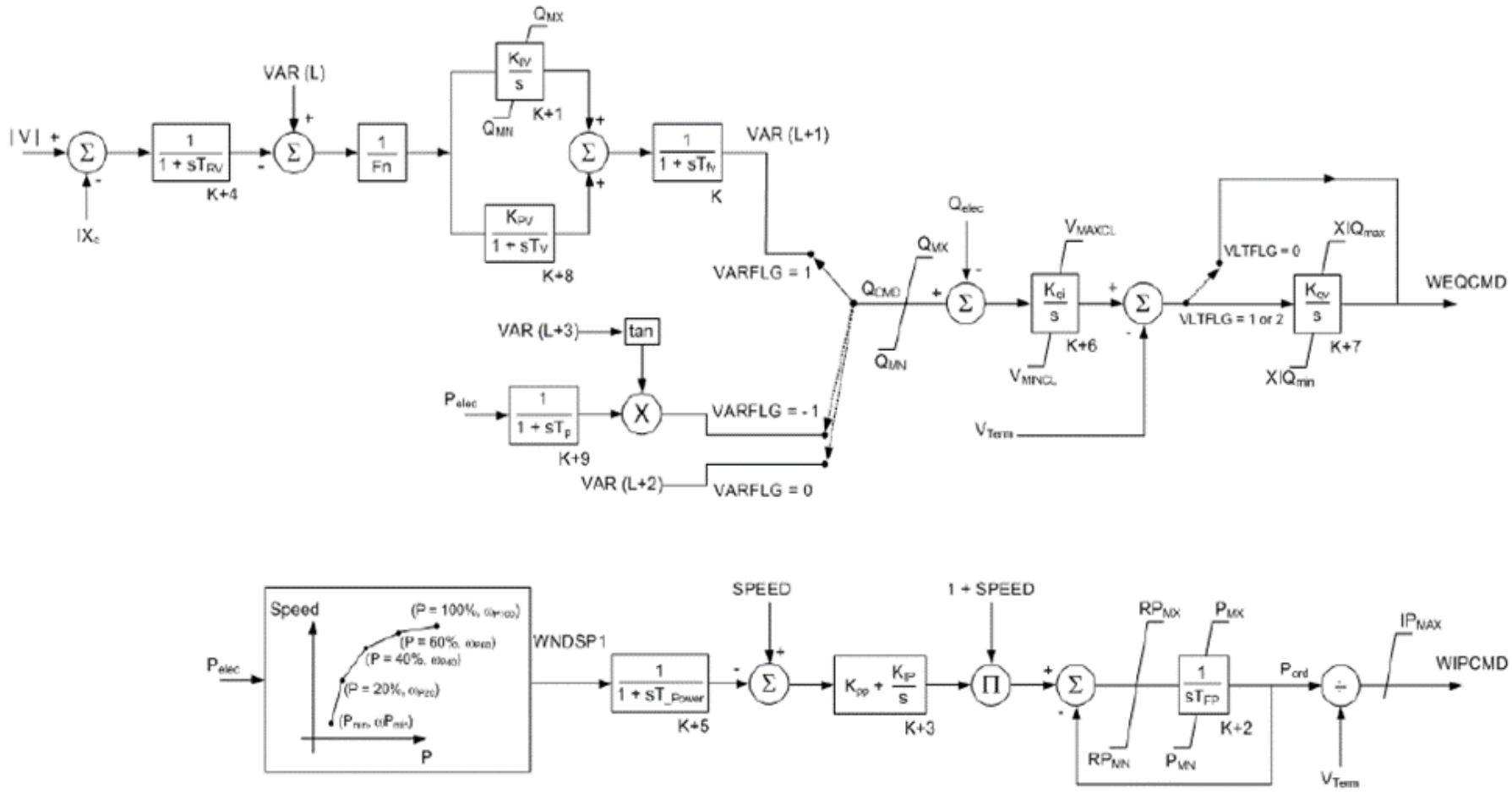
2. In steady-state, $V_Y = 0$, $V_X = V_{term}$, and $\delta = \theta$.

3. X_{eq} = Imaginary (ZSORCE)

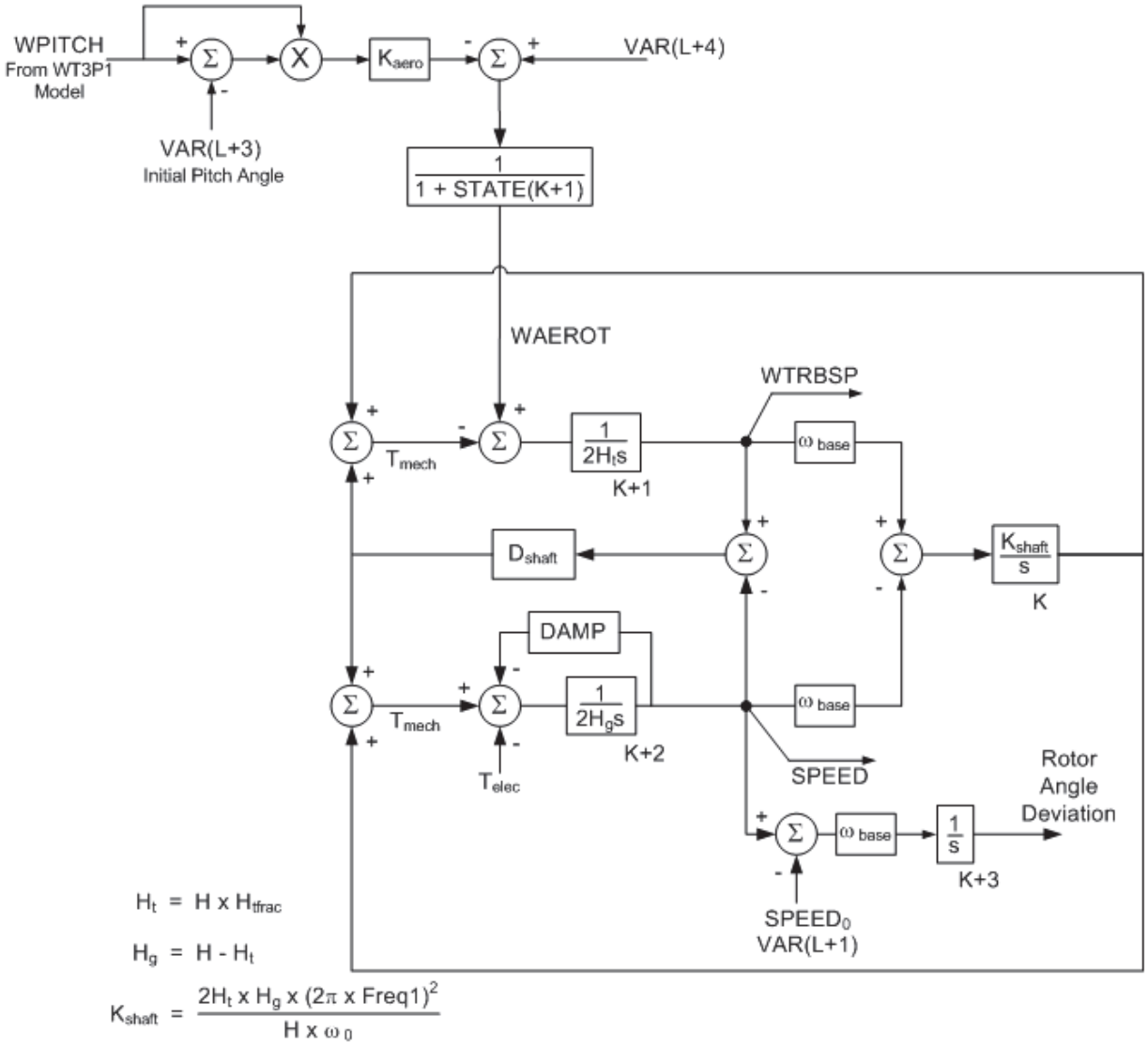
A.1.2 WT3G2


WEQCMD
1 / 1 + sT_Iqcmd
K + 1
-1 / X_eq
High Voltage Reactive Current Logic
Low Voltage Active Current Logic
T
I_sorc
R_Ip_LVPL
LVPL
WIPCMD
+
-
1 / sT_Ipcmd
K
V_T
LVPL
G_LVPL
V_LVPL1
V_LVPL2
V
1 / 1 + sT_LVPL
K + 4
K + 2
PLL_MAX
K_IPLL / s
PLL_MIN
PLL_MAX
V_T
T^-1
V_y
V_x
K_PLL / ω_o
+
+
PLL_MIN
ω_o / s
K + 3
Angle of V_T
if K_PLL = 0 and K_IPLL = 0

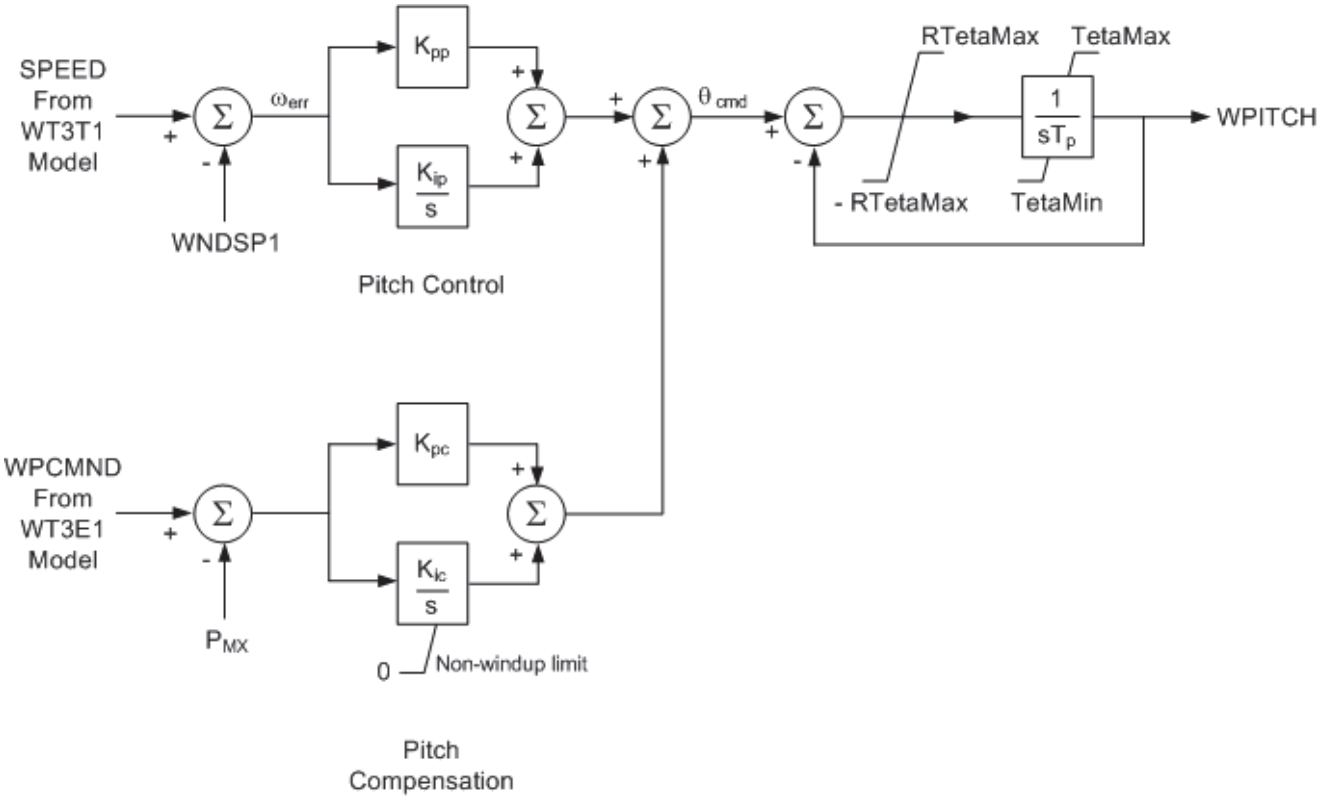
A.2 Electrical control for Type 3 wind generator (WT3E1)



A.3 Mechanical system model for Type 3 wind generator (WT3T1)

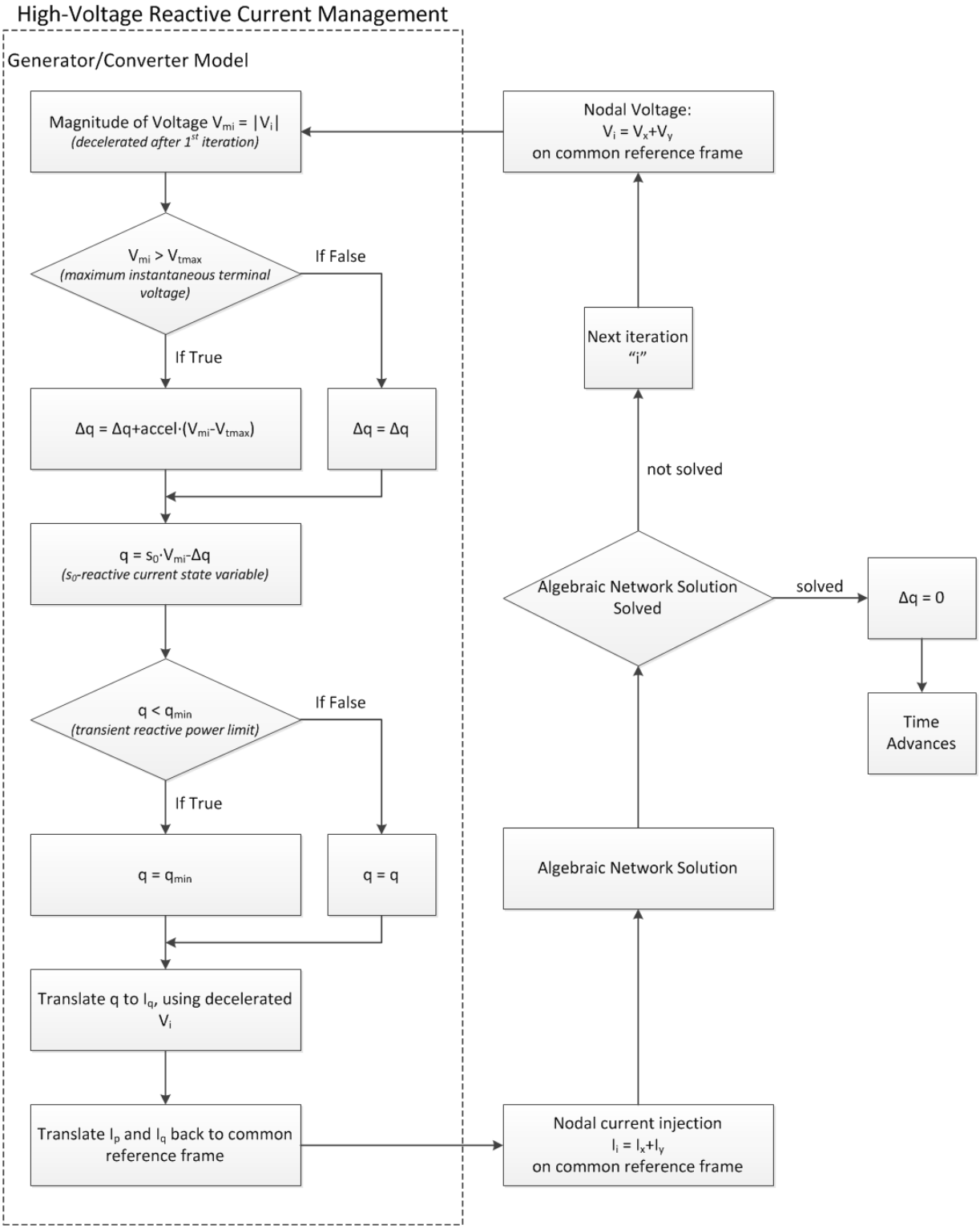


A.4 Pitch control model for Type 3 wind generator (WT3P1)

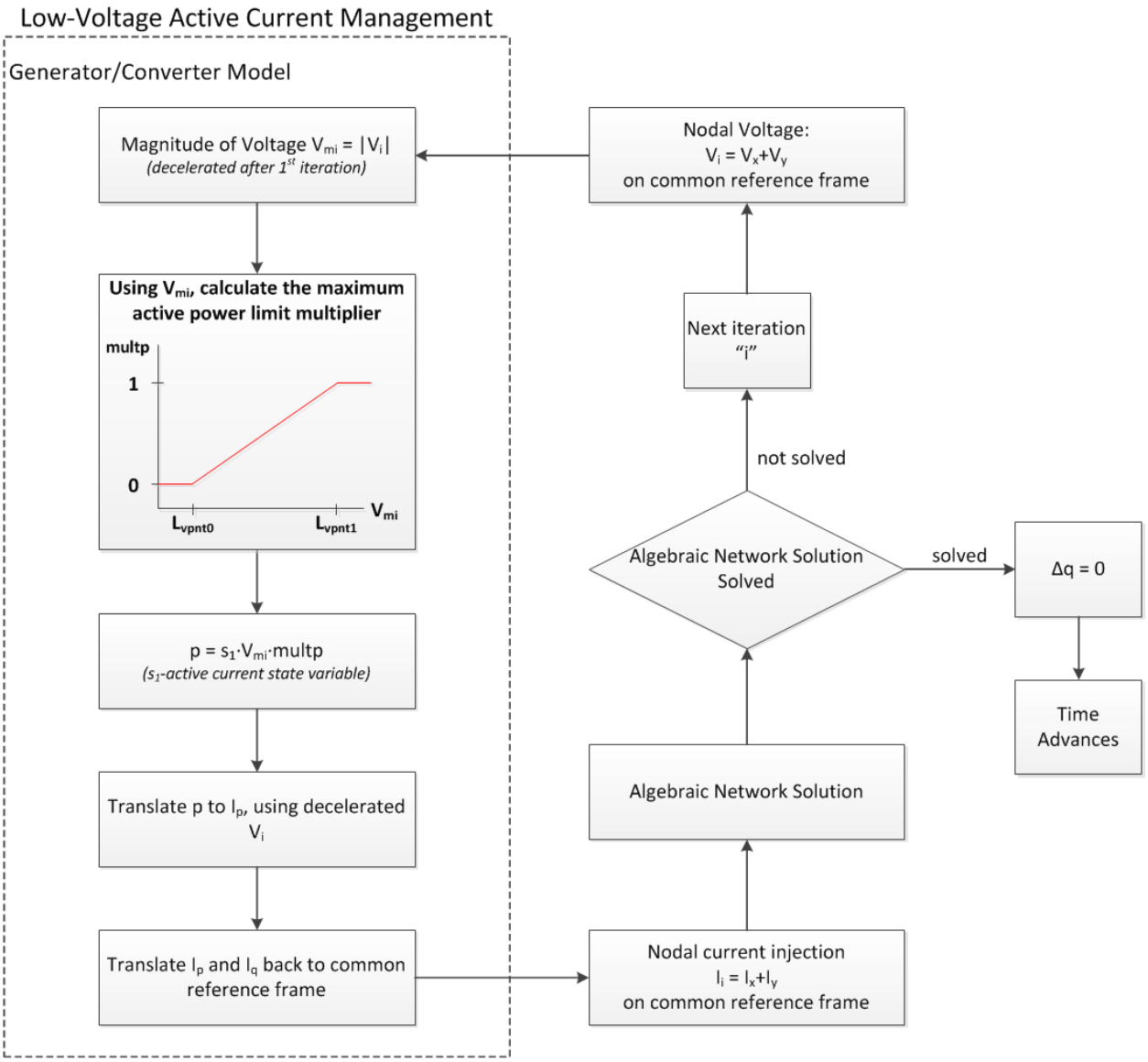


Appendix B Converter Model Grid Interface

B.1 High-Voltage Reactive Power Management



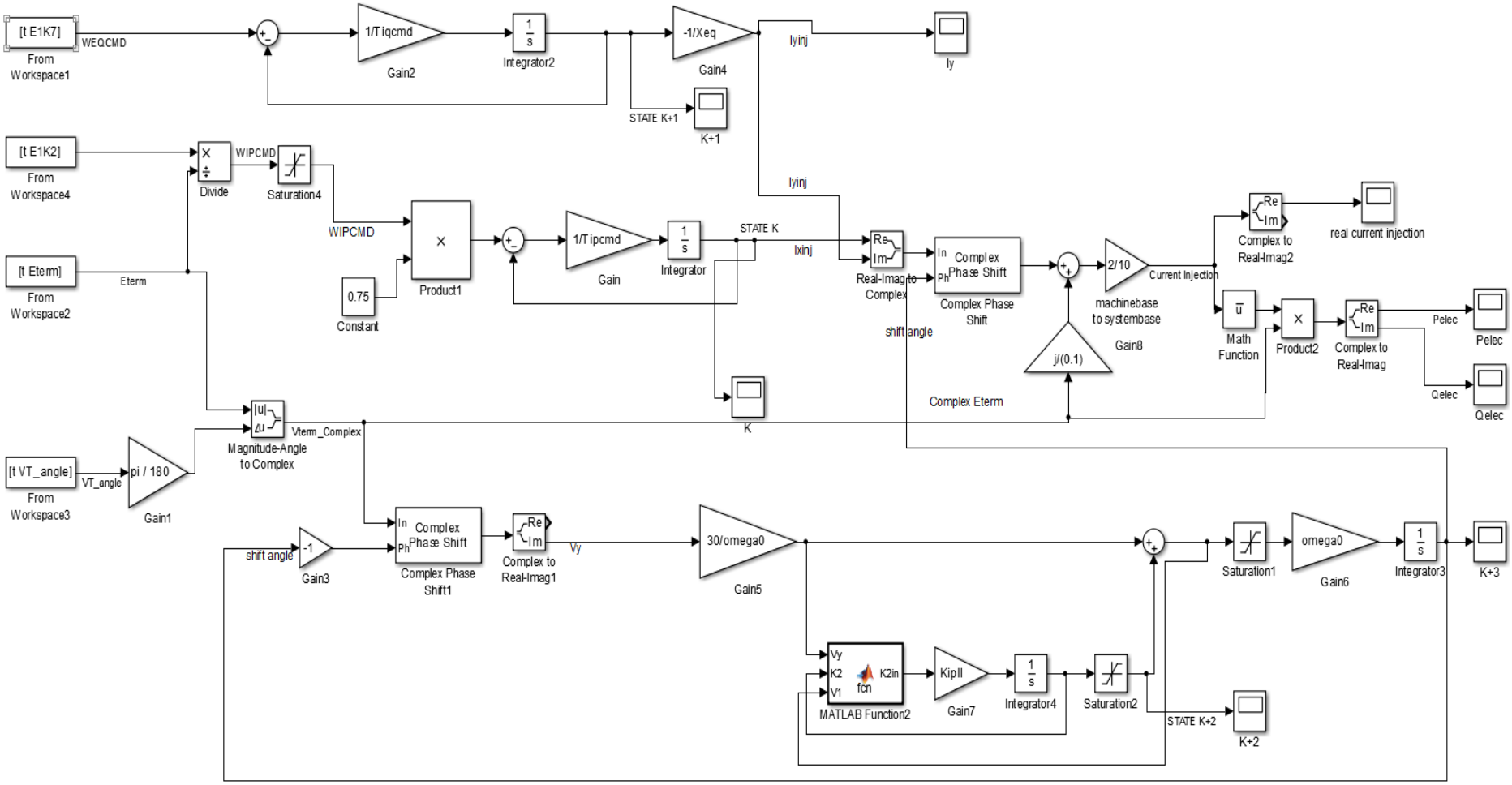
B.2 Low-Voltage Active Current Management



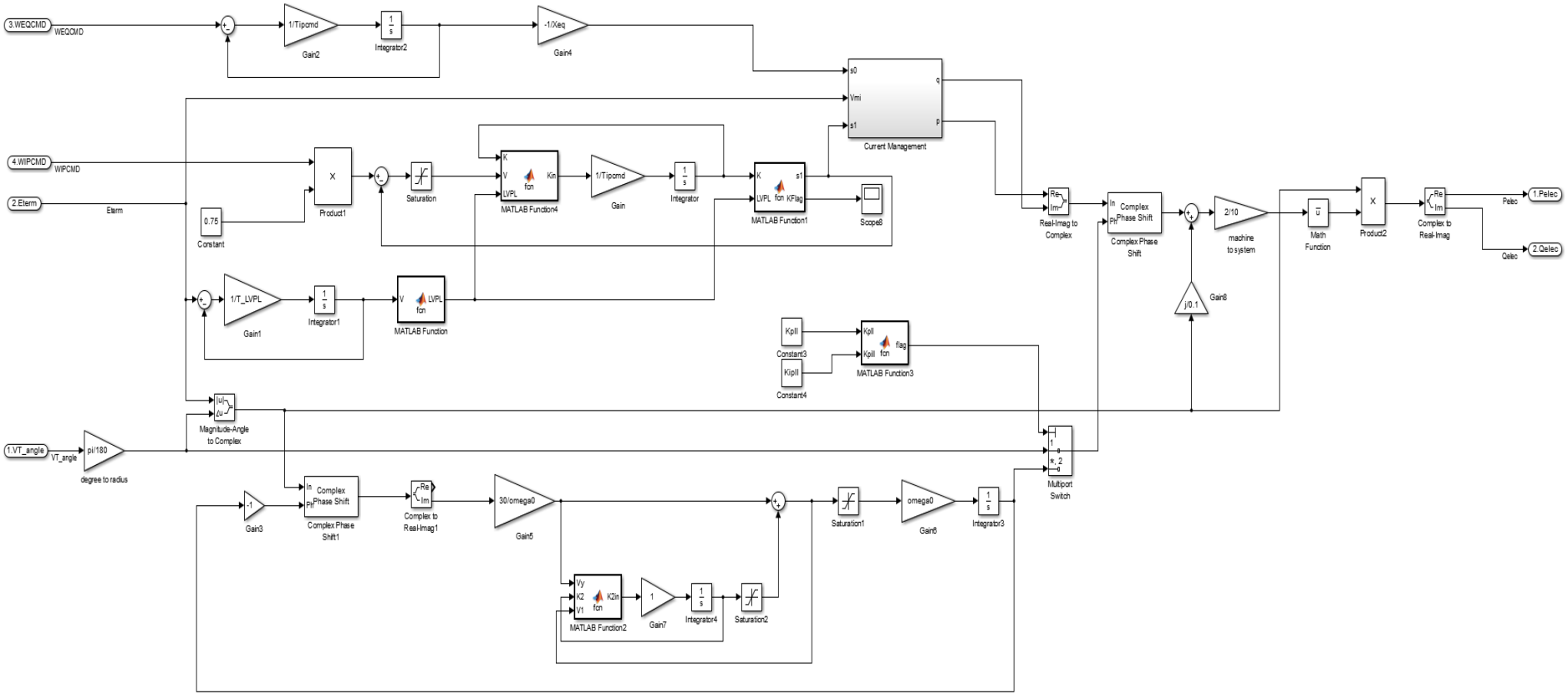
Appendix C MATLAB Simulink Model

C.1

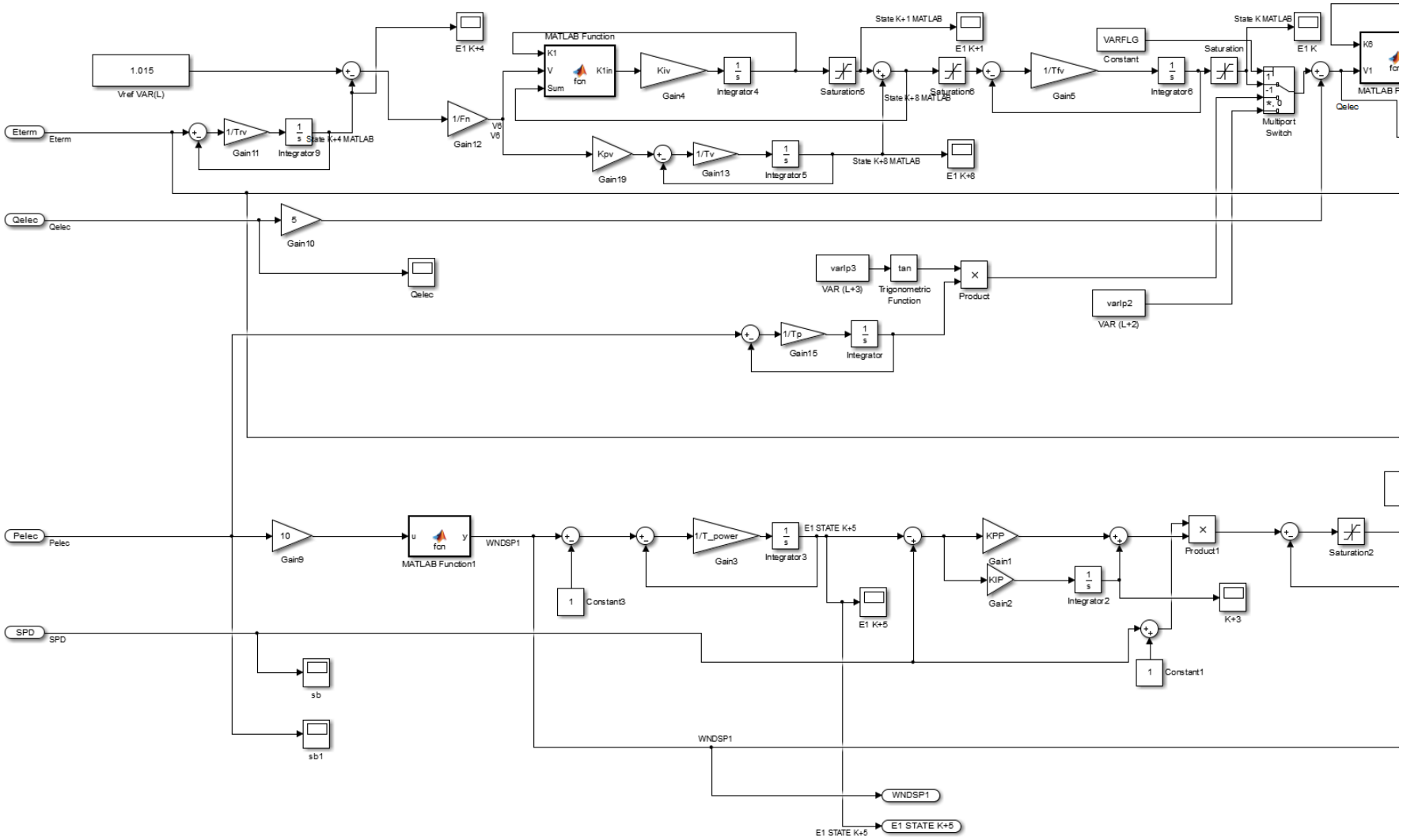
C.1.1 WT3G1

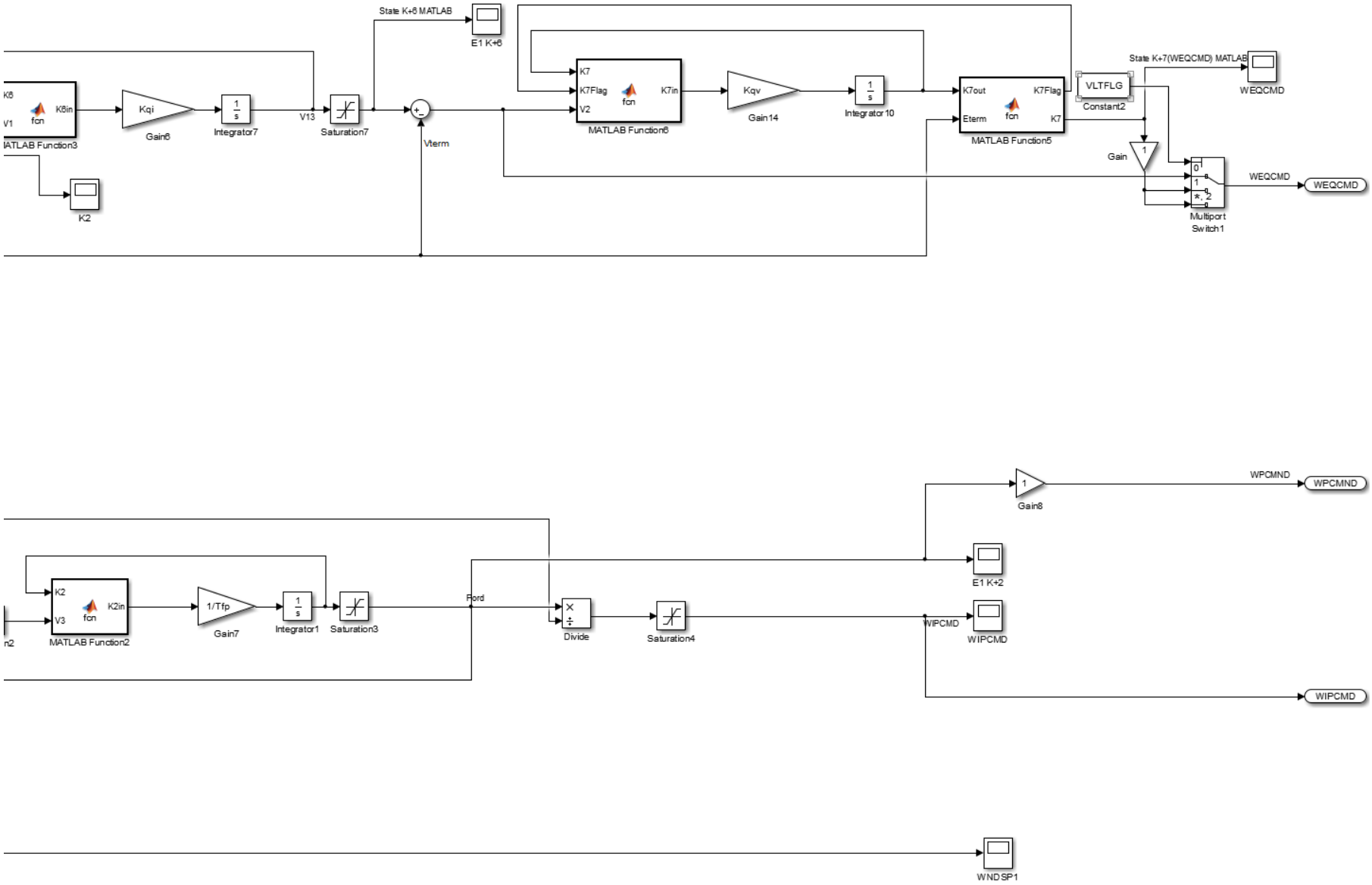


C.1.2 WT3G2

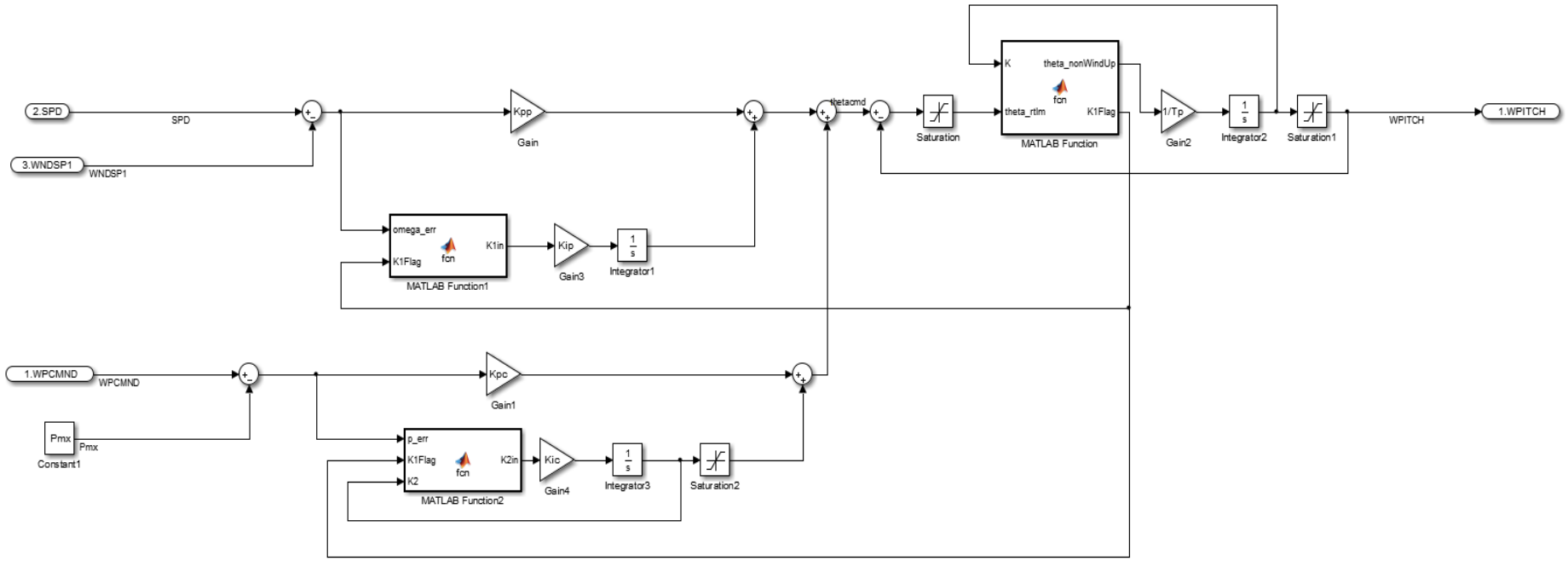

3.WEQCMD
WEQCMD
1/Tipcmd
Gain2
1
s
Integrator2
-1/Xeq
Gain4
s0
Vmi
s1
q
p
Current Management
4.WIPCMD
WIPCMD
2.Eterm
Eterm
0.75
Constant
X
Product1
Saturation
K
V
LVPL
fcn
Kin
MATLAB Function4
1/Tipcmd
Gain
1
s
Integrator
K
LVPL
fcn
s1
KFlag
MATLAB Function1
Scope8
Re
Im
Real-Imag to Complex
In
Pi
Complex Phase Shift
Complex Phase Shift
2/10
machine to system
u
Math Function
X
Product2
Re
Im
Complex to Real-Imag
1.Pelec
Pelec
2.Qelec
Qelec
1/T_LVPL
Gain1
1
s
Integrator1
V
LVPL
fcn
MATLAB Function
j/0.1
Gain8
KpII
Constant3
KipII
Constant4
KpII
KipII
fcn
flag
MATLAB Function3
|u|
Δu
Magnitude-Angle to Complex
1.VT_angle
VT_angle
pi/180
degree to radius
-1
Gain3
In
Pi
Complex Phase Shift
Complex Phase Shift1
Re
Im
Complex to Real-Imag1
30/omega0
Gain5
Vy
K2
V1
K2in
fcn
MATLAB Function2
1
Gain7
1
s
Integrator4
Saturation2
Saturation1
omega0
Gain6
1
s
Integrator3
1
2
Multiport Switch

C.2 WT3E1

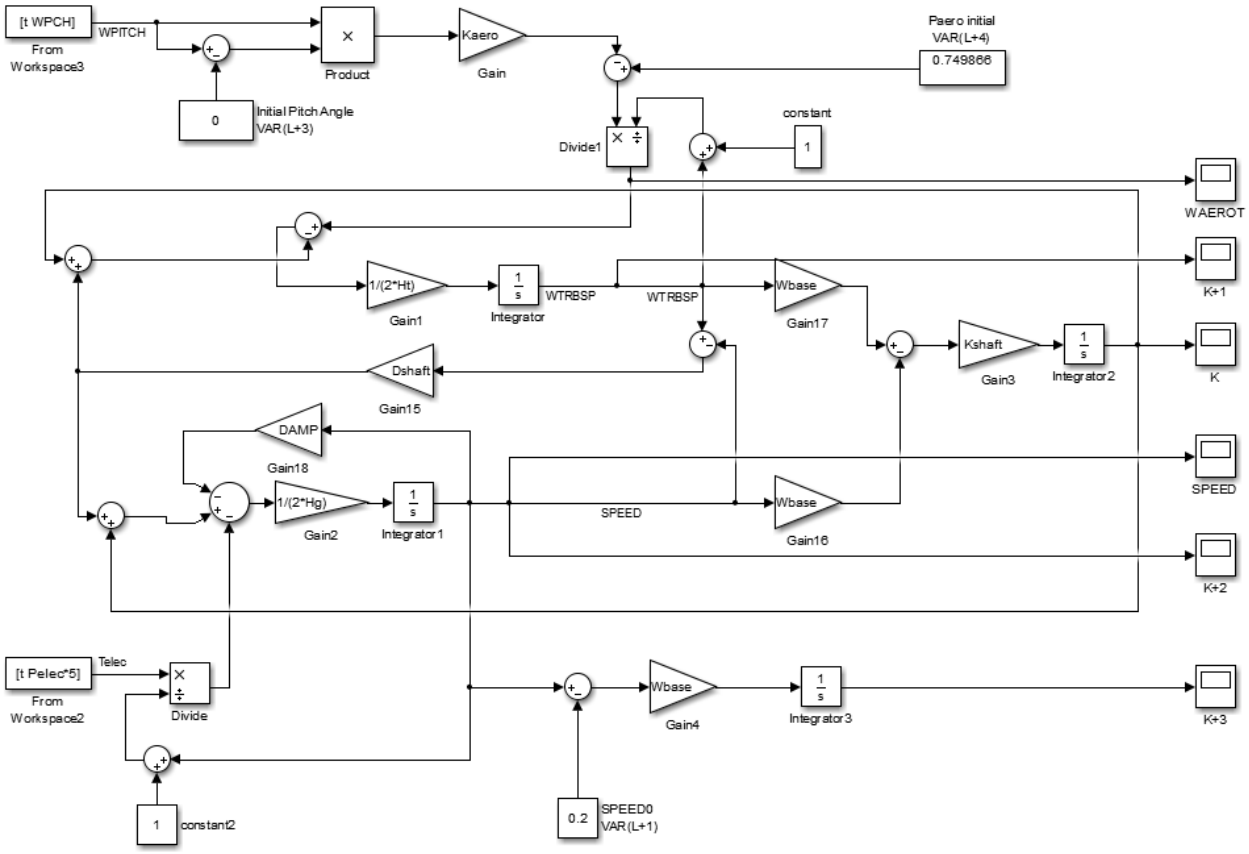

1.015
Vref VAR(L)
E1 K+4
MATLAB Function
K1
V
Sum
fcn
K1in
Kiv
Gain4
Integrator4
Saturation5
State K+1 MATLAB
E1 K+1
Saturation6
State K+8 MATLAB
1/Tfv
Gain5
VARFLG
Constant
Integrator6
Saturation
State K MATLAB
E1 K
Multiport Switch
MATLAB F
Qelec
Eterm
1/Trv
Gain11
Integrator9
State K+4 MATLAB
1/Fn
Gain12
V6
Kpv
Gain19
1/Tv
Gain13
Integrator5
State K+8 MATLAB
E1 K+8
Qelec
5
Gain10
varlp3
VAR (L+3)
tan
Trigonometric Function
Product
varlp2
VAR (L+2)
1/Tp
Gain15
Integrator
Pelec
10
Gain9
u
y
fcn
MATLAB Function1
WNDSP1
1
Constant3
1/T_power
Gain3
Integrator3
E1 STATE K+5
E1 K+5
KPP
Gain1
KIP
Gain2
Integrator2
Product1
K+3
Saturation2
1
Constant1
SPD
sb
sb1
WNDSP1
WNDSP1
E1 STATE K+5
E1 STATE K+5


State K+6 MATLAB
E1 K+6
K6
V1
K6in
fcn
ATLAB Function3
Kqi
Gain6
1/s
Integrator7
V13
Saturation7
Vterm
K2
K7
K7Flag
V2
K7in
MATLAB Function6
Kqv
Gain14
Integrator10
K7out
K7Flag
Eterm
K7
MATLAB Function5
VLTFLG
Constant2
State K+7(WEQCMD) MATLAB
WEQCMD
Gain
Multiport Switch1
WEQCMD
WEQCMD
K2
V3
K2in
MATLAB Function2
1/Tfp
Gain7
Integrator1
Saturation3
Ford
Divide
Saturation4
Gain8
WPCMND
WPCMND
E1 K+2
WIPCMD
WIPCMD
WIPCMD
WNDSP1

C.3 WT3P1


2.SPD
SPD
3.WNDSP1
WNDSP1
Kpp
Gain
thetacmd
Saturation
K
theta_nonWindUp
fcn
theta_rtlm
K1Flag
MATLAB Function
1/Tp
Gain2
1
s
Integrator2
Saturation1
WPITCH
1.WPITCH
omega_err
K1Flag
fcn
K1in
MATLAB Function1
Kip
Gain3
Integrator1
1.WPCMND
WPCMND
Kpc
Gain1
Pmx
Pmx
Constant1
p_err
K1Flag
K2
fcn
K2in
MATLAB Function2
Kic
Gain4
Integrator3
Saturation2

C.4 WT3T1


[t WPCH]
From Workspace3
WPITCH
Product
Kaero
Gain
Paero initial
VAR(L+4)
0.749866
0
Initial Pitch Angle
VAR(L+3)
Divide1
constant
1
WAEROT
K+1
1/(2*Ht)
Gain1
Integrator
WTRBSP
WTRBSP
Wbase
Gain17
Kshaft
Gain3
Integrator2
K
Dshaft
Gain15
DAMP
Gain18
SPEED
1/(2*Hg)
Gain2
Integrator1
SPEED
Wbase
Gain16
K+2
[t Pelec*5]
From Workspace2
Telec
Divide
Wbase
Gain4
Integrator3
K+3
1
constant2
0.2
SPEED0
VAR(L+1)